Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2000 Proceedings

Americas Conference on Information Systems (AMCIS)

2000

Task-Based Ontology for Information Systems

Somkiat Kitjongthawonkul
Latrobe University, somkiat@cs.latrobe.edu.au

Rajiv Khosla
Latrobe University, khosla@cs.latrobe.edu.au

Task-Based Ontology for Information Systems

Somkiat Kitjongthawonkul, Department of Computer Science and Computer Engineering, La Trobe University, Melbourne, Australia, somkiat@cs.latrobe.edu.au
Rajiv Khosla, Department of Computer Science and Computer Engineering, La Trobe University, Melbourne, Australia, khosla@cs.latrobe.edu.au

Abstract

Problem solving method describes the reasoning process and knowledge requirement for accomplishing a given task. There are a number of problems solving methods and other problem solving approaches currently in place. Apparently, these existing approaches demonstrate an excellent work done in the field. Several of these approaches are technology based or driven where the designer/developer has to force fit their design into these technologies. Further, it is not clear as to how such approaches account for external representations in problem solving, epistemological limitations that humans and computers have, and pragmatic constraints associated with real world problems. In this paper we have examined these perspectives and taken them into account for developing Task-Based Problem Solving Adapters (TPSAs). TPSAs developed by us lead toward human centeredness and help us to address the pragmatic task constraints through a range of technologies like neural networks, fuzzy logic, and genetic algorithms. We have also provided an example of applying the TPSAs to develop a working system for assisting sales engineers of an electrical manufacturing firm in monitoring the status of orders in the company.

Introduction

There are a numbers of problem solving approaches currently in place (e.g. Generic Task (Chandrasekaran et al., 1992), Components of Expertise (Steels, 1990), KADS and CommonKADS (Schreiber et al., 1993; Schreiber et al., 1994), MIKE (Angele et al., 1998)). These approaches are somewhat different in their underlying philosophy; yet, they share common phenomena in problem solving process (Fensel and Groenboom, 1999). These include: a) decomposition the entire reasoning task into more elementary inferences; b) defining the types of knowledge that are needed by the inference steps to be done; and c) defining control and knowledge flow between the inferences.

More recently, the concept of adapter pattern has been introduced to solve particular recurring design problems of large-scale and complex software systems. These adapters facilitate the linkages on design-level of objects that differ in their syntactical input and output descriptions. Inspired by this concept, (Fensel and Groenboom, 1997) employed adapters and applied them in the context of problem solving. The attempt is to use adapter for adjusting the tasks, methods, and domain model to the specific application problem. This is for the purpose of reusable components of knowledge based systems. Apparently, these existing approaches demonstrate excellent bodies of work in knowledge based system area. Nevertheless, some crucial issues still have been left unclarified. For example, it is not clear as to how such approaches account for external (perceptual) representations in problem solving, epistemological limitations that humans and computers have, and pragmatic constraints associated with real world problems. Further, none of them provides insight into how to deal with the complexity of large-scale real world problems. That is to what extent applications developed by using these problem-solving methods or ontologies will be scalable, evolvable and maintainable. In addition, most of the above approaches are embedded in the knowledge based system technology. They do not adequately address the pragmatic task constraints like noisy and imprecise data, which are satisfied by other technologies like neural networks, fuzzy logic, and genetic algorithms. A lack of this consideration has resulted in unsatisfactory results (in terms of satisfaction of constraints and quality of solution) from implementation of these problem-solving methods in the field.

In this paper, we describe a task-based problem solving adapters. The emphasis of the work reported here lies on the representation vocabulary of the overall problem solving adapters construct. The adapters are used for modeling user's or practitioner's task, domain and representation ontology in a specific domain. The adapters developed by us are human-centered in their design and underpinnings. That is, they are derived from the problem solving pattern or consistent problem solving structures/strategies employed by practitioners while designing solutions to complex problems like image processing, alarm processing, sales management and medical diagnosis. They facilitate use of perceptual (external) as well as conceptual (internal) representations for problem solving as advocated by the distributed cognitive science approach (Zhang and Norman, 1994). It constrains the perceptual and conceptual representations of the environment in the context of problem and task being studied with the help of five information processing phases.

