40

Rapid #: -3113833

Ariel
IP: 157.182.199.203

| Status | Rapid Code | Branch Name | Start Date |
|---|---|---|---|
| Pending | NED | Snell Library | 1/29/2010 5:44:33 AM |

**CALL #:** QA75 .K5000
**LOCATION:** NED :: Snell Library :: sp

| | |
|---|---|
| TYPE: | Article CC:CCL |
| JOURNAL TITLE: | Knowledge engineering review |
| USER JOURNAL TITLE: | The Knowledge Engineering Review |
| NED CATALOG TITLE: | The Knowledge engineering review. |
| ARTICLE TITLE: | Knowledge level modelling: concepts and terminology |
| ARTICLE AUTHOR: | Mike Uschold |
| VOLUME: | 13 |
| ISSUE: | 1 |
| MONTH: | March |
| YEAR: | 1998 |
| PAGES: | 5-29 |
| ISSN: | 0269-8889 |
| OCLC #: | |
| CROSS REFERENCE ID: | [TN:407973][ODYSSEY:157.182.232.29/EVL] |
| VERIFIED: | |

**BORROWER:** WVU :: Evansdale Library
**PATRON:** TIm Menzies

| | |
|---|---|
| PATRON ID: | timmenzies |
| PATRON ADDRESS: | |
| PATRON PHONE: | |
| PATRON FAX: | |
| PATRON E-MAIL: | |
| PATRON DEPT: | |
| PATRON STATUS: | |
| PATRON NOTES: | |

RAPID ILL


System Date/Time: 1/29/2010 6:11:16 AM MST

*The Knowledge Engineering Review*, Vol. 13:1, 1998, 5–29
*Printed in the United Kingdom*


# Knowledge level modelling: concepts and terminology

MIKE USCHOLD[1] (EDITOR)

*Artificial Intelligence Applications Institute (AIAI), 80 South Bridge, Edinburgh EH1 1HN, Scotland.*
*Email: m.uschold@ed.ac.uk*

**Abstract**

We address the problem of highly varied and inconsistent usage of terms by the knowledge technology community in the area of knowledge-level modelling. It is arguably difficult or impossible for any standard set of terms and definitions to be agreed on. However, *de facto* standard usage is already emerging within and across certain segments of the community. This is very difficult to see, however, especially for newcomers to the field. It is the goal of this paper to identify and reflect the most common usage of terms as currently found in the literature. To this end, we introduce and define the concept of a knowledge level model, comparing how the term is used today with Newell's original usage. We distinguish two major types of knowledge level model: ontologies and problem solving models. We describe what an ontology is, what they may be used for and how they are represented. We distinguish various kinds of ontologies and define a number of additional related concepts. We describe what is meant by a problem solving model, what they are used for, and attempt to clarify some terminological confusion that exists in the literature. We define what is meant by the term 'problem', and some common notions used to characterise and represent problems. We introduce and describe the ideas of tasks, problem solving methods and a variety of other important related concepts.

## 1 Introduction

In the past few decades, the emergence of knowledge-based technologies in general, and the idea of knowledge-level modelling in particular, has been accompanied by a proliferation of new concepts and terms for which there is no generally agreed usage. It is arguably difficult or impossible for any standard set of terms and definitions to be agreed on, especially in the early development of a field of study. However, the field now has sufficient history and maturity, that a core of *de facto* standard usage is already emerging within and across certain segments of the community. However, outside the core, there is highly varied and inconsistent usage. Thus, unless one is deeply immersed in the field, it is very difficult to realise that such a core exists, much less identify it.

It is the goal of this paper to identify and present this core set of concepts and terms for which *de facto* standard usage is beginning to emerge, and to as accurately as possible, give definitions which *reflect* the most common usage, as currently found in the literature.

The main focus of our attention is in the area of *knowledge-level modelling*, by which we mean, in broad terms, capturing and representing knowledge *without* specific attention being paid to how it will be *implemented*. This includes ontologies and problem solving models.

Very importantly, *this is a* **descriptive** *exercise, not a normative one.* We are not arguing that these are the right terms, or the right definitions, nor advocating that these are to be adopted as a

[1]Now at Boeing Applied Research and Technology, P.O. Box 3707 M/S 7L–40, Seattle, WA, USA 98124.

standard. Instead, we are attempting to describe a snapshot indicating how the current key concepts and terms are used today.

**Objectives**

The clarification of terminology in the area of knowledge-level modelling is expected to make the field more accessible in general, and to be of specific use to:

- people interested and/or involved in the development of methodologies for knowledge engineering;
- people interested and/or involved in the development and use of ontologies; and
- people in the related software engineering and knowledge engineering communities.

To ensure an accurate characterisation of the field, this has been a joint effort involving many people from all over the world. There was a core development team of a few individuals and approximately a dozen reviewers internal to the project. Many more provided insightful comments and criticisms in response to a prior draft circulated by email to special interest groups in related communities.

### *1.1 Background: EuroKnowledge*

The work reported here was funded in part by the EuroKnowledge Project in the CEU ESPIRIT Programme[2]. The aim of EuroKnowledge was to establish recommendations for knowledge representation standardisation, with concentration on the "knowledge level". The initiative viewed its role as a focal point for the consolidation of common views and best practice from knowledge technology users, theorists, and tool vendors. The glossary of terms presented in this paper was an early deliverable on this project. It's initial role was to facilitate communication among the project members. Subsequently, it was intended to be useful to the wider software engineering and knowledge engineering communities.

### *1.2 The approach*

This glossary is intended to be non-controversial, acting in a descriptive rather than normative role. We place primary emphasis on identifying the underlying *ideas* in the area of knowledge level modelling. This glossary is intended to be an accurate record of how terms are currently used to refer to these ideas.

Inevitably, various conflicts and inconsistencies arise with current usage, e.g.

- the same terms being used for many different concepts, or
- many different terms being used for the same concept.

Our emphasis is mainly on identifying such conflicts, rather than trying to resolve them. The aim is to reflect common usage, not dictate it.

**Scope**

Considerable care was taken in determining which terms need to be defined, for the purposes of this glossary. The decision was based on whether the underlying ideas are relevant to the area of standardisation and important to one or more of the following subject areas:

- knowledge level representation formalisms;
- problem solving methods; and
- domain ontologies.

[2]Project number 9806. The participating partners included: Artificial Intelligence Applications Institute (AIAI), the University of Edinburgh (UK); Intelligence Logicielle (ILOG / France); Cap Gemini Innovation (France); DTK Gesellschaft für Technische Kommunikation (Germany); Centro Informazioni Studi Esperienze (CISE / Italy); Coventry University / Lucas KBEC (UK); Swiss Bank Corporation (Switzerland); Digital Equipment (DEC / France) and The EuroKnowledge Association.

**Definitions**
Great care has been taken to produce clear and accurate definitions which best reflects common usage. Specifically, we have:

- put the definitions through a detailed feedback/review cycle [mainly] within the project reaching a true consensus of viewpoint[3];
- taken special care to ensure coherence with respect to other terms, in particular:
  — with minimal redundancy,
  — with minimal circularity,
  — with maximal cross-referring to already defined terms;
- identified and explicitly noted relationships between important related terms which are *not* defined in this glossary;
- taken special care to ensure uniformity of presentation style.

### *1.3 Presentation*

In this paper, we give detailed descriptions of the following core terms: Knowledge Level Model, Problem Solving Model, and Ontology. Each of these begins with a short summary definition, followed by an elaboration of the important aspects of the concept being defined. We also include some additional information, such as examples, key references and related terms.

The remainder of this paper consists of short definitions of a few lines each. To highlight important relationships between the concepts, other defined terms are presented in UPPER case throughout the document (except for in the long definitions of core terms, to increase readability).

A term is defined using a base word; however, for convenience of exposition, we use grammatical variations also in upper case as if they were themselves formally defined (e.g. PROBLEM TYPE, TYPE OF PROBLEM). For example, definitions will look like this:

**TECHNICAL TERM:** a key word or phrase with a special technical meaning in a given context. This glossary consists of TECHNICAL TERMS in the context of KNOWLEDGE TECHNOLOGY. Also, TECHNICAL TERMS comprise the vocabulary of an ONTOLOGY.

Occasionally, a word or phrase that is a defined term will appear in lower case; this means that either:

- it is as part of a larger phrase, e.g. "inference rule", not "INFERENCE rule";
- it is being used as a technical term, but the meaning cannot be assumed to be that given in the definition in this glossary; or
- it is being used in an informal, non-technical sense, in which case its appropriate dictionary meaning applies.

#### *1.3.1 Related terms*

To better understand this glossary, it is helpful to know how its terms and concepts relate to the terms and concepts widely used in other contexts (e.g. other sub-fields and even individual projects tend to have their own vocabularies). Therefore, we list a number of related terms that are fairly commonly used but are *not* defined in this glossary. Where possible, we specify the relationship between these terms and those in the glossary. We distinguish two kinds of related terms:

1. *Synonyms*: commonly used terms that are not defined in the glossary, but which are identical or very similar in meaning to specific defined terms.
2. *Borderline terms*: terms for concepts that are not defined in this glossary, but for which an

[3]This generated large quantities of email correspondence which had to be analysed.

attempt is made to indicate how such concepts might be defined using other defined terms in this glossary.

*1.3.2 Definition template*

For the three core terms, we used the following template as a guide.

**Term:** the word or sequence of words which is the referent
e.g. KNOWLEDGE LEVEL MODELLING;

**Short definition:** text which briefly defines the concept referred to by the term; one or two sentences only.

**Elaboration:** elaboration of Short Definition containing pertinent background, motivation, technical details and explanations helpful in gaining full understanding of the concept. Where appropriate, include direct quotes giving references.

**Example(s):** one or more examples illustrating the concept and/or how it is used.

**Reference(s):** one or more key references in which a definition, description, summary or use of the term is found. Should be listed in order of direct relevance to the term. Except where no better may be found, do not include references that merely mention the term.

