Timothy D. Korson and Vijay K. Vaishnavi

Managing Emerging Software Technologies: A Technology Transfer Framework

To minimize adoption risks and effectively manage emerging software technologies, corporations need access to a wide range of timely information. The problem is that much of this information is not available. It is either in a form that managers cannot use, or it simply does not exist because the necessary underlying research has not been done.

Software managers are often faced with two extremes: information explosion vs. near total lack of information. In the area of object-oriented technology for example, a keyword search of our university library catalog produces references to over 3,000 articles. Yet in some crucial areas, such as metrics for managing object-technology, there is an almost total lack of information [10].

We propose a framework for addressing this problem that has four key elements: comprehensive scope; knowledge engineering; basic research; and the use of an advanced knowledge delivery vehicle to build an automated Management Support System (MSS). This framework calls for integrating and synthesizing the full range of existing pertinent information, while at the same time identifying missing knowledge. The identification of the missing knowledge drives research efforts to fill the gaps. The results of this research and knowledge engineering are then delivered to managers in a form they can directly use via the MSS.

After elaborating this framework we will describe our experience with a prototype application of the framework to an instance of an emerging software technology: the area of object-oriented analysis. Finally, we consider the further development of this concept, both in terms of full implementation and also extension of the approach to areas other than emerging software technologies.

This technology transfer framework is the basis for the newly formed Consortium for the Management of Emerging Software Technologies (COMSOFT) [26], whose founding sponsor is AT&T/Human Resources Information Systems. Technology transfer is used to refer to a variety of different activities [24]. In this article, we use the term to refer to those activities necessary to enable a corporation to apply a new software technology to its own set of internal or external products.

Need for Help in the Management of Emerging Software Technologies

Information and information processing assets are occupying an increasingly important strategic and economic role. For many corporations, survival during the next decade will depend on a competitive edge in software technology. Unfortunately, most current computer information systems environments are characterized by:

- Excessive cost for inflexible systems that do not fully support business needs
- Data and information systems not managed as an asset
- Million-dollar decisions based on inadequate information
- The inability to handle complex data adequately
- Craftsman culture, in which each system is designed and hand-coded from scratch
- Data/Information quality problems (i.e., data are incomplete, incorrect, not timely)

These problems along with attempted solutions have been exhaustively chronicled in the literature [1, 30].

Competitive reality requires rapid response to changes in the business environment. Yet additions or changes to the computer systems that support businesses cannot be made in a timely or cost-effective manner with the software technology currently in use within most organizations. Fortunately, new software technologies, such as object-oriented, are emerging that allow an organization to have more responsive and cost-effective computer systems that more fully support their business needs. The potential advantages of these new technologies are compelling [11, 27]. However, typically little is known about how to manage these technologies successfully.

We do not claim that all the problems in the management of traditional software technology have been solved. But much has been written on the subject and a number of formal resource [23] and life-cycle models [4] are in use. One reason it is difficult to manage new software technologies is because they lack these underlying theoreti-

cal models and the corresponding experience base. Some existing models may be able to be extended or calibrated to work with new software technologies [19], but in other cases a new model is needed [13].

As an example, consider object-oriented technology (OOT). It was introduced at the implementation level with languages such as Smalltalk, C++, and Eiffel. Currently, object-oriented design and analysis techniques and notations are being explored in the research community and commercially tested on real projects. Many of these small projects have demonstrated significant benefits. When companies have attempted to apply OOT to large projects, however, many of them have failed to see the same benefits because of the lack of knowledge about how to manage OOT effectively [20].

In general, expertise is first gained at the implementation level of emerging software technologies. For example, a number of programmers are experts in C++ and Smalltalk. In addition, constant research and development activity at the technical level of computer science has produced a large number of support systems and tools for implementing object-oriented systems.

At the higher levels of software analysis and design, there are fewer experts. There is less research, and theoretical knowledge is not as strong. Since there are, however, an increasing number of researchers in object-oriented design and analysis at major universities, the infrastructure is improving.

The last area of software development to be understood is management. There is little academic research in the area of managing object-oriented technology, and most of what corporations know is kept proprietary. Management is a soft science and will always require human judgement and expertise. Software technology is, however, too complex and is changing too fast for software managers to manage effectively on an *ad hoc* basis. These problems are compounded by the fact that some of the new software technologies represent a paradigm change, which has a significant impact on the organization [27]. One class of new management problems is centered in the area of reuse and corporate class libraries [17].

Any well-managed software project will have milestones, deliverables and a quality assurance process. Managers must allocate resources, budget costs, and set a development schedule. To manage the development of a complex software system a rigorous process model is useful [12]. Emerging software technologies do not have a rigorous detailed process model—if they did they would not be classified as "emerging." This presents serious problems for managers. Object-oriented technology falls into this category. The object life cycle, analysis and design techniques, and basic principles of reuse give rise to fundamental changes in management principles and techniques [16]. Large-scale projects have been completed [14] using object-oriented technology, but the experiences gained during these projects have not been synthesized and codified. Most of the data gathered during these projects remains proprietary. Information that is made public is vague and at best is presented as a collection of management heuristics. The result is that managers do not have access to the information they need.