Preliminaries

In order to establish the background, we outline a wide spectrum of issues related to problem solving and

artificial intelligence in general. These different perspectives form the imperative part for a foundation of the Task-Based Problem Solving Adapters (TPSAs). Inspired by the work done of (Khosla and Dillon, 1997; Khosla and Dillon, 1994), our approach has incorporated various perspectives related to problem solving and artificial intelligence. These perspectives include human-centeredness, neurobiological control, cognitive science, man made physical systems, learning, conscious and automated behavior, knowledge representation, and others (see (Khosla and Dillon, 1993) for more details). Some of the perspectives are worth to be mentioned here.

From a neurobiological perspective, the highly connected and parallel design of the brain allows it to work on many different things at once (e.g. vision and speech). The brain, with its parallel design, is able to work on lots of different external stimuli/items of information in a distributed form and able to process these items of information in a parallel manner. This makes us intuitively feel that the complex real world problems that we are trying to solve are immensely parallel ones. Besides, research into the functional principles of the brain indicates that it exercises central and hierarchical control at different levels (sensory, motor, etc.) to provide both stability and adaptability. In addition, investigations into the visual nervous system of mammals reveal that the visual system works at different levels of abstraction. The underlying principles of abstraction and hierarchical control of the human brain form an intrinsic part of man made systems like power systems, organizational systems, manufacturing process control systems, telecommunication systems, etc. For example, in power systems, the power network is hierarchical decomposed into transmission, sub-transmission and distribution levels. The structure of these systems influences their behavior. Humans, while reasoning with these systems engage in structural and/or behavioral decomposition of the problem and perceive the solution process at different levels of abstraction.

From a knowledge representation perspective, knowledge can be broadly represented in three forms namely, sub-symbolic, symbolic and non-formal, and symbolic and formal. Sub-symbolic knowledge can be attributed to microfeatures of a particular concept or a set of concepts in a particular domain which necessarily cannot be articulated in terms of rules (e.g. pattern recognition), sometimes known as tacit knowledge and may involve parallel processing for inference. Symbolic and non-formal knowledge can be seen as knowledge represented by heuristic/rules which are an outcome of experience of the human expert and generally cannot be formalized in a rigorous fashion (e.g. heuristic taught by a piano teacher to a student). Symbolic and formal knowledge can be seen as knowledge represented by a more vigorous formalism (e.g. physics, mathematics, etc.) such as a mathematical/structural/behavioral model or formal logic representation. A particular application can use the above three forms of knowledge individually or in different combinations. All these forms of knowledge have also associated with them different levels of granularity ranging from coarse to fine-grain knowledge.

From human-centeredness perspective, human form an important part of solution to most artificial intelligent problems. Hence, it is essential that any systems, which is derived out of artificial intelligent methodologies, should result in reducing the cognitive barriers between human and computers. Without cognitive compatibility the followings may turn out: a) the system's behavior can appear surprising and unfriendly to user; b) effective interpretation of user's or expert's problem solving behavior is at risk which may result in unsatisfactory performance.

Ontology of Problem Solving Adapters

Prior to embarking on the description of the problem solving adapters construct, it is important to clarify the basic concepts used by us. The term ontology is used here to mean a representation vocabulary, typically specialized to some technology, domain or subject matter. In particular, here we are dealing with upper ontology, i.e., ontology that describes generic knowledge that holds across many domains. The problem solving adapters can be thought as analogous to a device that facilitates transformation of a human solution (obtained through activity-centered analysis) to a software solution (in form of a computer-based artifact). The use of adapters is thus for adjusting the impedance mismatch between the two different domains.

