Communications of the Association for Information Systems

Volume 14 | Article 9

8-14-2004

Computational Ontologies and Information Systems II: Formal Specification

Raj Sharman
State University of New York at Buffalo, rshrman@buffalo.edu

Rajiv Kishore
State University of New York at Buffalo, rkishore@buffalo.edu

Ram Ramesh
State University of New York at Buffalo

Communications of the Association for Information Systems

COMPUTATIONAL ONTOLOGIES AND INFORMATION SYSTEMS: II. FORMAL SPECIFICATION

Raj Sharman
Rajiv Kishore
Ram Ramesh
School of Management
State University of New York at Buffalo
rsharman@buffalo.edu

ABSTRACT

This paper extends the study of ontologies in Part I of this study (Volume 14, Article 8 [Kishore et al, 2004]) in the context of Information Systems. The basic foundations of computational ontologies presented in Part I are extended to formal specifications in this paper. This paper provides a review of the formalisms, languages, and tools for specifying and implementing computational ontologies. Directions for future research are also provided.

Keywords: formal ontologies, ontology, computational ontologies, ontology development tools, ontology-driven information systems, ontology representation formalisms, ontology specification languages, ontological engineering, ontology mining, ontology metrics.

I. INTRODUCTION

The focus of this paper is to provide a comprehensive review of the formalisms, languages, and tools used for specifying and implementing computational ontologies. Several examples and applications of specific ontologies are also provided. This paper is a sequel to Kishore et al. [2004]. The Kishore et al. paper should be read before this one.

For conceptualizations to be useful they must be communicated unambiguously. Often communication using natural languages leads to prose that is either verbose or ambiguous. Therefore representation languages have been developed that allow for communication succinctly and precisely. This paper is devoted to presenting a review of the state of the art about representation of ontologies for use in information systems.

This paper is organized into five sections as shown in Figure 1. Section 1 is the introduction. Section II deals with the issues that are relevant to choosing a specification formalism and an implementation language. Section III presents an in-depth analysis of specification formalisms and implementation languages for ontologies. Section IV includes a discussion on research directions and emerging issues in computational ontologies. Section V provides a conclusion to this paper.

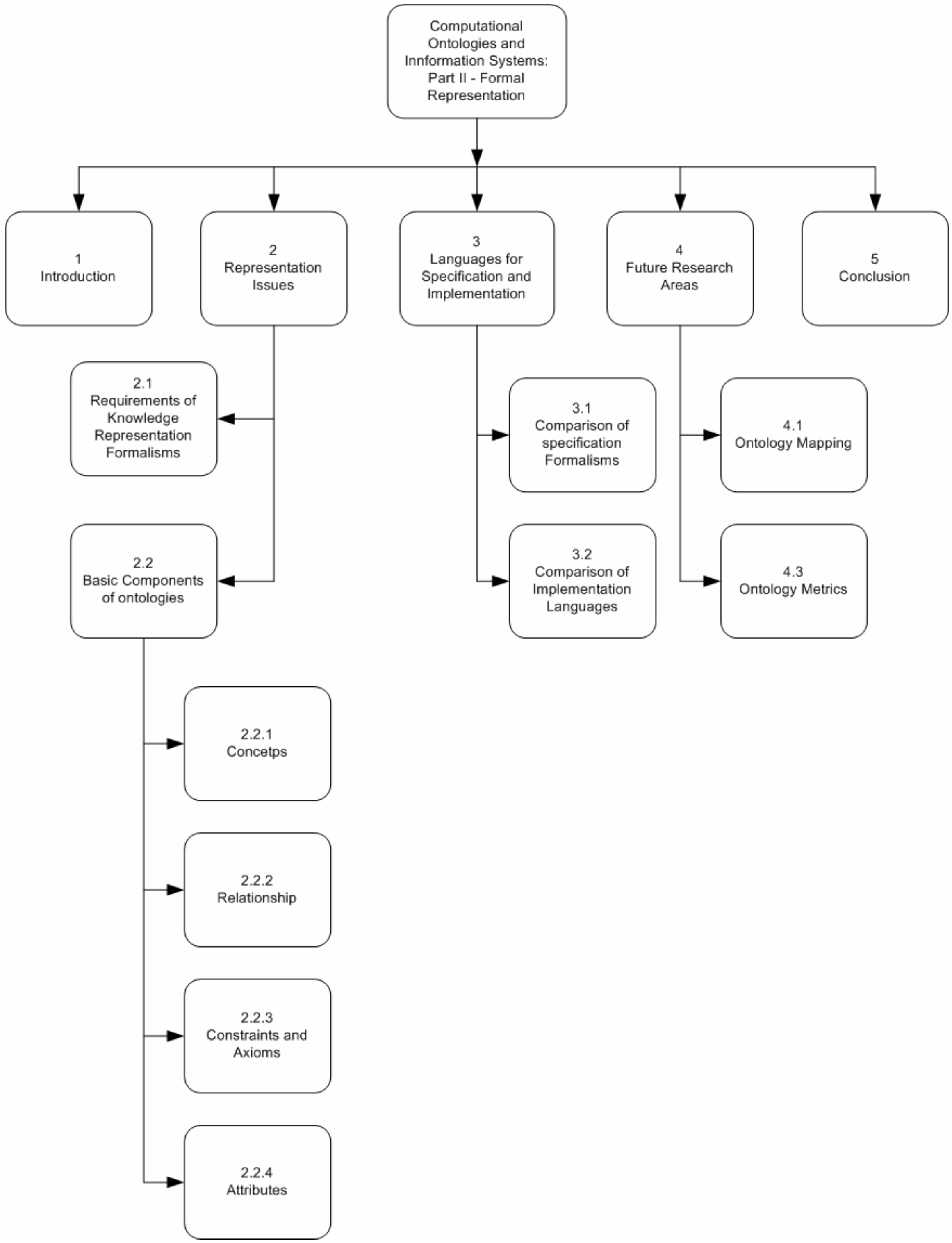

Figure 1: Organization of this Paper

II. REPRESENTATION ISSUES

An ontology, as indicated in Kishore et al. [2004], serves many purposes; for example it is a vehicle for a shared understanding of concepts and relationships, integration of heterogeneous information systems, intelligent information retrieval, and knowledge based machine translation. This section addresses the issues surrounding the choice of arepresentation formalism and an implementation language in developing ontologies.

The use of the ontology generally dictates the level of rigor with which the ontology is specified. Several languages have been used to represent ontologies. The limitation of the representation language imposes constraints on how and what can and cannot be specified. Therefore, we compare some of the popular formalisms and languages based on type of content and reasoning capabilities usually needed in ontologies. The framework that we used for this comparison was developed based on several research papers in the areas of ontology, logics, programming languages and artificial intelligence [Corcho and Gomez 2000a and 2000b], [Reichgelt 1991], [Russel and Norwig, 2003], and [Sowa, 1999].

This section is divided into two subsections as shown in Figure 1. The first subsection details issues, expectations, and requirements of knowledge representation languages. The second subsection provides an insight into the possible components of ontologies such as components and constraints.

REQUIREMENTS OF KNOWLEDGE REPRESENTATION FORMALISMS

This subsection presents issues to consider while choosing a representation formalism. Several criteria can be used to assess the value of a formalism. However, the most important criterion is adequacy of the language at the implementation, logical, epistemological and conceptual levels [Reichgelt, 1991]. This criterion includes qualifiers such as expressiveness and naturalness. At the implementation level, a language should provide efficient storage, quick inferencing capabilities, and consistent encoding of the ontology constructs. The language should allow for representations to be modular so that changes and evolutions in the domain can be managed by minimal changes to the ontology.

At the logical level, a representation language should allow for precise specification and interpretation of well-formed expressions (as in model theory). More specifically, this idea deals with the expressive power in terms of flexibility, explicitness, accuracy, and formality. These criteria imply

- at the meaning of complex expressions should be derivable from simpler expressions and
- that sound[1] inference procedures can be created.

Soundness ensures that statements do not contradict each other. Furthermore, it is important to recognize the trade-offs between expressive power and complexity.

