Knowledge Warehouse:
An Architectural Integration of Knowledge Management, Decision Support, Data Mining and Data Warehousing

Hamid R. Nemati
University of North Carolina at Greensboro
Greensboro, NC 27412

David M. Steiger
University of North Carolina at Greensboro
Greensboro, NC 27412

Lakshmi S. Iyer
University of North Carolina at Greensboro
Greensboro, NC 27412

Richard T. Herschel
St. Joseph's University
Philadelphia, PA 19131
Biography

Hamid R. Nemati is an assistant professor in the Information Systems and Operations Management Department at the University of North Carolina at Greensboro. He has also served on the faculty of College of Business Administration at Georgia State University. He received his Doctorate degree from the University of Georgia. He also holds a B. S. in Physics and an M.B.A from the University of Massachusetts. He has extensive professional experience as an analyst and has consulted with a number of major corporations. His current research and publications are in the areas of Decision Support Systems, Data Warehousing, Data Mining and Knowledge Management.
Address: 473 Bryan Bldg, Bryan School of Business and Economics, UNCG, Greensboro, NC 27412
Phone: (336)334-4993; Fax: (336)334-4083; email: nemati@uncg.edu

David M. Steiger is an associate professor in the Information Systems and Operations Management Department at the University of North Carolina at Greensboro. He received his B.S. in Electrical Engineering and M.B.A. from the University of Texas at Austin and a Ph.D. in Management Science/Information Systems from Oklahoma State University. Between his M.B.A. and Ph.D. degrees, he spent 15 years in various analysis and managerial positions in industry, applying the concepts of information systems and management science. Professor Steiger’s research interests include knowledge management, decision support systems, model analysis systems, and inductive artificial intelligence technologies. His articles have appeared in Information Systems Research, Management Science, Journal of Management Information Systems, Interfaces, ORSA Journal on Computing, European Journal of Operational Research and Annals of Operations Research.
Address: 479 Bryan Bldg, Bryan School of Business and Economics, UNCG, Greensboro, NC 27412
Phone: (336)334-4985; Fax: (336)334-4083; email: steiger@uncg.edu

Lakshmi S. Iyer is an assistant professor in the Information Systems and Operations Management Department at the University of North Carolina at Greensboro. She has also served on the faculty of the University Of Dayton and the University of Georgia. She obtained her B.S. in Chemical Engineering from Bangalore University in India, M.S. in Industrial Engineering from the University of Alabama at Tuscaloosa, and Ph.D. from the University of Georgia. Her research interests include Group Support Systems, Knowledge Management, Data Mining, Electronic Commerce, and Cluster Analysis. Her research work has been published in the Encyclopedia of ORMS, Journal of Information Technology and Management, Annals of OR, and Industrial Management and Data Systems.
Address: 482 Bryan Bldg, Bryan School of Business and Economics, UNCG, Greensboro, NC 27412
Phone: (336)334-4984; Fax: (336)334-4083; email: lsiyer@uncg.edu

Richard T. Herschel is an Associate Professor of Management & Information Systems at St. Joseph's University in Philadelphia. He received his Ph.D. in MIS from Indiana University, an M.A.S. in Business Management from John Hopkins University, and a B.A. in Journalism from Ohio Wesleyan University. Dr. Herschel's research interests include knowledge management, organizational communication, decision making, and group support systems. His articles have appeared in Information Strategy: The Executive's Journal, Computers in Human Behavior, and Small Group Research. He has also published a book entitled Organizational Communication in a Technological Society.
Address: 5600 City Avenue, Erivan K. Haub School of Business, St. Joseph’s University, Philadelphia, PA 19131-1395, USA
Phone: (610)660-1630; Fax: (610)660-1229; email: herschel@sju.edu
Knowledge Warehouse: 
An Architectural Integration of Knowledge Management, Decision Support, Data Mining and Data Warehousing

ABSTRACT

Data warehousing provides an infrastructure that enables businesses to extract, cleanse, and store vast amounts of corporate data. The basic purpose of a data warehouse is to empower the knowledge workers with information that allows them to make decisions based on a solid foundation of fact. However, only a fraction of the needed information exists on computers, while the vast majority of needed intellectual assets exists in the heads of the people and in their networks of associates and sources in the form of their collective knowledge. What is needed is a new generation of knowledge enabled systems that provide the infrastructure needed to capture, enhance, store, organize, create, and disseminate not only data and information but also knowledge. The purpose of this study is to propose, as an extension to the data warehouse model, a Knowledge Warehouse (KW) architecture that will not only facilitate the capturing and coding of knowledge but also enhance the retrieval and sharing of knowledge across the organization. The primary goal of a KW is to provide the decision-maker with an intelligent analysis platform that enhances all phases of the knowledge management process.