Based on the different perspective described in previous section, the TPSA is established on the five information processing phases, namely, Preprocessing Phase, Decomposition Phase, Control Phase, Decision Phase, and Postprocessing Phase. These five information phases represent the problem solving pattern or ontology being employed by human while designing solutions to a delegated task. The description of problem solving vocabulary of the five problem solving adapters are classified as follows:

- *Information Processing Phase* – a distinct step or event in problem solving
- *Goal* – a desire or desired outcome or state
- *Task* – is a goal directed process, which people consciously or unconsciously engage in.
- *Task Constraints* – are pragmatic constraints imposed by the stakeholders and the environment for successful accomplishment of a task.
- *Represented features* – are the conceptual and perceptual features of artifacts in a domain. Conceptual features are perceptual categories (e.g. high temperature, low temperature) which can have binary, structured, fuzzy or continuous values. Perceptual categories are derived from perceptual features. Perceptual feature is a stable signature in a raw sensory signal.

- *Representing dimension* – is the physical or abstract dimension used to represent a conceptual or perceptual feature.
- *Psychological Scale* – is the abstract measurement property of the physical or abstract dimension of a represented feature, types of scales including nominal, ordinal, interval and ratio.
- *Knowledge Engineering Strategy* – is a plan to select whether hard or soft methods are employed, depending upon the availability of domain knowledge.
- *Methods* – are ways of accomplishing a task. They can be of computational or algorithmic methods as well as perceptual or non-perceptual algorithmic methods.

An example of the Control phase adapter construct with its attributes is shown in Figure 1.

Figure 1. Control Phase Adapter Construct

Phase:	Control
Goal:	Establish domain decision control constructs for orthogonal concepts based on desired outcomes from the system
Precondition:	A. Orthogonal concept defined B. Concept related data available
Tasks:	A. Determine decision level concepts B. Problem formulation C. Decision level concept validation (optional - for relevance feedback systems) D. Conflict resolution (optional) - form - decision conflicts between decision categories, coordinating in vocation of decision level concepts
Task Constraints:	Scalability, Reliability, Learning, Adaptability, Domain model constraint
Domain Model:	Structured, Functional, Causal, Heuristic, Spatial, etc.
Represented Features:	*Quantitative/Linguistic* - Binary, Structured, Fuzzy data (optional) *Non-Linguistic* - Continuous data related to orthogonal concept
Psychological Scales:	Nominal & Ordinal, Internal, Ratio
Representing Dimensions (Perceptual):	Shape, Size, Length, Distance, Density, Location, Position, Color, Texture
Knowledge Engineering Strategy:	Top down/Bottom up
Methods:	Hard (e.g. Symbolic rule based), Soft (e.g. Neural network, Genetic algorithms), Perceptual

Preprocessing Phase Adapter

The main goal of the Preprocessing Adapter is to improve data quality so that they are suitable for processing. The tasks involved in this phase include, for example, noise filtering and input conditioning (e.g. input formatting, dimensionality reduction). Since the task like noise filtering is heuristic in nature. The represented features in the domain can be qualitative/linguistic (e.g. binary, structured and fuzzy or continuous). For example, in alarm processing problem, alarm may be filtered based on its existence (binary), based on multiple occurrences of it or based on the topology of the network (structured). Further, fuzzy variables (e.g. adjectives in a natural language query) may be used to eliminate particular type of queries. In domain like signal processing, fast fourier transforms are applied on continuous numeric data. The represented features can be analyzed based on the nominal, ordinal, interval or ration scales. These psychological scales have been used to derive perceptual and conceptual semantics of real world objects.

The perceptual dimensions on which the psychological scales are applied could be shape (e.g. eliminating noisy energy consumption profiles based on shape), distance (e.g. suppressing sympathetic alarms emerging from parts of network beyond certain threshold distance from the faulty component), color, etc.

Top down or bottom up knowledge engineering strategy can be assigned in this phase depending upon the availability of domain knowledge for accomplishing a task. In case the domain knowledge is available, the top down strategy is adopted resulting in a hard computing method like a symbolic rule based system. Otherwise, the bottom up strategy is employed resulting in use of soft computing methods like neural networks or genetic algorithms for accomplishing a given task.