At the epistemological level, the representation language should allow for representations to be constructed or organized in ways that are most natural to the domain. The language should provide flexibility in terms of the granularity of information at the epistemological level and support

[1] Property of logic system that every sentence derived from a set of valid sentences is also a valid consequence of that set of sentences. A deductive argument is sound if and only if it is both valid, and all of its premises are *actually true*. Otherwise, a deductive argument is *unsound.* Further, a deductive argument is said to be *valid* if and only if it takes a form that makes it impossible for the premises to be true and the conclusion nevertheless to be false. Otherwise, a deductive argument is said to be *invalid* [Fieser and Dowden, 2004].

the primitives at the conceptual level. The granularity dictates the chunks of knowledge that form the building blocks for organizing the knowledge.

At the conceptual level, the language or chosen representation should provide the modeler the ability to represent real world concepts, relationships, constraints and axioms in a concise and precise manner (i.e., expressiveness).

BASIC COMPONENTS OF ONTOLOGIES

To understand the usefulness and the limitation of languages it is fruitful to know the artifacts that need to be formally specified or implemented. This subsection provides a detailed discussion on the constructs that are part of most ontologies.

We restrict our discussion to formalisms that are useful for working with computer-based information systems. Regardless of whether the ontology is a top-level or a domain level ontology, ontologies in general include primitive constructs: concepts, relationships, and constraints as shown in Figure 2.

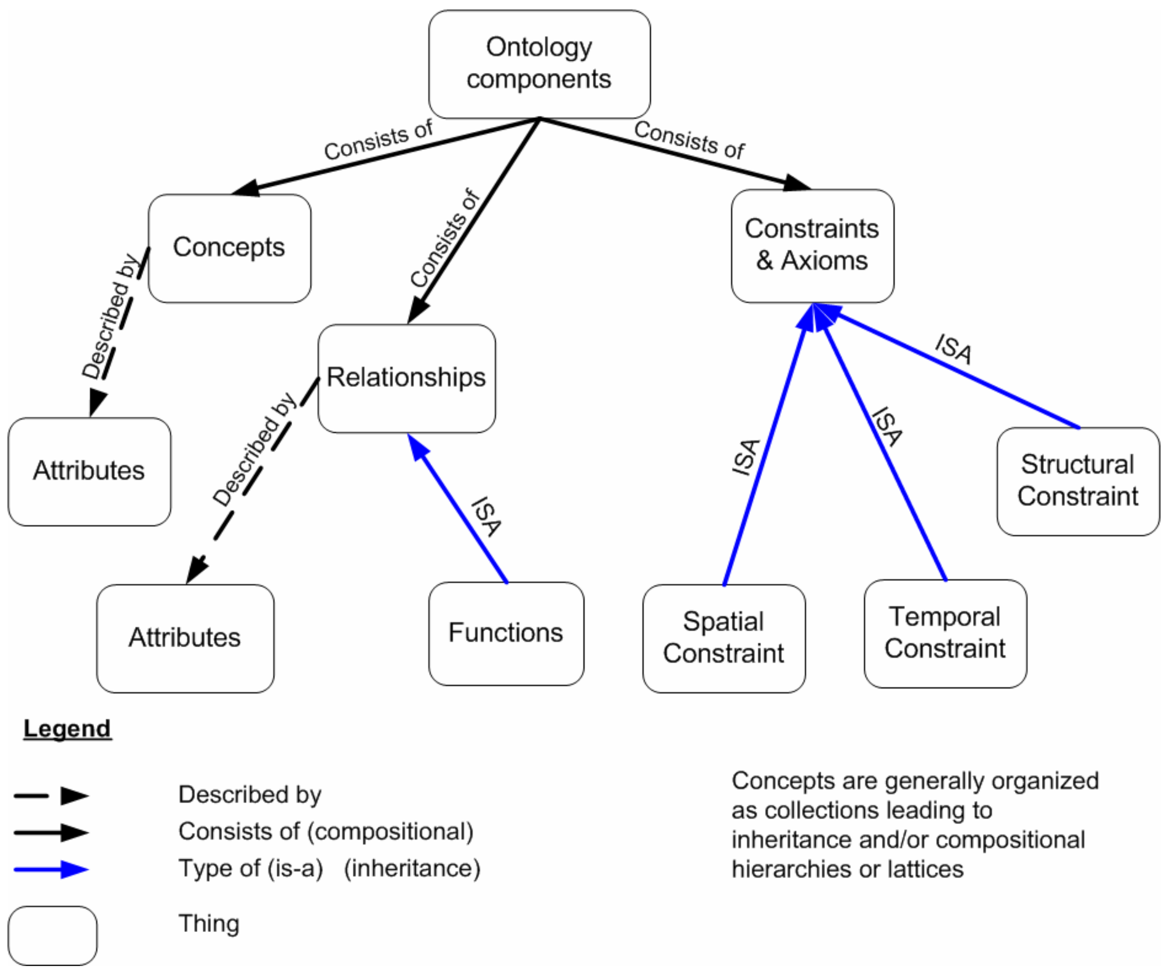

Figure 2: Components of an Ontology

Concepts

Merriam-Webster Online Dictionary defines a concept as an abstract or generic idea derived or inferred from particular instances. It is something conceived in the mind. From a computational ontology and an information systems perspective we are interested in only those abstract or generic ideas that are relevant and needs to be kept track of. Concepts can be categorized as either elementary or composite. Composite concepts are often viewed as consisting of elementary concepts grouped according to some logic. Concepts are often associated with attributes that need to be represented. Further, certain concepts exhibit polymorphic[2] behavior or show temporal properties. A ontology representation language must have the necessary constructs to support the representation of the needed features.

Concepts that are part of an ontology are usually organized into categories because much of the reasoning takes place at the level of categories [Russell and Norvig, 2003]. Categories also serve to make predictions about objects once they are classified. Categories serve to organize and simplify the knowledge in the ontology though inheritance (is-a relationship). Subclass relations organize categories into a taxonomy or a taxonomic hierarchy.

Concepts in an ontology may also be aggregated as ordered or unordered collections, often without an inheritance relationship (possibly creating compositional hierarchies[3]). Further partitioning of a concept into sub-concepts (has-a relationship) that are exhaustive or non-exhaustive and either non-overlapping or overlapping is an important aspect of most ontologies. Formalisms must provide features to support these features.

Relationships

Concepts are usually related to other components of the ontology through, for example, relationships and functions. Further, relationships may be unary, binary, ternary or of higher order. Representation languages must also provide facilities to represent attributes of relationships in the ontology. Relationships may be asserted or inferred. A *function,* shown in Figure 2, is a special type of relation which relates some number of terms to exactly one other term [Russell and Norvig, 2003]. We define a *term* as any object that is defined (e.g., concepts, instances, relationships, functions). An axiom is not considered a term. In a strict sense, a relationship can be viewed as a constraint.

Constraints and Axioms

Constraints provide a bound or restrictions on both static and dynamic systems, and objects. Constraints are a useful way of representing knowledge and inferencing. It is common to include *structural* (cardinality, integrity), *spatial,* and *temporal* constraints in most conceptualizations. Constraints can also be classified as hard (must be satisfied) or soft (should be satisfied). Constraints may represent concrete or inferred knowledge.

Every axiom is a constraint. An *axiom* is a sentence that is assumed to be true without proof. Axioms provide basic factual truth from which useful conclusions can be derived. Not all logical sentences are axioms. Further, not all axioms are definitions. Note that tautologies are not regarded as constraints by many schools of thought because they are trivially true.

Axioms are included in ontologies such reasons as verifying correctness, and deducing new facts. Often constraints and axioms are expressed using first and second order logic.

---

[2] Polymorphic means to have many forms
[3] Compositional hierarchies are nested hierarchies in which each level is composed by units (concepts) from the level below [Smith, 2001].

It may be necessary to express conditionality in an ontology. That is, it must be possible to express statements of the form 'if condition ….then action'. IF-THEN rules are also called *production rules* [Reichgelt, 1991].