**Variations:** Note any variations of this term used. These may fall under the following categories:

*synonyms*: terms that have the same meaning, or nearly so; if not an exact match, be sure to note differences. e.g. *domain level* and *domain layer*.

*grammatical variations*: *knowledge level model* vs *knowledge level modelling*; clarify any difference in meaning that is not obvious.

**Related terms/concepts:** a list of terms denoting closely related concepts to the term being defined. Where possible, indicate the nature of the relationship between the different terms (e.g. part-of, is-a, used-in, different-viewpoint-of).

**Annotation:** (optional) comments by the person filling in the template.

## 2 Fundamentals: Knowledge, reasoning and representation

The definitions given in the main body of this document pre-suppose an understanding of some fundamental notions and terms. In particular, readers are expected to know how the term KNOWLEDGE is being used, and what INFERENCE is. They should be familiar with fundamental notions of LANGUAGE in general and KNOWLEDGE REPRESENTATION LANGUAGES in particular, as well as what is meant by a KNOWLEDGE BASE.

Due to their extremely general nature, they are not defined in the main body of this paper. However, to be clear about how we are using these and other fundamental terms, we include some working definitions in Appendix A. We will use capital letters to refer to them to indicate that definitions are provided. All defined terms are listed in an index.

Here we summarise some of the key ideas and distinctions, introducing a few general terms. There are many ways to classify types of KNOWLEDGE, however we make no attempt to provide a general classification scheme. We do, however, draw attention to one particularly important distinction: whether or not the KNOWLEDGE is specifically about PROBLEM SOLVING. First, we clarify the idea of a DOMAIN.

**DOMAIN:** this is a highly ambiguous term. Two commonly used senses are:

**Sense 1**: the particular subject matter of interest in some context. The subject matter may be anything at all. It may relate to the real world, or be purely hypothetical; it may be concrete or abstract. It may be a general subject such as medicine, or a rather different sort of subject such as how to diagnose faults in a DOMAIN such as medicine.

The nature and scope of a DOMAIN are determined by what is of interest and the context. The context includes the purpose for delimiting a subject area (e.g. to build or design a KBS).

**Sense 2**: the particular subject matter of interest in some context *considered separately* from the problems or tasks that may arise relevant to the subject. For example, a DOMAIN (sense 2)

might be geology; a model of this subject might be referred to as the DOMAIN ONTOLOGY. Separate from this, there might be TASK MODELS describing how to solve problems that arise in geology, or other subjects.

Note: in this document we normally use the term "DOMAIN" ambiguously, failing to specify which of the above two senses in intended. The appropriate sense should be clear from context.

**DOMAIN KNOWLEDGE:** KNOWLEDGE, whose subject matter corresponds to a DOMAIN (Sense 2), i.e. as distinct from problems or tasks in that DOMAIN.

**PROBLEM SOLVING KNOWLEDGE:** KNOWLEDGE specifically about PROBLEM SOLVING. For our purposes, PROBLEM SOLVING KNOWLEDGE is chiefly distinguished from DOMAIN KNOWLEDGE.

A TASK MODEL in CommonKADS (Breuker & van de Velde, 1994) is an example of PROBLEM SOLVING KNOWLEDGE.

**KNOWLEDGE TECHNOLOGY:** collectively, all techniques, methods and tools for capturing, representing and using KNOWLEDGE. Although KNOWLEDGE TECHNOLOGY may be independent from any concern with software, it is probably in this context, that it is most widely applied.

## 3 Knowledge level model/conceptual model

First we briefly define the notion of a MODEL.

**MODEL:** "A purposeful abstraction that allows one to reduce complexity by focusing on certain aspects." (Karbach et al., 1990)

Importantly, no model is completely accurate, in that abstractions simplify reality; however, many models are useful.

### *3.1 What is a knowledge level model?*

**Short Definition** a MODEL constructed in a manner whereby no specific attention is paid to implementation issues and decisions. Typically, such a MODEL:

- expresses some portion of the KNOWLEDGE required by one or more agents to achieve an existing or required level of problem solving competence; or
- will be made during an early phase of KBS construction and serve to specify the requirements for subsequent design and implementation.

**Elaboration** The above definition gives a wide interpretation of the term. For example, there is no restriction on what is being modelled. The only hard constraint is that the process of creating a knowledge level model must *not* have been influenced by implementation concerns.

Knowledge level models vary along a number of dimensions, a key one being the *intended purpose*. We note two categories of purposes for knowledge level models (adapted from Uschold & Gruninger, 1996):

1. system engineering benefits for building KBS;
2. to enhance human understanding and communication.

Some important uses in the first category are:

- *Specification:* to assist the process of identifying requirements and defining a specification for an KBS;
- *Knowledge acquisition:* to drive model-based knowledge acquisition;
- *Reliability:* to assist evaluation of the correctness and/or completeness of knowledge;
- *Re-Usability:* a knowledge level model can serve as the basis for more than one KBS, e.g. a formal ontology represents domain knowledge that may be (or become so by automatic translation) a re-usable and/or shared component in a number of systems.

The most prevalent purpose of a Knowledge-Level Model (KLM) is probably the first; such models are sometimes referred to design models, or competence models. This usage is nicely summarised by Slater (1994) as follows:

> "The knowledge level is a competency type notion: it denotes a capacity for generating action and serves as a specification for what some symbolic systems should perform." (p. 134)

The benefits of implementation-independent descriptions and models (e.g. requirements and design specifications) are well established in the software engineering community. In this context, one can view the idea of a knowledge level model as an application of this same principle for a special class of software (KBS) (Wielinga, 1993).

> "[separating the knowledge and implementation levels] is seen as an important structuring and simplifying principle in KBS project management" (p. 8)

Importantly, a KLM need not be associated with any software. This is illustrated by the following two examples from the second category of purposes for a KLM (i.e. human communication):

- to increase understanding of some area of interest, e.g. of how a human solves problems in a given domain;
- to document knowledge in an unambiguous manner, e.g. corporate knowledge assets.

Note that increased understanding can result from the analysis carried out during the *process* of creating the knowledge-level model, irrespective of how or whether the model is later used.

Other important dimensions along which knowledge level models vary (to greater or lesser extents) include:

**Conceptual Distance** — a knowledge level model is expressed in conceptual terms that are usually (but not always) very close to how people think about what is being modelled—this is sometimes taken to be a defining characteristic of a knowledge level model;

**Formality** — a knowledge level model may be highly informal, expressed in natural language and/or loosely structured diagrams; or highly formal, expressed in a rigorous logic and/or executable; or anything in between.

Important advantages of a formal representation include: reduced ambiguity; capacity to automate analysis of various sorts (e.g. consistency, completeness, soundness) and capacity to automate subsequent KBS construction. For example, it may be possible to translate a knowledge level model for direct use in a KBS (e.g. as in Ontolingua (Farquhar et al., 1995; Gruber, 1995)).

**Problem Solving** — a knowledge level model may be:

- specifically for problem solving (e.g. KADS inference models (Schreiber et al., 1993));
- constructed independently from any specific use in a problem-solving context (e.g. domain models and ontologies);
- a combination of both of the above with equal or varying emphasis on each (e.g. a Model of Expertise includes both).

*Historical Note and Clarification* The knowledge level idea was originally introduced by Newell (1982), in part, to clear up confusion about the terms "knowledge" and "representation". Newell defines "representation" as "a symbol system that encodes a body of knowledge".

The idea was to distinguish that which is being represented (the knowledge) from its represented form (implementation). Taken literally, this implies that no representation can be at the knowledge level, and it is thus meaningless to speak of a language for representing knowledge level models. In practice, both the underlying conceptualisation (e.g. mental model) and the corresponding representation are commonly referred to as the knowledge level model, and this rarely causes confusion.

Though it may be important philosophically, we do not use this distinction to define the knowledge level; also we deliberately avoid use of the term "representation" to characterise a knowledge level model. The knowledge level is characterised in Wielinga (1993) as:

"the separation of the conceptual, knowledge-oriented, aspects from *representational* [our emphasis] and computational issues" (p. 8)

The use of the term "representation" here rather than "implementation" also seems to imply that there can be no representation of a knowledge level model—yet there are notations for representing knowledge level models in KADS. This apparent contradiction is resolved by recognising that the above use of the word "representation" is very restricted, referring to the implementation of the knowledge base in some particular representation language in a KBS. In that sense, a knowledge level model is independent from representation issues. Of course, the knowledge level model must itself be represented (in the wider sense of the word).

**Example(s)** Examples of Knowledge Level Models include:

- a Domain Ontology;
- a Problem Solving Model;
- a Model of Expertise (from KADS).

Note that the latter includes the first two as components.

**Variations**

*Elaborations:*

- Knowledge Level Analysis.
- Knowledge Level Modelling.
- Knowledge Level Representation Language.

**Related terms/Concepts**

*What and Why level*: David Marr's (1982) levels for computer vision encompass the (i) "what-and-why level", where Marr sees "computational theories'," and (ii) the "how level", where Marr sees data structure, algorithms and hardware. He then proceeds to define Type 1 theories with a level (i) and (ii) description, and Type 2 theories only with a level (ii) description. The idea is to distinguish a method from a hack (not his words). This can be readily applied to problem solving in general. A knowledge level model is similar to the what-and-why level.

*Logical, Conceptual, Epistemological & Ontological Levels*: Brachman (1979) distinguishes the logical, conceptual, and epistemological levels which are contrasted from the implementational level. Guarino (1993) further distinguishes an ontological level which is in between the conceptual and epistemological levels.

Insofar as we emphasise lack of implementational concerns as the distinguishing characteristic of a knowledge level model, this includes the conceptual, ontological, and epistemological levels. It is not clear how to view the logical level. From a programming perspective, it seems far from implementation concerns, and thus at the knowledge level. However, from the perspective of building a knowledge base, it is not unreasonable to view a formal representation at the logical level as an implementation of structures identified at the conceptual, ontological, and epistemological levels.

