ONTOLOGICAL ENGINEERING: A STATE OF THE ART

Asunción Gómez-Pérez

Facultad de Informática
Universidad Politécnica de Madrid
Campus de Montegancedo s/n
Boadilla del Monte, 28660. Madrid. Spain.
Tel: (34-1) 336-74-39, Fax: (34-1) 336-74-12
Email: asun@delicias.dia.fi.upm.es

1 INTRODUCTION

In 1991, the ARPA Knowledge Sharing Effort (Neches et al., 91) revolutionized the way in which intelligent systems were built. They proposed the following: "Building Knowledge-based systems today usually entails constructing new knowledge bases from scratch. It could be done by assembling reusable components. Systems developers would then only need to worry about creating the specialized knowledge and reasoners new to the specific task of their system, using them to perform some of its reasoning. In this way, declarative knowledge, problem-solving techniques and reasoning services would all be shared among systems. This approach would facilitate building bigger and better systems cheaply..."

Since them, considerable progress has been made in developing the conceptual bases for building technology that allows knowledge-component reuse and sharing. To enable sharing and reuse of knowledge and reasoning behavior across domains and tasks, Ontologies and Problem Solving Methods (PSMs) have been developed. Ontologies are concerned with static domain knowledge and PSMs with dynamic reasoning knowledge. The integration of ontologies and PSMs is a possible solution to the "interaction problem" (Bylander et al., 88) which states that "representing knowledge for the purpose of solving some problem is strongly affected by the nature of the problem and the inference strategy to be applied to the problem.". Through ontologies and PSMs this interaction can be made explicit and taken into consideration.

The aims of this paper are to provide answers to the following questions: What is an ontology? What principles should I follow to build an ontology? What are the components of an ontology? What types of ontologies already exist? How are ontologies organized in libraries? What methods should I use to build my own ontology? Which techniques are appropriate for each step? How software tools support the process of building and using ontologies? What are the most well-known ontologies? What are the uses of ontologies? Which principles should I use to select the best ontology for my application? Etc. To answer the above questions, the paper is organized as follows. First, the theoretical foundations of the ontological engineering field will be presented. This will be followed by a presentation of the most well-known ontologies, methodologies for building ontologies, tools and languages for building ontologies, and uses of ontologies in applications.

2 THEORETICAL FOUNDATIONS

2.1 What is an ontology?

The word ontology has been taken from Philosophy, where it means a systematic explanation of Existence. In the Artificial Intelligent field, first Neches and colleagues (Neches et al., 91) defined an ontology as follows An ontology defines the basic terms and relations comprising the vocabulary of a topic area as well as the rules for combining terms and relations to define extensions to the vocabulary. We can say that this definition tells us how to proceed to build an ontology, giving us vague guidelines: identify basic terms and relations between terms, identify rules to combine them, provide definitions of such terms and relations. Note that according this definition, an ontology includes not only the terms that are explicitly defined in it, but terms that can be inferred using rules. Later, in 1993, Gruber's definition (Gruber, 93) becomes famous "an ontology is an explicit specification of a conceptualization", being this definition the most referenced in the literature. In 1997, Borst (Borst, 97) slightly modify Gruber's definition saying that: "Ontologies are defined as a formal specification of a shared conceptualization." These two definitions have been explained by Studer and Colleagues (Studer et al., 98) as follows: "Conceptualization refers to an abstract model of some phenomenon in the world by having identified the relevant concepts of that phenomenon. Explicit means that the type of concepts used, and the constraints on their use are explicitly defined. Formal refers to the fact that the ontology should be machine-readable. Shared reflects the notion that an ontology captures consensual knowledge, that is, it is not private to some individual, but accepted by a group."

Since Gruber's definition, many definitions of what an ontology is have been proposed in the literature. In 1995, Guarino and Giaretta (Guarino et al., 95) collected seven definitions and provided syntactic and semantic interpretations of them. However, other authors provide definition based on the approach they take to build their ontologies. For Swartout (Swartout et al., 97) and colleagues, "an ontology is a hierarchically structured set of terms for describing a domain that can be used as a skeletal foundation for a knowledge base". This definition is based on the fact that they build domain-specific ontologies by identifying the relevant terms to a particular domain in the ontology SENSUS (which includes more than 50,000 terms). Then, they prune the SENSUS ontology using a kind of heuristics. A different approach is taken by Bernaras and colleagues (Bernaras et al., 96). They build a preliminary ontology from a knowledge base, which is refined and augmented with new definitions if new applications are built. This is the cause why the propose the following definition. "An ontology provides the means for describing explicitly the conceptualization behind the knowledge represented in a knowledge base.".

As main conclusion to this section, we can say that the literature provides a bunch of definitions of the word ontology. Different definitions provide different and complementary points of view of the same reality.

2.2 What principles should I follow to build ontologies?

Here we summarize some design criteria and a set of principles that have been proved useful in the development of ontologies.

