Encapsulation and Composition of Ontologies*

Jan Jannink, Srinivasan Pichai, Danladi Verheijen, Gio Wiederhold

Stanford University

Stanford CA, 94025, U.S.A.

{jan, vasan, verhe, gio}@db.stanford.edu

Abstract

Ontology concerns itself with the representation of the objects in the universe and the web of their various connections. The traditional task of ontologists has been to extract from this tangle a single ordered structure, in the form of a tree or lattice. This structure consists of the terms that represent the objects, and the relationships that represent connections between objects. Recent work in ontology goes so far as to consider several distinct, superimposed structures, which each represent a classification of the universe according to a particular criterion.

Our purpose is to defer the task of globally classifying terms and relationships. Instead, we focus on composing them for use as we need them. We define contexts to be our unit of encapsulation for ontologies, and use a rule-based algebra to compose novel ontological structures within them. We separate context from concept, the unit of ontological abstraction. Also, we distinguish composition from subsumption, or containment, the relationships which commonly provide structure to ontologies. Adding a formal notion of encapsulation and composition to ontologies leads to more dynamic and maintainable structures, and, we believe, greater computational efficiency for knowledge bases.

Introduction

There is growing interest in reusing and extending existing ontologies, both general purpose and domain specific. Until now, most of the focus has been on developing larger, static ontologies. Also, there are extensive debates about establishing a single top-level ontological taxonomy underpinning all terms in existence. We do not subscribe to the notion that one ontology can serve all objectives, rather we adopt a practical approach to ontology design and development. We aim to reduce the tensions between knowledge base builder and the applications or *knowledge customers* that use the knowledge base. To do so, we provide *domain experts* with a methodology for engineering application specific ontologies which simplify the task of maintaining the knowledge.

We define an *ontology* to be a set of terms and the relationships between them. Ontologies do not have to be explicitly defined, since many knowledge sources implicitly contain structure that constitutes a specification. For example, the terms and definitions in the Webster's dictionary constitute an ontology based on the relationships between the dictionary head words and the words in their definitions. Other examples of ontologies include, but are not limited to, object oriented class hierarchies, database schemas, semi-structured databases, definitional thesauri and knowledge bases.

Motivation

We assume domain specific ontologies are internally consistent. They are reusable if new applications are able to compose the existing ontologies using algebraic operations. The ability to compose ontologies reduces the overall cost of building and maintaining an application specific ontology. A domain expert should easily be able to modify ontologies constructed in this fashion to adapt to changing data or requirements. This is **not**, however, the current state of affairs.

Inferences in large knowledge bases are known to have poor termination characteristics. Typical workarounds include the imposition of external constraints on the duration and depth of inference. Knowledge engineers should have at their disposal a mechanism to encapsulate inferencing so as to

*This work was supported by a grant from the Air Force Office of Scientific Research (AFOSR).

explicitly limit it. Such a mechanism states the information organization, and the inferences performed over the knowledge, therefore also enabling a clearer ontological structure.

Our work begins to address these issues, by defining an application specific and modular framework for ontology composition. This framework enables a knowledge expert to verify the validity of data from multiple sources and prioritize the use of those sources. We use a running example throughout the paper to demonstrate the improved reliability achieved by combining information from multiple sources. Furthermore, the use of problem specific ontological structures rather than a static global structure results in fewer irrelevant inferences, thus optimizing computational efficiency and also improving query reliability.

The next subsection reviews approaches other researchers take to compose ontologies from multiple data sources. Then, we define the various concepts used in our work. The paper continues with a discussion of our use of these concepts and our methodology for composing contexts. Then, we examine a theoretical framework within which to express our technique. Finally, we discuss the role of our rule-based algebra in knowledge bases.

Related Work

A number of ontologies have been developed for usage in computer applications. Some of them such as CYC (Guha 1991) are general purpose while others like Unified Medical Language System (UMLS) (Humphreys & Lindberg 1993) are domain specific. Even amongst the general purpose ontologies there is no standard basis for the nature of the classification (Roy & Hafner 1997). Hence an ontology based on philosophical foundations (Sowa 1998) is quite different from the CYC ontology. Sheth considers the use of contexts to manage semantic heterogeneity in database objects (Kashyap & Sheth 1996). Wiederhold introduces the notion of an algebra over ontologies in (Wiederhold 1994). Hovy's (Chalupsky, Hovy, & Russ 1997) work on ontology alignment indicates that a low percentage of top level concepts are matched using semi-automated tools. Some researchers argue that differences in ontological structures arise due to the fact that they simultaneously model several relationships between the entities, such as **is-a** or **part-of** relationships. They suggest (Guarino 1997) that it would be better to model these relationships as different layers of an ontology and an entity participates one or more of these layers. Efforts are underway to standardize the interoperation of various knowledge representation schemes. For example, there are complete APIs such as OKBC (Chaudhri *et al.* 1998) and there are languages such as KIF (Genesereth & Fikes 1992) and Conceptual Graphs (Wille 1997). In the following section we present the basis for our work.