Technology Transfer Framework

Consider the needs of software managers in charge of an object-oriented project. Currently there is no single source for the information they need. It is scattered through five years of conference proceedings, journals, technical reports, vendor literature, and corporate records. Moreover, it is at the wrong level of abstraction and often unintelligible to those outside the research community. This problem is highlighted in the special report "Scaling up: A Research Agenda for Software Engineering" [8]. This report by the Computer Science and Technology Board emphasizes the need to "Codify software engineering knowledge for dissemination and reuse" and proposes that over the long term this information should be available in an automated form.

If all of this information were readily available to managers in a form they could use, would that be sufficient? No, because there is still much empirical and theoretical research that needs to be done to create and calibrate the models referred to previously.

The four elements of the proposed framework—comprehensive scope, knowledge engineering, basic research, and the advanced knowledge delivery vehicle—are key aspects of a complete strategy for addressing these needs. In the following subsections we explain how this strategy works.

Comprehensive Scope

The proposed framework calls for a thorough search of the research literature, technical reports, conference proceedings, commercial literature, and interviews with developers, consultants, managers and researchers to identify knowledge regarding "theory; practice; standards; and management" that are relevant to the safe transition to, and effective management of, emerging software technologies (see Figure 1). Most of this knowledge is hidden behind proprietary corporate walls, or in the information explosion of research literature and commercial propaganda. Our framework requires that one organization collect the full range of pertinent information from all available sources of useful information.

Knowledge Engineering

Further, our framework requires that this organization perform the knowledge engineering necessary to synthesize, index, integrate, structure, and validate this body of

knowledge. This process is illustrated in Figure 2.

In the area of design metrics, for example, the knowledge engineering component of our framework would identify all existing work on design metrics. References to this original source material will be maintained and made available, but the real value added is the synthesis and integration of these sources into a set or sets of design guidelines that can be used for formal reviews of object-oriented designs.

On top of the structured and validated knowledge base described previously, our technology transfer framework requires the construction of frameworks and models that relate the underlying knowledge to the individual needs of organizations. This leads to the development of higher-level features within the MSS that help in customizing emerging software technologies and related management strategies. In a later section we describe an example support tool.

Basic Research

In the process of knowledge engineering, gaps in knowledge are identified. Often basic research is needed to fill these gaps. Such research is an excellent candidate for university/industry cooperation. Universities benefit from industry funding of basic research. Industry benefits in that the results of research carried out under our framework are made available at the right level of abstraction and integrated into an advanced knowledge delivery vehicle which serves as a management support system.

Because of the way research is funded and university researchers are rewarded, much basic research is driven bottom-up (i.e., by the interests of individual researchers and research institutions). Bottom-up research will bring a new technology to a certain level, but to bring a technology to the level of commercial viability takes some top-down coordination and identification of gaps. It is the knowledge-engineering component of our framework, along with the needs of corporations, that drives the identification of gaps in basic and applied research.

The previously mentioned gaps in knowledge will always be there. Any adequate technology transfer strategy must address them. The gaps exist by definition of an emerging technology. The creation of the models and frameworks needed for the inferencing capabilities of the MSS in some cases requires a metalevel of research. The results of this level of research augment the inferencing capabilities of the MSS. This is in line with the stress placed on modeling for the development of knowledge-based systems by Chandrasekaran, Johnson and Smith [7]. Since users of the MSS are involved in a feedback loop, there is constant verification and revision of the knowledge and models in the MSS.