Key words: Knowledge warehouse, Data warehouse, Knowledge management, Decision support systems, Data mining, Intelligent analysis, Model analysis.
1. Introduction

Over the last three decades, the organizational role of information technology has evolved from that of efficient processing of large amounts of batch transactions to providing information in support of decision making activities. This paradigm shift is reflected in the fact that in the 1970’s most IS organizations changed their name from "data processing" to "management information systems” (Berson and Smith, 1997). The market forces in the 1990’s have accelerated the pace of this shift in focus. With the emergence and evolution of enterprise network computing and client/server architecture as the dominant computing paradigm and a set of significant new information processing concepts and tools, it is now possible for organizations to provide all the key people within the enterprise with access to needed information and the means to utilize them in a decision support context.

Data Warehousing, an integral part of this process, provides an infrastructure that enables businesses to extract, cleanse, and store vast amounts of corporate data from operational systems for efficient and accurate responses to user queries. A data warehouse empowers the knowledge workers with information that allows them to make decisions based on a solid foundation of fact (Devlin, 1997). However, only a fraction of the needed information exists on computers, while the vast majority is scattered throughout the organization as intellectual assets among its knowledge workers (Nonaka and Takeuchi, 1995). Hence, a data warehouse does not necessarily provide adequate support
for knowledge intensive queries in an organization. What is needed is a new generation of knowledge enabled systems that provide the infrastructure needed to capture, enhance, store, organize, create, and disseminate not only data and information but also knowledge which may include objects, mathematical models, metamodels, case models, text streams, etc. The existing enterprise-wide information delivery systems provided in a data warehouse can be leveraged and extended to create a knowledge warehouse (KW). This warehouse can be used as a clearinghouse of knowledge to be used throughout the organization by the knowledge workers to support their knowledge intensive decision-making activities. The KW can also evolve over time by enhancing the knowledge it contains.

Just as in a data warehouse environment where data mining techniques can be used to discover untapped patterns of data that enable the creation of new information, by extension then, use of technologies such as data warehousing and data mining are potentially critical technologies to enable the knowledge creation and management processes (Berson and Smith, 1997). However, for an effective KW to become a reality, different forms of knowledge (implicit and explicit) need to be captured, codified, and cataloged. Moreover, the codified knowledge must maintain its usability and meaning and contribute to the generation of additional knowledge.

The purpose of this study is to propose, as an extension to the data warehouse model, a KW architecture that will not only facilitate the capturing and coding of knowledge but also enhance the retrieval and sharing of knowledge across the organization. The primary goal of a KW is to provide the decision-maker with an
intelligent analysis platform that enhances all phases of the knowledge management process.

The remainder of this paper is organized into five additional sections. In Section 2 we provide some KM background. In Section 3 we discuss how Decision Support Systems (DSS), Group Support Systems (GSS), and Data Mining can enhance KM. We then present the foundations for the goals and requirements for a KW in Section 4. In Section 5 we present the proposed KW architecture and finally, in Section 6 we provide our summary and conclusions.

2. Knowledge Management

Knowledge management focuses on the harnessing of intellectual capital within an organization. The concept recognizes that knowledge, not simply information, is the greatest asset to an institution. Knowledge as a concept is distinguishable from data and information; e.g., Earl and Scott (1998) make the distinction based on several criteria and state that knowledge is more complex, subtle, and multivariate than information.

There are two forms of knowledge: explicit knowledge and tacit (implicit) knowledge. Explicit knowledge is defined as knowledge that can be expressed formally and can, therefore, be easily communicated or diffused throughout an organization (Nonaka and Takeuchi, 1995). Explicit knowledge is codified in strings of symbols (e.g., words, numbers, formulas), physical objects (e.g., equipment, documents, models), rules, routines, or standard operating procedures (Choo, 1998).