Working with Contexts

This section defines our terms and provides an overview of the algebra. We chose real problems and datasets, to expose ourselves to the full range of issues that arise when merging heterogeneous data. The example here is drawn from a set of challenge problems (CPs) put forth in the HPKB project (Teknowledge 1997). The CPs are questions –economic, social, political, geographical– pertaining to crisis management in the Middle East. Our running example uses the query below:

- Which OPEC member nations have also been on the UN Security Council?

Although the question may appear to be simple, arriving at the *correct* answer turned out to be a non-trivial task. Inconsistencies between the different sources of data as well as errors and irregularities in the data itself were the most significant problems we faced. The data sources we combined were: the on-line text version of the World Factbook for 1996 (Central Intelligence Agency 1997), the UN policy web site (Global Policy Organization 1997), and the OPEC web site (Org. Petroleum Exporting Countries 1997). We will refer to these data sources as **Factbook**, **UN** and **OPEC.**

Domains

A *domain*, as described in (Wiederhold 1995), is a semantically consistent body of information, maintained by a single organization. The OPEC website is an example of a domain. Since the Factbook aggregates a large number of domains, it is not fully consistent. Domains serve as information sources in our work. We do not expect domains to fully describe their contents, nor must they be error free. A further property of domains is that we rarely have control of their contents. By constructing *contexts* over domains we are able to assert correctness and consistency properties for the data.

Contexts

We define contexts to be our unit of encapsulation for well-structured ontologies. Contexts provide guarantees about the knowledge they export, and contain the inferences feasible over them. The *domain restricted context* encapsulates knowledge pertaining to a single domain. Domain restricted contexts are the primary building blocks which our algebra composes into larger structures. The ontology resulting from the mappings between two source ontologies is assumed to be consistent only within its own context. Such a context is defined as an *articulation context*.

In contrast, McCarthy does not syntactically or structurally distinguish contexts from abstract objects (McCarthy 1993). Contexts are simply mathematical entities used to define situations in which particular assertions are valid. McCarthy proposes the use of *lifting axioms* to state that a proposition or assertion in the context of one knowledge base is valid in another.

The CYC use of *microtheories* bears some resemblance to our definition of contexts. Every microtheory within CYC is a context that makes some simplifying assumptions about the world (Guha 1991). Microtheories in CYC are organized in an inheritance hierarchy whereby everything asserted in the super-microtheory is also true in the microtheory, unless explicitly contradicted. In contrast, we use contexts to encapsulate an application specific portion of an ontology. Also, relationships between ontologies are expressed via explicit mapping rules within a context, and no inheritance hierarchy is implied.

Semantic Mismatch

In knowledge bases, *Frames* or *Concepts* represent a specification of a typed set. This specification is an *intensional* one, that is, its instances, or *extension*, do not account for all the possible permutations of its attribute values. For example, our common sense notion of a nation is quite simple, an independent geopolitical region of the globe. However, in the UN security council membership data, the definition of nation also contains a historical component. Yugoslavia is a nation in the UN data, but not in the Factbook. The specification of a concept in knowledge bases is not a legislating one. There are instances that conform to the specification that belong to some other concept. Continuing our example, Switzerland is a nation, but not a UN member nation, therefore not in the UN data. Finally, the specification is, in general, semantically incomplete. There are implicit constraints on attribute values that would exclude an instance from the set. If we treat the specification as a test of membership, the excluded instances are *false positives*. Likewise, there are *false negatives*, exceptional instances that belong to the set, although they violate the specification. Referring to our example, the Factbook contains an entry for Taiwan, which for political reasons will no longer appear in UN data. Figure 1 below expresses the mismatch in coverage between the concept specification and its extension.