Attributes

Attributes represent semantic information associated with terms. An attribute, is sometimes referred to as a variable (in functions) or slot (in a frames context) or field (in a relational database context) [Parigot, 1999]. The information stored in the attributes can be of any type (such as symbol tables, pieces of generated code, types of expressions, values of constant expressions, or Boolean flags.) suitable for the purpose at hand. Further, the scope of an attribute may be global, concept, local, or instance level. These attributes may be part of the conceptualization and therefore may need to be represented in the ontology by the language.

Concepts and relationships contain ontological instances and may be included as part of an ontology. An assertion is any statement that is true in the ontology. Certain assertions are made based on instances and these assertions may lead to claims [Luke, 2000].

Besides their mechanisms of basic knowledge representation, most ontologies support some form of formal semantics and reasoning, Several languages have been developed for this purpose and all of them support some or all of the above constructs to different degrees. At a minimum, a knowledge representation mechanism should provide both syntax and logic support. While the syntax is concerned with how knowledge is stored, the logical component deals with its inferential capabilities [Reichgelt, 1991]. We address these requirements at the implementation, logical, epistemological, and conceptual levels below.

At the implementation level, we are primarily concerned with the tractability of the representation mechanism. These mechanisms relate to the ability of the representation language to aid the creation of computer information systems. Some examples of concerns at this level relate to how well the language supports inferencing, indexing, a large set of concepts, and relationships in an ontology. At the logical level, the expressive power of the language is the primary concern. This idea refers to the ability to represent logical properties unambiguously and with clarity from both syntactic and inferential points of view. Some examples of these concerns are:

- can we represent equivalence between concepts or instances?
- does an 'is-a' relationship between two instances x and y imply that every x is a y or that some x's are y's?

At the epistemological level, the main concern is with the types of primitive expressions and the types of inference strategies used. For example, these concerns translate to the following questions in a medical ontology:

- does the formalism support an inferencing strategy to help an expert physician to diagnose a physical ailment? and
- does the formalism also support a strategy for non-physicians to learn more about the ailment?

However, we do not make any decision about which actual primitives and inference strategies are used to represent knowledge about some domain at this stage.

At the conceptual level, the actual primitives that are part of the knowledge representation formalism are of concern. Examples of such concerns are:

- is an 'is-a' arc to support inheritance, or
- is there a 'part-of' arc to represent composition in the formalism.

In the next section we discuss the tools available to the analyst to represent the constructs that are usually part of most ontologies.

III.LANGUAGES FOR ONTOLOGY SPECIFICATION AND IMPLEMENTATION

In this section we provide a comprehensive and comparative analysis of specification formalisms and implementation languages for ontologies.

Ontologies are not all built the same way. A number of possible languages can be used to specify languages. Further, some languages allow for easier implementation of ontologies. The specification using general logic formalisms allow for expression of the conceptualization. However, different formalisms pose certain limitations. Languages have now been developed that specifically support ontology construction. Many of these languages use one or more logic formalisms as a basis. This section is divided into two subsections: (1) comparative analysis of formalisms and (2) languages for ontology representation

COMPARATIVE ANALYSIS OF FORMALISMS

This subsection is devoted to providing a comprehensive analysis of formalism for the specification of ontologies. In order to bring sharper focus to our discussion we used the common classification of ontologies as being: (a) informal, (b) semi-formal and (c) formal representations as shown in figure 3.

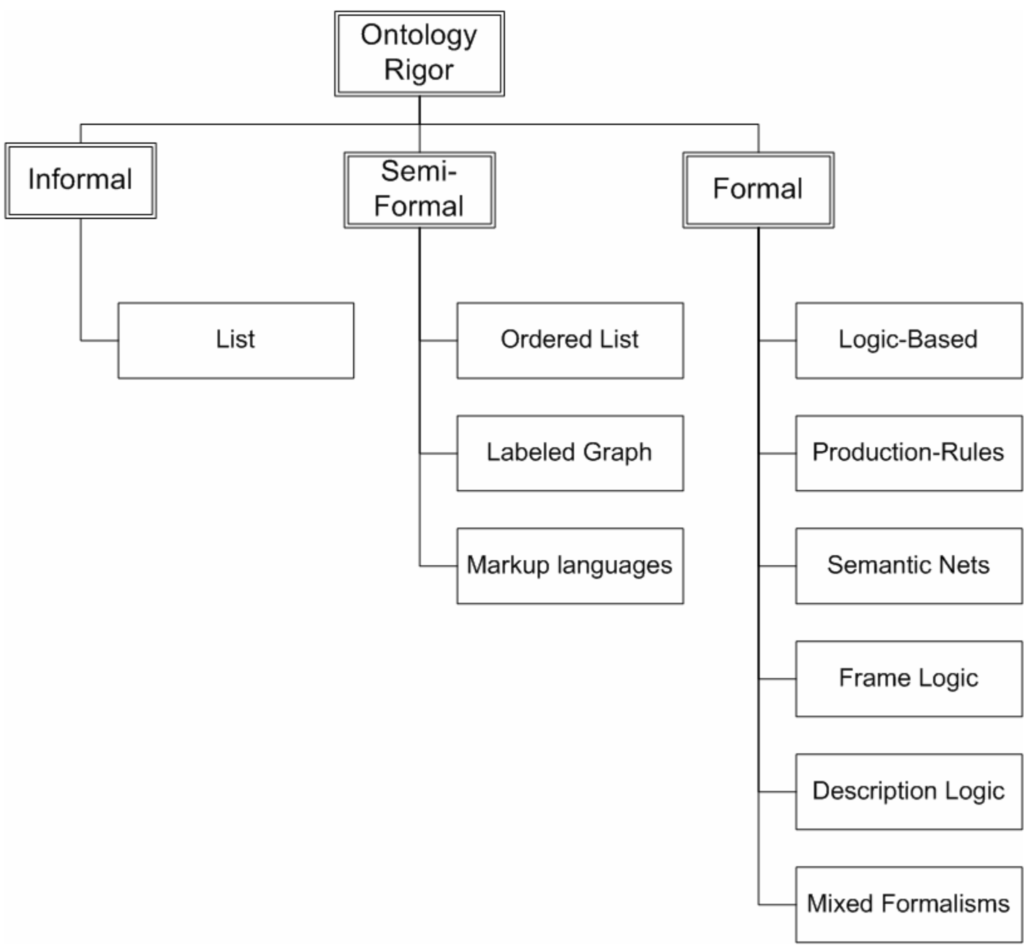

Figure 3: Ontology Categories: Informal, Semi-Formal and Formal

An informal ontology is one where the types are either not defined or defined in some natural language [Sowa, 1991]. This type of ontology contains neither rules nor structures. Semi-formal representations express content in a restricted and structured form of natural language or an artificial formally defined language [Sowa, 1991]. A formal ontology is one where the conceptualization is specified very rigorously using a specification or programming language. Please note that our notion of ontologies includes object-oriented class hierarchies, database schemas, semi-structured databases, definitional thesauri, and knowledge bases. Table 1 provides a comparative insight into the level of formality and formalisms normally used along with other useful details.

Formal ontology specifications are accomplished using one of these languages as shown in Figure 3

Logic-based languages (First order predicate logic, second-order predicate logic [Kelly, 1997], [Rogers, 1990]): Logic based languages provide a formal way to represent knowledge. A logic-based formalism consists of a set of primitive expressions (constant symbols, function symbols, predicate symbols, variables and connectives, quantifiers – universal and existential) and syntax or set of formation rules to create complex expressions [Russell and Norvig, 2003].

Production rules are a knowledge representation language with a pattern-directed inference system [Waterman and Hayes-Roth, 1979]. Production rules provide a natural representation for the kind of heuristic knowledge commonly used in many expert systems. Pattern-directed inference system is a system that consist of three main components, i.e., working memory, rule base and interpreter. The working memory contains the information that the system has gained about the problem thus far. The rule base contains information that applies to all the problems that the system may be asked to solve. The interpreter solves the control problem, i.e., decide which rule to execute on each selection-execute cycle.