Implicit (tacit) knowledge is knowledge that is uncodified and difficult to diffuse. Choo (1998) states that implicit knowledge is learned through extended periods of
experiencing and doing a task, during which the individual develops a feel for and a capacity to make intuitive judgements about the successful execution of the activity. Choo suggests that implicit knowledge is vital to an organization because organizations can only learn and innovate by somehow leveraging the implicit knowledge of its members. Despite it being uncodified, implicit knowledge can be and is regularly taught and shared. It can be learned by example. The ability to effectively act on knowledge coupled with the inability to articulate an explanation is exemplary of tacit knowledge.

Nonaka and Takeuchi (1995) view implicit knowledge and explicit knowledge as complementary entities (see Figure 1). They contend that there are four modes (Socialization, Externalization, Combination, and Internalization) in which organizational knowledge is created through the interaction and conversion between implicit and explicit knowledge.

[Insert Figure 1 here]

For a more comprehensive review of tacit to explicit knowledge conversion please refer to Nonaka and Takeuchi (1995) and Davenport and Prusak (1998).

3. DSS, GSS, and Data Mining Support of Knowledge Management

DSS, GSS, and Data Mining can be used to enhance knowledge management and its three associated processes: i.e., tacit to explicit knowledge conversion, explicit knowledge leveraging, and explicit to tacit knowledge conversion. These process enhancements are discussed individually below.
Converting Tacit Knowledge to Explicit Knowledge:

In tacit to explicit knowledge conversion, the literature of knowledge acquisition in expert systems-based DSS is well established (Kidd, 1987). Knowledge acquisition involves employing various techniques to elicit information (verbal and/or quantitative) from the knowledge worker, interpreting this information (more or less skillfully) in order to infer the underlying knowledge and reasoning processes, and using this interpretation to guide the construction of some model or language that describes (more or less accurately) the knowledge worker's performance (Johnson, 1985). Knowledge acquisition is a crucial state in the development of an expert system since the utility and power of the resulting system depends almost totally on the quality of the underlying knowledge.

DSS can also enhance the tacit to explicit knowledge conversion through the specification of mathematical models. Specifically, in the model building process (e.g., in linear programming models) the knowledge worker is asked to explicitly specify the goal or objective of the model, the decision variables, and perhaps the relative importance of the decision variables (in the case of a goal programming model). The knowledge worker also explicitly specifies the model constraints in terms of the decision variables, and estimates both the numerical coefficients of the decision variables in each constraint and in the objective function, as well as the right hand side constraint values. These model components (decision variables, coefficients, constraints and objective functions) reflect the tacit knowledge built up over the years of being immersed in the decision making environment. Such models may be stored in the form of a set of explicit mathematical inequalities (Fourer, 1983), as annotated graphs of arcs and nodes in
network flow models (Jones, 1990; Steiger, Sharda, and LeClaire, 1993), as a set of arc
descriptions (Kennington and Helgason, 1980) or as a condensed canonical model
formulation with links to relational tables for instantiation (Steiger and Sharda, 1993).

DSS can also enhance the tacit to explicit knowledge conversion by eliciting one
or more what-if cases (i.e., model instances) that the knowledge worker wants to explore.
That is, as the knowledge worker changes one or more model coefficients or right hand
side values (e.g., in a linear programming model) to explore its effect on the modeled
solution, s/he is estimating ranges of those parameters/values that better reflect the actual
decision making environment represented by the model. That is, the knowledge worker
is converting the tacit knowledge of various historical situations and/or decisions into
explicit knowledge that can be shared with other workers and leveraged to enhance
decision making. These multiple, related model instances can be stored, along with their
associated solutions, as tuples in a relational database, as objects in an object-oriented
database, or as sparse matrices.

A final source of tacit to explicit knowledge conversion occurs in the
brainstorming of GSS. GSS brainstorming sessions solicit the participants’ ideas and
concerns about a stated problem. The ideas are then anonymously relayed (without
evaluative comments) to the other participants for their enhancements and modifications,
generating a continual stream of related and tangential ideas directed toward solving the
stated problem. At some point of time, the session leader directs the participants to stop
generating new ideas and start evaluating, again anonymously, a specific idea. The
evaluations are given in the form of short lists of things the participant likes about the
idea and any concerns that may hamper implementation. The group then addresses the
concerns, evolving toward a valid and implementable solution to the stated problem. The ideas, likes and dislikes of GSS brainstorming sessions are stored as text streams for further processing and future use.

