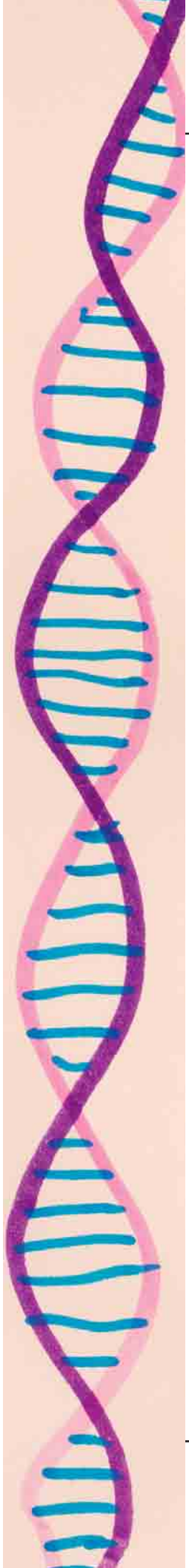


by RAJIV KISHORE, HONG ZHANG, AND R. RAMESH

A HELIX-SPINDLE MODEL *for Ontological Engineering*

Recently, there has been increasing recognition that ontological principles can be effectively applied to the broader field of information systems [5, 9]. This has led to the notion of "ontology-driven information systems" covering both the structural (the content including the database, user interface, and application program components) and the temporal (the development and operation processes) dimensions of IS [5]. In the structural dimension, ontological abstractions of IS artifacts will lead to improved communication through common vocabularies, enhanced software reuse, and systems interoperability [9]. In the temporal dimension, effective and efficient conceptual analysis and design processes can be obtained through new ontology-driven IS development approaches [5].

It is in this context that ontological engineering gains currency and relevance because the first step in ontology-driven IS development is to develop useful IS ontologies at appropriate levels of abstraction and granularity. In this article, we present a Helix-Spindle model for engineering special-purpose ontologies that has evolved from our experience in developing MibML, a special-purpose ontology for the discourse universe of Multiagent-based Integrative Business Information Systems (MIBIS). The Helix-Spindle model provides a new way of thinking about ontological

Using a forward-lockstep build-test process that combines theoretic and pragmatic approaches to ontology building.

CECILIA ROSENBLOOM

engineering and incorporates several unique features not found in current ontological engineering approaches.

Ontologies, Frameworks, and Systems

Information systems are essentially knowledge artifacts that capture and represent structural and behavioral knowledge about particular knowledge contexts. Knowledge contexts can be divided into three abstraction/generalization levels—a universe of discourse, a domain of interest, and an instance of domain of interest—and these three contexts occur at increasing levels of concreteness and specialization. If business is the subject of interest, then the world of business, an industry within the world of business, and a specific company within the chosen industry exemplify the three levels. The IS knowledge artifacts pertaining to these three levels are an ontology, a framework, and a system, respectively (see Figure 1). The guiding principle in all these artifacts has been the extraction of the commonalties in structures, functions, and processes into instantiatable knowledge abstractions at different levels of specialization.

