METHODOLOGIES FOR ONTOLOGY DEVELOPMENT

Dean Jones, Trevor Bench-Capon and Pepijn Visser[1]

It is now widely recognised that constructing a domain model, or ontology, is an important step in the development of knowledge based systems. What is lacking, however, is a clear understanding of how to build ontologies. In this paper we survey the work which has been done so far in beginning to provide a methodology for building ontologies. This work is still formative, and relies heavily on particular experiences. We also provide some discussion of this work, and identify the key issues that must be addressed if we are to move on from ontology construction being an art and to make it an understood engineering process.

1. Introduction

It is now widely recognised that constructing a domain model, or ontology, is an important step in the development of knowledge based systems. The advantages of such domain models have been widely canvassed, and include enabling the sharing of knowledge, the re-use of knowledge, and the better engineering of knowledge based systems with respect to acquisition, verification and maintenance. If, however, we examine ontologies, we find significant variety in them, even when they have been constructed for very similar purposes [32]. At present the construction of ontologies is very much an art rather than a science. This situation needs to be changed, and will be changed only through an understanding of how to go about constructing ontologies. In short what is needed is a good methodology for developing ontologies. So far a number of suggestions for such a methodology have emerged as people reflect on their experience of building

[1] Department of Computer Science, University of Liverpool, Liverpool, U.K., L69 7ZF.

ontologies. In this paper we will survey this work, and some other work giving insights into ontology development, and provide some comparative discussion.

2. Methodologies for Ontological Engineering

There are a growing number of methodologies that specifically address the issue of the development and maintenance of ontologies. In this section, we present a (fairly comprehensive) survey.

2.1. TOVE

Based on experiences in the development of TOVE (Toronto Virtual Enterprise) the following approach to engineering ontologies is developed ([15], [16], [17], [30]):

(1) *motivating scenarios*: the start point is a set of problems encountered in a particular enterprise, which are often in the form of story problems or examples.

(2) *informal competency questions*: requirements of the ontology, based on the motivating scenario, described as informal questions that an ontology must be able to answer; this phase acts as an evaluation on the ontological commitments made in the previous stage.

(3) *terminology specification*: the objects, attributes and relations of the ontology are formally specified (usually in first order logic).

(4) *formal competency questions*: the requirements of the ontology are formalised in terms of the formally defined terminology (see especially [16]).

(5) *axiom specification*: axioms that specify the definition of terms and constraints on their interpretations are given in first-order logic, guided by the formal competency questions as the axioms must be necessary and sufficient to express the competency questions and their solutions.

(6) *completeness theorems*: an evaluation stage which assesses the competency of the ontology by defining the conditions under which the solutions to the competency questions are complete.

The initial description of the tasks to be supported by an ontology in terms of motivating scenarios seems to result from the methodology being abstracted from the development process undertaken in the TOVE project. Motivating scenarios are only one of many ways in which such tasks could be described and in order to arrive at a more comprehensive methodology, it would probably be necessary to incorporate other types of representation.

The TOVE approach is most interesting for the emphasis on ontology evaluation, especially as a means of performing this evaluation is provided in the form of completeness theorems. These theorems are useful in an umber of ontology maintenance tasks, e.g. assessing the extendibility of an ontology - any extension must be able to preserve the validity of the completeness theorems - or to provide a benchmark for ontologies [16].

2.2. Enterprise Model Approach

In [29] a skeletal methodology for ontology construction is described, based largely on the experience of developing the Enterprise ontology ([27]). This is then considerably refined in [28] to give the following:
(1) *identify purpose*: determines the level of formality at which the ontology should be described.
(2) *identify scope*: a "Specification" is produced which fully outlines the range of information that the ontology must characterise. This may be done using motivating scenarios and informal competency questions, as in TOVE or by "brainstorming and trimming" i.e. produce a list of potentially relevant concepts and delete irrelevant entries and synonyms.
(3) *formalisation*: create the "Code", formal definitions and axioms of terms in the Specification.
(4) *formal evaluation*: the criteria used may be general, such as those given in [16] and [10], [11], [12], [14] (see 2.3 below), or specific to a particular ontology, such as checking against purpose or competency questions. This stage may cause a revision of the outputs of stages 2 and 3.