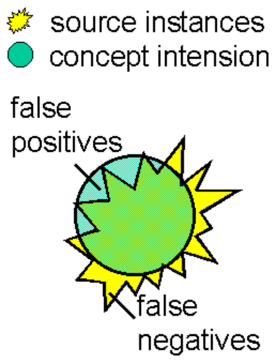

Figure 1: Concept Specification Mismatch

How then can we expect to define an algebra over incomplete specifications and irregular instances? Combining multiple specifications together and merging disparate instances seems fated to produce an increasing divergence between specification and extension. The problem is compounded by inaccuracy and erroneous information. The next subsection provides our framework for using contexts to manage the correspondence.

Interfaces

In order to better maintain a context's suitability for use or reuse, we specify four *interfaces* to the context. The interfaces are queryable by the knowledge engineer and are as follows:

Schema Interface provides templates for the queries that the context guarantees. These templates specify the set of concepts, types and relationships in the context.

Source Interface provides access to the input data sources used to answer the query. This access allows for verification and validation of the knowledge.

Rule Interface returns the rule sets used to transform the data from the sources so they conform to the items in the schema. The rules specify the context's transformations of the sources.

Owner Interface contains a time stamp, as well as the names of the context maintainers. Such information is useful for re-use of the context, because it frames its authority, and its validity.

Figure 2 expresses how the interfaces make a context self-describing, by enabling queries over all of its computation and metadata. This property makes on-line context development and maintenance a feasibility, as suggested in the figure.

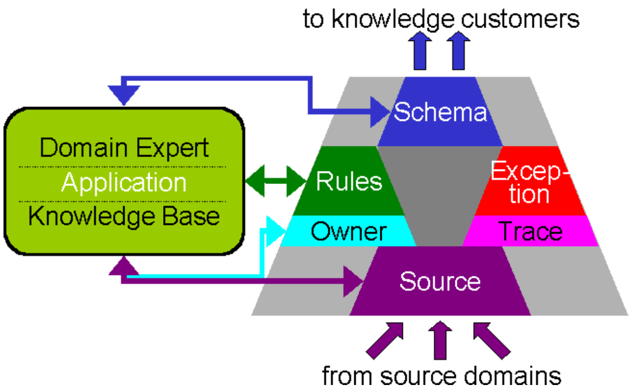

Figure 2: Context Interfaces

The relevance of the interfaces is better understood with an example context – "UN Security Council non-permanent members' years of membership". The interface components for the context are as follows, with the caveat that rules are expressed here as pseudo-code:

```
Schema
   terms
      1
         label  Non_permanent_member
         type   enum(UN_NATION)/string
      2
         label  Start_year
         type   enum(1945-1999)/integer
      ...
   relationships
      label  tenure
      source Non_permanent_member
      object
         1    Start_year
         ...
Source
   input_items
      host      http://www.globalpolicy.org/
      location  security/membship/mem2.html
Rules
   view
      Extract lines containing the pattern '19.*-.*19'
      For each line
         Remove HTML tags ('<[^>]*>') at start/end of line
         Replace all other tags with ', ' (comma and space)
         Split line using ', ' as delimiter
         Output the second segment onwards as term[1]
         Replace '-' with ' ' in the first segment
         Split the first segment using ' ' as delimiter
         Output resulting segments as term[2], term[3]
Owner
   title        UN Security Council [Non_permanent_member]
                        tenure [Start_year] [End_year]
   timestamp    03/10/98
   author       SKC group
```

The ruleset above selects lines with a pattern specifying the membership term in the security council. These lines are parsed to obtain the years of the term, and the member nations in those years. The parsing is done by splitting the line based on delimiters and processing the relevant segments.

Composing Contexts

Algebras offer uniform treatment of their operands, allow their composition, and enable transformations over the resulting expressions. Unfortunately, the operands we wish to manipulate are quite irregular. When there are no constraints on the exceptional instances allowed by a concept, and the concept specification itself is incomplete, it is difficult to imagine an undifferentiated set of operators to compose them.

A Rule-based Algebra

Instead of single operators, we define a two classes of mapping primitives, formed of sequences of simpler operations. Each simple operation is in fact a logical rule, belonging to one of three types. The rules are fired according to structural and lexical properties of the source data, i.e., position and string matching techniques. Note that while the rules effect syntactic changes at the level of bit patterns, the transformations correspond to semantic changes in the concepts and structures of the knowledge sources.

Instance rule: modifies a single item. Rewriting a name to another form, or transforming a type corresponds to such a rule.

Class rule: modifies a class of like organized items. An example is a standardized notation for names of nations wherein all separating spaces and commas are replaced with underscores.