- Clarity and Objectivity (Gruber, 93), which means that the ontology should provide the meaning of defined terms by providing objective definitions and also natural language documentation.
- Completeness (Gruber, 93), which means that a definition expressed by a necessary and sufficient conditions is preferred over a partial definition (defined only by a necessary or sufficient condition).
- Coherence (Gruber, 93), to permit inferences that are consistent with the definitions.
- Maximize monotonic extendibility (Gruber, 93). It means that new general or specialized terms should be included in the ontology in a such way as does not require the revision of existing definitions.
- Minimal ontological commitments (Gruber, 93), which means making as few claims as possible about the world being modeled, which means that the ontology should specify as little as possible about the meaning of its terms, giving the parties committed to the ontology freedom to specialize and instantiate the ontology as required.
- Ontological Distinction Principle (Borgo et al., 96)which means that classes in an ontology should be disjoint. The criterion used to isolate the core of properties considered to be invariant for an instance of a class is called the Identity Criterion.
- Diversification of hierarchies to increase the power provided by multiple inheritance mechanisms (Arpírez et al., 98). If enough knowledge is represented in the ontology and as many different classification criteria as possible are used, it is easier to enter new concepts (since they can be easily specified from the pre-existing concepts and classifications criteria) and to inherit properties from different points of view.
- Modularity (Bernaras et al., 96) to minimize coupling between modules.
- Minimize the semantic distance between sibling concepts (Arpírez et al., 98). Similar concepts are grouped and represented as subclasses of one class and should be defined using the same primitives, whereas concepts which are less similar are represented further apart in the hierarchy.
- Standardize names whenever is possible (Arpírez et al., 98).

2.3 What are the components of ontologies?

As we said before, ontologies provide a common vocabulary of an area and define — with different levels of formality — the meaning of the terms and the relations between them. Knowledge in ontologies are formalized using five kind of components: classes, relations, functions, axioms and instances (Gruber, 93). Classes in the ontology are usually organized in taxonomies. Sometimes, the definition of ontologies have been diluted, in the sense that taxonomies are considered to be full ontologies (Studer et al., 98).

- Concepts are used in a broad sense. They can be abstract or concrete, elementary (electron) or composite (atom), real or fictitious. In short, a concept can be anything about which something is said and, therefore, could also be the description of a task, function, action, strategy, reasoning process, etc.

- Relations represent a type of interaction between concepts of the domain. They are formally defined as any subset of a product of n sets, that is: $R : C1xC2x...xCn$. Examples of binary relations are: subclass-of and connected to.

- Functions are a special case of relations in which the n-th element of the relationship is unique for the n-1 preceding elements. Formally, functions are defined as: $F : C1xC2x...xCn - 1 \rightarrow Cn$. Examples of binary functions are Mother-of and square, and an example of a ternary function is price-of-a- used-car that calculates the price of a second-hand car depending on the car-model, manufacturing date and number of kilometers.

- Axioms are used to model sentences that are always true.

- Instances are used to represent elements.

Once the main components of ontologies has been presented, the next question is What does an explicit ontology look like? Uschold and Grninger (Uschold et al., 96) have distinguished four kind of ontologies depending on the kind of language used to build them. They are: Highly informal ontologies if they are written in natural language; Semi-formal ontologies if they are expressed in a restricted and structured form of natural language (i.e., using patterns); Semi-formal ontologies, which are defined in an artificial and formally defined language; and Rigorously formal ontologies if they are defined in a language with formal semantics, theorems and proofs of such properties as soundness and completeness.

2.4 What types of ontologies already exist?

This section does not seek to give an exhaustive typology of ontologies as presented in (vanHeijst et al., 97) and (Mizoguchi et al., 95) However, it presents the most commonly used types of ontologies so the reader can get an idea of the knowledge to be included in each type of ontology. Basically, the following categories are identified: Knowledge representation ontologies, meta-ontologies, domain ontologies, tasks ontologies, domain-task ontologies, application ontologies, index ontologies, tell and ask ontologies, etc.

- Knowledge Representation ontologies (vanHeijst et al., 97) capture the representation primitives used to formalized knowledge in knowledge representation paradigms. The most representative example of this kind of ontologies is the Frame-Ontology (Gruber, 93), which captures the representation primitives (classes, instances, slots, facets, etc.) used in frame-based languages.

- General/Common ontologies (Mizoguchi et al., 95) include vocabulary related to things, events, time, space, causality, behavior, function, etc.

- Meta-ontologies, also called Generic Ontologies or Core Ontologies (vanHeijst et al., 97), which are reusable across domains. The most representative example could be a mereology ontology (Borst, 97) which would include the term part-of.

- Domain ontologies (Mizoguchi et al., 95) (vanHeijst et al., 97) are reusable in a given domain. They provide vocabularies about the concepts within a domain (i.e., scalpel, scanner in a medical domain) and their relationships, about the activities that take place in that domain (i.e., anesthetize, give birth), and about the theories and elementary principles governing that domain.

- Task ontologies (Mizoguchi et al., 95) provide a systematized vocabulary of the terms used to solve problems associated with tasks that may or may not be from the same domain. These ontologies provide a set of terms by means of which to generically describe how to solve one type of problems. They include generic names (i.e., plan, goal, constraint), generic verbs (i.e., assign, classify, select), generic adjectives (i.e., assigned) and others in the scheduling tasks.
- Domain-Task ontologies are task ontologies reusable in a given domain, but not across domains. For example, a domain-task ontology in the medical domain could include the terms related with the schedule of a surgery: plan-surgery.
- Application ontologies (vanHeijst et al., 97) contains the necessary knowledge for modeling a particular domain.