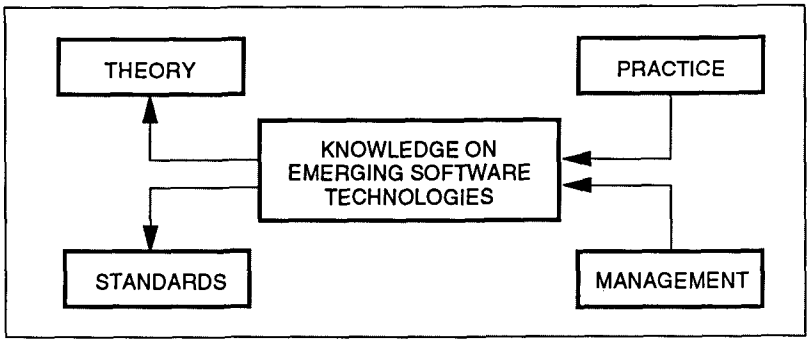
Management Support System

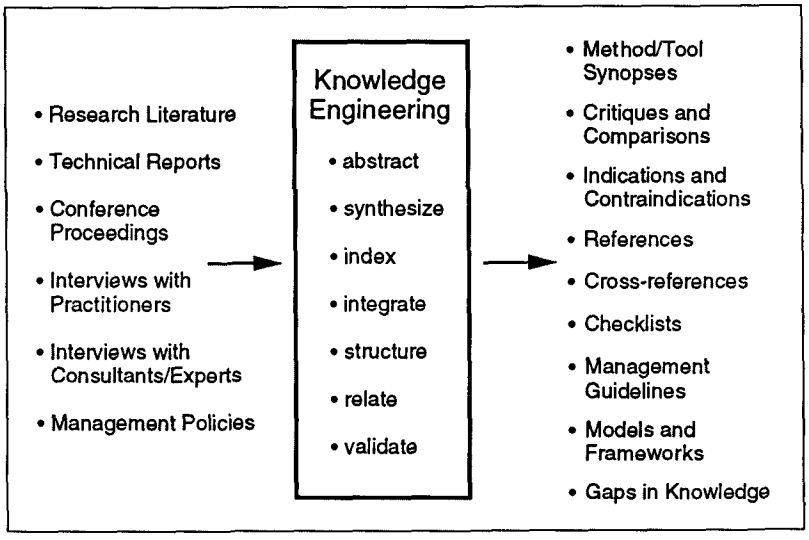
The MSS envisioned in our framework differs from a simple automated document retrieval system in several important ways. First, the content is different. The MSS contains annotated bibliographies, but not complete source documents. Instead of source documents, it contains the knowledge-engineering results (called components). These components are at several different levels of abstraction.

At the leaf level there could be a component that describes a given object-oriented analysis methodology along with such factors as tools that support it, available training, applicable metrics and standards,

Figure 1. Sources of knowledge on emerging software technologies

Figure 2. Knowledge engineering activity





available experience reports, and a summary of reviews it has received. At this level the goal is to present in a standard format and in a single component all the related information a manager might need about the given methodology.

At a higher level of abstraction there could be a component that compares different methodologies and presents their strengths and weaknesses. This component would synthesize all available information at this level of abstraction. As indicated previously, this process might highlight a gap in the technology (such as the lack of good object-oriented analysis techniques for real-time systems) that would lead to additional basic research as part of the technology transfer effort. The frameworks and models mentioned previously are implemented as high-level components.

Another way in which the MSS differs from a simple automated document retrieval system is that the interface is much richer. In addition to the standard indices and keyword searches, such a system provides a hypertext interface and an expert system interface. The hypertext interface interconnects related components and thus supports browsing through the MSS in meaningful ways. The expert system interface assists the user in selecting or adapting variants of new technologies to specific corporate needs. For example, a manager might wish to know what database model best fits a particular class of applications. A simple inferencing system could dialogue with the user and, based on knowledge in the components coupled with answers to the dialogue, recommend a particular database model.

A major advantage of the MSS is that it is directly available to the user via a personal computer/workstation. Software professionals can interact with and benefit from the MSS as needed, without having to leave their offices. Until recently, software professionals did not have sufficient computing power on their desktops to make the MSS practical. The MSS taps the computing resource available today to make not just a quantitative, but a qualitative difference in the empowerment of software professionals.

The Elements in Combination

Each of the four elements presented in the previous subsections is important in its own right. The importance of comprehensive scope and on-line availability is exemplified by the IBM Technical Information Retrieval Center [2]. Evidence for the importance of advanced knowledge delivery vehicles is supplied by the commercial systems that have been built using tools such as *Nextpert Object* from Neuron Data and *Level 5 Object* from Information Builders [6]. These products provide shells for the development of advanced knowledge delivery systems. The success of management consultants, and organizations that produce management summaries and newsletters, such as the Cutter Information Group and Ovum Ltd., attests to the importance of knowledge engineering. And, of course, all members of the high-technology community are cognizant of the importance of well-targeted research programs.

The contribution of our framework is the important synergies that result from bringing all four elements together under the direction of a single organization. The knowledge engineering combines with the rich interface and multiple levels of abstraction within the MSS to obviate the classical problem with comprehensive scope—*information overload*. The creation of models and frameworks for the MSS combines with knowledge engineering and comprehensive scope to provide a clear direction for basic research efforts. The results of these basic research results fill critical gaps in knowledge, enhancing the value of the knowledge base delivered via the MSS.

Other successful technology transfer efforts have recognized the value of combining several of these elements. For example, the Agricultural Extension Service combines scope and knowledge engineering, but typically uses a human agent for the delivery mechanism; the Research Institute for Computing and Information Systems has combined research and knowledge engineering to serve the Johnson Space Center [25]. Other examples exist that combine some of the elements, but we are not aware of any instantiation of our entire framework other than COMSOFT.

Organizational Issues and Critical Success Factors

The technology transfer activity, as any work process, can be presented in terms of its customers, suppliers, and principal functions. Figure 3 illustrates this view for a technology transfer organization that embodies the elements of our framework. The organization envisioned serves as a focal point for all parties with an interest in the continued development and adoption of emerging software technologies.

In our discussion of the proposed framework with corporate managers, we sometimes hear the comment, "All of this sounds good, but will it really work?" Much of the answer to this question lies in the organization structure of the envisioned technology transfer entity. While we do not downplay the importance of the components of our framework and the crucial interactions among them, similar attempts have succeeded or failed largely on political and organizational circumstances.