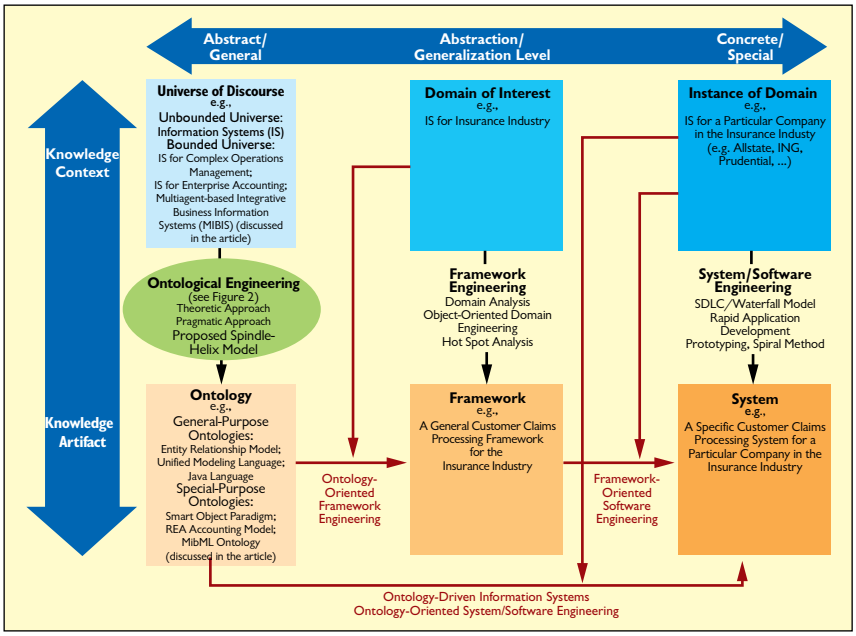


Figure 1. The knowledge-based view of information systems: Ontologies, frameworks, and systems.

Ontologies can be distinguished in terms of whether they provide a foundational representation vocabulary or capture a body of knowledge about a discourse universe [7]. An ontology as a representation vocabulary is equivalent to a conceptual modeling grammar [11] in the IS field. An ontology as a body of knowledge is similar to the notions of domain/application ontology [5] and application/domain/object-oriented framework [2] because all these terms refer to artifacts that capture and represent structural and behavioral knowledge about some domain of interest. These two types of ontologies—representation vocabulary versus body of knowledge—pertain to two different knowledge abstraction levels: a discourse universe and a domain of interest within the discourse universe, respectively (see Figure 1). Here, we use the term ontology to refer to a representational vocabulary for a discourse universe complete with appropriate syntax and semantics, and which can be utilized to model domains of interest within the discourse universe. Further, we use the term framework to refer to a body of knowledge within an application domain to maintain conceptual clarity about these notions.

Ontologies as representation vocabularies can be further classified into two types: general purpose or special purpose (see Figure 1). A general-purpose ontology is the knowledge artifact for an unbounded universe of discourse, while a special-purpose ontology is for bounded discourse universes [7]. Examples of general-purpose ontologies include the Unified Modeling Language (UML), the Entity-Relationship model, and the Java paradigm, while those for special-purpose ontologies are the Smart Object Paradigm [10], the REA Accounting Model [4], and the MibML ontology [8] for the bounded discourse universes of complex operations management, enterprise accounting, and MIBIS.

The Helix-Spindle Process Model for Ontological Engineering

Most extant ontological engineering approaches follow one of the two dominant approaches—the theoretic (deductive) approach or the pragmatic (inductive) approach [4]. The theoretic approach derives ontological constructs and their relationships from extant theories on the discourse universe while the pragmatic approach identifies and defines these constructs through an inductive problem-solving approach. For example, the REA ontology [4] has been developed predominantly following the theoretic approach, while the TOVE ontology [9] has resulted from mainly a pragmatic approach.

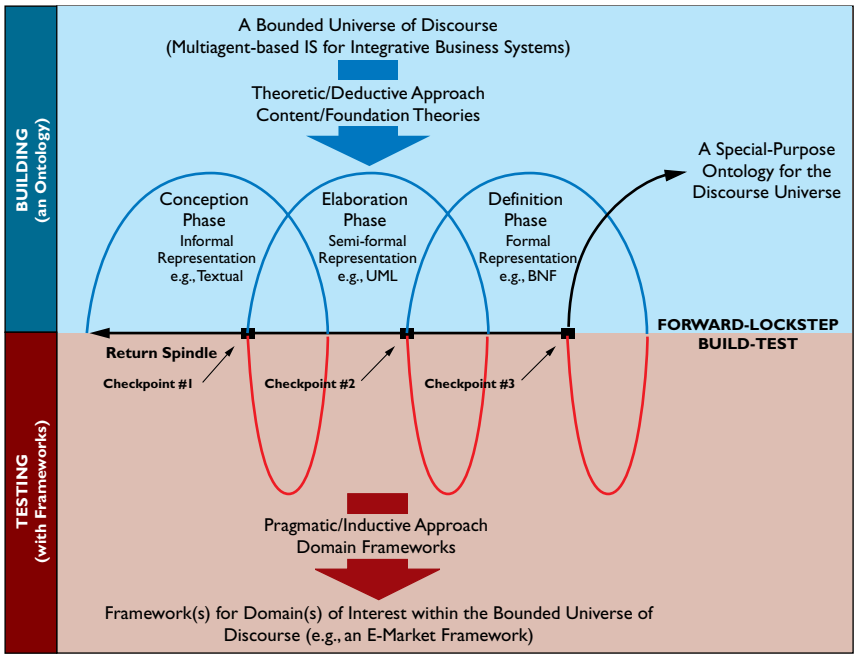
We have developed a new ontological engineering process model, the Helix-Spindle model (see Figure 2), as part of our research on developing the MibML (Multiagent-based Integrative Business Modeling Language) ontology for the MIBIS universe. One of the unique and distinguishing features of the Helix-Spindle model is it combines both the theoretic and the pragmatic approaches to ontology development in a single ontological engineering process model rather than following one or the other approach exclusively.