**Knowledge Leveraging: Converting Explicit Knowledge to New Knowledge**

Once a knowledge worker's tacit knowledge is converted to explicit knowledge and stored in an appropriate (computer readable) form, it can be leveraged by making it available to others who need it. In addition, analyzing explicit knowledge to produce new knowledge can further it. For example, explicit knowledge generated from GSS brainstorming sessions and stored as text streams can be analyzed by data mining software routines to provide key words, related concepts, clusters of similar ideas, etc.

However, since knowledge may appear in non traditional format (i.e., textual), mining it requires special considerations. The most frequently used data mining techniques require the use of highly structured data. These are the data that have a finite, well-defined set of possible values. Textual data is highly unstructured and not well behaved at all. Words can only be understood in their context. To put it another way, multiple words may convey the same meaning while a given word may have multiple meanings. Overcoming the problem unique to textual data is not a trivial task. Therefore, in order for data mining techniques to be useful in extracting knowledge from textual data that represents corporate knowledge, alternative approaches are needed.

The traditional approach to data mining from textual documents is based on the use of keyword searches. Most of these methods search the document and count the number of occurrences of a given word in that document. The keyword search approach to extracting concepts and information from textual documents has many limitations.
Artificial Intelligence (AI) based learning methods have shown to be very effective in improving the accuracy of the keyword search based methods of knowledge extraction from textual documents (Kupiec, Pedersen, and Chen, 1995; Jang and Myaeng, 1997). These methods use a set of keywords supplied by a knowledge generator to train the algorithm to extract information from textual documents.

Using keywords and key phrases may not be appropriate in extracting knowledge from textual documents. An alternative approach to summarization-based methods that extract keywords is information extraction (IE). Information extraction methods find specific information in a textual document according to a predefined set of rules and guidelines which are specific to a given topic area (Turney, 1997). However, since information extraction systems are specific to a single topic, they tend to be very labor intensive to develop and they are very project dependent. To address this problem a number of researchers and tool developers have use AI techniques. These learning methods use predefined information templates to extract knowledge. Soderland and Lehnert (1994) trained different decision trees to fill with information different parts of a template. AI techniques are also used in a number of other tools developed for learning templates. Examples include CRYSTAL (Soderland and Lehnert, 1994), RAPIER (Califf, 1997) and AutoSlog (Riloff, 1996).

On the other hand, explicit knowledge stored in the form of instances of a mathematical model (what-if cases) can be leveraged via deductive and/or inductive model analysis systems. Here, deductive model analysis systems (DMAS) apply paradigm- or model-specific knowledge to a single instance of the model, addressing such questions as "Why is this the solution?," "Why do the solutions to two model
instances differ so much?,” or, in the case of linear programming models, "Why is this instance infeasible?" Deductive model analysis systems exist for each of the three major modeling paradigms: linear programming, simulation and spreadsheet models (Greenberg, 1993; Kimbrough et al., 1990, 1993; Kosy and Wise, 1984).

Inductive model analysis systems (IMAS) operate on a set of many related model instances that represent historical situations familiar to the knowledge worker and/or several (if not many) what-if cases. The primary goal of IMAS is to help the knowledge worker develop insight(s) into the business environment represented by the model, insights spawned by key factor identification, simplified meta-model generation, etc (Sharda and Steiger, 1996). Inductive analysis systems are distinguished from deductive analysis systems by both the required input and the type of processing logic employed. That is, IMAS operate on many model instances and apply inductive analysis technologies (e.g., statistical analysis, the group method of data handling, genetic algorithms) to extract new knowledge (Piela et al., 1991; Saltelli and Homma, 1992; Saltelli and Marivoet, 1990; Sharda and Steiger, 1996; Stephanopoulos et al., 1990a, 1990b), whereas DMAS operate on one or two model instances and apply deductive analysis based on known paradigm- or model-specific knowledge (Steiger, 1998).

Another form of explicit knowledge leveraging is found in case-based reasoning (CBR). CBR is characterized by the knowledge worker making his or her inferences and decisions based directly on previous cases recalled from memory (Kolodner, 1987). That is, the knowledge worker tries to avoid, or reduce, the potential for failure by recalling previous similar failures and avoiding the associated pitfalls or changing key factors in those previous failures. He or she can also speed the decision making process by not
having to generate and evaluate all alternatives from scratch. Finally, the attributes of past cases can be generalized to improve decision making in the future (Hammond, 1988). CBR requires case storage capabilities (perhaps in the form of frames), a filtering of cases for relevancy of key factors, a sophisticated recall capability based on key factors, and a case-based inference capability based on those parts of the previous case which are appropriate for the current decision.