Meta-ontologies, domain ontologies and applications ontologies capture static knowledge in a problem-solving independent way, where as PSMs ontologies, task ontologies and domain-task ontologies are concerned with problem solving knowledge. All these kind of ontologies can be combined to build a new ontology. The reusability-usability trade-off problem (Klinker et al., 91) applied to the ontology field states that the more reusable an ontology is, the less usable is, and vice versa.

3 MOST WELL-KNOWN ONTOLOGIES

Nowadays, it is easy to get information from organizations that have ontologies on the WWW. Many ontologies like Ontolingua ontologies at the Ontolingua Server[1] (Farquhar et al., 96) and WordNet[2] (Miller, 90) at Princenton are freely available over the internet. Other ontologies, like Cyc ontologies [3] (Lenat et al., 90), are partially freely available on the web. However, the majority of ontologies have been developed by companies for their own use and are not available. The Ontology Page[4] (also known as TOP) and (Onto)2Agent[5] (Arpírez et al., 98) (an ontology-based www broker that helps to select ontologies) might help to select ontologies. Taking into account the previous typology of ontologies, this section presents the most well-known ontologies.

The most representative example of a **knowledge representation ontology** is the *Frame Ontology* (Gruber, 93). It captures the representation primitives used in frame-based languages, such as classes, subclasses attributes, class-partitions, relations and axioms. It allows other ontologies to be codified using frame-based conventions. It is implemented in KIF 3.0 (Genesereth et al., 92), and it is the basis upon which Ontology Server translators are built.

Top-Level Ontologies provide general notions under which with all the terms in existing ontologies are related. Examples of top level ontologies are: Sowa's boolean lattice (Sowa, 97), Penman Upper Level (Bateman et al., 90), Cyc (Lenat et al., 90), Guarino's top level proposal (Guarino, 97), etc. Some works on a sort of "standardized" top-level ontology started within ANSI in 1996.

The *Mereology ontology* (Borst, 97) could be the most typical example of a **meta ontology**. The Mereological ontology defines the part-of relation and its properties. This relation allows to express that devices are assembled of components.

Cyc ontology (Lenat et al., 90) is a **common sense ontology** that provides a vast amount of fundamental human knowledge. The ontology consists of a set of terms and assertions which relate those terms. The Cyc ontology is divided into many microtheories. Each microtheory only captures a consistent point of view of a given domain of knowledge. Some areas can handle several different microtheories, representing different perspectives and assumptions, levels of granularity and distinctions. Cyc Ontologies are implemented in CycL language.

The most illustrative **linguistic ontologies** are the Generalized Upper Model (Bateman et al., 95), Wordnet (Miller, 90) and Sensus (Swartout et al., 97). The *Generalized Upper Model* (GUM[6]) is a general task and domain-independent linguistic ontology. To make it portable across different languages, the GUM ontology only includes the main linguistic concepts and how they are organized across languages, and omits details that differentiate languages. This philosophy allowed the use of GUM to create

[1] http://www-ksl.stanford.edu:5915 or the european mirror site at http://www-ksl-svc- lia.dia.fi.upm.es:5915
[2] http://www.tio.darpa.mil/Summaries95/B370-Princenton.html
[3] http://www.cyc.com/
[4] http:// www.medg.lcs.mit.edu/doyle/top
[5] http://delicias.dia.fi.upm.es/OntoAgent/
[6] http://www.darmstadt.gmd.de/publish/komet/gen-um/newUM.html

ontologies about specific languages, like English, German, Spanish, and Italian by entering the semantic distinctions of each language. The ontology has been implemented in LOOM. *WordNet* is a lexical database for English based on psycholinguistic principles. Its information is organized in units called "synsets", which are sets of synonyms that are interchangeable in a particular context and are used to represent different meanings. WordNet contains a set of pairs (w, s), where w is a string of ASCII characters and the meaning s is an element of a set of meanings or synset. Most of the synsets are accompanied by explanatory glossaries, and they are all organized in a network by means of semantic relationships of the type: antonymy, hyponymy, metonymy and implication. WordNet is composed of 4 nets that represent the main syntactical categories: nouns, verbs, adjectives and adverbs. SENSUS is a natural language based ontology whose goal is to provide a broad conceptual structure for work in machine translation. It was developed by merging and extracting information form existing electronic resources like: PENMAN Upper model, ONTOS, Wordnet and electronic natural languages dictionaries. It contains more than 50,000 concepts.

In the domain of **engineering ontologies**, special mention deserves the EngMath ontology (Gruber et al., 94) and PhysSys (Borst, 97). *EngMath* is an Ontolingua ontology developed for mathematical modelling in engineering. The ontology includes conceptual foundations for scalar, vector, and tensor quantities, physical dimensions, units of measure, functions of quantities, and dimension quantities. PhysSys is an engineering ontology for modeling, simulating and designing physical systems. It consists of three engineering ontologies formalizing the three viewpoints on physical devices: system layout, physical process behavior and descriptive mathematical relations. Three engineering ontologies formalize each of these viewpoints: a component ontology, a process ontology and the EngMath ontology. The interdependencies between these ontologies are formalized as ontology projections. These ontologies use other meta-ontologies: mereology, topology and system theories.