*High-Level Specification Languages*: as regards programming/representation languages, being at the knowledge level is analogous to being at a very high level, so "high" that it no longer need be possible to implement/run/execute the knowledge level model; instead it plays the role of a specification for the eventual implementation.

*Conceptual Schema*: a conceptual schema for a database describes which entities can possibly exist in the universe of discourse, which in turn may be represented in a database. It may also describe what facts apply, or what may (or is required to) happen to various entities [van Griethuysen (1987)]. Insofar as it is intended to be independent from underlying implementation concerns, it is analogous to a Knowledge Level Model.

**Annotation** This definition is not intended to be used to determine whether a given model is or is not at the knowledge level. To make such a determination is rarely important; it is far more useful

to examine carefully what purpose any particular model is to be put and to construct the model accordingly.

**Reference(s)** See (Brachman, 1979; Van de Velde, 1993; Steels & McDermott, 1994; Vinkhuyzen, 1992; Fensel, 1993; Clancey, 1989; Guarino, 1993; Slomon, 1994; Karbach et al., 1990; Wielinga, 1992; Marr, 1982) for source material used to formulate this definition and additional reading.

### *3.2 Some important terms*

**KNOWLEDGE LEVEL REPRESENTATION LANGUAGE:** a KNOWLEDGE REPRESENTATION LANGUAGE suitable for expressing KNOWLEDGE LEVEL MODELS. Such a LANGUAGE may have been specifically designed for knowledge level modelling (e.g. CML (Schreiber et al., 1994)), or may be a general KNOWLEDGE REPRESENTATION LANGUAGE designed mainly for KNOWLEDGE BASE implementation.

KNOWLEDGE LEVEL REPRESENTATION LANGUAGES vary greatly in their degree of formality; e.g. CML is highly structured, yet informal; $(ML)^2$ (van Harmelen & Balder, 1992) is formal.

**ONTOLOGY REPRESENTATION LANGUAGE:** a KNOWLEDGE REPRESENTATION LANGUAGE suitable for expressing ONTOLOGIES. It may have been specifically designed for representing ONTOLOGIES (e.g. Ontolingua (Gruber, 1993)), or may be a general KNOWLEDGE REPRESENTATION LANGUAGE (e.g. Conceptual Graphs (Sowa, 1984; ANSI, 1995)).

## 4 Ontology

First we introduce the important idea of a CONCEPTUALISATION.

**CONCEPTUALISATION:** broadly, a CONCEPTUALISATION is a world view; it corresponds to a way of thinking about some DOMAIN. It can be seen as "a set of informal rules that constrain the structure of a piece of reality" (Guarino, 1997b). It is typically conceived and/or expressed "as a set of concepts (e.g. entities, attributes, processes), their definitions and their inter-relationships" (Uschold & Gruninger, 1996).

A CONCEPTUALISATION may be implicit, e.g. existing only in someone's head, or embodied in a piece of software. For example, an accounting package presumes some world view encompassing such concepts as invoice, and a department in an organisation. A CONCEPTUALISATION that is explicit, is usually called an ONTOLOGY.

### *4.1 What is an ontology?*

**Short Definition** An *explicit* account or representation of some part of a CONCEPTUALISATION (adapted from Guarino & Giaretta (1995)).

An ONTOLOGY may take a variety of forms, but necessarily it will include a vocabulary of terms, and some specification of their meaning. This includes definitions and an indication of how concepts are inter-related which collectively impose a structure on the DOMAIN and constrain the possible interpretations of terms.

An ONTOLOGY is virtually always the manifestation of a shared understanding of a DOMAIN that is agreed between a number of agents. Such agreement facilitates accurate and effective communication of meaning, which in turn leads to other benefits such as inter-operability, reuse and sharing.

**Elaboration** Although there is no universally agreed meaning for the term "ontology", (see Guarino & Giaretta (1995) for a competent analysis of this situation), we believe that our definition

conforms with the most common standard usage. Ours is not as broad as some definitions, e.g. we specifically require an ontology to be explicit, thus distinguishing it from a conceptualisation, for which 'ontology' is sometimes used as a synonym.

In the natural language community, the term "ontology" is frequently used to refer to taxonomies of terms for which there are no definitions (e.g. Miller, 1990). This is because, for the purpose of natural language understanding, it has not been found necessary specify the meaning of the terms, other than the "isa" links. This usage of the term "ontology" is not central to our main focus of knowledge level modelling, and thus we shall say no more about ontologies for natural language.

However, we intentionally make few other restrictions; for example, we do not require ontologies to be logical theories (as in Guarino & Giaretta (1995)). Instead, we identify a number of key dimensions along which ontologies vary. Implicitly these give rise to many "kinds" of ontologies. These are:

- *Formality*: the degree of formality by which a vocabulary is created and meaning is specified;
- *Purpose*: the intended use of the ontology;
- *Subject matter*: the nature of the subject matter (i.e. DOMAIN sense 1) that the ontology is characterising.

**Formality** Four somewhat arbitrary points along what might be thought of as a formality continuum are:

- highly informal: expressed loosely in natural language, e.g. many glossaries fit into this category;
- structured-informal: expressed in a restricted and structured form of natural language, greatly increasing clarity by reducing ambiguity, e.g. the text version of the "Enterprise Ontology" (Uschold et al., 1998) and the glossary of workflow terms produced by the Workflow Management Coalition (1994);
- semi-formal: expressed in an artificial formally defined language, e.g. the Ontolingua version of the Enterprise Ontology[4];
- rigorously formal: meticulously defined terms with formal semantics, theorems and proofs of such properties as soundness and completeness, e.g. TOVE (Gruninger, 1997; Gruninger & Fox, 1995).

The degree of formality required depends a great deal upon the intended *purpose* of the ontology.

**Purpose** The literature is currently rich with descriptions of ontologies and their intended purposes. At a high level, most seem to be intended for some manner of reuse. Some of these purposes are implicit in the various interpretations of the word "ontology" that are commonly found in the literature, as noted in Guarino & Giaretta (1995) (e.g. a vocabulary for Gruber (1995) vs. a meta-level specification of, a logical theory (Schreiber et al., 1995; Wielinga et al., 1994)).

Other factors include the nature of the software with which the ontology will be used, whether it is intended to be shared within a small group and reused within that context for a variety of applications, or whether it is intended to be reused by a larger community. Some view their ontologies mainly as a means to structure a knowledge base; others conceive an ontology to be used as part of a knowledge base, e.g. by loading it in as a set of sentences which will be added to as appropriate; still others view their ontology as an application-specific *inter lingua* (e.g. ATOS (Fuchs & Wheadan, 1995; Jones et al., 1995) and the Enterprise Ontology (Uschold et al., 1998)).

Based on these observations, we identify three main categories of uses for ontologies (see Figure 1; for further details and examples see Uschold & Gruninger (1996)). We have already seen this classification in the definition for KNOWLEDGE LEVEL MODEL. It is repeated and elaborated on here for emphasis, clarity and completeness:

***Communication*** between people. Here, an unambiguous but informal ontology may be sufficient.

[4]Available from `http://www.aiai.ed.ac.uk/entprise/enterprise/ontology.html`

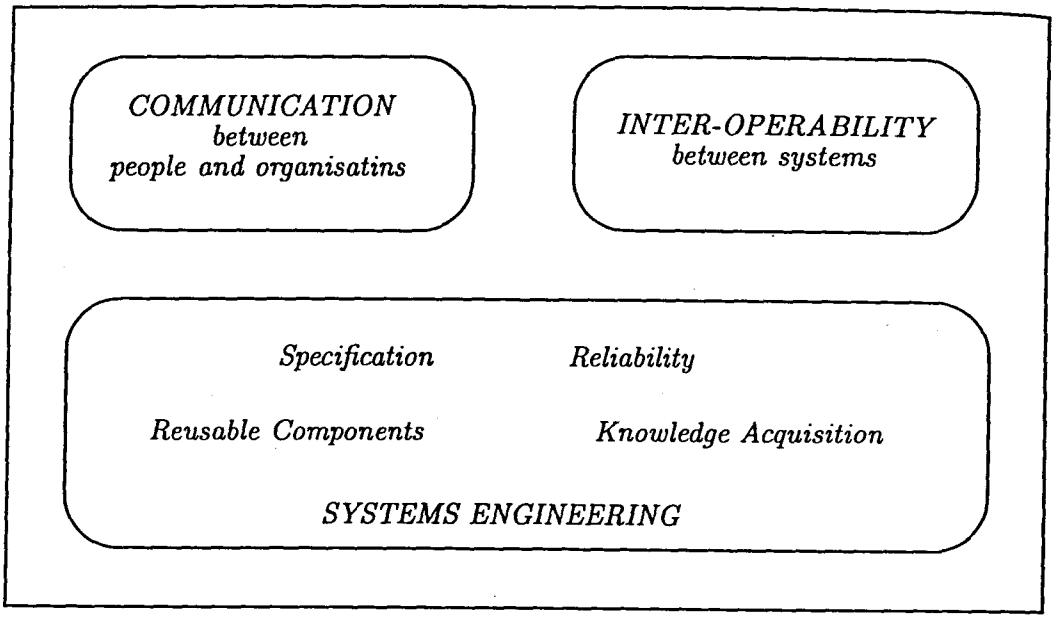


**Figure 1** We identify three main categories of uses for ontologies. Within each, other distinctions may be important, such as the nature of the software, who the intended users are, and how general the domain is.

***Inter-operability*** among *systems* achieved by translating between different modelling methods, paradigms, languages and software tools; here, the ontology is used as an interchange format (see Figure 2).

***Systems engineering benefits:*** in particular,

*Re-usability:* the ontology is the basis for a formal encoding of the important entities, attributes, processes and their inter-relationships in the domain of interest. This formal representation may be (or become so by automatic translation) a re-usable and/or shared component in a software system[5].