Learning New Knowledge: Converting Explicit Knowledge to Implicit Knowledge.

DSS/GSS can also provide valuable aids in internalizing explicit and new knowledge; i.e., in helping the knowledge worker to learn. One mode of internalizing explicit and/or new knowledge is through modifying the internal mental model that a knowledge worker uses to serve as a performance guide in specified situations. Such mental model modifications may occur in the building of a DSS model. For example, a knowledge worker might modify his or her mental model based on the discovery of new relationships between key factors during model development, the development of counterexamples of assumed relationships, and/or the acknowledgement of fallacies in deductive logic uncovered during modeling.

Another source of mental model modification may be the adjustment of the relative importance of various components of the mental model. DMAS can be helpful here; e.g., sensitivity analysis offered in some types of mathematical modeling (e.g., linear programming) can be used to help the knowledge worker understand and alter the relative importance of key parameters and how incremental changes in one parameter can affect the solutions (Greenberg, 1993).
A third source of mental model modification may come from the inductive analysis of multiple, related solved model instances. For example, if several model instances are specified in which two or more uncertain parameters are varied over appropriate ranges, an analysis of the multiple solved instances may provide new knowledge concerning not only the relative importance of key factors, but also how the key parameters interact, perhaps in a nonlinear fashion, to affect the model solution (Sharda and Steiger, 1996; Wagner, 1995).

Another aid in internalizing explicit knowledge occurs in expert system-based DSSs. Here the explanation capability of ES provides understandable and amplifying rationale(s) for a recommended course of action.

Still another way that DSS can help the knowledge worker internalize explicit knowledge is to enhance his understanding of the knowledge. Understanding, according to one theory of learning (Perkins, 1986), consists of knowing three things: 1) the purpose of the analysis, or what the knowledge worker wants to understand, 2) a design, or hypothesized (mathematical) model, of the process/system to be understood, and 3) arguments about why the design serves the purpose. These arguments can be of three different types. Evaluative arguments focus on the accuracy, sufficiency, necessity and consistency of a proposed model and its components. Simple explanatory arguments focus on explaining or defining the elements of the model and/or state what each element contributes. And, finally, deep explanatory arguments seek to explain a design or model in terms of basic underlying principles; e.g., the underlying formulae and interconnections between the balance sheet, income statement and funds flows statement in a business financial problem. The advantages of deep explanatory arguments include
their power of abstraction, generalization, and insight generation, resulting from the application of basic principles and relations applicable to the current analysis. The basic disadvantage of deep explanatory arguments is the difficulty of defining, storing and retrieving relevant basic principles, relating these basic principles to the model, and successfully communicating the relationships to the knowledge worker (Perkins, 1986; Steiger, 1998). Thus, this type of analysis requires not only the storage of multiple, related model cases, but also the storage, retrieval and processing of the purpose and underlying principles potentially applicable to the specific decision making environment, stored as text streams and referenced through key words and context.

4. Goals and Requirements for Knowledge Warehouse

The goal of a KW is to provide the decision maker with an intelligent analysis platform that enhances all phases of the knowledge management process. Several comments can be made to further amplify and explain the KW goal.

First, this goal assumes that the user of the KW is the decision maker. That is, we assume that the user is not an expert in the various technologies used to enhance knowledge management, but rather is an expert in the decision making field. Throughout the remainder of this paper, the terms 'decision maker' and 'user' refer to the actual decision maker.

Second, an intelligent analysis platform is defined as a PC-based platform that makes available to the decision maker an array of analytical tools, each of which utilizes various technologies to aid the articulation, integration, understanding and internalization of knowledge management. The purpose of including artificial intelligence is to amplify
the cognitive capabilities of the decision maker in converting tacit knowledge into explicit knowledge, integrating this explicit knowledge by analyzing it to detect new patterns and relations, and understanding the new knowledge by providing analogs and explanations.