The Helix-Spindle process model consists of three major phases captured and represented using the imagery of a helix and a spindle. The three phases—a conception phase, an elaboration phase, and a definition phase—are represented as one full-loop each of the helix. This three-phased process model is motivated by the notion of incremental development, which is not only a well-respected concept in software engineering, it also provides the capability for developing ontologies at increasing levels of formality [9]. During the conception phase, an ontology for the discourse universe is initially conceptualized and specified using an informal meta-ontology (notation), such as natural English language. This ontology is then further elaborated and refined during the elaboration phase and is expressed using a semiformal meta-ontology, such as UML or Ontolingua. The final definition phase is used to formally define the complete ontology using a rigorously formal notation, such as BNF or predicate logic.

Another important feature of the Helix-Spindle model is each of its three phases consists of both ontology-building and ontology-testing activities, and these are captured in individual half-loops of the helix model. The ontology-building half-loops utilize content and foundation theories about the bounded universe of discourse to derive the relevant constructs, relationships, and constraints theoretically. The resulting ontology is then tested during the ontology-testing half-loop to ensure it provides an adequate, coherent, and consistent representation vocabulary for modeling domain frameworks contained within the discourse universe. This testing is conducted following a pragmatic or inductive approach by actually constructing one or more domain frameworks using the representation vocabulary.

If testing within a phase concludes that the ontology conceived and constructed thus far is capable of modeling domain frameworks within the discourse universe, ontology development proceeds to the next phase. However, if any problems related to coherence or extendibility are detected with the ontology, then development slides back to the beginning of the phase, and this is represented by the imagery of a spindle that is used to reach the starting point for the particular phase. Knowledge gained from pragmatic testing of an unsuccessfully completed phase is then applied during the next iteration of the building half-loop to further refine and strengthen the ontology. Making testing an integral part of each phase ensures problems in ontology development are identified early and unnecessary and misguided efforts toward further development of a weak or an erroneous ontology can be avoided.

Figure 2. The Helix-Spindle process model for ontological engineering.



The Helix-Spindle process model is based on sound principles of software and ontological engineering and the experiences we have gained from our research efforts in developing the MibML ontology [8, 12]. Our objective is not to describe the MibML ontology per se—we use it only as an exemplar to explain the key concepts and unique features of the Helix-Spindle process model. The table here also

provides a comparison of the key features and characteristics of the Helix-Spindle model with three other commonly cited ontological engineering process models.

Helix-Spindle Model in Action: A Case Study from the GRITIKA Ontology

The need for a special-purpose ontology for the MIBIS discourse universe arises on three counts. First, frameworks and systems for the MIBIS universe require an ontological foundation for faster development of reliable, flexible, and extensible systems [5]. Second, our approach is in part motivated by similar efforts in other discourse universes, such as the Smart Object Paradigm [10] and the REA ontology [4]. Third, and equally important, is the fact that general-purpose ontologies commonly used in the IS field do not possess adequate and powerful enough constructs and semantics to model MIBIS [12]. While the OO modeling paradigm and the UML ontology are quite recent and powerful, there are several conceptual differences between objects and agents, including differences in the degree of autonomy, flexibility, and control between objects and agents, and these differences render the UML ontology somewhat inadequate for agent systems. Further, the OO paradigm provides a very weak conceptualization for the notion of roles, which is central to both agent architectures and integrative systems.