Decomposition Phase Adapter

The goals of the decomposition phase adapter are to restrict the context of the input from the environment at the global level and to reduce the complexity and enhance overall reliability of the computer-based artifact. The input context at the global level is restricted in terms of stakeholder's or user's perception of the task context. The user's task context can be used to restrict the input in terms of different types of users (e.g. medical researcher, and evolutionary biologist in a human genome application), different control models in an optimum control system modeling application, different subsystems in a sales management application. Thus, user's task context is captured with help of concepts that are generally orthogonal in nature. These concepts are abstract as do not provide direct solution to the task in hand.

The generic task constraints associated with decomposition phase adapter are scalability and reliability. The concepts used to restrict the input context in decomposition phase should be scalable vertically as well as horizontally. One way of satisfying this task constraint is to ensure that the concepts defined in this phase are orthogonal or un-correlated. This will also enhance the reliability and quality of results produced by other phase adapters (e.g. control) which depend on the competency of the decomposition phase adapter.

The qualitative or linguistic features employed in this phase by the users are coarse-grain features. These coarse-grain features may have binary and/or structured values. For example, coarse-grain binary features to partition a global concept like an animal (into mammal, bird) in the animal kingdom domain may be *has_feathers*, *gives_milk*, *has_hair*, etc. On the other hand, structured

features like *player_configuration* (with values like 1,2,3,4) may be used in a computer game application. The features representing concepts in this phase can also be numeric or continuous in nature. For example, in an image processing application, like face recognition, orthogonal concepts like skin regions and non-skin regions can be distinguished based on the skin color pixel data.

The psychological scale used by the decomposition phase adapter is the nominal scale. The nominal scale is the lowest psychological scale with formal property category. It is suitable for determining orthogonal concepts represented by binary and structured qualitative features.

The representing dimension of the represented features can be shape, position, color etc. measured on the nominal scale. For example, in a face recognition application, the representing dimension for distinguishing between orthogonal concepts like skin-region and non-skin region is the skin-tone color. In order to accomplish various tasks in this phase, soft or hard computing mechanisms can be used by the decomposition phase adapter, depending upon the knowledge engineering strategy and the task constraints.

Control Phase Adapter

The goal of the control phase adapter is to establish the decision control constructs for the domain based decision classes as identified by stakeholders/users. Decision level classes are those classed inference on which is of importance to a stakeholder/user. These classes or concepts represent the control structure of the problem. These decision-level classes generally explicitly exist in the problem being addressed. They could represent a set of group of network components in a telecommunication network, electric power network problem (e.g. faulty sections), or a set of control actions in a control application, possible face region in a face recognition application, possible behavioral categories in a sales recruitment application, etc.

The granularity of a decision level class may vary between coarse and fine, depending upon the context in which problem is being solved and the decision level priority in a given context. In one context, a decision level class may be less important to a problem solver, and thus a coarse solution may be acceptable. In another context, the same decision level class, however, may assume higher importance and thus a finer solution may be required. That is, if the decision level class priority is low, then its granularity is coarse, and the problem solver is satisfied with a coarse decision on that class. Otherwise, if the decision level class priority is high then the decision level class which would involve a number of microfeatures in the domain space. In case of coarse granularity, distinct control and decision phase adapters (described in the next section) may not be required and can be merged into one.

It is possible that the decisions made by a decision level class may conflict with the decisions by another decision level class. For example, in a telecommunication network diagnostic problem, two decision level classes may represent two sections of a telecommunication network. If these sections predict fault in two different network components (given that only one of them can be actually fault), then there is a conflict. The conflict may be resolved by looking at the structural, functional, spatial indisposition of the decision level classes or their components or even through concept/decision validation (which would involve validation/feedback from the stakeholder/user on the conflicting set decisions).