In common with most recent KBS development methodologies, the Enterprise approach distinguishes between the informal and formal phases of ontology construction. The informal phase involves identifying key concepts then giving text definitions for concepts and relationships. As the use of existing knowledge acquisition techniques is recommended for this informal stage, no advice is given on how to identify *ontological* concepts. In fact, this comment applies not only to the Enterprise approach but to all of methodologies described in this section, as none seem to address this issue although such guidelines are available (e.g. see section 3.9)

2.3. METHONTOLOGY

Initially described in [13] and updated in [8], METHONTOLOGY starts by identifying the following activities that are involved in the development of an ontology.

(1) *specification*: identify the purpose of the ontology, including the intended users, scenarios of use, the degree of formality required, etc., and the scope of the ontology including the set of terms to be represented, their characteristics and the required granularity. The output of this phase is a natural-language ontology specification document.

(2) *knowledge acquisition*: this occurs largely in parallel with stage (1). It is non-prescriptive as any type of knowledge source and any elicitation method can be used, although the roles of expert interviews and analyses of texts are specifically discussed.

(3) *conceptualisation*: domain terms are identified as concepts, instances, verbs relations or properties and each are represented using an applicable informal representation.

(4) *integration*: in order to obtain some uniformity across ontologies, definitions from other ontologies, e.g. Ontolingua standard units ontology, should be incorporated.

(5) *implementation*: the ontology is formally represented in a language, such as Ontolingua.

(6) *evaluation*: much emphasis is placed on this stage in METHONTOLOGY. The techniques used are largely based on those used in the validation and verification of KBSs. In [14] a set of guidelines is given on how to look for incompletenesses, inconsistencies and redundancies.

(7) *documentation*: collation of documents that result from other activities.

The life cycle of an ontology, in which these activities are ordered, is based on the refinement of a prototype. An ontology goes through the following states (which correspond to some of the activities identified above): specification, conceptualisation, formalisation, integration, implementation. Finally, the ontology enters the maintenance state, which. Knowledge acquisition, evaluation and documentation are carried during the entire life cycle.

Like TOVE, the most distinctive aspect of METHONTOLOGY is the focus on maintenance (see [10], [11], [12], [14]). The main difference between the two is that in METHONTOLOGY the focus is on comprehensively addressing the maintenance stage of the life cycle of an ontology whereas TOVE utilises more formal techniques to address a more limited number of maintenance issues.

2.4. KBSI IDEF5

The IDEF5 method is designed to assist in the creation, modification and maintenance of ontologies [19]. As ontological analyses are necessarily open-ended, it is suggested that it is not prudent to adopt a "cookbook" approach to ontology development. The IDEF5 methodology is thus a general procedure with a set of guidelines:

(1) *organising and scoping*: establishes the purpose, viewpoint, and context for the ontology development project. The purpose statement provides a set of "completion criteria" for the ontology, including objectives and requirements. The scope defines the boundaries of the ontology and specifies parts of the systems that must be included or excluded.

(2) *data collection*: the raw data needed for ontology development is acquired using typical KA techniques, such as protocol analysis and expert interview.

(3) *data analysis*: the ontology is extracted from the results of data collection. First, the objects of interest in the domain are listed, followed by identification of objects on the boundaries of the ontology. Next, internal systems within the boundary of the description can be identified.

(4) *initial ontology development*: a preliminary ontology is developed, which contains *proto-concepts* i.e. initial descriptions of kinds, relations and properties.

(5) *ontology refinement and validation*: the proto-concepts are iteratively refined and tested. This is essentially a deductive validation procedure as ontology structures are "instantiated" with actual data, and the result of the instantiation is compared with the ontology structure.

As the IDEF5 methodology is based on refinement of the outputs produced, even with regard to the initial scope and level of detail, again we have an evolving prototype model. This gradual refinement is facilitated through the use of two representation languages. The initial ontology is defined with the *schematic language*, a graphical notation that is used to express the most common forms of ontological information during a project. The schematic language is used mainly for communication between domain expert and the ontology developer. The initial representation is then analysed and recast into the more structured *elaboration language* based on KIF.