Semantic Nets (SNePS, Conceptual graphs, KL-One): Semantic Nets are formalisms based on the notions of associations among concepts and their related properties as the basic artifacts of knowledge [Reichgelt, 1991; Sowa, 1993].

Frame-based languages: In frame-based languages, knowledge is stored in larger chunks as a set of conceptual entities with associated descriptions. The chunks are structures that represent knowledge and are referred to as frames. The descriptions in a frame are called slots. There are usually many connections between the various chunks of knowledge [Reichgelt, 1991], [Minsky, 1975]. Frames are common in intentional knowledge representations.

Description Logics ([Borgida, 1995], [Borgida, 1996], [Baader et al., 2003], [Donini, 1996]) provide a language for capturing declarative knowledge about a domain and a classifier that allows reasoning about that knowledge. Information captured using description logics is classified in a hierarchical lattice of concepts (comparable to classes, or frames), their inter-relationships or roles (comparable to slots in frame systems) and individual objects (instances).

Mixed Formalisms: Most formalism involve advantages and disadvantages. For example default reasoning is a problem with logic-based languages while semantic-nets and frame-based representations provide a natural way to deal with this type of reasoning. On the other hand, semantic-nets and frames encounter problems defining new concepts and expressing arbitrary disjunctions. To overcome such problems, several hybrid representations such as KL-TWO, KRYPTON were developed.

Table 1 Representation Formality Continuum
(Informal, Semi-Formal and Formal)

Rigor of Ontology	**Advantages**	**Disadvantages**	**Formalism**	**Ontology**
Informal	Quick	No Structure Maintenance difficult Interpretation problems	List	None well known. However many glossaries fit into this category.
Semi-Formal	Quick Better clarity than informal formalism Good for intermediate representations	No common formal semantics No model or proof theory possible	Lists, Labeled graphs, Markup languages	Chemicals [Lopez et al. 1999]. Early version of GRITIKA [Zhang et al., 2003; Zhang et al., 2004] Enterprise Ontology (Ontolingua version) [Ushold 1998]
Formal	Meticulously defined. Least ambiguous Constraints well defined. Most suitable for automatic integration of information systems. Reasoning systems possible.	No naturalness Requires the reader to understand the specification language.	FOPC, First order predicate logic, Second order predicate logic, Semantic Nets Frame Logic Description Logic	TOVE [Fox 1996] EngMath [Gruber and Olsen 1984], EcoCyc [Karp 2000] GRITIKA [Zhang et al., 2004] GALEN [Rector 2002]

A brief comparative assessment of these specification languages is provided in Table 2.

Table 2. A Comparison of Representation Languages for Formal Ontologies

Formalism	**Advantages**	**Disadvantages**	**Implementation Examples**	**Ontology Based on Formalism**
Logic-based	High expressive power Allows for creation of arbitrary attributes and constraints. No overt ontological content	No naturalness with expert knowledge Semi-decidable	FOPC, First order predicate logic, Second order predicate logic, KIF, OML, etc.	TOVE [Fox, 1996]
Production Rules	Naturalness with expert knowledge Modularity Restricted syntax Problem-solving process	Limitations in expressive power. Difficult to express structure	OPS	
Semantic Nets	Conceptually simple representation	No semantics to support interpretation No axioms to support reasoning	SNePS [Shapiro, 1979], Conceptual Graphs [Sowa, 1984], KL-One [Brachman, 1985]	
Frame-based	Naturalness with the way domain experts think, Hierarchical structure, Supports default reasoning	Absence of clear semantics (Implementations have provides some mechanisms to overcome this disadvantage)	KRL [Bobrow, 1970], CLIPS, XOL [Chaudhri, 1998] Ontolingua [Gruber, 1993]	EngMath [Gruber and Olsen, 1984], EcoCyc [Karp, 2000]
Description Logics	Well understood theoretical principles, Logic can be precisely expressed, Automatic derivation of classification taxonomies	One has to build sanctions or restrictions as needed. Formalism does not provide it.	GRAIL [Rector, et al 1997], Classic [Borgida, 1989], LOOM [MacGregor, 1991]	GALEN [Rector, 2002]

Mixed-Formalisms	Removes many problems of other formalisms	Depends on hybrid	F-Logic [Kifer, 1995], OIL [Fensel et al., 2000]	TAMBIS (uses: OIL, GRAIL) [Stevens et al., 1998]

Given the scope of this paper we did not discuss other formalisms that include deontic logics and modal logics.

LANGUAGES FOR ONTOLOGY REPRESENTATION

This subsection provides a comparative analysis of the languages that are most widely used to implement ontologies. The framework used for this comparison is based on whether the language supports the constructs often specified in an ontology. All the languages we discuss in this section are based on the specification formalism discussed in the previous section. We use the term language here to refer to those formalisms that can be used directly to create computer implementations. Table 3 provides an introduction to the languages that we compare in Table 4. The information in Table 3 and Table 5 was synthesized by reviewing many different sources. However, the information in Table 4 has been adapted from Corcho and Gomez-Perez [2000a] and Corcho and Gomez-Perez [2000b]. The symbol ☺ used in the tables implies that the attribute concerned is not determinable based on the published information.

Table 3. Brief Introduction to Commonly Used Ontology Specification and Implementation Languages

Language	Description	Reference
Ontolingua	Based on Frame Ontology and KIF. Besides providing for the constructs to represent objects, functions and relations, it includes declarative semantics that allows for representation of constraints. Further, it can represent meta-knowledge and non-monotonic reasoning rules.	[Gruber, 1993]
GRAIL	GRAIL stands for GALEN Representation and Integration Language. It is based on description Logics. GRAIL was developed to represent medical terminology. It is now also used for a range of other purposes from indexing DNA and Protein sequences for Bioinformatics to helping sort out the terminology used in art history.	[Rector et al., 1997]
XOL	XOL stands for XML-based Ontology Exchange Language. Although XOL was designed for use in the bioinformatics domain, it can be used to develop ontologies in any domain. It provides excellent features to support exchange of ontology definitions over the world wide web. It allows the user to define the XML syntax that is a subset of OKBC (Open Knowledge Base Connectivity)	[Chaudhri, 1998]
SHOE	SHOE stands for Simple HTML Ontology extension. It was developed to extend HTML so that machine-readable semantic knowledge could be encoded with documents. It allows the user to define classes, class hierarchies, relations and inference rules.	[Luke, 2000]
OML	OML stands for Ontology Markup Language. OML is explicitly oriented towards the representation of abstract semantics. All features of XML Schema are also in OML. Further OML allows the user to represent several types of constraints such as	[Kent, 2002]

	general assertions and sequences from information flow logic.	
RDF	RDF stands for Resource Definition Schema. It is a declarative language that provides explicit mechanisms to represent the relationships between attributes and resources, classes, hierarchies and constraints. It was developed by W3C for describing web resources. RDF integrates a variety of applications from library catalogs and world-wide directories to syndication and aggregation of news, software, and content to personal collections of music, photos, and events using XML as an interchange syntax.	[Miller, 1998]
LOOM	LOOM is a high level programming language and environment for knowledge representation and for constructing intelligent applications. It is based on first-order logic which belongs to the KL-ONE family. The knowledge representation system in Loom is used to provide deductive support to enable reasoning.	[MacGregor, 1991]
OIL	OIL stands for Ontology Interchange Layer. OIL provides a layered approach to specifying ontologies. The layers are: the ontology container level (contains information about the features of the ontology), ontology definition layer (contains ontology definitions) and the object layer (contains instances). Concepts, relations, functions and axioms can be easily defined. It combines modeling primitives from frame-based languages with the formal semantics and reasoning services provided by description logics	[Fensel et al., 2000]
DAML + OIL	DAML stands for DARPA Agent Markup Language. It is a semantic markup language that extends RDF and RDF Schema. It is written in RDF which in turn is written in Extensible markup Language (XML).	[Horrocks, 2001]

Table 4. Comparison of Representation Languages based on Constructs Needed in an Ontology