Many attempts within large companies to collaborate on technology transfer and other issues are foiled by turf problems and continual corporate reorganization. Thus it is important that the envisioned entity:

- be an independent legal entity sponsored by multiple organizations. This ensures the long-term stability required to realize fully the

benefits sought.

- have a balance of power between the organization's leadership and its corporate sponsors. Too little power for the sponsors leads to unresponsiveness; to much leads to infighting among the sponsors. Our specific strategy for achieving this balance is beyond the scope of this article, but the existence of such strategy is crucial for the success of the consortium.
- have a lean and small organization. Large organizations tend to become self-serving.

Another important success factor is setting correct expectations for such an entity. The benefits must neither be understated nor oversold. As an example, consider the benefits of the automated MSS. An advanced knowledge delivery vehicle makes a qualitative improvement in the empowerment of employees utilizing or introducing new technologies. Access to complete information, at the right level of abstraction, enables a corporation to minimize adoption risks and avoid false starts. Armed with the knowledge delivered by the MSS, advocates of change can mobilize their organizations to adopt appropriate emerging technologies in the most effective manner. An automated system, however, will not do away with the need for human interaction—technology is primarily transferred by people [15]. The MSS will not eliminate the need for training courses, consultants, meetings and lectures. These are needed to complete the assimilation of the chosen technology.

The AT&T Object Modeling Resource Base

Corporations have long recognized that specialized knowledge is required in areas such as personnel management and financial management, but have not invested in the specialized knowledge necessary for effective management of software technology. In recognition of this problem, AT&T has been working with Clemson University and Georgia State University in the area of managing software technology. One of the specific goals of this collaborative research effort is a management support system for object-oriented analysis [5].

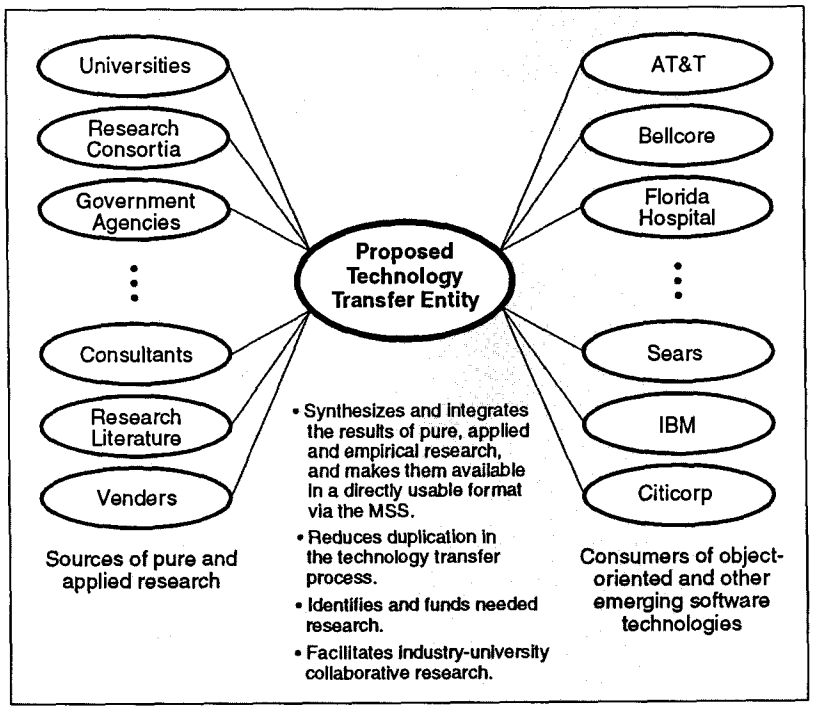


Figure 3. Technology transfer framework

Following the technology transfer framework discussed previously, we have built an instance of a MSS for AT&T called the Object Modeling Resource Base (OMRB). In its current form the OMRB is only a prototype, but experience with the OMRB is sufficient to verify the usefulness of our technology transfer framework.

The first phase in development of the components for the OMRB is the collection and structuring of the information (see Figure 2) from the four areas depicted in Figure 1. In general, a combination of a top-down and bottom-up approach is taken for the identification and development of the components. One of the first steps in this phase is the identification of existing object-oriented analysis and design methodologies (see Figure 4). The activities in each methodology are analyzed in order to determine which models, methods, and techniques are used in which phase of the methodology. Eventually, from this analysis, a framework is developed for classifying object-oriented methodologies based on their similarities [9, 22, 28]. The framework and the methodologies are high-level components in the OMRB that drive its inferencing capabilities.

In addition to storing knowledge about models, methods, and techniques **(theory),** the resource base contains such knowledge as what companies and/or industries are using what methodologies and methods, and what their experiences are concerning performance and use **(practice).** The resource base also contains knowledge about terminology, analysis and design notations, documentation of objects, object classes, and interface mechanisms **(standards).** The Object Management Group [21] is addressing many standards in the area of interoperability and has a task group focused on standards for object-oriented analysis and design methodologies. Effective **management** of OOA requires

practical metrics for assessing reuse, quality, productivity, and value in the new paradigm. Team organization and incentive structure, as well as project funding will be affected by the reuse considerations inherent in OOT. Existing knowledge in this area is, however, incomplete and further research is required.