To develop a special-purpose ontology for the MIBIS universe, we started with a literature review of the appropriate content and foundation theories in the key reference disciplines (including agent systems, integrative and workflow systems, and coordination, organization, and role theories), thereby utilizing the theoretic/deductive approach for iden-

Comparison of the Helix-Spindle model with other common ontological engineering process models.

Process Model Details	Uschold and Gruninger [9]	METHONTOLOGY [3]	Holsapple and Joshi [6]	Helix-Spindle Model (This article)
Goals of the Process Model: *Development of ontologies for:*				
Creating a shared understanding	Partially	Yes	Yes	Yes
Improving communication among people and organizations	Partially	Yes	Yes	Yes
Providing a content-base for ontology-driven information systems	Yes	Yes	No	Yes
Supporting analysis and design of ontology-driven information systems	Yes	No	No	Yes
Reuse of ontological content in ontology-driven information systems	Yes	Perhaps	No	Yes
Supporting interoperability of information systems	Yes	Perhaps in a limited domain	No	Yes
Process Model Output Characteristics:				
Ontology Type	Representation vocabulary	Body of Knowledge	Body of Knowledge	Representation vocabulary
Relevant Knowledge Context	A particular bounded universe of discourse	A particular domain of interest	A particular domain of interest	A particular bounded universe of discourse
Ontology Specification	Formal specification using first-order logic	Semi-formal and formal specification using database tables and logical axioms	Informal specification using textual and tabular descriptions, and taxonomic hierarchies	Formal specification using formal representation tools such as BNF and first-order logic
Intermediate Representations	Allowed through informal competency questions and informal terminology	Allowed through such informal representations as a glossary of terms	Not specified; the final ontology specification is itself informal	Allowed through informal (textual) and semi-formal (UML) representations
Key Features and Characteristics of the Process Model				
Ontology Development Approach	Uses a pragmatic/inductive approach	Uses a theoretic/deductive approach	Uses one of inspiration, induction, deduction, synthesis, or collaboration for initial seeding of the ontology, collaborative approach subsequently	Integrates a formal theoretic/deductive approach with a formal pragmatic/inductive approach in a single process model for development of the ontology
Process Iteration	Some iteration, but process model is linear predominantly	Highly iterative; iteration can occur after each phase of the process model	Iterative; iteration occurs only after a complete version of the ontology is developed	Highly iterative; iteration can occur after each phase of the process model
Ontology Evaluation	Evaluation methods and techniques are not described explicitly	Evaluation occurs during each development state; however, no formal evaluation methods and techniques are defined explicitly	Evaluation occurs only after a version of a particular ontology has been completed; it is conducted through a collaborative "Delphi" exercise	Evaluation occurs within each development stage; it is conducted formally by developing domain framework(s) using intermediate representation forms
Support for Evolutionary Growth and Monotonic Extendibility of Ontologies	Not specified but the process model may provide limited support for growth and extendibility	Yes	Yes	Yes
Detection of Errors/Inadequacies in ontologies being developed	Not clear; some error detection may occur during ontology specification, but most errors will only be detected during ontology use	Although this is not an explicit goal of the process model, it may help minimize incorrect early commitments to erroneous and/or inadequate ontologies	Ontological errors and inadequacies are detected during iterative improvement phase through collaborative evaluation	An explicit goal of the process model is to minimize incorrect early commitments to erroneous and/or inadequate ontologies; errors are detected early through ontology testing by building frameworks in each phase of the process model
Ontology(ies) Developed using the Process Model	TOVE and Enterprise ontologies for enterprise integration	Chemicals ontology	Ontology for the knowledge management phenomenon in organizations	MibML ontology for Multiagent-based Integrative Business Information Systems