CRITERIA	**Onto-lingua**	**GRAIL**	**XOL**	**SHOE**	**OML**	**RDF(S)**	**LOOM**	**OIL**	**OIL+ DAML**
Concepts									
Subclass of	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Not subclass of	Yes	Yes	No	No	No	No	Yes	Yes	Yes
Exhaustive decompositions	Yes	Yes	No	No	No	No	Yes	Yes	Yes
Disjoint decomposition	Yes	Yes	No	No	No	No	Yes	Yes	Yes
Instance of Concepts	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Relations									
IS-A (Inheritance hierarchy)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
HAS-A (Compositional hierarchy)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Functions	Yes	Yes	No	Yes	Yes	No	Yes	Yes	Yes
Constraints									
Cardinality Constraints	Yes	Yes	Yes	No	No	No	Yes	Yes	Yes
Integrity Constraints	No	Yes	No	No	Yes	No	No	No	No
Axioms									
Facts	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
First-order logic	Yes	Yes	No	☺	Yes	No	☺	☺	☺
Embedded logic	No	No	No	☺	No	No	No	No	No
Claims	Yes	Yes	No	Yes	No	☺	☺	☺	☺
Attributes									
Instance Attributes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Class Attributes	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes
Local Scope	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Global Scope	Yes		Yes	No	Yes	Yes	Yes	Yes	Yes
Default Slot Value	Yes	Yes	Yes	No	No	No	No	No	No

Adapted from [Corcho and Gomez 2000a] and [Corcho and Gomez 2000b])

Ontologies can and have been categorized in several ways. This categorization is extensively discussed in the companion paper[Kishore, Sharman, and Raman, 2004]. An ontology can be viewed as being either top-level or domain or application as shown in Figure 4 adapted from Guarino[1998].

- A top-level ontology describes very general concepts that are not specific to any domain. Any top level ontology can be used in multiple domains.
- A domain ontology define concepts, relationships, and other elements that are specific to a domain. A domain ontology is described using top-level ontologies and/or other domain ontologies to describe constructs within the ontology.
- Application ontologies are ontologies that define concepts, relations, and other elements that are specific to processes or tasks that have to be accomplished in a domain or between domains (which, in and of itself, can be considered as a domain). Application ontologies make use of top-level ontologies, domain ontologies and sometimes other application ontologies to describe constructs within the ontology.

The boundaries between Top-level, Domain and Application are often subject to loose interpretations and in that sense a specific ontology may overlap multiple categories.

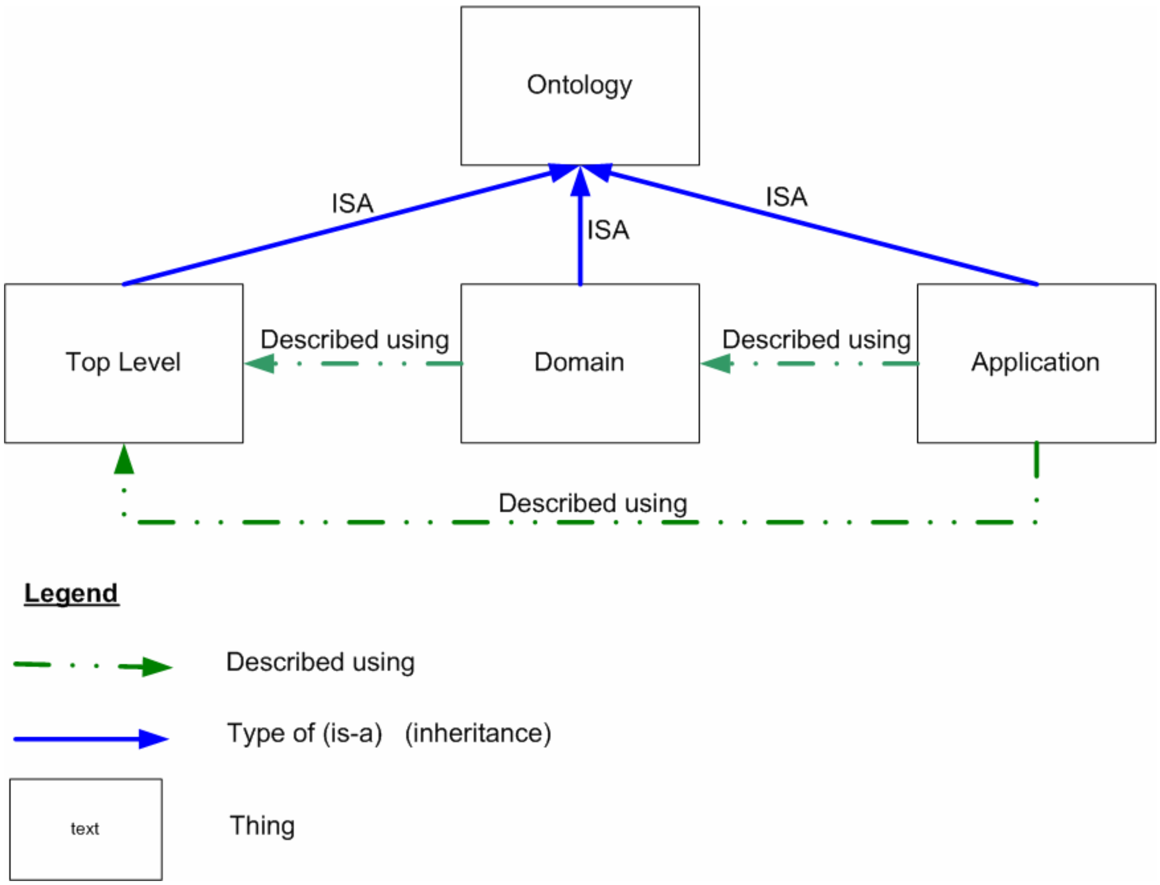

Figure 4: Ontology Types.

Table 5 exemplifies a few common ontologies and the language in which they are implemented.

Table 5. Languages Used to Implement Ontologies

Ontology Type	Ontology	Purpose	Reference	Language
Top Level (Linguistics)	CYC	The Cyc Knowledge Server is a very large, multi-contextual knowledge base and inference engine. Cyc is intended to provide a "deep" layer of understanding that can be used by other programs to make them more flexible.	[Lenat, 1990]	Cycl (based on first-order predicate calculus (FOPC), with extensions to handle equality, default reasoning, solemnization, and some second-order features)
Top Level (Linguistics)	GUM (Generalized Upper Ontology)	Linguistic categories	[Bateman et al., 1995]	LOOM
Top Level (Linguistics)	SENSUS	Provides vocabulary to describe various senses of a word and the relationship between senses.	[Swartout, 1997]	Ontolingua

Top Level (General)	EngMath	Mathematical modeling in Engineering.	[Gruber, 1994]	KIF
Top Level (General)	PhysSys	Modeling , Simulation and Designing Physical System.	[Borst, 1996]	
Domain	TAMBIS	Integration of heterogeneous bioinformatics sources.	[Baker, 1999]	GRAIL and OIL
Domain	GALEN	Provide coherence in medical terminology, for applications such as medical record keeping, etc.	[Rector, 1999]	GRAIL
Domain	ONIONS	Integration of terminological ontologies in medicine	[Gabgemi et al., 1996]	Ontolingua Formalism – Conceptual Graph
Domain	Gene Ontology [GO]	Provide structured vocabularies for the description of molecular function, biological processes and cellular component of gene products in any organism.	[Ashburner, 2002]	☺ Copyrighted
Domain	RiboWeb	Describe ribosomal components, associated data and computations for processing those data. The ontology contains structural data pertaining to the entire ribosome of prokaryotes (but primarily E. coli)	[Bada, 2000]	Java using the Protégé editor
Domain	EcoCyc	Describes the genes and intermediary metabolism of E.coli. Covers E. coli. genes, metabolism, regulation and signal transduction.	[Karp, 2000] [Karp, 1999]	
Domain	OZONE	It is a transportation planning and scheduling ontology. It provides a language for describing those aspects of the scheduling domain that are relevant to construction of an application system, and a set of constraints on how concepts in the language fit together to form consistent domain models. OZONE ontology adopts an activity centered modeling viewpoint and is biased towards constraint-based scheduling generation	[Smith, 1994] [Smith, et. al., 1996]	
Doman	Chemicals	To provide information about Chemicals [Elements from the periodic table].	[Lopez et al., 1999]	Semi-Formal. Implemented using Ontolingua
Doman (Enterprise)	TOVE	Enterprise Modeling	[Fox, 1996]	First-order predicate logic; Implemented using Quinus Prolog (axioms), and the rest in C++
Doman (General Information Systems Integration, Data Warehousing)	MOMIS	Integrates the schemas of heterogeneous information systems into a shared ontology. Provides a framework to perform information extraction and integration from both structured and semi-structured data sources.	[Bergamaschi et al., 2001] [Beneventano et al., 2001]	Uses an object oriented language ODL-I3 which has some description logics foundation.
Application (Agent based Information Systems Integration.	InfoSleuth	Agent based system to access to heterogeneous information sources and service. Provides a unifying framework for selectively and dynamically leveraging and	[Nodine et. al., 2000]	Agents coded in Java. Communicate with Ontologies via KQLM. Also uses Open Knowledge Based Connectivity (OKBC).