Exception rule: modifies an item or class of items to conform to a specification. Fixing malformed instances and type inconsistencies are actions that fall in this category.

A sequence of rules, executing over a knowledge source, corresponds to an algebraic operator. The first of these operators is denoted an *extraction mapping*, the second is an *articulation mapping*.

Extraction Mappings

We need a mechanism to initially construct contexts from knowledge sources. The class of *extraction maps* provides the primitive for creation of domain restricted contexts. First we perform any necessary restructuring to bring the data into an internal format. We follow this extraction with a *refinement step* that fixes spelling errors, adds underscores to names where necessary, etc.

To illustrate the usage of extraction mappings we return to our example. We observed that the Factbook contained a heading for membership in international organizations, without noticing UN security council membership information. Thus the first context consists of a relation mapping country names to international organizations of which they are members. A second context, summarized in the pseudo-code above, includes historical security council data from the UN data set. This context is a relation mapping country names to their years of membership in the security council.

Articulation Mappings

We combine multiple sources using articulation mappings, such as suggested by Figure 3 to create articulation contexts. Articulation mappings further improve the concordance between specification and extension, because we can correct errors present in single sources. We choose a mapping on the basis of expertise provided by one of the sources. Again, the example illustrates mapping problems we encountered while attempting to utilize heterogeneous data from different sources.

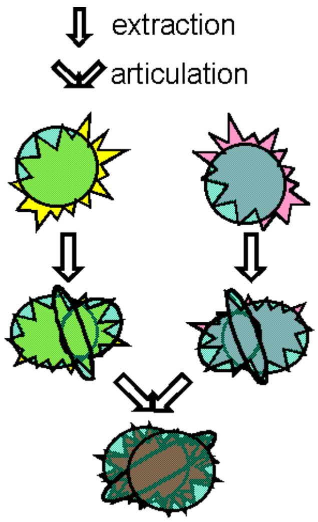

Figure 3: Extractions and Articulation

Our first query context maps country names between the Factbook and UN contexts. We map **Gambia** to **The Gambia**, and chose a null mapping for **Yugoslavia**. Other queries such as "Name the country which contains the site of the 1984 Winter Olympics", require a context that maps Yugoslavia to the nations that resulted from its breakup, in order to correctly answer **Croatia**. These examples show that in general there is no static mapping of multiple concept instances that is universally valid.

While verifying the latest year an OPEC member was in the security council we retrieved the instance data **Gabon 1999**. The Factbook reveals that Gabon is indeed listed as an OPEC member, and the UN data contains security council membership information through 1999, since membership is determined in advance.

We verified OPEC membership at the source, only to discover that Gabon left OPEC in 1994. We extracted an OPEC membership context, a relation that contains member countries with years of membership, and extended the query context to explicitly prioritize the OPEC context over the Factbook's OPEC membership data.

A repeat of the previous query returned the correct answer **Indonesia 1996**. At this time a complete re-examination of the data sets revealed that the Factbook's UN membership attribute contains observer nations as well as security council membership, although two years out of date.

In retrospect, the Factbook purported to contain sufficient information to answer the query, but was not properly maintained. Although its contents are updated annually, its size increases the problem of maintenance. Also, OPEC and the UN have much more at stake in maintaining up to date membership information for their own organization.

Source Prioritization and Correctness

By using rules to explicitly state our preferences with regard to source accuracy, and completeness, we avoid the pitfalls of using a fixed heuristic to determine the choice of source data. The previous example illustrates the importance of source accuracy. The use of domain restricted sources with up to date and accurate information, compensate for our primary sources' deficiencies.

Figure 4 indicates that the conjunctive model which reject all items absent from a source, is over restrictive. Likewise, a disjunctive model that accepts any item which appears in any source keeps too many source instances. When we handled queries about membership in the UN general assembly, we built a table of country names that were used differently in the Factbook and the UN pages. We included about twenty rules specifically enumerated to deal with different naming conventions.

The definitions we have used are convenient, but they do not appear, at first glance, to have a firm grounding. Category theory (Pierce 1991) provides a foundation for discussions about algebras and their properties, but their objects are completely defined abstract entities. The following section explores a theoretical basis for our framework.