The represented features involved in this control phase are qualitative/linguistic (binary, structured, fuzzy) and non-linguistic (continuous features). The qualitative or linguistic features include semi-coarse grain binary, structured and fuzzy features. The granularity of the binary and structured features used by the control phase adapter is finer than those used in the decomposition phase. In the decomposition phase, binary and structured features are used for determination of abstract independent orthogonal concepts at the global level. In the control phase adapter, the binary and structured features are used at the local level within each abstract concept. In addition, the binary and structured features are used many times with fuzzy features in order to identify the decision level concepts in a domain. The fuzzy features are used in the control phase rather than the decomposition because fuzzy features cannot be used to distinguish between abstract orthogonal concepts. For example, let us assume mammal and bird are two abstract concepts in an animal classification domain. Then, the interpretation of a large mammal is not the same as a large bird. That is, the fuzzy variable *large* qualifying a mammal and bird carry different perceptual as well as conceptual meanings and this cannot be used universally at the global level for discriminating between abstract concepts. Continuous features used by the control phase adapter are limited to an abstract concept determined in the decomposition phase. For example, in a face recognition application, pixel data related to the skin region concept is analyzed.

The psychological scale used by the control phase adapter can be nominal, ordinal, interval and ratio scales. In addition, the features used by the control phase adapter can be seen to represent information on the ordinal, interval or ratio scales. The representing dimension of the represented features can be shape, position, color etc. measured on the nominal and/or ordinal, interval and ratio scales. For example, in a face recognition application, area and shape of the skin-regions are the representing dimensions of the various face recognition decision classes. Plan to execute the task in this phase can be done in top down or bottom up strategy. If top down strategy is employed, it is assumed that qualitative data is available, whereas, if bottom up strategy is used it is assumed case data is available. The computing mechanisms can be hard (e.g. symbolic) or soft (e.g. neural networks, fuzzy logic, genetic algorithms) depending upon the task constraints and the knowledge engineering strategy.

Decision Phase Adapter

The granularity of the decision level class obtained from the control phase adapter is rather coarse grain and heterogenous in nature. It cannot provide instance solutions to the problem. It is then the responsibility of the Decision Phase Adapter to provide specific outcomes/solutions to satisfy the requirement of users/stakeholders. The main task engaged by the decision phase adapter involves determination decision instance. Decision instance or instances represent partly or wholly user defined outcomes from a computer-based artifact. These outcomes are realized within each decision class invoked by the control phase adapter. These outcomes are, for example, specific faulty components in an electronic circuit board, actual faces in a face recognition problem, a product with desired features in an electronic commerce application, etc.

The qualitative or linguistic features employed by the decision phase adapter can be fine grain fuzzy or even binary. For example, in the alarm processing problem two properties of the alarm data are used. First, existence or absence (i.e. binary property) of a circuit breaker alarm in a decision class (candidate faulty section) is determined. Second, fine grain fuzzy contribution value of a circuit breaker alarm and associated relay towards a fault in a particular network component is modeled in terms of their protection proximity to a possible faulty component (Khosla and Dillon, 1997). This contribution value is determined in terms of activity level of a path (consisting of alarm and relay). The non-linguistic represented features employed can be continuous decision data. For example, in the face recognition problem, color pixel data related to a face candidate and spatial coordinates of facial features like eyes, mount and nose are used to identify actual faces and track eye movements in the decision phase.

The nominal scale can be used to measure binary features like existence or non-existence of an alarm, whereas fine grain fuzzy features can be measured on the ordinal, interval or ratio scales depending on the scale properties by the fuzzy features. For example, in an animal classification (more specifically, tiger classification) domain, some of the scale properties of fuzzy features *heavy cheek hair* are category (cheek hair), magnitude (heavy > light) and absolute zero (no cheek hair). These properties represent the ratio scale.

The computing mechanisms can also be hard (symbolic), soft (e.g. neural network, fuzzy logic, and genetic algorithm) or other statistical/mathematical algorithms. Broadly, hard symbolic computing mechanisms like rule based systems can be used for high level tasks like problem formulation and noise filtering subject to availability of qualitative domain knowledge for the task. On the other hand, soft computing mechanisms can be used for decision instance task that may involve pattern recognition, learning, generalization and adaptability. As a consequence of satisfying task constraints like learning, generalization and adaptability, optimization may be another constraint, which may need to be satisfied. Genetic algorithms are ideal for satisfying the optimizing learning and generalization characteristics of soft computing mechanisms like neural networks.