Note:
This table has been compiled based on publicly available information in published papers about the three process models being compared with the Helix-Spindle model. Where information about particular aspects of a process model is sketchy or is not available in published papers, the authors of this article have formed their judgment about those aspects based upon applications of the process model in developing one or more ontologies, outlined here.

tifying the core concepts and relationships within the MIBIS universe. These were expressed using informal English language textual representations. In this initial phase, we did not have a formal Helix-Spindle process model for ontological engineering and we, therefore, relied upon existing ontological and software engineering principles. Following common practice, we planned to first develop the MIBIS ontology fully following a theoretic/deductive approach and specify it formally in an executable form. We planned to "test" our ontology only after its complete specification by building multiagent-based integrative systems for different domains. Following this plan, we moved to the next phase of refinement of our ontology and chose UML as the meta-ontology for its semiformal specification that resulted in the RITA (Role, Infrastructure, Taskflow, and Agent) ontology. We should mention here that while UML is not a "good" ontology for MIBIS for reasons mentioned earlier, it is quite adequate as a meta-ontology to be used for a semiformal intermediate representation of the MIBIS ontology.

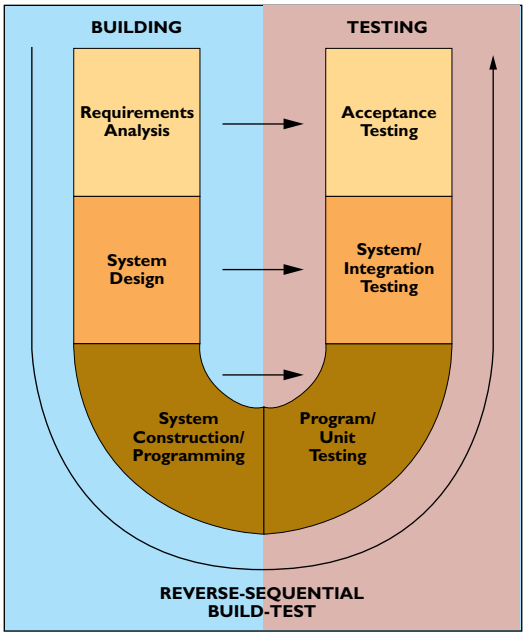


Figure 3. The U process model for software engineering.

However, we were finding it increasingly difficult to specify the ontology "completely" following a purely theoretic/deductive approach. There were numerous concepts in the reference disciplines pertaining to the MIBIS universe and the tasks of selection and abstraction of the relevant concepts were becoming very difficult. To overcome this difficulty, we decided to use the pragmatic approach for ontology development. We therefore decided to develop an e-market framework for the B2C e-commerce domain within the MIBIS universe using RITA with the intent of testing the viability and validity of the RITA constructs. One of the major outcomes of this exercise was to develop an explicit specification of goals as trees. While we had goals represented as part of an agent's attributes in RITA, this structure was neither very helpful in our conceptual analysis of the e-market framework nor powerful enough to capture and model various e-market processes.

This pragmatic validation of the RITA ontology proved to be an extremely valuable experience. It not only helped us get out of our mental staleness that set in following the theoretic approach for too long, it also planted the seeds for the Helix-Spindle process model. We identified obvious weaknesses in the RITA ontology through this pragmatic validation and started working on the next version of the ontology—the GRITA ontology—including the goal construct. More importantly, we realized that if we had continued with the theoretical development and formal specification of the ontology, we would have ended up with a sub-par ontology and a lot of wasted time and effort. We thus formalized the Helix-Spindle process model, which combines theoretic/deductive ontology building with pragmatic/empirical ontology validation during each phase of ontology development.

Development of the GRITA ontology followed the Helix-Spindle model, but it again encountered difficulty during the validation half-loop in the elaboration phase. Interagent communication and coordination requirements were modeled as tasks to be performed by agents in both this and the earlier versions of the ontology. However, as we started developing the e-market framework to validate the ontology, it became evident that a higher-level construct that identifies and defines various types of communication/coordination mechanisms/protocols for interactions among agents will greatly aid in conceptual analysis and modeling of the e-market framework. Once again, the Helix-Spindle model was utilized and we used checkpoint #2 and the spindle in the model to slide back to the very beginning of the ontological engineering process and started reconceptualizing the ontology with a new higher-level construct termed conversation giving rise to the GRITCA ontology.