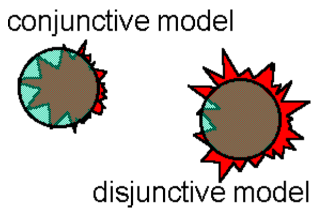

Figure 4: Lexically Driven Mappings

Theoretical Framework

We differentiate our problem space from that of specification morphisms (Smith 1993). Ontologies suffer from implicit semantics, incompleteness of source specifications and irregularity of their extensions. Morphisms allow translations from one specification to another, when there is no semantic mismatch. Therefore they are applicable when intension and extension are not distinguishable, such as in mathematical structures. In the absence of a rigorous and fully abstract notion of context algebra, we are nonetheless able to provide a framework with conditional guarantees of correctness.

Informal Categories

We introduce the notion of an *informal category* to represent the union of a number of concept specifications and the instances that represent their extensions. We define *mapping primitives* to be a combination of rule sets that augment a specification, along with mappings of the instances to values that conform to the new specification.

Informal categories correspond to our definition of contexts from the previous sections. We continue by describing the mapping primitives that we use in our algebra. We will match each of our algebraic operations to an operation in category theory using the above definitions.

Translation Operator

In the category theoretical representation, the extraction mapping, or *translation*, corresponds to the definition of a member of the informal category. The refinement step consists of an informal map that is similar to the identity mapping. An example of translation is a retrieval of OPEC member nations and their years of membership from the organization's web site. Translation is related to McCarthy's lifting axioms (McCarthy 1993).

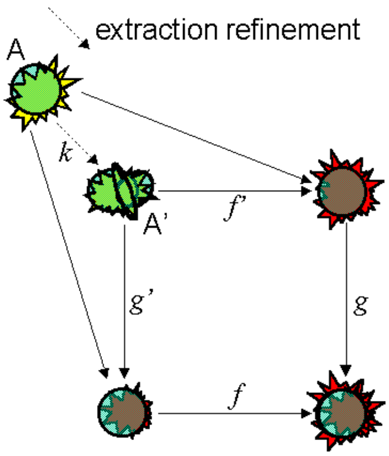

Figure 5: Translation Mappings

In category theory, translation refinement corresponds to the definition of *pullback*. The essential difference here is that we do not have a fully specified category to map from. The pullback starting from an informal category appears as in Figure 5. The arrow k represents levels of refinement for the extraction process. A' is the limit object having an arrow f' that commutes in the diagram.

Combination Operator

Here we present the category theoretical *product* diagram as it applies to the definition of intersection of ontologies. The *intersection*, shown in Figure 6, $A \times B$, projects along π_1, π_2, to both A and B. As above, we start from incompletely specified sources A and B. The interpretation of *intersection* from the diagram means: for every context that maps into all of the source contexts, there exists a unique mapping to the intersection of the sources.

The intersection is therefore identified as being the context whose unique mapping is the identity mapping. In our informal model we guarantee this property for the specified extension of the source contexts' definitions. This distinction separates our definition from the ordinary product diagram. In fact, we see that the intersection varies with the definition of the source interface of the context. The section on articulation mappings describes an *intersection* operator.

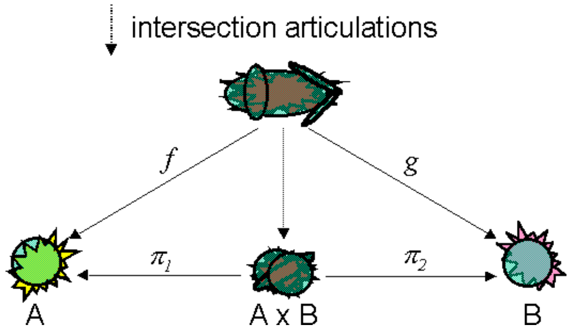

Figure 6: Intersection Operation

Knowledge Base Role

In the previous sections, we have defined context as the unit of encapsulation for ontologies, and an algebra to serve as constructors and composition operators for contexts. In this section we examine the role of contexts within knowledge base systems with an eye towards their practical benefits.

Efficiency and Performance

We perform single pass preprocessing of sources on demand, to maximize efficiency. This restriction limits the number of passes through the source data to the degree of nesting of contexts from a source to any particular query context.

We are using an OKBC interface to assert the results of our computations to a knowledge base. We limit extraction only to the concepts which are most relevant to the context. Thus, we restrict the amount of inference we allow to achieve an answer. By fully specifying the concepts required from each source, we are able to decompose queries, and issue only the parts of the query relevant to the source.

Encapsulation allows knowledge bases with differing knowledge representation to interoperate. In particular, a knowledge base's representation is optimized for its own inference engine. Porting a different representation to another inference engine will not in general result in equivalent inferencing performance. Encapsulating multiple knowledge base inferences and combining them through the algebra ensures that inferences occur where they will be most efficiently performed.