Maps shared and local Ontologies)		combining functionality provided by disparate classes of systems		
Application (Agent based Information Systems Integration. Maps shared and local Ontologies)	KRAFT	Agent based system for integration of heterogeneous information systems. Knowledge is integrated in the form of constraints. Maps information between shared and local ontologies	[Visser et al., 1999]	Uses common command and query language (subset of KQML), Constraint Interchange format language, and Prolog/Functional data Model.

Implementation languages that are based on logic formalisms carry over the shortcomings of the formalism. However at the implementation level, most of these languages provide mechanism to overcome the shortcomings. These fix-ups gives the appearance that all languages are functionally equal. However the elegance of the construct and the way the constructs are implemented and overcome shortcomings affects the ease of use of the implementation language. Further not all the shortcomings may have been overcome. Many of the languages are in continuous development and more features are being added.

IV. FUTURE RESEARCH AREAS

Research on ontology development and ontological engineering in general is a fertile area of academic pursuit with tremendous practical implications. While a complete enumeration of all these research directions is daunting, we highlight some of the important areas and provide guidelines in the following discussion. We organize these areas along the key dimensions of ontology mapping and ontology metrics. These dimensions form the two subsections of this section.

ONTOLOGY MAPPING

For an ontology to be useful over the long haul the issue of mapping an ontology to other parts of the system such as databases, user-interfaces, organizational processes needs to be addressed better [Ding and Foo, 2002]. Theoretical and empirical foundations need to be established for this kind of mapping.

Mapping refers to the connection of an ontology to different parts of an application system. Two important issues to consider are:

- mappings between ontologies and the information they describe and
- mapping between different ontologies used in an integrated system.

Mapping an ontology to the actual content of an information source is an open area of research. There are no good frameworks. Integrating information systems that use different ontologies requires an inter-ontology mapping. Some work exists in this area in projects such as the TAMBIS effort [Baker, 1999].

Unaddressed questions also include how ontology maps to the stages in the life cycle of an information system. As an ontology driven information systems matures through its life cycle, representation tools should provide features to capture the needs of the information system from an ontological perspective so that the shared understanding between the different users is maintained. Language features should support easy growth of the ontology as the domain changes and as the ontology driven information system evolves.

ONTOLOGY METRICS

Ontology metrics are an open research issue. How do we evaluate ontologies? Not much research activity is going on at this time in this area. We as a community need to develop more tools, and better languages that could help in creating and integrating ontologies to information systems. A computational ontology can be evaluated from both the ontology developers' perspective and the ontology users' perspective. Gomez-Perez differentiates these two perspectives by using terms "evaluation" and "assessment" to represent them [Gomez-Perez, 1994]. Evaluation judges technically the features of ontologies with respect to a frame of reference when ontologies are being developed [Gomez-Perez, 1995]. Assessment refers to usability and utility of the ontologies when they are used within a given organization or by software agents [Gomez-Perez, 1995]. Gomez-Perez [1994] proposes that the activities of ontology evaluation includes

- evaluation of each individual definition or axiom;
- evaluation of the set of definitions and axioms gathered in the ontology; and
- evaluation of the definitions and axioms that are imported from other theories.

Evaluation further subsumes verification and validation.

- *Ontology verification*: ontology verification refers to building the ontology right; that is, insuring that the ontology correctly implements its requirements, its competence questions [Gruninger and Fox, 1994] or the real-world.
- *Ontology validation*: Ontology validation refers to whether the meaning of the ontologies' definitions represents the real world for which it was created. The validation of the ontologies against the frame of reference provides information about whether the ontology definitions are necessary and sufficient to represent the tasks and their solutions for different uses.
- *Ontologies assessment:* Ontology assessment addresses computational ontologies from the users' perspective. This perspective encapsulates the users' needs to communicate, to share knowledge, or to develop application systems. As a result, ontologies assessment includes the understanding, usability, adequacy in the representation of behavioral knowledge and constraints, generality, granularity, quality, well-defined (both logically and syntactically) properties, portability, incrementalism, maintainability and uniformity of the definitions and axioms given by the ontology.

The verification and validation of an ontology in terms of architecture, lexicon and syntax, and content translates to the development of criteria for bounded completeness and soundness of the ontology. These criteria can be obtained from similar other contexts as theorem proving in the AI literature or rules of design and normalization used in the traditional database design area or through some innovative combinations and adaptations of Gruber's (1993a) criteria for ontology design. As a result, significant research on the development of metrics to assess the developmental processes, ontology constructs and structures, content and application methodologies is critically needed.

V. CONCLUSION

In this paper we discussed issues that deal with formal representations. We provided a detailed discussion of languages for specification and implementation of ontologies. We also provided examples of ontologies and the languages in which the ontologies are specified.

The goal of this paper and its companion [Kishore, Sharman, and Raman, 2004] is to provide a comprehensive state-of-the-art review about computational ontologies. The companion paper discussed the foundations and definitions of computational ontologies. This paper provided a comprehensive review of the formalisms, languages, and tools used for specifying and implementing computational ontologies. Both parts included directions for future research.

Editor's Note: This article was fully peer reviewed. It was received on December 5, 2003. It was with the authors for three and a half months for three revisions. It was published on August __, 2004

REFERENCES

EDITOR'S NOTE: The following reference list contains hyperlinks to World Wide Web pages. Readers who have the ability to access the Web directly from their word processor or are reading the paper on the Web, can gain direct access to these linked references. Readers are warned, however, that

1. these links existed as of the date of publication but are not guaranteed to be working thereafter.
2. the contents of Web pages may change over time. Where version information is provided in the References, different versions may not contain the information or the conclusions referenced.
3. the author(s) of the Web pages, not AIS, is (are) responsible for the accuracy of their content.
4. the author(s) of this article, not AIS, is (are) responsible for the accuracy of the URL and version information.

Ashburner, M. e. a. (2000) "Gene Ontology: tool for the unification of biology," *Nature Genetics* (25) 1, pp. 25-29.

Baader, F., D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider. (2003) *The Description Logic Handbook: theory, implementation, and applications*, First edition, Cambridge, Cambridge University Press.

Bada, M., and Altman, R. B., (2000) "Computational Modeling of Structured Experimental Data" *Methods in Enzymology* 317(A):470-491.

Baker, P. G., C.A. Goble, S. Bechhofer, N.W. Paton, R. Stevens, and A Brass. (1999), "An Ontology for Bioinformatics Applications" *Bioinformatics*, 15(6), pp. 510-520.

Bateman, J. A., B. Magnini, and G. Fabris. (1995) "The generalized upper model knowledge base: Organization and use." *Proceedings of the conference on Knowledge Representation and Sharing, Twente, the Netherlands, 1995*.

Beneventano, D., S. Bergamaschi, F. Guerra, and M. Vincini. (2001) "The MOMIS approach to Information Integration." *IEEE and AAAI International Conference on Enterprise Information Systems, Setubal, Portugal, 2001*.