Figure 4. Object-oriented analysis and design methods

- Alabiso:
  "Transformation of Data Flow Analysis Models to Object-Oriented Design", SIGPLAN *Notices*, November 1988
- Ballin:
  "An Object-Oriented Requirements Specification Method", *Communication of the ACM*, May 1989
- Booch:
  Object-Oriented Iwth Applications, The Benjamin/Cummings Publishing Company, Inc., 1991
- Coad/Yourdon:
  Object-Oriented Analysis, 2nd Edition, Prentice Hall, Inc., 1991
- Hood (Hierarchical OOD):
  Used by the European Space Agency (no reference/article)
- Jacobson:
  "Object-Oriented Development in an Industrial Environment", *ACM OOPSLA '87*
- Kreamer:
  "Production Development Using Object-Oriented Software Technology", *Hewlett-Packard Journal*, August 1989
- Kurtz/Ho/Wall:
  "An Object-Oriented Methodology for Systems Analysis and Specification", *Hewlett-Packard Journal*, August 1989
- Manfredi/Orlando/Tortorici:
  "An Object-Oriented Approach to Systems Analysis", ESEC '89, *2nd European Software Engineering Conference Proceedings*, 1989
- McIntrye/Higgins:
  "Object-Oriented Systems analysis and Design: Methodology and Application", *Journal of Management Information Systems*, January 1988
- Mozaffari/Tanaka:
  "ODM: an Object-Oriented Data Model", *New Generation Computing*, August 1989
- Nerson:
  "Experiencing Object-Oriented Analysis and Design", *Communications of the ACM*, September 1992
- Rubin/Goldberg:
  "Object Behavioral Analysis", *Communications of the ACM*, September 1992
- Rumbaugh/Blaha/Premerlani/Eddy/Lorensen:
  Object-Oriented Modeling and Design, Prentice-Hall, Inc., 1991
- Seidewitz:
  "General Object-Oriented Software Development", *Journal of System Software*, February, 1989
- Shlaer/Mellor:
  Object-Oriented Systems Analysis: Modeling thhe World in Data, Prentice-Hall, Inc., 1988
- Wirfs-Brock/Wilkerson/Weiner:
  Designing Object-Oriented Software, Prentice-Hall, Inc., 1990

The system is implemented in Level V Object, an object-oriented expert systems shell with hypertext capabilities. This makes the knowledge in OMRB readily available through navigation through the system in different ways. The primary purpose of the OMRB is to serve as a support system for an organization needing to choose an object-oriented analysis methodology. The knowledge about actual modeling techniques and methodologies, the knowledge about standards, practice, management issues, and the heuristics and contextual knowledge embedded in a component all work together to support the decision-making process.

We are currently working on inferencing capabilities. When this is complete, users can browse the system and select a methodology based on their analysis of information in the components, or they can choose to answer a set of questions and have the OMRB recommend the "best fitting" methodology. These questions characterize many areas, including the developer organization and experience, user organization, and problem domain.

Example Component in OMRB

The rich interface provided by the OMRB is awkward to convey in a 2D diagram; appreciation of its multidimensionality is best gained by interaction with the system. However, the knowledge components are amenable to paper presentation. Figures 5a–5c show the textual description of a component in the OMRB. The component shown is the entity-relationship diagram (ERD). The ERD is representative of a class of graphical diagramming techniques used in almost every methodology to model the structural relationships among objects in an application domain [9]. Figure 5a gives the name, type, description, notation, and an example of the ERD. Figure 5b describes one set of extensions to the original ERD and gives an example using these new abstraction constructs. Figure 5c illustrates how the ERD is related to other components in the resource base.

The section of Figure 5c "Used in Methodologies" lists several object-oriented analysis and design methodologies in which some form of enhanced ERDs are used in one of the phrases of the methodology. The "Remarks and Bibliographic Notes" section of Figure 5c provides contextual links with other components in the resource base and points to references in the an-

notated bibliography. The purpose of this section is to provide insights about ERD and how it relates to various techniques and models and the role of ERD in object-oriented methodologies. Feedback from "real world" use of a technique or methodology will also go into the Remarks section. The "Cross Reference" section of Figure 5c lists other links in the resource base where the user can find more information about a related topic. This example is typical of version 1.0 components. Our current emphasis is on populating the OMRB. Future versions will add more contextual, management, practice and heuristic knowledge.

The AT&T OMRB is a useful instantiation of our technology transfer framework for supporting the management of object-oriented software development. This framework is mirrored in the management support system shell we have developed in Level V Object. This shell facilitates synthesizing, structuring, indexing, and cross referencing knowledge in a given area.

Figure 5. Entity-relationship diagram

Figure 5a

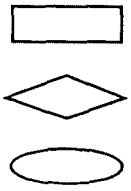
Name: Entity-Relationship Diagram (ERD)

Type: Graphical Modeling Technique

Description:

An *entity-relationship diagram* is a graphical technique that depicts the enterprise being modeled as a collection of entity sets, relationships between entity sets, and attributes associated with both entity and relationship sets.