Ontology Structure and Maintenance

A crucial aspect of ontology maintenance is its clarity of structure. Knowledge engineers must comprehend the shape of the ontology and its interconnections in order to accurately gauge how changes will affect the knowledge base's performance. At a recent workshop, a presentation (Chalupsky, Hovy, & Russ 1997) described preliminary efforts to align and merge two ontologies containing over 3000 concepts apiece. From an observers perspective it became evident that the original structure of the ontology was fragile enough that no substantial transformation or enhancement was feasible.

Adding encapsulation and composition to knowledge bases provides a dual benefit to ontology maintenance as well. Ontologies may be decomposed into application specific units that are reused only as necessary. Maintenance taking place within context boundaries will not affect performance of external components, so long as the context interfaces are maintained. A dual structure emerges from this re-engineering: first, a traditional ontology based on frames and related by subsumption or containment, second, an orthogonal structure based on contexts related by application driven composition. Furthermore, since composition is an inherently dynamic operation, the context structure is open-ended and evolves as requirements change. The resulting ontology has a cleaner structure, since relationships defined by concepts' functional roles are no longer shoehorned into the ontology alongside inheritance and instance relationships.

Conclusion

We have introduced a new formalism for context in knowledge bases, that enables localized and controlled inferences, resulting in more efficient processing of queries. We use a constructive mechanism in the form of a rule-based algebra, which creates a novel open-ended taxonomy of dynamic relationships between contexts. Our method declaratively reduces some of the ontological semantics to a syntactic structure. Together, encapsulation and composition enable problem specific optimizations, improving accuracy, simplifying maintenance, and creating new concepts from existing data.

Acknowledgments

We would like to thank Neetha Ratakonda for her coding contributions and proofreading. Thanks also to Rudi Studer and Erich Neuhold for their helpful comments and feedback.

References

Central Intelligence Agency. 1997. CIA site. http://www.odci.gov/cia. CIA Factbook 1996.

Chalupsky, H.; Hovy, E.; and Russ, T. 1997. NCITS.TC.T2 ANSI ad hoc group on ontology. Talk on Ontology Alignment.

Chaudhri, V. K.; Farquhar, A.; Fikes, R.; Karp, P. D.; and Rice, J. P. 1998. Open knowledge base connectivity 2.0.2. Draft standard proposal.

Genesereth, M., and Fikes, R. 1992. *Knowledge Interchange Format,*. Reference Manual.

Global Policy Organization. 1997. Global Policy site. http://www.globalpolicy.org. UN information and statistics.

Guarino, N. 1997. NCITS.TC.T2 ANSI ad hoc group on ontology. Talk on Formal Ontology.

Guha, R. V. 1991. *Contexts: A Formalization and Some Applications.* Ph.D. Dissertation, Stanford University.

Humphreys, B., and Lindberg, D. 1993. The UMLS project : Making the conceptual connection between users and the information they need. *Bulletin of the Medical Library Association.*

Kashyap, V., and Sheth, A. 1996. Semantic and schematic similarities between database objects: a context-based approach. *VLDB Journal* 5(4):276–304.

McCarthy, J. 1993. Notes on formalizing context. *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence.*

Org. Petroleum Exporting Countries. 1997. OPEC site. http://www.opec.org. OPEC membership data.

Pierce, B. C. 1991. *Basic Category Theory for Computer Scientists.* The MIT Press.

Roy, N. F., and Hafner, C. D. 1997. The state of the art in ontology design. *AI Magazine* 18(3):53–74.

Smith, D. R. 1993. Constructing specification morphisms. *Journal of Symbolic Computation* 15:571–606.

Sowa, J. F. 1998. *Knowledge Representation Logical, Philosophical and Computational Foundations.* Boston, MA: PWS Publishing Company.

Teknowledge. 1997. High Performance Knowledge Base site. http://www.teknowledge.com/HPKB. DARPA sponsored project.

Wiederhold, G. 1994. An algebra for ontology composition. In *Proceedings of 1994 Monterey Workshop on Formal Methods*, 56–61. U.S. Naval Postgraduate School.

Wiederhold, G. 1995. Objects and domains for managing medical knowledge. In *Methods of Information in Medecine.* Schattauer Verlag. 1–7.

Wille, R. 1997. Conceptual graphs and formal concept analysis. Technical Report 1903, Technische Hochschule Darmstadt.