Postprocessing Phase Adapter

The goal of the Postprocessing Phase Adapter is to explain and validate the outcomes made by the decision phase adapter. The validation task may involve decision instance result validation (subjected to task constraints like provability and reliability). For example, in a face recognition application, the actual faces and facial movements as determined by the decision phase adapter need to be validated by the user. In a control system application, feedback from the environment established whether the selected/executed control action has produced the desired results. In an alarm processing application, the operator may instruct the system or computer-based artifact to explain how certain components in the network have been identified as faulty.

The validation task can be accomplished by perceptual and/or hard/soft computing mechanisms. In a real time alarm processing application, for example, a power system control center operator may validate a decision made in the decision phase by using graphic display of the power network and by querying the system on the fault model of the faulty component. In fact, postprocessing phase represents logic and provability of our conscious reactions to the external environment. In problems where logical and provable models of the various problem components exist, the postprocessing phase adapter can be performed without need of any feedback from its environment to accomplish the task. However, in problems where logical and provable models of the various problem components do not exist or are too complex to be built, it is necessary to have feedback from its environment. This can be done with the help of the human agent/user or other external agent especially to validate the decision made in the decision level.

Order Monitoring System: An Application Example

The TPSAs have been applied to develop the Order Monitoring System (OMS) for an electrical manufacturing company to help marketing engineers to keep track the different stages of an order like indent preparation, costing, reservation, production status, etc. (Kitjongthawonkul and Khosla, 1999). The motivation for order monitoring is based on the coordination problems between marketing engineers and manufacturing and consequential delay in shipment and loss of future orders from customers. The OMS is designed to better monitor and identifying the bottlenecks in manufacturing and delivery of customer orders.

The task context in which the stakeholders of the OMS (e.g. marketing manager and marketing engineers) solve their problems are identified and captured in a generic form using the five phase adapters. Figure 2

shows an example of the problem solving adapter in control phase. These problem solving adapters have been realized on software model through the integration with software modeling paradigms of agent and object. Agents are used to model various tasks associated with the adapters. The features of agent-orientation like percepts, goals/tasks, reactive, collaboration, etc. facilitate each problem solving agent to achieve its goals/tasks. Objects and classes are used to structure the represented features used by problem solving agents.

Figure 2. Control Phase Adapter Construct for Accomplishing Indent Preparation Task of the OMS

Phase:	Indent preparation control
Goal:	A. Monitor indent preparation of Capacitor, Motor, Transformer, Lighting B. Monitor changes in indents
Precondition:	A. Order processing agent initialized B. Product catalog & related Customer order loaded
Tasks:	A. Check prices B. Check indent completing C. Indent dispatch D. Conflict resolution - rules for determining mismatch between order specification & technical drawings - rules for mismatch between customer's credit rating & payment schedule
Task Constraints:	A. Products need to be classified according to Manufacturing division format B. Indents need to be prepared as for Manufacturing division format C. Scalability - new products can be added
Domain Model:	Structural model of Manufacturing divisions
Represented Features:	Preparing indents based on type of structural data (e.g. 66KV, 220KV, 400KV Distribution transformer)
Psychological Scales:	Ordinal (e.g. Different products)
Representing Dimensions (Perceptual):	Size (e.g. Transformer rating)
Knowledge Engineering Strategy:	Top down
Methods:	Symbolic rule based

In the context of OMS, for example, the goals of Indent preparation agent are: a) to monitor the preparation of Capacitor, Motor, Transformer indents, b) to monitor changes in the indents, c) to coordinates the information with indents. To accomplish such goals, the indent preparation agent needs to engage a number of sequences including establishing communication constructs with other agents and objects like Capacitors indent agent, Motor indent agent, Transformer indent agent, and Order object, in forms of request, inform, and command. The messages are flown through interface channel and sensed by the indent preparation agent which in turn mapping this percepts into a series of actions such as identifying product/article, checking order type, and checking indents. These sort of actions establish a clear relationships between tasks of the indent preparation agent and services offered by linked to this agent in OMS object structure. These objects are Customer, Order, Product technical document, and Indent form, Product catalog. More details of modeling and design of OMS can be found in (Kitjongthawonkul and Khosla, 1999).