A useful inclusion in the IDEF5 methodology is the library of commonly used relations. This comprises definitions and characterisations of classification relations (including class inclusion relations), meronymic relations, temporal relations, spatial relations, influence relations, dependency relations and case relations.

3. Additional Methods in the Development of Ontologies

In addition to the development methodologies discussed in the previous section, many other approaches address a specific aspect of ontology development. As these descriptions do not claim to be comprehensive, it would be unfair to compare them with those described in the previous section. They do however provide valuable insights into the development of ontologies.

3.1. Ontolingua

The guides for use of the Ontolingua server [5], [6], [7] contain advice on browsing, developing, maintaining and sharing ontologies stored at the server. The Ontolingua language is based on the syntax and semantics of KIF. One of the main benefits in using the Ontolingua server is the access it provides to a library of previously defined ontologies. This library is extended as developers of new ontologies add to the repository. Thus, ontology construction in Ontolingua is based on the principle of modular development. Ontologies from the library can be re-used in four different ways:

(a) *inclusion*: ontology A is explicitly included in ontology B. The vocabulary of ontology A is translated into the vocabulary of ontology B. This translation is then applied to the axioms of A and the translated axioms are added to B [6]. Multiple inclusion is supported.

(b) *polymorphic refinement*: where a definition from an ontology is included and refined. For example, the addition operator, defined in a number ontology, can be included in ontology A and extended to apply to strings and included in ontology B and extended to apply to vectors.

(c) *restriction*: a restricted version of one ontology is included in another, e.g. a number ontology included in an integer ontology with the restriction that all numbers are integers [6].

(d) *cyclic inclusion*: as ontology inclusion (i.e. (a)) is transitive, situations such as the following are allowed: ontology A is included in ontology B, ontology B is included in ontology C and ontology C is included in ontology A.

These distinctions are very useful in the re-use of ontologies, although it is not clear whether they exhaustively specify the relationships between ontologies, e.g. mapping functions that convert from one ontology to another do not seem to be covered. Ontolingua is the *de facto* standard means of implementing ontologies although a more comprehensive methodology should be used in conjunction with the server.

3.2. CommonKADS and KACTUS

CommonKADs is a widely used methodology for the development of knowledge based systems, in which ontologies play an important role. The KACTUS project was a follow-up project which focused on the issue of ontology development. An engineering approach is adopted, stressing modular design, redesign and reuse [23], [33]. An ontology is constructed from a library of small-scale ontologies, which requires mapping between the various ontologies included in the development of the new ontology. Two kinds of mapping functions are defined between the vocabularies of ontologies:

(i) there is no change in the semantics of expressions of the mapped ontology.

(ii) a change occurs in the semantics of the mapped ontology as an interpretation of it occurs.

The selection of relevant ontologies from a library is supported by an indexing schema for ontologies. This characterises the interpretation context of the use of an ontology according to three dimensions: task type; problem-solving method; and domain-type. The modular development principle is common in ontological engineering and follows from the focus on re-use. Although the work on mapping functions is currently quite limited, the approach seems to have potential in tailoring ontologies to particular applications.

3.3. PLINIUS

The Plinius project [22] attempts the semi-automatic extraction of knowledge from natural-language texts, namely the title and abstracts of bibliographic document descriptions in the on-line version of Engineered Materials Abstracts. The Plinius ontology was developed to support the translation of natural-language sentences into expressions in a knowledge representation language [31]. Those design decisions taken during the development of the ontology which appeared to be domain-independent have been proposed as general ontology development principles. These are:

(1) conflicting assertions about the same entity can be more readily discovered if the concepts are defined as fully as possible.
(2) pre-existing formal theories are taken as given and a domain ontology does not specify the semantics of logical constants.
(3) an ontology should be independent of any particular knowledge representation language.
(4) the principle of the conceptual construction kit states that an ontology consists of primitives concepts and construction rules that allow the definition of all other concepts in terms of these primitives.
(5) a bottom-up approach is taken in order that the ontology exhibits sufficient completeness for the intended task (see principle 7). Thus the conceptual construction kit consists of the bottom-level concepts together with rules for the construction of higher-level concepts from these.
(6) the development of an ontology should be based on engineering decisions, e.g. decisions about the inclusion of particular concepts should be based on a kind of cost-benefit analysis.
(7) the ontology should be assessed in terms of its completeness with respect to the intended task. This completeness criterion can be decomposed into two sub-criteria: coverage (is every concept of interest covered?) and granularity (is every relevant distinction made?).
Although the authors claim that these principles apply generally in the development of ontologies, they also suggest that they are not always sensible or even feasible. Guidelines on when these principles do or do not apply would therefore be a useful addition.