Notation:



Entity

Relationship

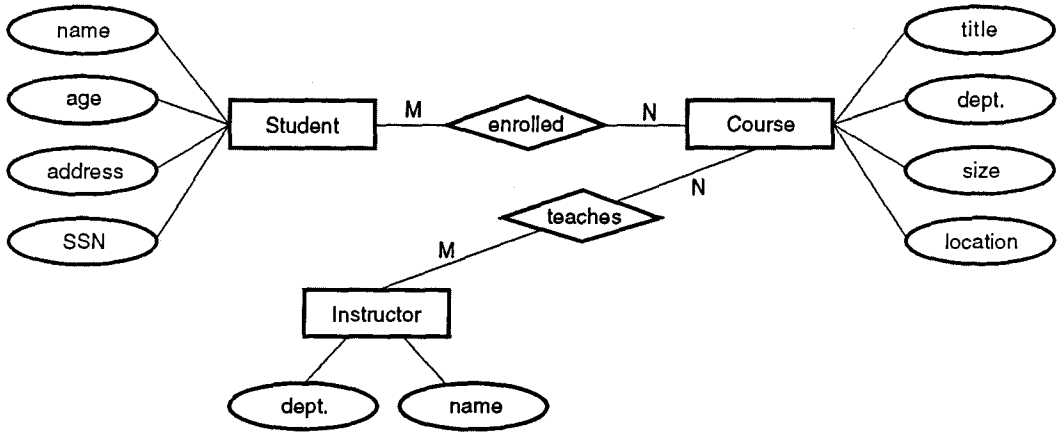
Attribute

1:1 = one-to-one
1:N = one-to-many
M:N = many-to-many

Cardinality of Relationship

In the standard ER notation, rectangles represent entity sets, diamonds represent relationships between entity sets, and ovals represent attributes of entity or relationship sets. The cardinality of a relationship is represented by the number 1, M, and N indicated on the arc connecting entities. The cardinality of a relationship can be: one-to-one, one-to-many, or many-to-many.

Example:



In the diagram above, there are three entity sets: **Student, Course,** and **Instructor.** The entity **Instructor** has two attributes: the instructor's name and the department he/she teaches for. There are two relationships interconnecting entire sets: **enrolled** and **teaches.** In the first relationship, each student may be enrolled in several courses and each course may have many students enrolled in it. This many-to-many relationship between students and courses taught is represented by the **M** and **N** on the arc connecting entities.

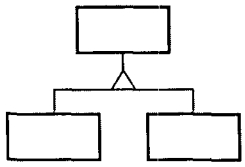
Figure 5b

Enhancements:

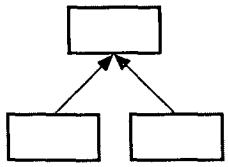
Description:

The most common extensions to the ER diagram include the abstraction concepts of aggregation, generalization/classificaiton, and association. *Aggregation* is a form of abstraction in which a collection of objects (entities) are viewed as a single higher level object. *Generalization* is a form of abstraction which captures the commonalities among objects while at the same time ignoring the differences among objects. *Classification* is a form of abstraction in which an object class is defined as a set of objects which have the same attributes. Classification provides the mechanism for the specification of the class or type of an instance of an object. Then given instance of an object, its attributes, and the values of those attributes, an object can be classified based on the generalization/classification hierarchy. *Association* is a form of abstraction in which a relationship between member objects is considered a set of higher level objects. Objects are considered elements of an abstraction set when they satisfy the set's membership properties.

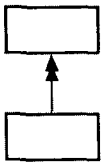
Notation:



Aggregation

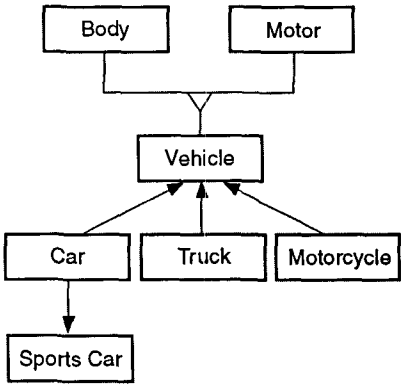


Generalization/Classification



Association

Example:



In the example, the object **Vehicle** consists of the separate lower level objects: **Motor** and **Body**. With the aggregation construct the collection of elements objects (component) becomes the attributes of the higher level object. In this case, **Motor** and **Body** are attributes (or parts) of **Vehicle**. The class **Vehicle** is also a generalization of the classes **Car, Truck,** and **Motorcycle**, and represents the common properties of the three classes. The class Vehicle is called the *superclass* of the *subclasses* **Car, Truck,** and **Motorcycle.** All subclasses of a superclass inherit the common attributes or properties of the superclass and can add attributes that specialize the subclass from its superclass. For example, a **Truck** not only has the attributes defined for the class **Vehicle**, but also those attributes that are particular to the subIcass **Truck**. The class **Truck** represents a classification of a set of objects with the same attributes. The set **Sports Car** is an association of members of **Car** which satisfy the set property that the value of attribute **Horsepower** is greater than 200.