The most representative **ontologies for enterprise modeling process** are the Enterprise Ontology (Uschold et al., 96) and TOVE ontology (Gruninger et al., 95). The Enterprise Ontology[7] is a collection of terms and definitions relevant to business enterprises and includes knowledge about activities and processes, organizations, strategies, marketing, etc. Ontologies built at the TOVE (Toronto Virtual Enterprise)[8] project are: Enterprise Design Ontology, Project Ontology, Scheduling Ontology, or Service Ontology.

An illustrative example of **ontologies for Knowledge Management** is the $(KA)^2$ ontology[9] (Benjamins et al., 99), to be used by the Knowledge Annotation Initiative of the Knowledge Acquisition Community. This ontology will form the basis to annotate www documents of the knowledge acquisition community in order to enable intelligent access to these documents. This ontology is being built jointly and distributively with people at different locations.

4 METHODOLOGIES

The ontology building process is a craft rather than an engineering activity. Each development team usually follows its own set of principles, design criteria and phases in the ontology development process. The absence of consenesuated guidelines and methods hinders the development of shared and consensuated ontologies within and between teams, the extension of a given ontology by others and its reuse in other ontologies and final applications.

It is common practice among ontology developers to switch directly from knowledge acquisition to implementation, which poses the following problems (Fernández et al., 99): Ontology conceptual models are implicit in the implementation codes; ontological commitments and design criteria are implicit and explicit in the ontology code; domain experts and human end users have no understanding of formal ontologies codified in ontology languages; as with traditional knowledge bases, direct coding of the result of knowledge acquisition is too abrupt a step, especially in the case of complex ontologies; ontology developer preferences in a given language condition the implementation of the acquired knowledge; and ontology developers (who are unfamiliar with or simply inexperienced in the languages in which ontologies are coded) may find difficult to understand implemented ontologies or even to build a new ontology, because traditional ontology tools focus too much on implementation issues rather than on questions of design.

[7] http://www.aiai.ed.ac.uk/project/enterprise
[8] http://www.ie.utoronto.ca/EIL
[9] http://www.aifb.uni-karlsruhe.de/WBS/broker/KA2.html

4.1 What methods should I use to build my own ontology?

The ontology development process refers to which tasks you should carry out when building ontologies (Fernández et al., 97).There exist three kind of activities: project management activities, development oriented activities and integral activities. Project management activities goal is to assure the well-running ontology, which include planning, control and quality assurance tasks. Development oriented activities goal is to build the ontology by performing specification, conceptualization, formalization, implementation and maintenance tasks. Finally, integral activities give support to development oriented activities and includes: knowledge acquisition, integration, evaluation, documentation and configuration management. If ontologies are built on a small scale, some ontology tasks can be skipped. But, if you mean to build large ontologies on a large scale and with some guarantees of correctness and completeness it is advisable to steer clear of anarchic constructions.

Uschold and King's (Uschold et al., 95) methodology is based on the experience of building the Enterprise Ontology, which includes a set of ontologies for enterprise modeling. They propose the following steps: (1) identify the purpose and scope of the ontology; (2) build the ontology by capturing knowledge, coding knowledge and integrating such knowledge with existing ontologies; (3) evaluate the ontology; (4) documentation; and (5) guidelines for each phase.

Grninger and Fox's (Grninger et al., 95) methodology is based on the experience of building an enterprise modeling ontology in the framework of the TOVE project. Essentially, it involves building a logical model of the knowledge that is to be specified in the ontology. This model is not built directly. First, the specifications that are to be met by the ontology are described informally by identifying a set of competency questions, and this description is then formalized in a language based on first-order predicate calculus. The competency questions are the basis for a rigorous characterization of the knowledge that the ontology has to cover, and they specify the problem and what constitute a good solution to the problem. By a composition and decomposition mechanism, competency questions and their answers can be used to answer more complex competency questions in other ontologies, allowing the integration of ontologies

The METHONTOLOGY framework (Gómez-Pérez, 1998), (Fernández et al., 99) enables the construction of ontologies at the knowledge level. It includes: the identification of the ontology development process, a proposed life cycle and the methodology itself. The ontology development process identifies which tasks should be performed when building ontologies (planning, control, quality assurance, specification, knowledge acquisition, conceptualization, integration, formalization, implementation, evaluation, maintenance, documentation and configuration management). The life cycle based on evolving prototypes identifies the stages through which the ontology passes during its lifetime. Finally, the methodology itself specifies the steps to be taken to perform each activity, the techniques used, the products to be outputted and how they are to be evaluated. The main phase in the ontology development process using the METHONTOLOGY approach is the conceptualization phase. During both specification and conceptualization, a process of integration was completed using in-house and external ontologies. This framework is partially supported by a software environment called Ontology Design Environment (ODE) (Blázquez et al., 98), (Fernández et al.., 99). Several ontologies has been developed using this methodology: CHEMICALS, an ontology in the domain of chemical elements; Environmental pollutants ontologies (Gómez-Pérez et al., 99) which represent the methods of detecting the different pollutants components of various media: water, air, soil, etc., and de maximum permitted concentrations of these components, taking into account all the legislataion in effect (European Union regulations, Spanish, German, US regulations, etc.); The Reference-Ontology (Arpíez et al., 98), an ontology in the domain of ontologies that plays the role of a kind of yellow pages of ontologies; and The restructured version of the $(KA)^2$ ontology (Blzázquez et al., 98). This methodology has been proposed to build ontologies by the Foundation for Intelligent Physical Agents (FIPA[10]) that promotes interoperability across agent-based application.