Bergamaschi, S., S. Castano, D. Beneventano, and M. Vincini (2001) "Semantic Integration of Heterogenous Information Sources," *Special Issue on Intelligent Information Integration, Data & Knowledge Engineering* (36) 1, pp. 215-249.

Bernstein, A., S. Hill, and F. Provost. (2002) *Intelligent Assistance for the Data Mining Process: An Ontology-based Approach*. Stern School of Business, New York University IS-02-02.

Borgida, A. (1989) "Type Systems for Querying Class Hierarchies with non-strict Inheritance." *PODS 1989, Philadelphia, PA, 1989*, pp. 394-400.

Borgida, A. (1996) "On the relative expressive power of Description Logics and Predicate Calculus", Artificial Intelligence 82 pp 353-367

Borgida, A. (1995) "Description Logics in Data Management," *Knowledge and Data Engineering* (7) 5, pp. 671-682.

Borst, P., Bengamins, J., Wielinga, B., and Akkermans, H. (1996) “An Application of Ontology Construction.” *Workshop on Ontological Engineering, ECAI '96, 1996*, pp. 5-16.

Brachman, R. J. and H. J. Levesque (eds.) (1985) *Reading in Knowledge Representation*, Los Altos, CA: Morgan Kaufmann.

Cannataro, M. and C. Comito (2003) "A Data Mining Ontology for Grid Programming," University of Calabria, 2003).

Cannataro, M., A. Conguista, C. Mastroianni, A. Pugliese et al. (2003) “Grid-Based Data Mining and Knowledge Discovery,” in, vol. 90 N. Zhong and J. Liu (Eds.) *Handbook of Intelligent Information Technology*: IOS Press.

Chaudhri, C., A. Farquhar, R. Fikes, D. Karp et al. (1998) “OKBC: A Foundation for Knowledge Base Interoperability.” *Proceedings of the National Conference on Artificial Intelligence, 1998*, pp. 600-607.

Corcho, O., and Gomez-Perez, A. (2000a) “A Roadmap to Ontology Specification Languages.” *2nd International Conference on Knowledge Engineering and Knowledge Management, 2000a*, pp. 80-96.

Corcho, O., and Gomez-Perez, A. (2000b) “Evaluating Knowledge Representation and Resourcing Capabilities of Ontology Specification Languages.” *ECAI'00 Workshop on Application of Ontologies and Problem Solving Methods, 2000b*.

Ding, Y. and S. Foo (2002) "Ontology Research and Development: Part 1," *Journal of Information Science* (28) 2.

Donini, F., Lenzerini, M., Nardi, D., Schaerf, A., (1996) “Reasoning in Description Logics”, in Principles of Knowledge Representation and Reasoning, edited by G. Brewka; Studies in Logic, Language and Information, CLSI Publications, pp 193-238.

Fensel, D., I. Horrocks, F. Harmelen, S. Decker et al. (2000) “Oil in a Nutshell.” *12th International Conference of Knowledge Engineering and Knowledge Management, Juan-les-Pins, France, 2000*.

Fieser, J. and B. Dowden (2004) "The Internet Encyclopedia of Philosophy," http://www.iep.utm.edu/ (Current August 3, 2004).

Fox, M., Barbuceanu, M., and Gruninger, M. (1996) "An Organisation Ontology for Enterprise Modelling: Preliminary Concepts for Linking Structures and Behaviour," *Computers in Industry* (29pp. 123-134.

Gabgemi, A., Steve, G., Giacomelli, F. (1996) “ONIONS: An Ontological Methodology for Taxonomic Knowledge Integration.” *ECAI '96 Workshop: Ontological Engineering, 1996*.

Gomez-Perez, A. (1994) *From Knowledge Based Systems to Knowledge Sharing Technology: Evaluation and Assessment*. Knowledge Systems Laboratory, Stanford University KSL 94-73.

Gomez-Perez, A. (1995) “Some ideas and examples to evaluate ontologies.” *Eleventh IEEE Conference on Artificial Intelligence for Applications, New York, 1995*, pp. 299 -305.

Gruber, T. R. (1993a) “Toward Principles for the Design of Ontologies Used for Knowledge Sharing,” in N. Guarino and R. Poli (Eds.) *Formal Ontology in Conceptual Analysis and Knowledge Representation*: Kluwer Academic Publishers.

Gruber, T. R. (1993b) "A Translation Approach To Portable Ontology Specifications," *Knowledge Acquisition* (5pp. 199-200.

Gruber, T. R. and G. Olsen. (1984) "An ontology for engineering mathematics." *Proceedings of the Fourth International conference on Principles of Knowledge Representation and Reasoning, San Mateo, CA, 1984*, pp. 258-269.

Gruber, T. R. a. O., G. (1994) "An Ontology for Engineering Mathematics." *Fourth International Conference on Principles of Knowledge Representation and Reasoning, Gustav Stresemann Institute, Bonn, Germany, 1994*.

Gruninger, M. and M. S. Fox. (1994) "The Role of Competency Questions in Enterprise Engineering." *IFIP WG5.7 Workshop on Benchmarking - Theory and Practice, Trondheim, Norway, 1994*.

Guarino, N. (1998), 'Formal Ontology in Information Systems', Proceedings of FOIS'98, Trento, Italy, 6-8 June 1998. Amsterdam, IOS Press, pp. 3-15.

Horrocks, I., F. Harmelen, P. Patel-Schneider, T. Berners-Lee et al. (2001) *DAML+OIL Web Ontology Language: Reference Descriptions*. Report submitted to W3C, March 2001.

IEEE (2003) "P1600.1 Standard Upper Ontology (SUO) Working Group," IEEE, (November 14, 2003, 2003).

Jennings, N. R., T. J. Norman, P. Faratin, P. O'Brien et al. (2000) "Autonomous Agents For Business Process Management," *Journal of Applied Artificial Intelligence* (14) 2, pp. 145--189.

Karp, P., Riley, M., Saier, M., Paulsen, I., Paley, S., and Pellegrini-Toole, A. (2000) "The EcoCyc and MetaCyc Databases," *Nucleic Acids Research* (28pp. 56-59.

Karp, P. D. e. a. (1999) "EcoCyc: Encyclopedia of E. coli Genes and metabolism," in, vol. 27*Nuclear Acidic Research*, pp. 1-55.

Kelly, J. (1997) The Essence of Logic, *1st edition*, Upper Saddle River, NJ: Prentice Hall.

Kent, R. E. (2002) "Conceptual Knowledge Markup Language (version 0.2)," http://www.ontologos.org/ckml/ckml%200.2.html) (Current July 3, 2004).

Kifer, R., Lawsen, G., and Wu, J. (1995) "Logical foundation of Object-Oriented and Frame-Based Systems," *Journal of the ACM* (42) 2, pp. 741-843.

Kishore, R., Ramesh, R., and Sharman, R., (2004), "Computational Ontologies and Information Systems: I. Foundations", Communication of the AIS, 2004.

Lenat, D. B. and R. V. Guha (1990) *Building Large Knowledge-Based Systems: Representation and Inference in the CYC project*. Reading, Massachusetts: Addison-Wesley Publishing Company, Inc.

Li, Y. and N. Zhong (2003) "Web Mining Model and its Applications for Information Gathering," *forthcoming in Knowledge-Based Systems, special issue on Web Intelligence*.

López, M. F., A. Gómez-Pérez, J. P. Sierra, and A. P. Sierra (1999) "Building a Chemical ontology using Methontology and the Ontology Design Environment," *IEEE Intelligent Systems* (14) 1, pp. 37-46.

Luke, S. a. H., J. (2000) "SHOE 1.01 Proposed Specification," SHOE Project, http://www.cs.umd.edu/projects/plus/shoe/spec1.01.htm (Current June 20, 2004).

MacGregor, R. (1991) “The Evolving Technology of Classification-Based Knowledge Representation Systems,” in J. Sowa (Ed.) *Principles of Semantic Networks; Explorations in the Representation of Knowledge*, San Mateo, CA: Morgan Kaufmann Press.

Maedche, A. and S. Staab. (2000) “Mining ontologies from Text.” *Proceedings of 12th International Workshop on Knowledge Engineering and Knowledge Management (EKAW 2000), Juan-Les-Pins, France, 2000*.