*Knowledge acquisition:* speed and reliability may be increased by using an existing ontology as the starting point and basis for guiding knowledge acquisition when building knowledge-based systems (Abu-Hanna & Jansweijer, 1994).

*Reliability:* a formal representation also makes possible the automation of consistency checking resulting in more reliable software.

*Specification:* the ontology can assist the process of identifying requirements and defining a specification for an IT system (knowledge based, or otherwise).

Related to purpose is the notion of *genericity*, which is the extent to which an ontology can or is intended to be reused in a range of different situations. Very generic ontologies (e.g. (Bateman et al., 1990; Lenat & Guha, 1990; Sowa, 1995)) are sometimes referred to as upper-level models and are used for organising substantial portions of human knowledge, e.g. for natural understanding. Less generic ontologies for particular applications are sometimes referred to as "application ontologies" (van Heijst et al., 1996).

The problem solving framework described in Abu-Hanna & Jansweijer (1994) uses genericity an explicit principle for organising and using ontologies.

Note that while there is a strong relationship between genericity and the domain specificity of the subject matter of an ontology, they are not the same. For example, an ontology that most would see as being highly domain-specific can be viewed and used as being generic with respect to any ontologies that are based on it and get still more specific.

**Subject Matter** The subject matter that an ontology characterises can be anything at all. Some

[5]NB. The up front cost of reuse can be considerable, this must be traded off against the longer term benefits.

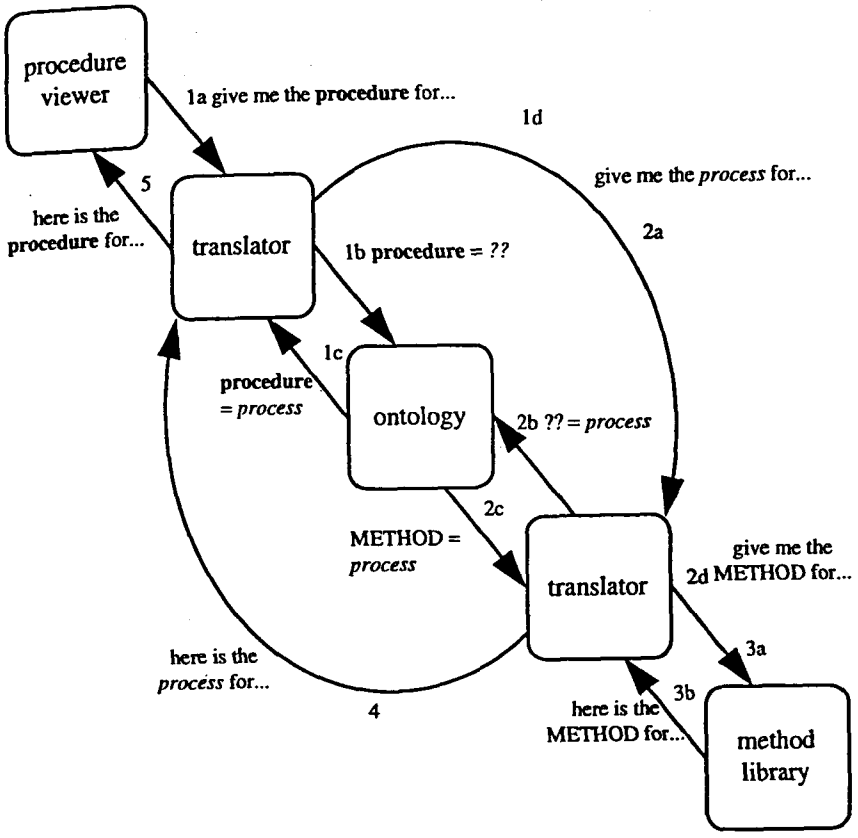


**Figure 2** Interchange format example. This illustrates the use of an ontology as an interchange format to integrate different software tools. The term *procedure*, used by one tool is translated into the term, *method* used by the other via the ontology, whose term for the same underlying concept is *process*.

widely accepted categories are listed below. The first two are subjects considered *separately* from the problems or tasks that may arise relevant to the subject:

1. Specialised subjects such as medicine, geology, or finance.
2. General world knowledge (e.g. Bateman et al., 1990; Lenat & Guha, 1990; Miller, 1990; Sowa, 1995; Guarino, 1997a)
3. the subject matter of Problem Solving,
4. the subject matter of Knowledge Representation Languages.

An ontology in the first category is frequently called a *domain* ontology; an ontology for the second is often called an upper model. An ontology in the third category is called a *task*, *method*, or *problem solving* ontology. The terms *representation ontology* or *meta-ontology* are used to refer to ontologies in the last category.

The boundary between the first and second is very fuzzy in principle, though in practice they are usually quite distinct. Indeed, one of today's research challenges is to create ontologies which bridge the gap between the upper models (Sowa, 1995; Guarino, 1997a) and the more specialised domains.

This is by no means intended to be a complete characterisation of how subject matter may differ. Many sub-dimensions are possible such as uncertainty, or imprecision in the domain. In what follows, we are mainly concerned with subject matter in the first two categories.

The following quote from the SRKB (Shared Re-usable Knowledge Bases) electronic mailing list nicely summarises what an ontology is and the various forms and contexts it arises in:

> "Ontologies are agreements about shared conceptualisations. Shared conceptualisations include conceptual frameworks for modeling domain knowledge; content-specific protocols for communication among

inter-operating agents; and agreements about the representation of particular domain theories. In the knowledge sharing context, ontologies are specified in the form of definitions of representational vocabulary. A very simple case would be a type hierarchy, specifying classes and their subsumption relationships. Relational database schemata also serve as ontologies by specifying the relations that can exist in some shared database and the integrity constraints that must hold for them." (Tom Gruber, 1994)

**Ontologies and Knowledge Bases** There is an important distinction between a *language* for expressing knowledge, and the knowledge itself, as expressed in the language. Some ontologies are conceived and used as the vocabulary with which a knowledge base will be specified. For example, a medical ontology might include terms for disease, symptom, treatment, etc. A particular knowledge base built using these terms might include a taxonomy of particular diseases, examples of treatments, etc. In this situation, one often refers to the ontology *of* or *for* the knowledge base.

In practice, there may be no clear boundary between the ontology and the knowledge base, if both are specified in the same language. Furthermore, *it does not matter!* The difference may be merely one of what part of the knowledge base is shared and agreed on, and what part is more specific. Furthermore, this may change over time.

Time is ill-spent arguing about what is or is not part of the "ontology". What matters is whether the repository of knowledge that you are constructing serves your intended purpose.

*Historical Note* In the discipline of philosophy, the term "ontology" originally refers to a systematic account of existence.

**Example(s)** The terms and definitions comprising the informal version of the Enterprise Ontology (Uschold *et al.*, 1998) may be characterised as follows:

- Level of formality: structured informal.
- Purpose: to facilitate communication between members of the project by giving a consistent terminology; to document and specify a subsequent formal encoding to facilitate inter-operation of enterprise modelling tools (see Figure 2).
- Subject matter: business enterprises (fairly generic).

Other examples of recent and/or ongoing projects in this area are:

- KSL Knowledge Sharing Effort: development of technologies to support the creation and dissemination of libraries of ontologies (Farquhar et al., 1995; Gruber, 1993; Fikes et al., 1991).
- KACTUS Esprit project; deals with reusable technical components in the oil, electrical and ship fields (Schreiber et al., 1995; Wielinga et al., 1994).
- CYC (Lenat & Guha, 1990) is an attempt to model huge portions of human knowledge to facilitate common-sense reasoning.
- UMLS (Lindberg *et al.*, 1993) deals with terminology in the medical fields.
- Conceptual Schema Modelling Facility (CSMF) (van Griethuysen, 1987). This is an ongoing ISO effort.
- NCITS.TC.T2—National Committee for Information Technology Standardization (NCITS), Technical Committee T2, Information Interchange & Interpretation. This committee is currently engaged in an effort to merge various upper level models. See Guarino (1997a) for a discussion of this problem and a proposed approach to address it.

**References** In addition to the references cited above, see (Gennari et al., 1994; Gruber, 1995; Mark et al., 1992; Steels, 1993; van Heijst, 1995; Tu et al., 1994; Moril et al., 1991; Wielinga & Schreiber, 1993) for source material used to formulate this definition and additional reading.

### *4.2 Kinds of ontologies*

As indicated above, there are a number of dimensions which can serve to classify ontologies. There are many terms in use for referring to this or that type of ontology, but very few both are reasonably well defined and have a fairly widely agreed meaning. Some exceptions are given below.

**DOMAIN ONTOLOGY:** an ONTOLOGY whose subject matter corresponds to a DOMAIN (Sense 2), i.e. as distinct from problems or tasks in that DOMAIN.

**PROBLEM SOLVING MODEL:** (see section 5).

**REPRESENTATION ONTOLOGY:** an ONTOLOGY whose subject matter is concerned with KNOWLEDGE REPRESENTATION LANGUAGES. Terms such as class, object, relation, attribute, slot have to be identified and defined. The Frame-Ontology part of Ontolingua (Gruber, 1993) is an excellent example of a REPRESENTATION ONTOLOGY. Another example, currently under development is CDIF[6], an ONTOLOGY for Case tools.

### *4.3 Other important concepts*

**ONTOLOGICAL COMMITMENT:** informally, an agreement on viewpoints on the world being talked about among agents (either human or software). In practical terms, an ONTOLOGICAL COMMITMENT is an agreement to use a vocabulary in a way that is consistent with respect to the terms and definitions that comprise an ONTOLOGY (Gruber, 1993).

The ONTOLOGY is in essence, a specification of ONTOLOGICAL COMMITMENTS that must be made by agents who will use the ONTOLOGY. Being at the knowledge level, ONTOLOGICAL COMMITMENTS must be documented in a way independent from a particular agent's internal representation.