The shell also provides a framework for inferencing capabilities.

Applications

The United States excels in the production of basic research results. With the help of federal agencies such as the National Science Foundation, the U.S. system of federal labs, corporate research centers, universities, and research consortia produce an enormous amount of basic research results. It is often difficult for researchers to keep up even with their own specialized areas of interest, due to the volume of research results continually produced.

But there is a big gap between research results and the needs of corporations. To bridge this gap most major corporations have a technology transfer division. These technology transfer divisions assess emerging technologies such as CASE, object-oriented technology, fuzzy logic, expert systems, and visual programming for applicability within their companies. When useful technologies are identified, the activities outlined in our technology transfer framework end up being carried out by each and every company wanting to use the technology.

This results in an enormous duplication of effort. In the area of object-oriented technology, for example, the authors are personally aware of substantial duplication of corporate effort in both knowledge engineering and research. This duplication of effort reduces our national productivity.

The Japanese have demonstrated the value of addressing technology transfer at the national level [15]. We believe the U.S. could benefit from similar national attention to technology transfer for the commercial sector. One striking feature of most U.S. software tech-

Figure 5c

Used in Methodologies

- OMT (Rumbaugh)
  Identify objects, attributes, and associations. Use Object Diagrams (based on ERDs) to build Object Model.
- OOA (Shlaer/Mellor)
  Identify objects, and attributes. Use extended ERDs (*class, aggregation, association*) to build Information Models.

⋮

- OOD (Schrefl/Kappel)
  Develop Static Model using extended ERDs (*class, aggregation*).

Cross References

For more information on

Rumbaugh Methodology ..... see ROMT
Shlaer/Mellor Methodology ..... see SMOOA

⋮

Schrefl/Kappel Methodology ..... see SKOOA

Remarks and Bibliographic Notes

The ER model was one of the first semantic data models (SDMs) that provided constructs for representing structurally complex relationships among data. One of the goals of the ER diagram was to simplify the design and use of databases by providing modeling structures closer to the way designers and users perceive the data of an application. The original ER model was limited in the abstraction concepts it supported and has evolved to support other relationships between entities.

References

Ballin, S.C., "An Object-Oriented Requirements Specification Method", *Communications of the ACM*, Vol. 32, No. 5, May 1989, pp. 608–623.

Chen, P.P., "The Entity-Relationship Model - Toward a Unified View of Data", *ACM Transactions on Database Systems*, Vol. 1, No. 1, 1976, pp. 9–34.

Coad, P., and Yourdon, E., *Object-Oriented Analysis*, 2nd. Edition, Yourdon Press, 1991.

Hull, R., and King, R., "Semantics Database Modeling: Survey, Applications, and Research Issues", *ACM Computing surveys*, Vol. 19, No. 3, September 1987, pp. 201–260.

Kappel, G., and Schrefl, M., "A Behavior Integrated Entity-Relationship Approach for the Design of Object-Oriented Databases", in *A Bridge to the User, Proceedings of the 7th International Conference on Entity-*

⋮

nology transfer activities is that they are almost all funded for and by the military. Most business leaders we have spoken with claim these organizations do *not* meet their needs. This is in contrast to Japan, where most of the high-technology research is, "for commercial purposes. Furthermore, Japanese government agencies and professional organizations take a more active role in organizing and energizing the civilian technology transfer process than do the counterpart organizations in the U.S." [15].

We envision a number of national consortia charged with the technology transfer in a given area. Most of the existing consortia and government labs are not at the level of abstraction outlined in our framework. They perform basic and applied research, but lack the knowledge-engineering component and advanced knowledge delivery vehicle that would make them a national center for technology transfer in a given area. The existence of organizations such as the SEI, SPC, MCC, OMG, and SERC is a crucial part of a national research infrastructure. We applaud the valuable work they are doing, but see the need for additional kinds of organizations.

The concepts presented in this article can be further developed along two lines: realizing a full implementation of the technology transfer framework within the area of emerging software technologies, and extending this approach to technology transfer to other areas. The following subsections discuss these two lines of development.

Implementation for Emerging Software Technologies

The newly formed Consortium for the Management of Emerging Software Technologies (COMSOFT)

[26] is an instance of the kind of national technology transfer center described in Figure 3. This consortium addresses many of the concerns outlined in the "Research Agenda For Software Engineering" produced by the Computer Science and Technology Board [8].

The current focus of COMSOFT is on object-oriented technology and the related area of reuse. A number of broad areas are listed to give a flavor of the current research and knowledge acquisition interests of the consortium.

- managing reuse across and at different levels of an organization
- factors crucial to success in adopting object technology
- productivity metrics appropriate to object-oriented technology
- testing object-oriented systems
- object-oriented process technology, including project planning, resource allocation and costing in the object-oriented life cycle
- policies and tools and metrics for the management of object-oriented technologies
- metrics, guidelines and methodologies for object-oriented analysis and design
- determining value and accounting procedures for information assets
- effect of object technology on structure and strategy of a corporation at all levels: corporate, division, project, and team.
- reengineering existing systems and hybrid systems

COMSOFT is a natural extension of the AT&T OMRB to the whole area of process infrastructure for emerging software technologies.