All these methodologies have in common that they start from the identification of the purpose of the ontology and the need for domain knowledge acquisition. However, having acquired a significant amount of knowledge, Uscholds methodology proposes coding in a formal language and METHONTOLOGY proposes expressing the idea as a set of intermediate representations (IR). Then the ontology is generated using translators. These Irs bridge the gap between how people see a domain and the languages in which ontologies are formalized. These intermediate representations provide a user-friendly approach for both knowledge acquisition and evaluation by computer scientists and domain experts who are not knowledge engineers (Aguado et al., 98).

[10] http://www.fipa.org

The need for ontology evaluation (Gómez-Pérez, 96) is also identified in the three above methodologies. Uscholds methodology includes this activity but it does not state how it should be carried out. Grninger and Fox propose identifying a set of competency questions. Once the ontology has been expressed formally, it is compared against this set of competency questions. Finally, METHONTOLOGY proposes that evaluation activities be carried out throughout the entire lifetime of the ontology development process. Most of the evaluation is done in the conceptualization phase.

5 LANGUAGES AND ENVIRONMENTS FOR BUILDING ONTOLOGIES

5.1 Which are the most commonly used languages to build ontologies?

Basically, several representation systems have been reported for formalizing ontologies under a frame-based modeling approach, a logic-based approach or even both. The most representative languages are Ontolingua (Gruber, 93), CycL (Lenat et al., 90), Loom (MacGregor, 91) and Flogic (Kifer et al., 95).

Ontolingua is a language based on KIF and on the Frame Ontology, and is the ontology-building language used by the Ontolingua Server. KIF (Knowledge Interchange Format) is an interlingua, having declarative semantics, sufficient expressive power to represent the declarative knowledge contained in typical applications system knowledge base, and a structure that enabled semiautomatic translations into and out of typical representation languages. It is a prefix version of first-order predicate calculus, with extensions to improve its expressiveness, like: definition of terms, representation of knowledge about knowledge, reifying functions and relations, specifying sets, and non-monotonic reasoning. The Frame Ontology, which, as mentioned above, is a knowledge representation ontology for modelling knowledge under a frame-based approach, was built on the basis of KIF and a series of extensions to this language. The Ontolingua language allows ontologies to be built in any of the following three manners: (1) using KIF expressions; (2) using exclusively the Frame Ontology vocabulary (it is not possible to represent axioms); (3) using both languages at the same time, depending on ontology developer preferences. In any case, the Ontolingua definition is composed of a heading, an informal definition in natural language and a formal definition written in KIF or using the frame ontology vocabulary. A GFP (Chaudhri et al., 97) application is required in order to reason with Ontolingua Ontologies. CycL is Cyc's knowledge representation language. CycL is a declarative and expressive language, similar to first-order predicate calculus with extensions to handle equality, default reasoning, skolemization, and some second-order features. CycL uses a form of circumscription, includes the unique names assumption, and can make use of the closed world assumption where appropriate. The Cyc inference engine performs general logical deduction, best-first search using a set of proprietary heuristics, uses microtheories to optimize inferences in restricting domains, and includes several special-purpose inferencing modules for handling specific classes of inferences.

LOOM is a high-level programming language based on first order logic and environment, which belongs to the KL-ONE family. The LOOM language provides: an expressive and explicit declarative model specification language; powerful deductive support, which includes both strict reasoning and reasoning by omission, and automatic consistency checking; several programming paradigms, which act as an interface with the declarative model specification; and knowledge-base services.

FLogic is an integration of frame-based languages and first-order predicate calculus. It includes objects (simple and complex), inheritance, polymorphic types, query methods and encapsulation. Its deductive system works with the theory of predicate calculus and structural and behavioral inheritance.

5.2 How software tools support the process of building and using ontologies?

The main tools for building ontologies are: The Ontolingua Server (Farqhuar et al., 96), Ontosaurus[11] (Swartout et al., 97) ODE (Blázquez et al., 98) and (Fernández et al., 99) and Tadzebao y Webonto (Domingue, 98).

The Ontolingua Server is the best known environment for building ontologies in Ontolingua language. It is a set of tools and services that support the building of shared ontologies between geographically distributed groups. It was developed in the context of the ARPA Knowledge Sharing Effort by the Knowledge Systems Laboratory at Stanford University. The ontology server architecture provides access

[11] http://indra.isi.edu:8000

to a library of ontologies, translators to languages (Prolog, CORBA's IDL, CLIPS, Loom, KI) and an editor to create and browse ontologies. There are three modes of interaction: remote collaborators that are able to write and inspect ontologies; remote applications that may query and modify ontologies stored at the server over the internet using the generic frame protocol; and stand-alone applications. The ontology server may be accessed through the URL: `http://www-ksl-svc.stanford.edu:5915/`.