Third, understanding is defined by Perkins's (1986) theory of learning as the knowledge of three things: 1) the purpose of this analysis or what the decision maker is trying to understand, 2) a model (or design) of what we want to understand; i.e., its structure, components, properties, relations, etc., and 3) various arguments about why the design serves the purpose. Arguments can be thought of as evidence showing that the model or a hypothesized metamodel does, or does not, support the purpose. There are three general types of arguments: 1) evaluative arguments that focus on comparing and evaluating two or more competing models to find which is superior with respect to accuracy, simplicity, conceptual validity, sufficiency, necessity and/or consistency, 2) simple explanatory arguments that explain or define the elements of the model and/or state what each element contributes, and 3) deep explanatory arguments that explain a model in terms of basic underlying principles or knowledge of the modeled system (Perkins, 1986). These arguments provide a primary source of insight and understanding in knowledge management.

Fourth, the process of data mining is often able to find important information that would not have been found using standard analysis techniques such as optimization models or regression analysis. Data mining helps uncover important facts hidden in warehouses of data. That is, information is often revealed by data forming patterns that enable sense making and knowledge creation. The knowledge gained can then be used to
aid decision-makers in determining organizational action (Skyrme, and Amidon, 1997). The data mining process can be divided into two distinct categories - verification driven and discovery driven. In verification driven data mining, a prior hypothesis is formed about the nature of relationships among data. The result of the mining process is then used to reach a conclusion regarding the validity of this hypothesis. Discovery driven data mining starts without any preconceived notion regarding the nature of relationships among data. It is the task of the data mining system to find significant patterns in the data. Two sub-categories of discovery-driven data mining are supervised learning (classification) and unsupervised learning (clustering) (Fayyad et al., 1996). Supervised learning is equivalent to learning with a teacher and involves building a model for the specific purpose of optimally predicting some target field in the historical database (the value of which can be used to gauge whether the right or wrong prediction was made). In contrast, unsupervised learning does not have any well-defined goal or target to predict (and, thus, no particular supervision over what is a right or wrong answer). Techniques such as clustering and detection of association rules fall into the category of unsupervised learning (Berson and Smith, 1997).

And finally, in a mathematical modeling (DSS) environment, converting tacit to implicit knowledge may take the form of specifying and creating mathematical models and associated what-if cases; integrating explicit knowledge may take the form of mining multiple, related model cases for new knowledge (e.g., simplified metamodels, key parameters, new relations); and understanding new knowledge may take the form of generating arguments, especially from underlying principles, and/or suggesting analogies (Steiger, 1988).
This goal suggests three functional requirements for KW: 1) an ability to efficiently generate, store, retrieve and, in general, manage explicit knowledge in various forms, including mathematical models, solved model instances, text streams, film clips, etc., 2) an ability to store, execute and manage the analysis tasks and their supporting technologies, and 3) a computer-assisted ability to generate natural language arguments concerning both the comparable validity of the models, metamodels and relations produced by analysis tasks, and how this new knowledge relates to the decision maker's purpose. Each of these three functional requirements is discussed individually below.

**Knowledge Storage and Retrieval**

The KW must provide the same services for knowledge that a data warehouse provides for data. That is, for knowledge stored in the form of models and solved model instances, the KW is required to efficiently generate, store, retrieve and manipulate multiple solved model instances. Further, each instance must be tied (logically) to its associated model in order to simplify the generation of additional related instances from the same model and to enhance storage and retrieval efficiency; i.e., two related instances normally exhibit a high degree of commonality in parameter values and can thus be stored and retrieved more efficiently if logically related.

**Analysis Task Management**

The analysis of knowledge is not a simple process. Specifically, an analysis frequently utilizes various inductive and deductive technologies; e.g., neural networks, GMDH, statistics, production rules, genetic algorithms, case-based reasoning. Each task has its own requirements with respect to 1) input instances (e.g., the number and domain coverage of related instances), 2) execution parameters required by the analysis
technologies (e.g., step-size and node architecture for neural networks, the complexity factor and number of layers for GMDH), and 3) output format (e.g., weight matrix, polynomial equations, production rules, quality measures). Further, some analysis technologies are limited to specific knowledge paradigms, whereas others are equally applicable to all paradigms; e.g., the explanation task implemented in ROME/ERGO (Kosy and Wise, 1984) is limited to spreadsheet models, whereas the causation task implemented in INSIGHT is applicable to all mathematical models (Sharda and Steiger, 1996).

KW must efficiently support the storage, initiation, execution and management of knowledge analysis tasks and the associated implementation technologies. Specifically, the analysis tasks and the associated technologies must not only be stored in KW, but also be logically tied to the appropriate knowledge paradigm, if required. Thus, the sensitivity analysis task implemented in ANALYZE must be tied to linear programming models.