3.4. ONIONS

The ONIONS (ONtologic Integration Of Naive Sources) methodology [9], [24] is motivated by the knowledge integration problem, i.e. how to integrate heterogeneous sources of information in knowledge acquisition. This problem is addressed through the creation of a formal domain ontology by the integration of existing repositories of knowledge. One of the most distinctive aspects of the ONIONS approach is the method of ontology acquisition. This produces a non-formal ontology, a schematic account of the conceptualisation of a domain. Rather than focus on the issue of a final representation of an ontology, ONIONS focuses on problems in ontology acquisition such as modelling stopover¸(‘how do we stop over -refining an ontology?’) and knowledge relevance (‘how do we state what is conceptually relevant?’). The solution used in ONIONS is that stop-over and relevance are both dependent on the intended task(s) of the ontology, although no means of determining the adequacy of an ontology with respect to task has yet been given.

3.5. Mikrokosmos

A total of thirty guidelines were developed to assist in the development of the Mikrokosmos ontology [20], [21]. While most make sound suggestions, some are obviously specific to the particular use of the ontology or even to the Mikrokosmos project (e.g. “Keep other languages in mind. Just because English names are used for the concepts in the ontology, we must not forget that the ontology must be equally useful for interpreting or generating any of a set of natural languages”, [20] p.60) and therefore cannot be generalised to form part of an approach to ontology development. However, some of the guidelines can be applied more generally. Examples of the guidelines are: “if something has a fixed position in time and/or space, it's an instance; if not, it's a concept” and “do not create specialised events with new arguments as concepts unless they are significantly different from existing concepts (e.g. walk_to_airport, walk_to_car_park, etc.)”. We thus have a set of interesting heuristics that are used in making fine detail design choices when building an ontology; such heuristics could be useful in any methodology.

3.6. MENELAS

The MENELAS ontology was designed as part of a natural language understanding system ([3], [4]). Four principles useful in the development of taxonomic knowledge in ontologies are described:

(1) *similarity*: a subclass must be of the same type as its parent.

(2) *specificity*: a subclass must have some difference that distinguishes it from it parent (the differentiae). This difference with the definition of the parent form necessary and sufficient conditions for the definition of the subclass.

(3) *opposition*: the subclasses of a concept are incompatible with each other.

(4) *unique semantic axis*: the subclasses of a concept can be constrained to differ from the parent in some common property or ‘axis’.

Again this offers assistance mainly at a fine level of detail. It does, however, take a rather idealised view of taxonomies which may not be applicable to many domains.

3.7. PHYSSYS

This is an approach that address problems in the use of libraries of ontologies, such as structuring of the library, retrieval of applicable components and maintenance of the library [1], [2]. The method aims to facilitate the selection of pre-existing ontologies based on dynamic knowledge construction rather than the simple selection of knowledge components. Two types of ontology are distinguished:

(a) *primary*: represent a domain from some point of view (e.g. functional, behavioural, etc.)

(b) *secondary*: introduce additional distinctions that can be applied to objects in (a).

The term 'ontological mapping' is used to describe the process whereby an entity of a primary ontology is further differentiated through the application of a dimension from a secondary ontology. The mapping exists between the secondary and the primary ontology. Constraints on a secondary ontology restrict the application of its elements. An ontological commitment is added as a result of ontological mappings, giving a new ontology. This technique derives from an engineering domain, and may not be applicable to less well defined domains.

3.8. SENSUS

Described in [26], this approach is based on the assumption that if two knowledge bases are built on a common ontology, knowledge can be shared between them more readily since they share a common underlying structure. Firstly, two kinds of ontology are distinguished:

(a) *domain ontology*: provides a set of terms for describing some domain.

(b) *theory ontology*: provides a set of concepts for representing some aspect of the world (e.g. time, space, causality, etc.)