Ontosaurus is being developed by the Information Sciences Institute at the University of South California. It consists of two parts: an ontology server that uses Loom as knowledge representation system and an ontology browser server that dynamically crates html pages (including image and textual documentation) that display the ontology hierarchy and it uses html forms to allow the user to edit the ontology. Translators from loom to Ontolingua, KIF, KRSS and C++ have also been developed.

ODE (Ontology Design Environment) is being developed by the Computer Science School at Universidad Politecnica de Madrid. The main advantage of ODE are the conceptualization module for building ontologies and the module for building "ad hoc" conceptual models. The conceptualization module allows the ontologist to develop the ontology at the knowledge level using a set of intermediate representations that are independent of the target language in which the ontology will be implemented. Once the conceptualization is complete, the code is generated automatically using ODE code generators. Current code generators include: Ontolingua, Flogic and a relational database. So, non-experts in the languages in which ontologies are implemented could specify and validate ontologies using this environment. The module for building "ad hoc" conceptual models includes a language called Lenguage for building intermediate representations (LBIR) that allows the ontologists to specify the kind of model they need for their ontology.

Tadzebao and WebOnto are complementary tools that are being developed by the Knowledge Media Institute at The Open University. Tadzebao enables knowledge engineers to hold synchronous and asynchronous discussion about ontologies and WebOnto supports the collaborative browsing, creation and editing of ontologies on the web.

6 APPLICATIONS THAT USE ONTOLOGIES

Although ontologies can be used (Uschold et al., 96) to communicate between systems, people, and organizations, interoperate between systems, and support the design and development of knowledge- based and general software systems, the number of applications built that use ontologies to model the application knowledge is small. That is, many times such ontologies have been built just for a given application without special consideration for sharing and reuse. Several problems make difficult the reuse of existing ontologies in applications[12] (Arpírez et al., 98): Ontologies are dispersed over several servers; the formalization differs depending on the server on which the ontology is stored; ontologies on the same server are usually described with different levels of detail; and there is no common format for presenting relevant information about the ontologies so users can decide which ontology best suits their purpose. These problems are probably the cause for the relatively small number of known applications until now in areas such as knowledge management, ontology-based brokers, natural language generation, enterprise modeling, knowledge-based systems, and interoperability between systems. A bunch of applications that use ontologies can be found at the proceedings of the workshop on Applications of ontologies and PSMs held in conjunction with ECAI98 (see `http://delicias.dia.fi.upm.es/WORKSHOP/ECAI98/index.html`).

There exist several applications that use **natural language** ontologies. The GUM is being used in natural language processing applications in different languages: *Penman* (Bateman et al., 90), a generator of text in different domains; KOMET (Bateman et al., 94), which generates text in english, german and dutch; *TechDoc* (Rosner, 94), that provides multilingual generation of technical texts; *AlFresco* (Stock et al., 93) an information retrieval system in the domain of italian art history; GIST, a multilingual system for generating bureaucratic texts in English, German and Italian; *OntoGeneration* (Aguado et al., 98), a system that reuse domain ontologies (chemicals), GUM and natural language generation technology (KPML (Bateman et al., 94)) for the generation of spanish texts in the domain of chemical substances; and the framework for using multiple ontologies in a natural lenguage generation pipeline architecture presented in (Frohlich et al., 98). *Wordnet* is used by Hoenkamp (Hoenkamp, 98) to spott a potential lacuna in an ontology that represents user information needs by analyzing retrieved documents.

In the domain of **enterprise modeling**, the *Enterprise toolset* is the most relevant environment built using the Enterprise ontology. It uses an agent-based architecture to integrate off-the-shelf tools in a plug-

[12] http://delicias.dia.fi.upm.es/OntoAgent/

and- play style. The components of the Enterprise Toolset are: a Procedure Builder for capturing process models, an Agent Toolkit for supporting the development of agents, a Task Manager for integration, visualisation, and support for process enactment, and an Enterprise Ontology for communication (See `http://www.aiai.ed.ac.uk/project/enterprise` for more information). Other applications use TOVE ontologies. The *Enterprise Design Workbench* is a design environment that allows the user to explore a variety of enterprise design. It provides a comparative analysis of enterprise design alternatives, and guidance to the designer. At the **Integrated Supply Chain Management Project**, a network of cooperating intelligent agents perform one or more supply chain functions, and each agent coordinates its actions with other agents. The TOVE virtual enterprise provides the unified testbed used by the agents they built for the major supply chain functions: logistic, transportation, management, etc.

Recently, ontologies are being used by **www brokers** in different domains. *Ontobroker*[13] (Fensel et al., 98) is an ontology-based brokering service for knowledge management that is being used in the context of the Knowledge Annotation Initiative of the Knowledge Acquisition Community. (Onto)^{2}Agent[14] (Arpírez et al., 98) is an ontology-based WWW broker about ontologies that uses the Reference-Ontology as a source of its knowledge and retrives descriptions of ontologies that satisfy a given set of constraints. It is available at (`http://delicias.dia.fi.upm.es/OntoAgent/`). *Chemical OntoAgent* (Arpírez et al., 98)is an ontology-based WWW Chemistry teaching broker that allows students to learn chemistry and to test their skills on this domain. It uses as a source of its knowledge CHEMICALS.