As originally conceived, the idea of ONTOLOGICAL COMMITMENT applied mainly to DOMAIN ONTOLOGIES (Gruber, 1993). Guarino generalised this notion and applied it to REPRESENTATION ONTOLOGIES (Guarino, 1993), arguing for the use of modal logic as the means for expressing the commitments. For example, an agent committing to an ONTOLOGY of unary predicates would agree to use the primitives of an KNOWLEDGE REPRESENTATION LANGUAGE in a certain way.

Similarly, one can view high level specifications of software as commitments: "An agent commits to a knowledge-level specification if its observable actions are logically consistent with the specification" (Mark et al., 1992). For a software agent, the observable actions are characterised by the produced outputs according to the inputs.

**KNOWLEDGE SHARING; KNOWLEDGE REUSE:** the situation whereby a single KNOWLEDGE BASE can be used as a component across multiple applications and/or be used by multiple agents simultaneously in a single application. One important way to achieve KNOWLEDGE SHARING and REUSE is via KNOWLEDGE INTERCHANGE.

**KNOWLEDGE INTERCHANGE:** the exchange of the information content of two or more independently defined KNOWLEDGE BASES. This enables dynamic co-operation of software agents for problem solving tasks. The transmission of information content requires a communication protocol which may be based on either:

- *ad hoc* translations between the various internal representations, or
- an INTERCHANGE FORMAT (see Figure 3).

Development of an INTERCHANGE FORMAT for KNOWLEDGE INTERCHANGE raises knowledge modelling and ontological issues. KIF (Genesereth & Fikes, 1992) is probably the best example of such an INTERCHANGE FORMAT.

NB: an extremely difficult and unsolved problem is the development of sufficiently powerful translators to convert between knowledge bases. Some initial experiments have been performed, and there some useful syntactic translation has been demonstrated (Gruber, 1993; Farquhar et al., 1995). Many difficult semantic problems of translation remain largely unaddressed.

**INTERCHANGE FORMAT:** a LANGUAGE used as a *lingua franca* to enable translation between any of a number of separate LANGUAGES, e.g. for the purpose of KNOWLEDGE INTERCHANGE. To translate from LANGUAGE $l_i$ to $l_j$ and *vice versa*, a translator is required

[6]See `http://www.cdif.org/intro.html`

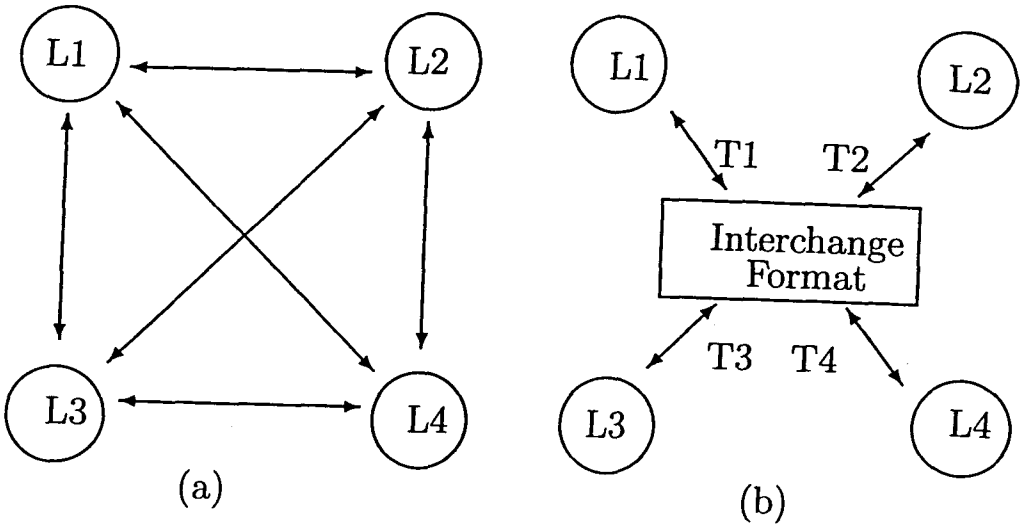


**FIgure 3** Interchange format. To translate from language $L_i$ to $L_j$ and *vice versa*, a translator is required between $L_i$ and the interchange format and another between the interchange format and $L_j$. Thus, given $n$ languages, only $O(n)$ translators are required, not $O(n^2)$.

between $l_i$ and the INTERCHANGE FORMAT and another between the INTERCHANGE FORMAT and $l_j$. Thus, given $n$ LANGUAGES, this approach requires only $O(n)$ translators, rather than $O(n^2)$ (see Figures 2 and 3).

There is a wide range of important issues for which there are no agreed terms that we believe are worth introducing and defining. Two important ones are listed below:

- Methodologies for ontology design and evaluation. See Uschold & Gruninger (1996), Gruber (1995) and Gómez Pérez et al. (1995) for some preliminary work in this area.
- The extent to which the so-called "interaction problem" limits the reuse and sharing of KNOWLEDGE BASES. The problem says [DOMAIN] KNOWLEDGE BASES cannot be generic insofar as their nature and content may depend on the tasks that an application is intended to perform. See van Heijst (1995), van Heijst et al. (1996) and Guarino (1997b) for a discussion of this matter.

See Uschold & Gruninger (1996) for a comprehensive introduction to the emerging field concerned with the development and use of ONTOLOGIES.

### *4.4 Related terms*

**Synonyms**

*Inter-Lingua*: INTERCHANGE FORMAT

*Conceptual schema*: virtually the same in meaning as an ONTOLOGY, though of different historical origins.

*Meta-ontology*: a REPRESENTATION ONTOLOGY where the representation language is an ONTOLOGY REPRESENTATION LANGUAGE (not quite but nearly synonymous).

**Borderline terms**

*Glossary*: a set of defined terms, may be an informal ONTOLOGY.

*Application ontology*: an ONTOLOGY intended to be used in a single application rather than across many applications.

*Generic ontology*: an ONTOLOGY intended to be used across many applications.

*Domain typology*: a classification of DOMAIN KNOWLEDGE in to various kinds of DOMAINS.

## 5 Problem solving

### *5.1 What is a problem solving model?*

**Short definition** A Problem Solving Model (PS model) is a description of problem solving at the knowledge level. A PS model specifies which bodies of knowledge participate in problem solving and how they relate to each other. A PS model may specify the problem solving methods to be applied to specific problem solving tasks.

**Elaboration** The development of PS models reflects the growing awareness of an inherent structure in knowledge-based problem solving and the need to observe this structure when developing knowledge-based problem solving systems. By expressing the structure of problem solving at the knowledge level, abstracting from implementation details, a PS model may provide the basis for a reusable framework for developing knowledge-based problem solving systems and for obtaining reusable solution components.

A PS model may also be useful for a single application, not intended for reuse. This is because the explicit account of the problem solving process can serve as the basis for designing and documenting a KBS which uses it.

The following are the main advantages which a PS model might promise for knowledge-based system development:

- A PS model guides the implementation of a knowledge-based system up to the extent that the low level software structure may reflect the structure of the PS model. This is called structure-preserving design (Schreiber et al., 1993).
- A PS model provides explanatory power and thus facilitates debugging and modifications.
- A PS model identifies domain independent problem solving structures, thus reducing the development efforts for problems of the same type and facilitating software reuse (Breuker & van de Velde, 1994).
- A PS model may support knowledge acquisition (together with a domain ontology). At the conceptual level of a PS model, the data which must be provided for problem solving may be specified in a manner compatible with human thinking and understanding (Benjamins, 1995).
- A PS model may result in increased efficiency (Benjamins et al., 1996; Fensel & Straatman, 1996).

A generally accepted format for PS models does not exist at this time. PS models may differ with respect to:

- the scope of the problem-solving context which is covered;
- the level of abstraction;
- the description formalism;
- the categorisation of problems into problem types; and
- the problem solving methods employed.

**Historical note and clarification** An early example of a PS model is the state-space model for problem solving which essentially consists of states, operators (or actions) and a search control structure.

The need for more differentiated PS models became apparent when distinct problem solving behaviour could be identified for particular problem types (Stefik et al., 1992), e.g. interpretation or construction problems. Much of the subsequent scientific work related to PS models focussed on distinctions expressed in terms of problem types, tasks and problem-solving methods. An elaborated set of PS models has been proposed in terms of "Generic Tasks" (Bylander & Chandrasekaran, 1988). Other approaches include "Role-Limiting Methods" (McDermott, 1988) and "Components of Expertise" (Steels, 1990).

Based on more recent developments, the CommonKADS (Breuker & van de Velde, 1994) methodology provides a much wider framework for PS models. Problem-solving is modelled in

terms of a "product model" which encompasses all knowledge bodies pertinent to developing and employing a knowledge-based problem solving system. These are:

- an organisational model which provides the organisational context for problem-solving activities,
- a task model defining the tasks and activities which are to be performed,
- an agent model describing the properties of agents carrying out the tasks,
- a communication model for inter-agent communication,
- an expertise model which describes the knowledge of an agent relevant to a particular task, and
- a design model which describes the realisation of problem-solving behaviour in computational and representational terms.

PS models in a more narrow sense, are contained in the CommonKADS model of expertise which includes a domain model, a task model and an inference model. The task model relates a problem definition to problem solving methods. It may be structured as a hierarchy of tasks, where each task is specified by a goal and by what is required to achieve the goal.

**Example(s)** Examples for PS models applicable to the problem types assignment, diagnosis, configuration and design are given in Breuker and van de Velde (1994).

**Variations**

- problem solving theory: formal structure for problem solving, as presented, for instance, in AI textbooks.
- product model: as defined in CommonKADS.

**Related terms/concepts** As indicated in the elaboration above, the scope of PS models as presented in the literature may vary widely from problem solving methods which constitute the core of problem solving activities to distributed component models which encompass the complete problem solving context. If one adopts the wide-sense notion of a PS model as in CommonKADS, a problem solving method is but one component of a PS model. Similarly, tasks and domain knowledge are sometimes viewed as PS model components.

*5.2 General*

Here we define some important general terms related to PROBLEM SOLVING MODELS.