SENSUS is a broad coverage ontology that includes both high and intermediate level (but not domain-specific) terms. It was developed by merging the Penman Upper Model, Ontos, WordNet and semantic categories from electronic dictionaries (English, Spanish and Japanese). The terms are organised into an AKO (subsumption) lattice but do not include slots as the application is expected to dictate what relations are most appropriate.

In the development of a domain-specific ontology, some "seed" terms are selected as representative of relevant domain-specific concepts. These terms are then linked (by hand) to SENSUS. All of the concepts from the seed terms to the root of SENSUS are included in the final ontology. Other terms are also included, such as (i) relevant semantic categories, and (ii) entire sub-trees of (non-high level) nodes that have a large number of paths running through them from domain-specific terms to the root. SENSUS terms not included are deemed irrelevant and are pruned from the domain-specific ontology in order to reduce storage and increase efficiency.

3.9. Guarino *et al.*

As discussed in [18], [25] defines sortal predicates, i.e. those that allow us to identify a thing as a particular kind, as providing a principle for distinguishing and counting individuals, e.g. 'apple(X)'. This contrasts with non-sortal predicates, which supply such a principle only for individuals already distinguished, e.g. 'red(X)'. In [18] a predicate is deemed to be sortal if it is (a) countable, i.e. the predicate allows a given object to be identified amongst other kinds of objects, and (b) temporally stable, i.e. if the predicate holds for an object at a given time, it also holds for the same object at another time. Further, a substantial sortal is one that is ontologically rigid i.e. it cannot lose the property without losing its identity. Substantial sortals are predicates that identify the type of an entity and should therefore be defined as classes. Non-substantial sortals should be defined as roles on those classes.

This kind of analysis needs to be incorporated into a comprehensive ontology development methodology in order to enable one to determine those predicates that should be included as classes and those that should be specified as roles. Indeed, this type of analysis seems to be one of the main principles that would separate an ontology development methodology from the domain modelling aspects of a KBS development methodology.

4. Discussion

Although there are considerable differences between the methodologies described above, a number of points clearly emerge:

(1) many of the methodologies take a task as a starting point. This is obviously very useful, if it is possible: it focuses the acquisition, provides the potential for evaluation and provides a useful description of the capabilities of the ontology, expressed as the ability to answer well defined competency questions. On the down side, however, it seems to provide limitations to the re-use of the ontology; tying the ontology closely to a task seems to admit defeat in the face of the interaction problem. It raises the question of whether ontologies are homogeneous, or whether reuse ontologies, sharing ontologies and design ontologies may turn out to be rather different things, needing to be built in different ways.

(2) the comprehensive development methodologies described in section 2 seem to be split between mainly stage-based models (TOVE and ENTERPRISE) and evolving prototype models (METHONTOLOGY and IDEF5), although the distinction is not quite as clean as this might suggest. Both approaches have benefits and drawbacks. Where the purpose and requirements are clear at the outset, a stage-based approach seems more appropriate whereas where no clear purpose has been identified, the evolving prototype model may be more applicable. It may be useful to have a set of guidelines that advise on the overall approach that should be taken or for a particular methodology to specify the kinds of analyses to which it is most suited.

(3) there are typically separate stages to produce first an informal description of the ontology, and then its formal embodiment in an ontology language. The existence of these two descriptions is an important characteristic on ontologies, and the informal description often carries through to the formal description (for example, the crucial role of comments in Ontolingua). Ontologies are intended to help bridge the gap between the executable system and the real world it models, and for this an identifiable nexus between the formal and informal descriptions is essential.

(4) there is an expectation that a library of ontologies will be accumulated, and form the basis for further ontology development. This is thematic with the ideas of re-using and sharing knowledge. However, it does raise issues of how to describe ontologies, how to select a suitable ontology, and how to extend an ontology. How to select an ontology needs a clear answer from a satisfactory methodology, if a development is not to run the risk of being “shoe-horned” into an unsuitable structure. Extending ontologies is addressed by several of the methodologies, but is not a straightforward matter. The principled inclusion and refinement of library ontologies, perhaps based on distinctions such as those described in the discussion of Ontolingua, should form part of any comprehensive ontology development methodology.