KACTUS (Schreiber et al., 95) is an ESPRIT project on modeling knowledge about complex technical systems for multiple use and the role of ontologies to support it. Ontologies in the domain of electrical-networks, off-shore oil-production, and ship design and assessment have been already built.

Plinius (van de Vet et al., 95) is a semi-automatic knowledge acquisition system from natural language text in the domain of ceramic materials, their properties and their production processes.

7 CONCLUSION

In this paper, we reviewed recent work in the area of ontologies. The current state of the art is that there is now a fairly good understanding of what ontologies are and what they do. Current works take this body of existing work and start from that in new directions.

In the ontology word, emphasis is now put on integration of heterogeneous ontologies, on characterizing and brokering ontologies on the web, on integrating ontologies and problem solving methods, as well as using ontologies for natural language analysis and generation. Also, efforts are made to connect to the object-oriented world and to databases. Ontologies are clearly becoming more and more important in a large variety of areas.

8 REFERENCES

1. Aguado G., Bateman J., Bañón A., Bernardos S., Fernández M., Gómez-Pérez A., Nieto. E, Olalla A., Plaza R., Sanchez A. ONTOGENERATION: Reusing domain and linguistic ontologies for Spanish. Workshop on Applications of Ontologies and PSMs. Brighton. England. August 1998. pp.1-10.

2. Arpírez, J.;. Gómez-Pérez, A.; Lozano, A.; Pinto, S. (ONTO)2Agent: An ontology-based WWW broker to select ontologies. . Workshop on Applications of Ontologies and PSMs. Brighton. England. August 1998. pp. 16-24.

3. Bateman, J.A.; B. Magnini, G. Fabris. The Generalized Upper Model Knowledge Bare: Organization and Use. In Towards Very Large Knowledge Bases. Pages 60-72. IOS Press. 1995.

4. J. A. Bateman. KPML: The KOMET-Penman (Multilingual) Development Environment. Technical Report, GMD/IPSI, Darmstadt (Germany), 1994.

5. J. A. Bateman, R. T. Kasper, J. D. Moore and R. A. Whitney. A General Organization of Knowledge for Natural Language Processing: the Penman Upper Model. Technical Report, USC/ISI, Marina del Rey, CA (USA), 1990.

[13] http://www.aifb.uni-karlsruhe.de/WBS/broker/
[14] http://delicias.dia.fi.upm.es/OntoAgent/

6. R. Benjamins, D. Fensel. Decker, Gómez-Pérez (KA)2 Building Ontologies for the Internet: A mid term report. To be published in the International Journal of Human Computer Studies. 1999.

7. A. Bernaras, I. Laresgoiti and J. Corera. Building and Reusing Ontologies for Electrical network Applications. Proceedings of the 12th ECAI", 1996. Pages: 298-302

8. Blázquez M., Fernández M., García-Pinar J. M., Gómez-Pérez A., Building Ontologies at the Knowledge Level using the Ontology Design Envirnoment, Proceedings of the Eleventh Knowledge Acquisition Workshop, KAW98, Banff, 1998.

9. Borgo, S.; Guarino, N.; Masolo, C. Stratified Ontologies: the case of physical objects. In proceedings of the Workshop on Ontological Engineering. Held in conjunction with ECAI96. Pages 5-15. Budapest. 1996.

10. Borst, W. N. Construction of Engineering Ontologies. University of Tweenty. Enschede, NL- Centre for Telematica and Information Technology. 1997.

11. Bylander, T.; Chandrasekaran, B. Generic Tasks in Knowledge-based reasoning: The right level of abstraction for Knowledge Acquisition. In B. Gaines and J. Boose Editors. Knowledge Acquisition of Knowledge Based Systems. Volume 1. Pages:65-77 Academic Press London. 1988.

12. Chaudhri Vinay K., Farquhar Adam, Fikes Richard, Karp Peter D., Rice James P. The Generic Frame Protocol 2.0 , Technical Report, Stanford.1997.

13. Domingue, J. Tadzebao and Webonto: Discussing, Browsing and editing ontologies on the web. In Proceedings of the Eleventh Knowledge Acquisition Workshop, KAW98, Banff, 1998.

14. Farquhar A., Fikes R., Rice J., The Ontolingua Server: A Tool for Collaborative Ontology Construction, Proceedings of the 10th Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Alberta, Canada, PP. 44.1-44.19, 1996.

15. Fensel, D, Decker, S. Erdman M. Studer, R. Ontobroker: The Very High Idea. In Proceedings of the 11th International Flairs Conference (FLAIRS-98), Sanibal Island, Florida, May 1998.

16. Fernández, M.; Gómez-Pérez, A.; Pazos, J.; Pazos, A. Building a Chemical Ontology using methontology and the ontology desing environment. IEEE Intelligent Systems and their applications. !4 (1):37-45. 1999.

17. Fernández, M., Gómez-Pérez, A. Juristo, N. METHONTOLOGY: From Ontological Art Toward Ontological Engineering. Spring Symposium Series on Ontological Engineering. AAAI97. Stanford. USA. March 1997.