**PROBLEM SOLVING:** the activity of identifying, formulating, and obtaining a solution to a PROBLEM.

PROBLEM SOLVING is frequently viewed as finding a set of ACTIONS, which if performed, will achieve a GOAL STATE. PROBLEM SOLVING may or may not entail performance of the ACTIONS in the real world (e.g. planning).

**PROBLEM:** very generally, a PROBLEM is a "collection of information that some agent will use to decide what to do" (Russel & Norvig, 1995). Typically, a PROBLEM is viewed as a discrepancy between an INITIAL STATE and a GOAL STATE.

A formulation (i.e. description or definition) of a PROBLEM:

- specifies how to represent STATES;
- identifies an INITIAL STATE and a GOAL STATE;
- defines ACTIONS which (hopefully) can transform the INITIAL STATE into the GOAL STATE.

This view may be recursively applied; thus we may speak of the [next-level] PROBLEM of identifying and/or formulating the [base-level] PROBLEM. The GOAL STATE of this [next-level] PROBLEM is to have formulated the [base-level] PROBLEM.

**STATE:** informally, one or more facts relevant to PROBLEM SOLVING. Formally, the facts may

be PROPOSITIONS or SENTENCES in some KNOWLEDGE REPRESENTATION LANGUAGE (e.g. has-symptom(runny-nose)).

STATES may contain information referring to the real world (external) and/or to data in the PROBLEM SOLVING agent (internal).

**ACTION:** a basic activity of a PROBLEM SOLVING agent that transforms one STATE into another. ACTIONS are described in terms of the STATE changes which they effect.

**GOAL (or GOAL STATE):** a STATE which some agent wants, needs or intends to hold; or is responsible for making sure it holds. A GOAL is a critical part of a PROBLEM formulation.

**INITIAL STATE:** the current STATE at the beginning of PROBLEM SOLVING. It includes both external information about the real world and internal information possessed by the PROBLEM SOLVING agent.

**STATE SPACE:** the set of STATES which can be reached from an INITIAL STATE by successively carrying out ACTIONS taken from a specific repertory.

### *5.3 Methods and models*

**TASK:** the job of applying one or more PROBLEM SOLVING METHODS to a PROBLEM in a particular domain. For example, the application of hierarchical classification to fault finding in four-stroke four cylinder engines of a special manufacturer (Karbach *et al.*, 1990).

A TASK is said to be *executed* when the PROBLEM SOLVING METHODS are applied to solve the PROBLEM.

**PROBLEM SOLVING METHOD:** a PROBLEM SOLVING METHOD (PS METHOD) is a reusable, domain independent description of how to reach a solution to a PROBLEM (Karbach et al., 1990). A PS METHOD is characterised by the type of ACTIONS supported, how a type of ACTION is selected and instantiated under a given set of circumstances, and the execution of the ACTIONS. The basic procedure consists of:

1. *identifying* what type of ACTION comes next, and which particular ACTIONS of that type are available;
2. *selecting* a particular ACTION to execute;
3. *executing* the ACTION.

A PROBLEM SOLVING METHOD will typically indicate how a particular TASK can be decomposed into sub-TASKS, how sub-TASKS contribute to higher-level TASKS, how their execution is controlled, and which requirements have to be met by DOMAIN KNOWLEDGE for the PS METHOD to work.

A PROBLEM SOLVING METHOD is reusable in different DOMAINS, and it may also may be applicable to more than one PROBLEM TYPE. For example, "Propose-and-Revise" is a PS METHOD which can be applied for design and planning problems, among others. "Cover-and-differentiate" is another example of a PS METHOD.

**Relationship to PS MODEL** 'PS MODEL' is a very general term; any PS METHOD may itself be viewed as an example of a PS MODEL. A MODEL OF EXPERTISE in CommonKADS is an example of a PS MODEL that is not *itself* a PS METHOD, though it may contain PS METHODS as components. Other components might be a DOMAIN ONTOLOGY, various tasks, etc.

**PROBLEM TYPE:** a category of PROBLEMS with similar characteristics. PROBLEM typologies may differ depending on the emphasis on application, task, architecture or method similarities. For example, PROBLEM TYPES in CommonKADS are defined according to the kind of solution which must be obtained. Diagnosis is the PROBLEM TYPE where one wants to find components or structures conflicting with a behavioural model (Breuker & van de Velde, 1994).

Other PROBLEM TYPES from various approaches to PROBLEM SOLVING include:

analysis, synthesis, assessment, design, abstraction, planning, matching, assignment, prediction and monitoring.

*5.3.1 CommonKADS*

CommonKADS is a particularly important effort in the development of PROBLEM SOLVING MODELS. Here we define some terms that originated in that project, but are very widely used.

**MODEL OF EXPERTISE:** in CommonKADS (Breuker & van de Velde, 1994), a formal structure for representing and processing expert knowledge in knowledge based systems. The range of proposed models includes Heuristic Models (typically rule-based), Deep Models (based on structural and functional information), Implicit Models (nonsymbolic, connectionist), Competence Models (representation-independent high-level descriptions of expertise) and Distributed Models (for multiagent problem solving). The CommonKADS methodology is a systematic approach to expertise modelling.

**INTERPRETATION MODEL:** a description of a PROBLEM SOLVING METHOD in CommonKADS.

**TASK MODEL:** in CommonKADS, a central component of the MODEL OF EXPERTISE. It specifies the control structure of PROBLEM SOLVING behaviour in terms of subtasks and their relations.

NB: "TASK MODEL" is a wholly separate technical term from "TASK"; it is not possible to derive the meaning of the larger term from the meaning of the separately defined core term. In other words, there is overloading on the word "TASK".

*5.4 Related terms*

**Synonyms**

*Operator*: ACTION.
*Situation*: STATE.
*World state*: STATE restricted to external information (i.e. referring to the real world).

**Borderline terms**

*Generic task:* a PROBLEM TYPE in the so-called Generic Task approach to PROBLEM SOLVING (Bylander & Chandrasekaran, 1988; Chandrasekaran, 1988).
*Role-limiting method:* a PROBLEM TYPE in the Role-Limiting Method approach to PROBLEM SOLVING (McDermott, 1988).

## 6 Summary and conclusions

In this paper, we have addressed the problem of highly varied and inconsistent usage of terms in the area of knowledge level modelling. We have identified and presented a core set of concepts, terms and definitions which we believe reflects emerging *de facto* standard usage.

This is a descriptive, not normative, exercise. We are not recommending that these terms and definitions be adopted as a standard, but instead are providing a snapshot of how terms are most widely used today. Our main contribution is to provide for the first time, a consistent and coherent summary of the main ideas in the field.

## Acknowledgements

This work was funded in part, under the CEU ESPRIT programme (project 9806). The production of this glossary was a joint effort by the members of the EuroKnowledge Project team which includes Ian Filby, Massimo Gallanti, Mari Georges, Philippe Gobinet, Alex Goodall, Uwe

Haferstroh, Stephen Kneebone, Bernd Neumann, Louis F. Pau, Catherine Peyralbe, Paolo Pogliano, Austin Tate, Mike Uschold and Michael Wolf.

Ian Filby assisted in developing the overall structure of the document. The definition of 'knowledge level model' benefits from feedback provided by Mari Georges, Nicola Guarino, Paolo Pogliano, Giovanni Guida, Paul van der Vet and Michael Wolf.

Various terms and definitions related to ontologies were contributed by Phillipe Gobinet and based directly on the work of Nicola Guarino and Gertjan van Heijst who kindly checked the definitions for accuracy and gave helpful comments. Useful feedback on the ontology terms was also provided by Paul van der Vet. Martin King and Tony Sarris helped clarify the relationship between conceptual schema and ontologies.

Bernd Neuman authored the majority of the section on problem solving; Ian Filby provided much useful input and feedback regarding the final choice of terms and definitions in this section.

I am grateful to Francois Arlabosse, Richard Benjamins, Mari Georges, Masahiro Hori, Brice LePape, Bob Malcolm, Tony Sarris, Austin Tate, Walter Van de Velde, Paul van der Vet and Michael Wolf for carefully reading and giving detailed feedback on previous drafts. Pat Cassidy and Natalya Fridman also contributed helpful comments.

## Appendix A: Fundamentals: Knowledge, Reasoning and Representation

### *A.1 What is knowledge?*

**KNOWLEDGE:** anything that can be known or believed about a real or hypothetical world. Two important kinds of KNOWLEDGE are matters of fact and ways of reasoning (e.g. transitivity). There are many other kinds.

This is an extremely broad interpretation of the term, intended to incorporate what is commonly referred to as either "data" or "information". What KNOWLEDGE includes that is normally different from data and information, is difficult to precisely characterise. However, the following give some indications:

- represents expert human problem solving mechanisms;
- often more natural to represent symbolically rather than numerically;
- often has a heuristic aspect, rather than being [just] routine mathematical algorithms;
- representations of reasoning and processing mechanisms that can themselves be processed as data as opposed to, say, a hard-wired equation in a spreadsheet program.

**Notes**

1. the term "KNOWLEDGE" is also frequently used to refer to the KNOWLEDGE BASE contents, i.e. the *represented* KNOWLEDGE. Although, strictly speaking, this is ambiguous usage, it rarely causes a problem. The distinction is crucial, however, when considering the semantics of a representation language.

### *A.2 Reasoning*

**INFERENCE:** the generation of new KNOWLEDGE from existing KNOWLEDGE, e.g. using modus-ponens.

One may speak of the *process* of INFERENCE, or alternatively one may speak of *an* INFERENCE, which refers to a particular case of carrying out the process.

In logic, an INFERENCE is achieved by manipulation of logical formulas using inference rules; e.g. deduction. More generally, an inference can be viewed as a primitive action type, defined through its name, relating input and output knowledge roles.