(5) A complete methodology must provide guidelines to assist the ontological engineer in making choices at a variety of levels, from the high level structure of the ontology, to the fine detail of whether or not to include some particular distinction. Currently there exist a number of heuristics, derived from experience in ontology building, and other guidelines based on ontological analysis, such as those described in section 3.9, but there is plenty of scope for refinement here.

5. Conclusions

It is quite clear that building ontologies is still a matter of craft skill rather than an understood engineering process. If ontologies are to realise their potential, it is important to clarify this practice, taking into account the variety of experience that is available, rather than basing the methodology too much on the experience of one or two projects. The five issues outlined above must be addressed if a proper methodology, with a clearly defined range of application, is to emerge.

6. References

[1] BORST, P., BENJAMIN, J. WIELINGA, B and AKKERMANS, H. (1996) "An Application of Ontology Construction", *ECAI-96 Workshop on Ontological Engineering*, Budapest, August 13th.

[2] BORST, P. and AKKERMANS, H. (1997) "An Ontology Approach to Product Disassembly", *EKAW 97*, Sant Feliu de Guíxols, Spain, October 15-19th.

[3] BOUAUD, J. BACHIMONT, B., CHARLET, J. and ZWEIGENBAUM, P. (1994) "Acquisition and Structuring of an Ontology within Conceptual Graphs", in *Proceedings of ICCS'94 Workshop on Knowledge Acquisition using Conceptual Graph Theory*, University of Maryland, College Park, MD, 1-25.

[4] BOUAUD, J. BACHIMONT, B., CHARLET, J. and ZWEIGENBAUM, P. (1995) "Methodological Principles for Structuring an Ontology", *IJCAI-95 Workshop on Basic Ontological Issues in Knowledge Sharing*, August 19-20th, Montreal.

[5] FARQUHAR, A., FIKES, R., PRATT, W. and RICE, J. (1995) "Collaborative Ontology Construction for Information Integration", Technical Report KSL 95-63, Knowledge Systems Laboratory, Stanford University.

[6] FARQUHAR, A., FIKES, R. and RICE, J. (1996) "The Ontolingua Server: A Tool for Collaborative Ontology Construction", Technical Report KSL-96-26, Knowledge Systems Laboratory, Stanford University.

[7] FARQUHAR, A., FIKES, R. and RICE, J. (1997) "Tools for Assembling Modular Ontologies in Ontolingua", Technical Report KSL-97-03, Knowledge Systems Laboratory, Stanford University.

[8] FERNANDEZ, M., GOMEZ-PEREZ, A. and JURISTO, N. (1997) "METHONTOLOGY: From Ontological Art Towards Ontological Engineering", *AAAI-97 Spring Symposium on Ontological Engineering*, Stanford University, March 24-26th.

[9] GANGEMI, A., STEVE, G. and GIACOMELLI, F. (1996) "ONIONS: An Ontological Methodology for Taxonomic Knowledge Integration", *ECAI-96 Workshop on Ontological Engineering*, Budapest, August 13th.

[10] GOMEZ-PEREZ, A. (1994a) "Some Ideas and Examples to Evaluate Ontologies", Technical Report KSL 94-65, Knowledge Systems Laboratory, Stanford University.

[11] GOMEZ-PEREZ, A. (1994b) "From Knowledge Based Systems to Knowledge Sharing Technology: Evaluation and Assessment", Technical Report KSL 94-73, Knowledge Systems Laboratory, Stanford University.

[12] GOMEZ-PEREZ, A. (1995) “Criteria to Verify Knowledge Sharing Technology”, Technical Report KSL 95-10, Knowledge Systems Laboratory, Stanford University.

[13] GOMEZ-PEREZ, A., FERNANDEZ, M. and DE VICENTE, A.J. (1996) “Towards a Method to Conceptualize Domain Ontologies”, *ECAI-96 Workshop on Ontological Engineering*, Budapest.

[14] GOMEZ-PEREZ, A., (1997). “A Framework to Verify Knowledge Sharing Technology”, *Expert Systems with Applications*, **11**(4), 519-529.

[15] GRÜNINGER, M. and FOX, M.S. (1994a) “The Design and Evaluation of Ontologies for Enterprise Engineering”, *Workshop on Implemented Ontologies, European Conference on Artificial Intelligence 1994,*, Amsterdam, NL.