Argument Generation

Arguments are pieces of evidence that show a specific hypothesized model does, or does not, support its purpose. In mathematical model analysis, hypothesized models refer to either the metamodels generated by various technologies employed in an analysis task, or the mathematical relations hypothesized and specified by the decision maker based on human expertise and/or mathematical manipulations (Steiger, 1998).

In preparing evaluative arguments, KW must compare one or more hypothesized metamodels based on internal evaluation measures such as accuracy of a metamodel over some set of instances, model complexity with respect to the number of terms or the order of the metamodel's polynomial, or some combination of both (Barron, 1984).
In preparing simple explanatory arguments, KW must provide a link to the model dictionary and data dictionaries, both a part of the MMS knowledge base (Dolk and Konsynski, 1984). Information from these dictionaries can be combined with a natural language processor to generate English production rules for explaining arguments (Greenberg, 1994; Parsaye et al., 1990). KW should also support simple explanations based on a base case and examples generated to demonstrate the response of the metamodel to both small and large changes in input parameter values.

In preparing deep explanatory arguments, KW must support the interactive collection, storage and application of basic, underlying principles relevant to the modeled environment. For example, to generate deep explanatory arguments for financial models, KW could obtain and store the formulae behind balance sheets, income statements and funds flow statements, and the interconnections between the basic components of these financial statements. KW must also provide a link to the model/knowledge base to retrieve key terms and common synonyms for use in the deep explanatory arguments.

5. Knowledge Warehouse Architecture

These goals and requirements of a KW can be implemented via an extension of the data warehouse architecture. The proposed extension consists of three major components: 1) a knowledge base management system module, 2) a knowledge analysis workbench, and 3) a communication manager. The KW also includes a feedback loop to enhance its own knowledge base with the passage of time, as the tested and approved results of knowledge analysis is fed back into the KW as an additional source of knowledge. The details of the components are described below.
**Knowledge Base Management System Module**

One of the primary components of the KW architecture is an object-oriented knowledge base management system (KBMS) that integrates the knowledge base, model base, and analysis tasks. A KBMS is a system that manages the integration of a wide variety of knowledge objects into a functioning whole producing, maintaining, and enhancing a business. These knowledge objects include data, text streams, validated models, metamodels, movie clips, animation sequences, as well as the software used for manipulating them. The KBMS is implemented in an object-oriented environment.

The KBMS must not only manage data, but all of the objects, object models, process models, case models, object interaction models and dynamic models used to process the knowledge and to interpret it to produce the knowledge base.

Object-specific knowledge is stored as part of the appropriate object. The specific form of the knowledge storage mechanism may include frames, semantic nets, rules, etc. Stores of knowledge include, but are not limited to, metadata, metamodels and instances of metamodels. For example, a model's purpose is stored as part of the associated model whereas the basic underlying principles may be stored with a more general model class.

Messages sent to the objects are generic in form, independent of the method's technology. If additional information is required to execute a specified method, a message is sent to other appropriate object(s).

The object-oriented database technology provides several advantages for this application. One advantage is that a existing knowledge is integrated with 1) it own metaknowledge, 2) examples or instances of the knowledge, and 3) methods including
the analysis tasks. This enhances storage efficiency; e.g., if the knowledge is in the form of a model and its instances, related instances may differ from a base case by only one or two parameter values and the solution vector, and all common parameter values can be inherited from the base case or other parent instance for storage efficiency. A second advantage is that some analysis tasks (e.g., the linear programming sensitivity analysis task in ANALYZE) can be logically tied to a specific class of models, whereas other analysis can be tied to a super class of all models and be independent with respect to modeling paradigms. A third advantage is that method overloading allows a single user-specified command to call several different implementations of a given task and apply the appropriate technology to different forms of knowledge; this reduces the cognitive burden on the decision maker by providing him/her with independent execution calls (i.e., messages) for all analysis tasks. It also provides a primary prerequisite for effective management of technology; i.e., overloading, in conjunction with encapsulation, makes the changing of implementation technologies transparent to the user.

Knowledge Analysis Workbench

The second primary component of the KW architecture is the knowledge analysis workbench. The analysis workbench handles all interaction with the analysis tasks, including task control, and argument generation, and management of technology.