Miller, D. (1998) "An Introduction to the Resource Description Framework," *D-Lib Magazine*.

Minsky, M. (1975) “A Framework for Representing Knowledge,” in P. Winston (Ed.) *The Psychology of Computer Vision*, New York: McGraw-Hill.

Nodine, M., D. Chandrasekara, and A. Unruh. (2000) “Task Coordination Paradigms for Information Agents.” *Proceedings of the International Workshop on Agent Theories, Architectures and Languages, 2000, 2000*.

Noy, N. F., and McGuinnss, D.L. (2002) "Ontology Development 101: A Guide to Creating Your First Ontology," Stanford Medical Informatics Report SMI Technical SMI-2001-0880, http://smi-web.stanford.edu/pubs/) (Current June 20, 2004).

Parigot, D. (1999) “Using Attribute Grammars in Industry”, 2nd International Workshop on Attribute Grammars and their Applications, ETAPS'99, Amsterdam, The Netherlands, March, 1999.

Rector, A., and Rogers, J. (1999) “Ontological Issues in Using a Description Logic to Represent Medical Concepts: Experience from GALEN.” *International Medical Informatics Association (IMIA), Working Group 6: Workshop: Terminology and Natural Language in Medicine, Phoenix, AZ, 1999*.

Rector, A. (2002) “Analysis of propagation along transitive roles: Formalisation of the GALEN experience with medical ontologies.” *Proceedings of DL 2002, 2002*.

Rector, A., S. Bechhofer, C. A. Goble, I. Horrocks et al. (1997) "The GRAIL concept modelling language for medical terminology.," *Artificial Intelligence in Medicine*, pp. 139-171.

Reichgelt, H. (1991) *Knowledge Representation: An AI Perspective*. Norwood, NJ: Ablex Publishing Corporation.

Rogers, M. H. and Galton. A., (1990) Logic for Information Technology, New Jersey, John Wiley & Sons Inc.

Russell, S., and P. Norvig, (2003), “Artificial Intelligence – A Modern Approach”, Prentice-Hall, New Jersey.

Shapiro, S. (1979) “The SNePS semantic network processing system,” in Findler (Ed.) *Associative Networks: Representation and Use of Knowledge by Computers*, New York: Academic Press, pp. 179-203.

Simoff, S. J. and M. L. Maheri (2002) "Ontology-based Multimedia Data Mining for Design Information," Key Centre of Design Computing, University of Sydney, 2002).

Smith, R., (2001) “Report from ECOMAS 2001”, http://www.csm.uwe.ac.uk/~rsmith/ECOMAS/ECOMAS2001/report.htm (current July 31, 2003).

Smith, S. (1994) “Opis: A methodology and architecture for reactive scheduling” In Zweben, M., and Fox, M., eds., Intelligent Scheduling, Morgan Kaufmann Publishers.

Smith, S.; Lassila, O.; and Becker, M. (1996) “Configurable,mixed-initiative systems for planning and scheduling”. In Tate, A., ed., Advanced Planning Technology. Menlo Park: AAAI Press.

Sowa, J. F. (1984) *Conceptual Structures: Information Processing in Mind and Machine*. New York: Addison-Wesley.

Sowa, J. F. (1991) “Logical Structures in the Lexicon.” *Proceedings of the Special Interest Group on the Lexicon Workshop, Berkeley, CA, 1991*, pp. 39-60.

Sowa, J. F. (1993) “Relating Diagrams to Logic.” *International Conference of Conceptual Structures, Quebec City, Canada, 1993*, pp. 1-35.

Sowa, J. F. (1999) *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Pacific Grove, CA: Brooks/Cole.

Stevens, R., R., Baker, G. P., Bechhofer, S., Ng, G., Jacoby, A., and Paton, N. (1998) “TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources.” *Sixth International Conference on Intelligent Systems for Molecular Biology, Montreal, Quebec, Canada, 1998*.

Swartout, B., Patil, R., Knight, K. and Russ, T. (1997) “Toward Distributed Use of Large-Scale Ontologies.” *American Association of Artificial Intelligence Spring Symposium Series, Workshop on Ontological Engineering, Providence, RI, 1997*.

Visser, P., D. Jones, M. Beer, T. Bench-Capon et al. (1999) “Resolving ontological heterogeneity in the KRAFT project.” *10th International Conference and Workshop on Database and Expert Systems Applications (DEXA'99), University of Florence, Italy, 1999*.

Waterman, D. A. a. F. H.-R. (1979) “Exemplary Programming in RITA,” in D. A. W. a. F. Hayes-Roth (Ed.) *Pattern Directed Inference Systems*, New York: Academic Press.

Winograd, T. (1987-88) "A language/action perspective on the design of cooperative work," *Journal of Human-Computer Interaction* (3) 1, pp. 3-30.

Wrobel, A., O. Wuermli, J. Joller, and S. C. Hui (2003) "Data Mining for Ontology Building," *forthcoming in IEEE Intelligent Systems*.

Zhang, H., Kishore, R., Ramesh, R. and Sharman, R., “The GRITICKA ontology for modeling multiagent-based integrative business information systems,” *2003 Americas Conference on Information Systems (AMCIS 2003),* Tampa, FL, August 2003.

Zhang, H., R. Kishore, R. Sharman, and R. Ramesh. (2004) “The GRITIKA ontology for modeling e-service applications: formal specification and illustration.” *Proceedings of the 37th Hawaii International Conference on System Sciences (HICSS-37), Big Island, Hawaii, 2004*.

ABOUT THE AUTHORS

Rajiv Kishore is assistant professor in the School of Management at the State University of New York at Buffalo. His research interests are in IT outsourcing, adoption of IT innovations, and conceptual modeling. His papers are published or accepted for publication in *Communications of the ACM, Decision Support Systems, Information Systems Frontiers, Journal of Healthcare Information Management, and Journal of Management Information Systems.* Rajiv presented his research at ICIS, HICSS, AMCIS, SIM, and other international conferences and workshops, and received a best paper award at AMCIS 2001. He is the recipient of a multi-year National Science Foundation research grant as a co-principal investigator in the area of IT outsourcing. Rajiv is also a founding board member of AIS SIG on Ontology-Driven Information Systems (SIG-ODIS). He is currently guest-editing a special issue of the *Journal of AIS* on Ontologies in the Context of Information Systems.

Raj Sharman is assistant professor in the School of Management at State University of New York at Buffalo. He received his Bachelors degree in Engineering and Masters Degree in Management from the Indian Institute of Technology, Bombay, India. He also received a Masters in Industrial Engineering, and a Doctorate in Computer Science from Louisiana State University. His research interests are in Conceptual Modeling and Knowledge Representation, Information Security, Web Caching and Medical Informatics. He is the recipient of several grants, both internal and external, to support his research. His publications appear in refereed journals and conferences in both the Information Systems and the Computer Science disciplines. Prior to his appointment at The State University of New York at Buffalo, Raj Sharman was active in the technology sector both as an entrepreneur and as a developer of Enterprise Integration Software.

Ram Ramesh is professor in the School of Management at the State University of New York at Buffalo. His research streams include conceptual modeling (ontologies, connectionist modeling and nonmonotonic reasoning, economics and technologies of internet capacity provision networks (CPN), and database systems and distributed computing frameworks. He is a founding board member of AIS SIG on Ontology-driven Information Systems (SIG-ODIS) and a board member of AIS SIG on Semantic Web and Information Systems (SIG-SEMIS). He serves as an Associate Editor for *INFORMS Journal on Computing, Communications of the AIS* and several more. He is co-Editor-in-Chief of *Information Systems Frontiers* and edited volumes in *Annals of OR* and *CACM.* He is currently guest-editing an issue of the *Journal of AIS* on Ontologies in the context of Information Systems. He publishes extensively in journals such as *INFORMS Journal on Computing, Information Systems Research, IEEE/TKDE, ACM/TODS* and *CACM* among others

Communications of the Association for Information Systems

ISSN: 1529-3181