[16] GRÜNINGER, M. and FOX, M.S. (1994b) “The Role of Competency Questions in Enterprise Engineering”, *IFIP WG5.7 Workshop on Benchmarking - Theory and Practice*, Trondheim, Norway.

[17] GRÜNINGER, M. and FOX, M.S. (1995) “Methodology for the Design and Evaluation of Ontologies”, *IJCAI-95 Workshop on Basic Ontological Issues in Knowledge Sharing*, Montreal, August 19-20th.

[18] GUARINO, N., CARRARA, M. and GIARETTA, P. (1994) “An Ontology of Meta-Level Categories”, in J. Doyle, E. Sandewall and P. Torasso (eds.) *Principles of Knowledge Representation and Reasoning: Proceedings of the 4th International Conference*, Morgan Kaufmann, San Mateo, CA.

[19] KBSI (1994) “The IDEF5 Ontology Description Capture Method Overview”, KBSI Report, Texas.

[20] MAHESH, K. (1996) “Ontology Development for Machine Translation: Ideology and Methodology”, Technical Report MCCS 96-292, Computing Research Laboratory, New Mexico State University, Las Cruces, NM.

[21] MAHESH, K. and NIRENBURG, S. (1995) “A Situated Ontology for Practical NLP”, *IJCAI-95 Workshop on Basic Ontological Issues in Knowledge Sharing*, Montreal, August 19-20th.

[22] MARS, N.J.I., TER STAL, W.G., DE JONG, H., VAN DER VET, P.E. and SPEEL, P.-H. (1994) “Semi-automatic Knowledge Acquisition in Plinius: An Engineering Approach”, in *Proc. 8th Banff Knowledge Acquisition for Knowledge-based Systems Workshop*, Banff, January 30th-Febraury 4th, 4.1-4.15.

[23] SCHREIBER, A.TH., WIELINGA, B.J. and JANSWEIJER, W.H. (1995) “The KACTUS View on the ‘O’ Word”, *IJCAI Workshop on Basic Ontological Issues in Knowledge Sharing*, Montreal, August 19-20th.

[24] STEVE, G. and GANGEMI, A. (1996) “ONIONS Methodology and the Ontological Commitment of Medical Ontology ON8.5”, *Proc.10th Knowledge Acquisition Workshop - KAW’96*, Banff, Canada, November 9-14th.

[25] STRAWSON, P.F. (1959) *Individuals: An Essay in Descriptive Metaphysics*, Routledge, London.

[26] SWARTOUT, W.R., PATIL, R., KNIGHT, K. and RUSS, T. (1997) “Towards Distributed Use of Large-Scale Ontologies”, *AAAI-97 Spring Symposium on Ontological Engineering*, Stanford University.

[27] USCHOLD, M. (1996) “Converting an Informal Ontology into Ontolingua: Some Experiences”, *ECAI-96 Workshop on Ontological Engineering*, Budapest, August 13th.

[28] USCHOLD, M. (1996) “Building Ontologies: Towards A Unified Methodology”, *Proc. Expert Systems 96*, Cambridge, December 16-18th.

[29] USCHOLD, M. and KING, M. (1995) “Towards A Methodology for Building Ontologies”, *IJCAI-95 Workshop on Basic Ontological Issues in Knowledge Sharing*, Montreal, Canada.

[30] USCHOLD, M. and GRÜNINGER, M. (1996) “Ontologies: Principles, Methods and Applications”, *Knowledge Engineering Review*, **11**(2), 93-137.

[31] VAN DER VET, P.E. and MARS, N.J.I. (1995) “Bottom-up Construction of Ontologies: The Case of an Ontology of Pure Substances”, Technical Report 95-35, Department of Computer Science, University of Twente, Netherlands.

[32] VISSER, P.R.S., AND BENCH-CAPON, T.J.M., (1998) “A Comparison of Four Legal Ontologies”, *Artificial Intelligence and Law*, in press.

[33] WIELINGA, B.J., SCHREIBER, A.TH., JANSWEIJER, W.H., ANJEWIERDEN, A. and VAM HARMELEN, F. (1994) “Framework and Formalism for Expressing Ontologies”, KACTUS Project Deliverable DO1b.1, University of Amsterdam.