**INHERITANCE:** a form of inference whereby either:

- the characteristics of a kind of thing (often called a "class") are assumed to be characteristics of things of that kind (often called "instances"); or
- the characteristics of a more general kind of thing (often called a "super-class") are assumed to be characteristics of less general kinds of things (often called "sub-classes"), with respect to a generalisation/specialisation hierarchy.

**Notes**

1. In logic-based representations, INHERITANCE is achieved by simple deductive INFERENCE and a "kind of thing" is typically represented and referred to as a "sort" or a "type" rather than a "class".

*A.2.1 Other important terms*

**KNOWLEDGE BASE:** a set of SENTENCES in some KNOWLEDGE REPRESENTATION LANGUAGE.

What is stored in a KNOWLEDGE BASE is often called just KNOWLEDGE, but strictly speaking, it is *represented* KNOWLEDGE. Typically, it is in the form of facts or rules, although there are other possibilities.

**LOGICAL THEORY:** a collection of ASSERTIONS. In AI and applied logic, it is often restricted to a particular DOMAIN (e.g. time, measurements, actions). In the discipline of mathematical logic, this is usually referred to simply as a "theory". ONTOLOGIES and KNOWLEDGE BASES may be expressed as LOGICAL THEORIES.

*A.3 Language*

*A.3.1 What is language?*

**LANGUAGE:** a means of communication and/or representation consisting of (1) a set of basic expressions; (2) rules and conventions for combining expressions into more complex expressions; and (3) the association of meaning with expressions. (1) and (2) comprise the SYNTAX of a LANGUAGE, and (3) is the SEMANTICS.

Expressions are usually in the form of written symbols or spoken words (e.g. English, C++), but anything else is possible (e.g. movements used in sign language).

**NATURAL LANGUAGE:** a language such as English, which people speak and which develops in some sense automatically rather than being consciously invented. Chiefly contrasted with ARTIFICIAL LANGUAGE.

**ARTIFICIAL LANGUAGE:** a LANGUAGE invented by humans for some specific purpose; chiefly contrasted from a NATURAL LANGUAGE. Three important ways to categorise an ARTIFICIAL LANGUAGE are:

- what it is used for;
- degree of formality;
- degree and nature of how it may be automatically processed.

Two key uses of ARTIFICIAL LANGUAGES are KNOWLEDGE representation and programming[7].

ARTIFICIAL LANGUAGES range from extremely *ad hoc* to highly formal. For some, neither their SYNTAX nor SEMANTICS is both explicit and well-defined. Others have rigorously defined SYNTAX (e.g. using BNF), but loosely (if at all) defined SEMANTICS. Finally, a logic has a rigorously defined SYNTAX *and* SEMANTICS.

The degree of formality impacts on, but does not fully determine, the automated processing capabilities. These include analysis, consistency checking, INFERENCE, and executability.

[7]NATURAL LANGUAGE also can be seen as representing knowledge, however, we are restricting our attention to more formal languages.

*A.3.2 Syntax and semantics*

**SYNTAX:** the set of basic (i.e. primitive) expressions (e.g. symbols) and combination rules which together determine what expressions may be written in a LANGUAGE. The SYNTAX of an ARTIFICIAL LANGUAGE is often expressed using a context-free grammar (e.g. BNF notation).

**SEMANTICS:** the association of meaning with expressions in a LANGUAGE, e.g. how do SENTENCES in a LANGUAGE relate to states of affairs in the world that are being represented.

**FORMAL SEMANTICS:** a characteristic of an ARTIFICIAL LANGUAGE whereby the SEMANTICS is rigorously defined. A meaning function is defined which maps TERMS and SENTENCES in the LANGUAGE to what they represent in the world. For example, this function might map the symbol "1" to the number 1, the symbol "+" to the mathematical function addition and the symbol "Albert Einstein" to the brilliant physicist who invented the general theory of relativity.

The meaning of a SENTENCE is often referred to as its "Interpretation", *i.e.* the interpretation of a SENTENCE is the fact to which it refers. If the fact referred to is part of the actual world, the SENTENCE is said to be *true*.

**SENTENCE:** any expression allowed by the SYNTAX of a LANGUAGE.

**ASSERTION:** (in logic) a SENTENCE added to a KNOWLEDGE BASE. This term is sometimes used to refer to the operation of adding the SENTENCE, rather than the SENTENCE itself. Whether "ASSERTION" refers to the activity of asserting or to the thing asserted is usually clear from context.

**PROPOSITION:** (in propositional logic) a non-decomposable SENTENCE, e.g. "Nixon is a liar", which may be said to be true or false. Often referred to as a "Fact".

**PREDICATE:** (in logic) a property of an object, or a relationship between two or more objects (e.g. *liar*(*nixon*); *loves*(*mike*, *karen*)).

**TERM:** (in logic) a logical expression that refers to an object in the universe of discourse.

*A.3.3 Kinds of languages*

**PROGRAMMING LANGUAGE:** an ARTIFICIAL LANGUAGE that can only refer to lexical object types (LOT). A LOT is anything that can be completely represented in symbols on a sheet of paper or a computer storage device, e.g. numbers, character strings, and lists, arrays, records, or other structures made up of them.

A NOLOT is anything that can *not* be so represented, such as physical objects, events, situations, and abstractions like justice or happiness. If you can store it on a disk and recover the same thing you stored, it's a LOT. If you can't, it's a NOLOT (e.g. the person John is a NOLOT, and the string 'John' is a LOT that represents the name used to refer to John).[8]

A PROGRAMMING LANGUAGE is chiefly distinguished from a KNOWLEDGE REPRESENTATION LANGUAGE which mainly refers to NOLOTS, though it can also refer to LOTS.

**KNOWLEDGE REPRESENTATION LANGUAGE:** a KRL is an ARTIFICIAL LANGUAGE whose SENTENCES represent KNOWLEDGE. This is essentially equivalent to:

> "a symbol system that encodes a body of knowledge." (Newell, 1982)

Primarily, a KRL is used to refer to NOLOTS (things that are not lexical object types), thus

[8]The distinction between LOTS and NOLOTS is made by the NIAM system. The idea to use it to distinguish a programming language from a representation language was brought to my attention in an email message from John Sowa (available on the www from `http://www-ksl.stanford.edu/email-archives/srkb.index.html`; the message title is: "knowledge languages vs. programming languages". I have quoted from this and related messages. For further information on NIAM, see `http://wwwedu.cs.utwente.nl/misop020/`

distinguishing it from a PROGRAMMING LANGUAGE. A KRL can also refer to LOTS, and it is sometimes necessary to do so.

Two main uses of a KRL are:

1. as an aid in the process of conceptualisation, i.e. in producing KNOWLEDGE LEVEL MODELS;
2. to construct a KNOWLEDGE BASE, which in turn is used to enable automated reasoning and/or behaviour of an artificial agent to solve problems or achieve tasks for the benefit of humans. This is the more prevalent use of a KRL.

As is the case for any ARTIFICIAL LANGUAGE, KRLs vary considerably as to their degree of formality and the nature of automatic processing that is supported. Formal KRLs are equally suitable for both above-mentioned uses. If the KRL is informal, it is largely unsuitable for use (2).

*A.3.4 Related terms*

**Synonyms**

*Fact*: PROPOSITION, ASSERTION, and/or SENTENCE

*Formula*: a SENTENCE in a logic-based LANGUAGE.

**Borderline terms**

*Formalism:* a very informal term used in a wide variety of ways from very general to fairly specific interpretations. These include:

- an ARTIFICIAL LANGUAGE which to some reasonable extent is defined in a careful and precise manner;
- an ARTIFICIAL LANGUAGE with a formal SYNTAX and SEMANTICS;
- an ARTIFICIAL LANGUAGE which is executable.

*Representation formalism:* synonym for KNOWLEDGE REPRESENTATION LANGUAGE. Variations on interpretations of this term are similar to that for "Formalism", above. We have defined KNOWLEDGE REPRESENTATION LANGUAGE to be sufficiently broad, to include relatively informal as well as highly rigorous LANGUAGES.

**References**

Abu-Hanna, A and Jansweijer, W, 1994. "Modelling domain knowledge using explicit conceptualisation" *IEEE Expert*, **9**(5) 53–64.

ANSI, 1995. "Conceptual graphs, a presentation language for knowledge in conceptual models; working draft of proposed american national standard" *Technical Report X3T2/95-019r1*, ANSI. (Contact person: J.F. Sowa.)

Bateman, JA, Kasper, RT, Moore, JD and Whitney, RA, 1990. "General organisation of knowledge for natural language processing: the Penman upper model" *Technical report*, USC/Information Sciences Institute, Marina del Rey, CA.

Benjamins, VR, 1995. "Problem-solving methods for diagnosis and their role in knowledge acquisition" *International Journal of Expert Systems: Research and Applications* **8**(2) 93–120.

Benjamins, VR, Fensel, D and Straatman, R, 1996. "Assumptions of problem-solving methods and their role in knowledge engineering" In W Wahlster, editor, *Proc. ECAI-96*, 408–412. Wiley.

Brachman, RJ, 1979. "On the epistemological status of semantic networks" In NV Findler, editor, *Associative Networks: Representation and Use of Knowledge by Computers*, Academic Press.

Breuker, J and van de Velde, W, 1994. *CommonKADS Library for Expertise Modeling*, IOS Press.

Bylander, T and Chandrasekaran, B, 1988. "Generic tasks in knowledge-based reasoning: The right level of abstraction for knowledge acquisition" In B Gaines and J Boose, editors, *Knowledge Acquisition for Knowledge-Based Systems: Volume 1* 65–77. Academic Press.

Chandrasekaran, B, 1988, " Generic tasks as building blocks for knowledge based systems: The diagnosis and routine design examples" *The Knowledge Engineering Review* **3**(3) 183–210.

Clancey, WJ, 1989. "The knowledge level reinterpreted: Modeling how systems interact" *Machine Learning* **4** 285–291.

Van de Velde, W, 1993. "Issues in knowledge level modelling" In J-M David, J-P Krivine and R Simmons, editors, *Second Generation Expert Systems* 211–231. Springer-Verlag.