This iterative ontological engineering process led to the development of successively refined versions of the ontology. GRITCA evolved into GRITICKA (Goals, Roles, Interactions, Tasks, Information, Capabilities, Knowledge, and Agents) ontology as it was found necessary to convert information, capabilities, and knowledge into foundation-level constructs. The GRITICKA ontology was subsequently

changed into GRITIKA (Goals, Roles, Interactions, Tasks, Information, Knowledge, and Agents) ontology as role/agent capabilities were subsumed under the knowledge construct due to a high degree of overlap between these two constructs [12]. This has evolved into MibML ontology [8], which is currently being specified formally using BNF and predicate logic.

Ontological Engineering Versus Software Engineering

Ontological engineering in general is closely related to the disciplines of framework and software engineering [1]. The Helix-Spindle model for ontological engineering is no exception. Although the goals of these disciplines could be considered similar, there are certain basic differences. While framework and software engineering are concerned with building domain frameworks and systems, the objective of ontological engineering is to abstract knowledge from a discourse universe and make those abstractions communicable, sharable, and reusable across domains and organizations.

However, a larger and a more profound difference between ontological engineering and software engineering lies in the basic engineering process itself. A generic software engineering process, regardless of the specific methodology being considered, can be conceptualized as a U-shaped process model that provides for three major phases of building the software artifact—requirements analysis, system design, and system construction/programming (see Figure 3). Development of the software artifact is an incremental refinement process. The software artifact is fully and formally specified only during the programming phase when it is written using a formal and executable language. However, testing in this case occurs in a reverse sequence. Programs created during the last phase of building are tested first via program or unit testing. Next, the artifact of the second phase—the design for the entire system—is tested. Finally, at the very end of the testing cycle, the software artifact gets tested with respect to the requirements that were established in the first phase: this test is typically called the acceptance test. This reverse sequence in building and testing is the crux of software engineering and is captured in the image of a U-shaped model.

Conversely, the Helix-Spindle model provides for building and testing the ontology artifact in three successive phases, but where the testing activities in each phase closely follow the building activities of that phase in a lockstep manner, while continuously moving forward from the first to the last phase, disregarding the iterations as in the case of the U-model. Therefore, the Helix-Spindle methodology is a Forward-Lockstep Build-Test model while software engineering models are Reverse-Sequential Build-Test models. This is a consequence of two simple facts. First, the Helix-Spindle model provides a process for creating a representation vocabulary, while a software engineering model provides a process for using a given representation vocabulary to create a functional system that has to meet some prespecified requirements. While one can perform an acceptance test only after the software system is created but against the first-phase specifications, one can assess the adequacy and consistency of a representation vocabulary only after it is completely specified in the third phase but not against the first-phase specifications, as they will be, by nature and by design, incomplete, and perhaps inconsistent. Second, while the software engineering U-model operates on a single level of knowledge abstraction—the instance of a domain, the Helix-Spindle model for ontological engineering operates on two different levels—the bounded discourse universe and a domain of interest within the universe.

The intent in software engineering is to test whether the needed functionality in a system has been captured and represented adequately using a particular representation vocabulary; it is not to test the adequacy, consistency, or correctness of the vocabulary itself. On the contrary, the intent in ontological engineering is precisely that, and the only empirical/pragmatic way to evaluate whether a vocabulary is adequate, consistent, and correct is by assessing whether the vocabulary can be used to build a framework or a system, a lower-level knowledge abstraction.