The task controller handles all requests for data and run-time interactions (e.g., complexity factors in GMDH algorithms, step sizes in neural networks) required by the analysis technologies. That is, the task controller acts as an AI-based surrogate decision maker for task interactions, shielding the real decision maker from the requirements of knowing the technologies, their nuances, interactions, etc.
The argument generation sub-module evaluates the outputs of the various analysis tasks, especially the causation task, filtering out implausible or inconsistent results based on relative measures of accuracy, simplicity, conceptual validity, sufficiency, necessity, and consistency. It then generates simple and deep explanatory arguments that (hopefully) enhance the decision makers understanding of the modeled environment. In generating these arguments, the argument generation module interfaces with the knowledge base, the instance base and model base, applying deductive knowledge, analogical reasoning, and other technologies, as appropriate.

The management of technology module manages the repository of analysis technologies. Specifically, it provides for the encapsulation of new analysis algorithms into object model classes, integration of legacy data mining applications, incorporation of new analytical models and metamodels into the object model repository, etc.

**Communication Manager**

The third primary component of the KW architecture is the communication manager. This module, which handles all analysis communication between KBMS and the user interface, includes five functional sub-modules: a knowledge engineer, what-if interface, query processor, results presentation manager, and on-line help.

The knowledge engineer sub-module is an expert system-based sub-system responsible for interacting with the decision maker to develop the purpose of the analysis and the basic underlying principles of the modeled environment. Both types of knowledge are used in the development of arguments. This knowledge may be stored in the knowledge base in the form of frames, rules, semantic nets, etc.
The what-if interface is designed to efficiently and effectively help the decision maker specify one or more what-if cases to be investigated. It includes an analogical component that is used to suggest pertinent instances by varying one or more parameter values. It also includes one or more interactive graphical displays, or summaries, of instances already available, so that the decision maker can see at a glance what has already been tried and what instance(s) might lead to additional insights. The what-if interface also includes a capability to suggest potentially valuable cases based on the planning analysis task.

The query processor provides the interface between the decision maker and the analysis task. It translates natural language, QBE or SQL-like queries specified by the decision maker into machine executable queries.

The result representation manager selects the most appropriate presentation view for each analysis result; e.g., graphics, natural language production rules, polynomials, decision trees, etc. The selection is based on a combination of the analysis task output and the decision maker's preference which, in turn, is based on an adaptable machine learning algorithm which analyzes previous uses of models and analysis tasks by the current decision maker (Liang, 1988).

The help sub-module provides the user with information concerning the model (e.g., assumptions, parameter ranges, units of measurement, internal model structure), instances (differences from base case, key decision variable values), pertinent knowledge (e.g., metamodels, metadata, basic principles, analysis purpose), and analysis tasks (e.g., applicable technology, technology description, explanatory traces of results, technical parameters used, advantages and limitations of technologies).
6. Summary and Conclusions

In this paper we have proposed a Knowledge Warehouse (KW) architecture as an extension to the data warehouse (DW) model. The KW architecture will not only facilitate the capturing and coding of knowledge but will also enhance the retrieval and sharing of knowledge across the organization. Essentially, the KW will provide the same service for knowledge that a DW provides for data. The primary goal of the KW is to provide the decision maker with an intelligent analysis platform that enhances all phases of knowledge.

In order to accomplish these goals the KW should efficiently generate, store, retrieve and, in general, manage explicit knowledge in various forms. Secondly, the KW should be able to store, execute and manage the analysis tasks and its supporting technologies. Finally, the KW should provide computer-assisted support to generate natural language arguments concerning both the comparable validity of the models, metamodels and relations produced by analysis tasks, and how this new knowledge relates to the decision maker's purpose.

The proposed KW architecture consists of an object-oriented knowledge base management system module (OO-KBMS), a knowledge analysis workbench, and a communication manager. The OO-KBMS module integrates a wide variety of knowledge objects and analysis tasks. The knowledge analysis workbench handles the interaction with the analysis tasks, including task control, argument generation, and encapsulation of new analysis algorithms into object models. The communication manager handles all analysis communication between the OO-KBMS and the user interface. The communication manager accomplishes this effectively through the use of
five functional sub-modules: a knowledge engineer, what-if interface, query processor, results presentation manager, and on-line help.

The KW will also include a feedback loop to enhance its own knowledge base with the passage of time, as the tested and approved results of knowledge analysis is fed back into the KW as an additional source of knowledge.
References


Figure 1: Conversion of tacit to explicit knowledge and vice versa.

(or A cyclical conversion of tacit to explicit knowledge)