Farquhar, A, Fikes, R, Pratt, W and Rice, J, 1995. "Collaborative ontology construction for information integration" *Technical Report KSL-95-63*, Stanford University Knowledge Systems Laboratory.

Fensel, D, 1993. "The reconciliation of symbol and knowledge level" *Technical report*, University of Karlsruhe.

Fensel, D and Straatman, R, 1996. "Problem-solving methods: Making assumptions for efficiency reasons" In N Shadbolt, K O'Hara and G Schreiber, editors, *Lecture Notes in Artificial Intelligence, 1076: 9th European Knowledge Acquisition Workshop, EKAW-96* 17–32. Springer-Verlag.

Fikes, R, Cutkosky, M, Gruber, T and van Baalen, J, 1991. "Knowledge sharing technology project overview" *Technical Report KSL-91-71*, Stanford University, Knowledge Systems Laboratory.

Fuchs J and Wheadon, J, 1995. "Prospective applications of ontologies for future space missions" In *The Impact of Ontologies on Reuse, Interoperability and Distributed Processing* 83–96. Unicom Seminars.

Genesereth, MR and Fikes, RE, 1992. "Knowledge interchange format, version 3.0 reference manual" *Technical Report Logic-92-1*, Computer Science Department, Stanford University.

Gennari, JH, Tu, SW, Rothenfluh, TE and Musen, MA, 1994. "Mapping domains to methods in support of reuse" *International Journal of Human-Computer Studies*.

Gómez-Pérez, A, Juristo, N and Pazos, J, 1995. "Evaluation and assessment of knowledge sharing technology" In NJ Mars, editor, *Towards Very Large Knowledge Bases—Knowledge Building and Knowledge Sharing 1995* 289–296. IOS Press.

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

Gruber, T, 1995. "Towards principles for the design of ontologies used for knowledge sharing" *International Journal of Human-Computer Studies* **43**(5/6) 907–928.

Gruninger, M, 1997. "Integrated ontologies for enterprise modelling" *International Conference on Enterprise Integration Modeling Technology*, Torino.

Gruninger, M and Fox, MS, 1995. "The logic of enterprise modelling" In J Brown and D O'Sullivan, editors, *Reengineering the Enterprise* 83–98. Chapman & Hall.

Guarino, N, 1993. "The ontological level" In R Casati, B Smith and G White, editors, *Philosophy and the Cognitive Sciences* Hlder-Pichler-Tempsky. (Presented at IV Wittgenstein Symposium, Kirchberg, Austria, 1993.)

Guarino, N, 1997a. "Some organizing principles for a unified top-level ontology" Revised version of a paper appeared at *AAAI 1997 Spring Symposium on Ontological Engineering* Available from `http://www.ladseb.pd.cnr.it/Infor/Ontology/Papers/OntologyPapers.html`.

Guarino, N, 1997b. "Understanding, building and using ontologies—a commentary to 'Using explicit ontologies in KBS development' by van Heijst, Schreiber and Wielinga" *International Journal of Human and Computer Studies* **46**(3/4) 293–310.

Guarino, N and Giaretta, P, 1995. "Ontologies and knowledge bases- towards a terminological clarification" In NJ Mars, editor, *Towards Very Large Knowledge Bases—Knowledge Building and Knowledge Sharing 1995* 25–32. IOS Press.

van Griethuysen, JJ (editor), 1987. "Information processing systems—concepts and terminology for the conceptual schema and the information base" *Technical Report ISO TR9007:1987*, International Standards Organisation, Geneva, Switzerland.

van Harmelen, F and Balder, JR, 1992. "$(ML)^2$: a formal language for KADS models of expertise" *Knowledge Acquisition* **4**(1). (Special issue on 'The KADS approach to knowledge engineering'.)

van Heijst, G, 1995. "The role of ontologies in knowledge engineering" *PhD thesis*, University of Amsterdam.

van Heijst, G, Wielinga, B and Schreiber, Th, 1996. "Using explicit ontologies for KBS development" *International Journal of Human and Computer Studies* (accepted).

Jones M, Wheadon, J, Whitgift, D, Niezatte, N, Timmermans, R, Rodriguez, I and Romero, R, 1995. "An agent based approach to spacecraft mission operations" In NJ Mars, editor, *Towards Very Large Knowledge Bases—Knowledge Building and Knowledge Sharing 1995* 259–269. IOS Press.

Karbach, W, Linster, M and Voß, A, 1990. "Models of problem-solving: One label—one idea?" In *Current Trends in Knowledge Acquisition* Frontiers in Artificial Intelligence and Applications, 173–189. IOS Press.

Lenat, D and Guha, RV, 1990. *Building Large Knowledge-based Systems: Representation and Inference in the CYC Project*. Addison-Wesley.

Lindberg, DAB, Humphreys, BL and McCray, AT, 1993. "The unified medical language system" *Methods of Information in Medicine* **32** 281–291.

Mark, W, Tyler, S, McGuire, J and Schlossberg, J, 1992. "Commitment-based software development" *IEEE Transactions on Software Engineering*. October.

Marr, D, 1982. *Vision: A Computational Investingation into the Human Representation and Processing of Visual Information*. WH Freeman.

McDermott, J, 1988. "Preliminary steps towards a taxonomy of problem-solving methods" In S Marcus, editor, *Automating Knowledge Acquisition for Expert Systems* 225–255. Kluwer.

Miller, GA, 1990. "Wordnet: An on-line lexical database" *International Journal of Lexicography* **3**(4) 235–312.

Morik K, Causse, K and Boswell, R, 1991. "A common knowledge representation integrating learning tools" *Proceedings of MSL.*

Newell, A, 1982. "The knowledge level" *Artificial Intelligence* **18** 87–127.

Russel, S and Norvig, P, 1995. *Artificial Intelligence: A Modern Approach.* Prentice Hall.

Schreiber, AT, Wielinga, BJ, Akkermans, JM, Van de Velde, W and Anjewierden, A, 1994. "CML: The CommonKADS conceptual modelling language" In L Steels, AT Schreiber and W Van de Velde, editors, *A Future for Knowledge Acquisition: Proceedings of the 8th European Knowledge Acquisition Workshop EKAW 94* 1–25. Springer-Verlag. (Volume 867 of *Lecture Notes in Artificial Intelligence.*)

Schreiber, ATh, Wielinga, BJ and Breuker, JA, editors, 1993. *KADS: A Principled Approach to Knowledge-Based System Development* Vol 11 of *Knowledge-Based Systems Book Series.* Academic Press.

Schreiber, G, Wielinga, B and Jansweijer, W, 1995. "The Kactus view on the 'o' word" *Workshop on Basic Ontological Issues in Knowledge Sharing: International Joint Conference on Artificial Intelligence.*

Slater, PE, 1994. "Models of expertise in knowledge engineering" 130–156.

Sloman, A, 1994. "Towards a general theory of representations" *Technical Report CSRP-94-13*, School of Computer Science, University of Birmingham.

Sowa, J, 1984. *Conceptual Structures: Information Processing in Mind and Machine.* Addison-Wesley.

Sowa, J, 1995. "Top-level ontological categories" *International Journal of Human-Computer Studies* **43**(5/6) 669–686.

Steels, L, 1990. "Components of expertise" *AI Magazine* **11**(2). (Also available as *AI Memo 88-16*, AI Lab, Free University of Brussels.)

Steels, L, 1993. "The componential framework and its role in reusability" In JM David, JP Krivine and R Simmons, editor. *Second Generation Expert Systems* 273–298. Springer-Verlag.

Steels, L and McDermott, J, editors, 1994. *The Knowledge Level In Expert Systems: Conversations and Commentary* Vol 10 of *Perspectives in Artificial Intelligence.* Academic Press.

Stefik, M, Aikins, J, Balzer, R, Benoit, J, Birnbaum, L. Hayes-Roth, F and Sacerdoti, E, 1992. "The organization of expert systems: A tutorial" *Artificial Intelligence* **18** 135–173.

Tu, SW, Eriksson, H, Gennari, JH, Shahar, Y and Musen, MA, 1994. "Ontology-based configuration of problem-solving methods and generation of knowledge acquisition tools: The application of Protege-II to protocol-based decision support" *Technical Report KSL-94-22*, Knowledge Systems Laboratory, Medical Computer Science Stanford University.

Uschold, M, King, M, Moralee, S and Zorgios, Y, 1998. "The enterprise ontology" *The Knowledge Engineering Review* **13**(1). (Also available as *AIAI-TR-195* from AIAI, The University of Edinburgh. This ontology was developed as part of the Enterprise Project, see `http://www.aiai.ed.ac.uk/~entprise/enterprise/` for further information.)

Uschold, M and Gruninger, M, 1996. "Ontologies: Principles, methods and applications" *The Knowledge Engineering Review* **11**(2). (Also available as AIAI-TR-191 from AIAI, The University of Edinburgh.

Vinkhuyzen, R, 1992. "On the non-existence of knowledge level models" In B. Neumann, editor, *ECAI 92. 10th European Conference on Artificial Intelligence* 620–622. Wiley.

Wielinga, B, 1993. "Expertise model: Model definition document" *CommonKADS Project Report*, University of Amsterdam, October.

Wielinga, BJ and Schreiber, AT, 1993. "Reusable and sharable knowledge bases: A European perspective" *Proceedings of International Conference on Building and Sharing of Very Large-Scaled Knowledge Bases* 103–115.

Wielinga, R, Schreiber, G, Jansweijer, W, Anjewierden, A and van Hamelen, F, 1994. "Framework and formalism for expressing ontologies" *Technical report*, University of Amsterdam. (Esprit Project 8145 Deliverable DO1b1, available from `http://www.swi.psy.uva.nl/projects/Kactus/Reports.html`.)

Workflow Management Coalition Members, 1994. "Glossary—a workflow management coalition specification" *Technical report*, The Workflow Management Coalition.

**Index**