Don't Play Snakes and Ladders

Snakes and Ladders, an ancient board game similar to the modern-day version Chutes and Ladders, is a game of chance. While ladders are fun because they take you up, the board also contains several snakes that take you down, just like the chutes. The board contains 100 squares in 10 rows of 10 squares each. Tension builds when a player comes close to finishing the game as there is a snake positioned before the finish line on square 99 ready to swallow anyone landing there, and depending on the board, a player may plunge down through several rows of squares. Ontological engineering is a much more serious endeavor and it should not be conducted in a manner similar to a game of chance. Planned, careful,

and rigorous validation in a forward-lockstep manner during every stage of the ontology development process will greatly reduce the risk of getting "swallowed by the snake at square 99" and having to start again. And that is precisely what the Helix-Spindle model provides: a forward-lockstep build-test process that combines the theoretic and the pragmatic approaches resulting in a highly effective ontological engineering model. C

References

1. Devedžic, V. Understanding ontological engineering. *Commun. ACM 45*, 4 (Apr. 2002), 136–144.
2. Fayad, M.E., Schmidt, D.C., and Johnson, R.E., Eds. *Building Application Frameworks: Object-Oriented Foundations of Framework Design*. Wiley, New York, 1999.
3. Fernández, M., Gómez-Pérez, A. and Juristo, N. Methontology: From ontological art towards ontological engineering. In *Proceedings of Workshop on Ontological Engineering: AAAI-97 Spring Symposium Series*, (Stanford, CA, 1997), 33–40.
4. Geerts, G. and McCarthy, W.E. *The Ontological Foundation of REA Enterprise Information Systems*. Tech. Rep., University of Delaware and Michigan State University, 2000.
5. Guarino, N. Formal ontology and information systems. In N. Guarino, Ed., *Proceedings of FOIS '98*, (Trento, Italy, June, 1998). IOS Press, Amsterdam, 1998, 3–15.
6. Holsapple, C.W. and Joshi, K.D. A collaborative approach to ontology design. *Commun. ACM 45*, 2 (Feb. 2002), 42–47.
7. Kishore, R., Ramesh, R. and Sharman, R. Computational ontologies: Foundations, representations, and methods. In *Proceedings of Ninth Americas Conference on Information Systems (9th AMCIS)*, (Tampa, FL, 2003). Association for Information Systems, 3178–3189.
8. Kishore, R., Ramesh, R., Sharman, R., and Zhang, H. *MibML: A Conceptual Modeling Language for Multi-Agent-based Integrative Business Information Systems*. SUNY-Buffalo, NY, 2003.
9. Uschold, M. and Gruninger, M. Ontologies: Principles, methods and applications. *Knowledge Engineering Review 11*, 2 (Feb. 1996), 93–155.
10. Vaishnavi, V.K., Buchanan, G.C. and Kuechler, W.L. A data/knowledge paradigm for the modeling and design of operations support systems. *IEEE Transactions on Knowledge and Data Engineering 9*, 2 (Feb. 1997), 275–291.
11. Wand, Y. and Weber, R. Research Commentary: Information systems and conceptual modeling—A research agenda. *Information Systems Research 13*, 4 (2002), 363–376.
12. Zhang, H., Kishore, R., Sharman, R. and Ramesh, R. The GRITIKA ontology for modeling e-service applications: Formal specification and illustration. In *Proceedings of 37th Hawaii International Conference on System Sciences (HICSS-37)*, (Big Island, Hawaii, 2004), IEEE Computer Society.

Rajiv Kishore (rkishore@buffalo.edu) is an assistant professor in the Department of Management Science and Systems at the State University of New York at Buffalo.
Hong Zhang (hoz565f@smsu.edu) is an assistant professor at Southwest Missouri State University.
R. Ramesh (rramesh@acsu.buffalo.edu) is a professor in the Department of Management Science and Systems at the State University of New York at Buffalo.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

© 2004 ACM 0002-0782/04/0200 $5.00

Stay on Top of ACM News with

MemberNet

Coming in future issues of MemberNet:

- A new graphic design
- Findings from a recent readership survey
- New computing initiatives to reach underserved groups
- ACM's involvement in the RoboCup Jr. competition and the Grace Hopper Celebration of Women in Computing.

And much more!

All online, in MemberNet: www.acm.org/membernet.