Conclusions

The role of artificial intelligent in information systems has arisen because conventional information systems cannot cope with such complex and fuzzy problems. These problems range from psychology, linguistics, business to engineering. One thing common in these problems is that all of them address one or more aspects of human cognition, namely, information processing, knowledge representation and learning.

Our task-based problem solving adapters has recognized such aspirations and reflected them through the integration of various perspectives like neurobiological perspective, expert systems perspective, physical systems perspective, form of knowledge perspective, symbolic perspective, artificial neural networks perspective, learning perspective, and user perspective. The characteristics of the task-based problem solving adapters are reflected in software model by employing the technological artifact of objects and agents (not shown in this paper, however, please refer to (Kitjongthawonkul and Khosla, 1999) for more information). The problem solving adapters consist of five different phase adapters, namely, preprocessing phase adapter, decomposition phase adapter, control phase adapter, decision phase adapter, and postprocessing phase adapter. We have outlined an example of the application of the problem-solving adapter in Sales and Marketing.

References

Angele, J., Fensel, D., Landes, D. and Studer, R. "Developing Knowledge-Based Systems with MIKE," *Journal of Automated Software Engineering*, (5)**,** 1998, pp. 389-418.

Chandrasekaran, B., Johnson, T. R. and Smith, J. W. "Task-Structure Analysis for Knowledge Modeling," *Communications of the ACM*, (35)**,** 1992, pp. 124-136.

Fensel, D. and Groenboom, R. "Specifying Knowledge-Based Systems with Reusable Components," In *Proceedings of the 9th International Conference on Software Engineering & Knowledge Engineering (SEKE-97)*, Madrid, Spain, 1997.

Fensel, D. and Groenboom, R. (1999) "A Software Architecture for Knowledge-Based Systems," www.cs.vu.nl/~dieter/publications99, (Current Dec. 16, 1999).

Gamma, E., Helm, R., Johnson, R. and Vlissides, J. *Design Patterns*, Addison-Wesley Publication, 1995.

Khosla, R. and Dillon, T. "Perspectives for Integration of Artificial Neural Networks and Expert Systems," In *IEEE*

Australia-New Zealand Conference on Information Systems, Perth, Australia, 1993, pp. 624-628.

Khosla, R. and Dillon, T. "Cognitive and Computational Foundations of Symbolic-Connectionist Integration," In *ECAI'94 Workshop Proceedings on Symbolic-Connectionist Processing*, Netherlands, Holland, 1994.

Khosla, R. and Dillon, T. *Engineering Intelligent Hybrid Multi-Agent Systems*, Kluwer Academic Publishers, Norwell, MA 02061, USA, 1997.

Kitjongthawonkul, S. and Khosla, R. "Modeling Information Systems Using Objects, Agents, and Task-Based Problem Solving Adapters," In *Proceedings of the 10th Australasian Conference on Information Systems*, Wellington, New Zealand, 1999, pp. 474-483.

Schreiber, A. T., Wielinga, B. J. and Breuker, J. A. *KADS: A Principle Approach to Knowledge-Based System Development*, Academic Press, London, 1993.

Schreiber, G., Wielinga, B., de Hoog, R., Akkermans, H. and de Velde, W. V. "CommonKADS: A Comprehensive Methodology for KBS Development," *IEEE Expert*, 1994, pp. 28-37.

Steels, L. "Components of Expertise," *AI Magazine*, (11**),** 1990, pp. 28-49.

Zhang, J. and Norman, D. A. "Representations in Distributed Cognitive Tasks," *Cognitive Science*, (18)**,** 1994, pp. 87-112.