18. Frohlich, M.; van de Riet, P. Using Multiple ontologies in a framework for Natural language generation. Workshop on Applications of Ontologies and PSMs. Brighton. England. August 1998.Pages: 67-77-

19. Genesereth M., Fikes R., Knowledge Interchange Format, Technical Report, Computer Science Department, Stanford University, Logic-92-1, 1992.

20. Gómez-Pérez, A.; Rojas-Amaya, M.D. Ontological Reengineering for Reuse. Knowledge Acquisition Modeling and Management. 11th European Workshop, EKAW99. Dagstuhl Castle, Germany, May 26-29, 1999. Pags 139-156.

21. Gómez-Pérez A., Knowledge Sharing and Reuse, The Handbook of Applied Expert Systems, Edited by J. Liebowitz, CRC Press, 1998.

22. Gómez-Pérez, A. A Framework to Verify Knowledge Sharing Technology. Expert Systems with Application. Vol. 11, N. 4. 1996. PP: 519-529.

23. Gruber, T. and Olsen, R. An Ontology for Engineering Mathematics, Technical Report KSL-94-18, Knowledge Systems Laboratory, Stanford University, CA, 1994.

24. Gruber, T. A translation Approach to portable ontology specification. Knowledge Acquisition. 5: 199- 220. 1993.

25. Gruber T., Toward Principles for the Design of Ontologies Used for Knowledge Sharing. International Journal of Human Computer Studies. 43:907-928. 1995

26. Gruninger M., Fox M., Methodology for the Design and Evaluation of Ontologies, Proceedings of IJCAI95's Workshop on Basic Ontological Issues in Knowledge Sharing, 1995.

27. Guarino, N. Some Organizing Principles for a unified top-level ontology. In Spring Symposium Series on Ontological Engineering. Stanford. 1997. Pages: 57-63.

28. Guarino N.; Giaretta. P. Ontologies and Knowledge Bases: Towards a Terminological Clarification. In N.J.I. Mars Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing.. IOS Press. pages 2532. 1995.

29. van Heist G., Schreiber A. Th., Wielinga B. J., Using explicit ontologies in KBS development, International Journal of Human-Computer Studies, 45, PP. 183-292, 1997.

30. Hoenkamp. E. Spotting Ontological Lacunae through spectrum analysis of retrieved documents. In Workshop on Applications of Ontologies and PSMs. Brighton. England. August 1998.Pages: 73-77.

31. Kifer M., Lausen G., Wu J., Logical Foundations of Object-Oriented and Frame-Based Languages, Journal of the ACM, 1995.

32. Klinder, M.; Lausen, G.; Wu, J. Logical Foundations of Object oriented and frame-based languages. Journal of ACM, 1995.

33. Lenat D.B., Guha, R V. Building Large Knowledge-based systems. Representation and Inference in the Cyc project. Addison-Wesley, Reading, Massachusetts. 1990.

34. MacGregor, R. Inside the LOOM classifier. SIGART bulletin, 2(3):70-76. June, 1991.

35. Miller G. A., WordNet: An On-line Lexical Database, International Journal of Lexicography 3, 4: 235- 312, 1990.

36. Mizoguchi, R.; Vanwelkenhuysen, J.; Ikeda, M. Task Ontology for reuse of problem solving knowledge. In N.J.I. Mars Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing.. IOS Press. pages 46-57. 1995.

37. R. Neches, R.E. Fikes, T. Finin, T.R. Gruber, T. Senator, and W.R. Swartout. Enabling technology for knowledge sharing. AI Magazine. 12(3)::36-56- 1991.

38. D. Rösner. Generating Multilingual Documents from a Knowledge Base: The TECHDOC Project. Technical Report FAW Ulm, Ulm (Germany), 1994.

39. Sowa J. F., Knowledge Representation: Logical, Philosophical, and Computational Foundations, Boston:MA, PWS Publishing Company, Forthcoming, 1997.

40. O. Stock, G. Carenini, F. Cecconi, E. Franconi, A. Lavelli, B.Magnini, F. Pianesi, M. Ponzi, V. Samek- Lodovici and C. Strapparava. ALFRESCO: Enjoying the Combination of Natural Language Processing and Hypermedia for Information Exploration. In Mark T. Maybury, editor, Intelligent Multimedia Interfaces, The MIT Press, pp. 197-224, chapter 9. Extended and revisedversion of a paper previously published at IJCAI-91. 1993.

41. R. Studer, V.R. Benjamins, D. Fensel. Knowledge Engineering: Principles and Methods. Data and Knowledge Engineering. 25: 161-197. 1998.

42. B. Swartout, R. Patil, K. Knight and T. Russ. Towards Distributed Use of Large-Scale Ontologies. Spring Symposium Series on Ontological Engineering. Stanford University, CA. 1997, Pages: 138148.

43. Uschold M., Grüninger M., ONTOLOGIES: Principles, Methods and 16Applications, Knowledge Engineering Review, Vol. 11, N. 2, June 1996.

44. Van der Vet P.E., Speel P.-H., Mars N. J. I., The Plinius ontology of ceramic materials. Proceedings of ECAI94's Workshop on Comparison of Implemented Ontologies, Amsterdam, 1994.