Application of Framework to Other Areas

The technology transfer framework that we have described could easily be instantiated in other areas. To create a MSS in a new area requires the following steps:

1. Creating a comprehensive annotated bibliography covering the four areas of theory, practice, standards and management,
2. Performing knowledge engineering and basic research to fill gaps,
3. Creating necessary models and frameworks,
4. Using the results of steps 1–3, build the databases, indices, hypertext links, knowledge base components, and rules required by the MSS shell. The open architecture of the shell simplifies this process.

Evidence that our framework is of general applicability comes from our having been able to instantiate the MSS in two very different areas, object modeling and assessment of information value. Work is ongoing in both of these areas.

Acknowledgments

We would like to thank David Abbott, on leave from AT&T Bell Laboratories, for his invaluable input and assistance with this article. We would also like to thank Rick Foster and Tom Cooper, also of AT&T, for their assistance and support in developing the concepts presented in this article. ■

References

1. Bayer and Malone. A critique of diffusion theory as a managerial framework for understanding adoption of software engineering innovations. *J. Syst. Softw.* (Sept. 1989), 161–166.
2. Berman, M. Infogate—The information gateway: A tool for employee empowerment. In *Proceedings of the Thirteenth National Online Meeting* (May 1992), pp. 41–50.
3. Bobrow, D.G., Mittal, F. and Stefik, M. Expert systems: Perils and promise. *Commun. ACM 29,* 9 (Sept. 1986), 880–894.
4. Boehm, B. A spiral model for software development and enhancement. *Computer 21,* 5 (May 1988), 61–72.
5. Buchanan, Korson, T., and Vaishnavi, V. OMRB: Object modeling resource base. A support tool for object-oriented analysis. In *Proceedings of the fourth International Conference on the Technology of Object-Oriented Languages and Systems.* Prentice Hall, 1991, Englewood Cliffs, N.J., 215–226.
6. Carlson, D. and Ram, S. Hyperintelligence: The next frontier. *Commun. ACM 33,* 3 (Mar. 1990), 311–321.
7. Chandrasekaran, B., Johnson, T.R. and Smith, J.W. Task-structure analysis for knowledge modeling. *Commun. ACM 35,* 9 (Sept. 1992).
8. Computer Science and Technology Board. Scaling up: A research agenda for software engineering. *Commun. ACM 33,* 3 (Mar. 1990), 281–293.
9. de Champeaux, D. and Faure, P. A comparative study of object-oriented analysis methods. *J. Object-Oriented Programming, 5,* 1 (Mar./Apr. 1992).
10. Chidamber, S. and Kamerer, C. Towards a metrics suite for object-oriented design. In *Proceedings of OOPSLA'91.* ACM Press, Oct. 1991, pp. 197–211.
11. Cox, B. Planning the software revolution: The impact of object-oriented technologies. *IEEE Softw.,* (Nov. 1990).
12. Curtis, B., Kellner, J. and Over, J. Process modeling. *Commun. ACM 35,* 9 (Sept. 1992).
13. Harrold, M-J and McGregor, J. Incremental testing of object-oriented class structures. In *Proceedings of the International Conference on Software Engineering* (May 1992).
14. Hazeltine, N. Panel Position Paper: Managing the transition to object technology. *Addendum to the Proceedings of OOPSLA'91.* ACM Press, Oct. 1991.
15. Cutler, R. A comparison of Japanese and U.S. high-technology transfer practices. *IEEE Trans. Eng. Manage. 36,* 1 (Feb. 1989), 17–24.
16. Korson, T. Panel Position Paper: Managing the transition to object technology. In *Proceedings of OOPSLA'91.* ACM Press, Oct. 1991.
17. Korson, T. and McGregor, J. Technical criteria for the specification and evaluation of object-oriented libraries. *IEE Softw. Eng. J.* (Mar. 1992), 85–94.
18. Kumar, K. and Welke, R. Method-

ology engineering: A method for situation specific methodology construction. *Syst. Analysis and Design: A Research Agenda,* W. Cotterman and J. Senn, Eds. Wiley, N.Y. To be published.
19. Laranjeira, L. Software size estimation of object-oriented systems. *IEEE Trans. Softw. Eng. 16,* 5 (May 1990), 510–521.
20. Leathers, B. Panel Position Paper: OOP in the real world. In *Proceedings of OOPSLA/ECOOP'90,* N. Meyrowitz, Ed. ACM Press, Oct. 1990, pp. 299–302.
21. Martin, W. Object modeling: What business has been waiting for. *First Class* (June/July 1991), 7–9
22. Monarchi, D. and Puhr, G. A research typology for object-oriented analysis and design. *Commun. ACM 35,* 9 (Sept. 1992).
23. Putnam, L. and Myers, W. *Measures for Excellence: Reliable Software on time, Within Budget.* Yourdon Press, 1992.
24. Reichert, W. and Sello, H. Whole-earth technology. *High Tech. Bus.* (July-Aug. 1989), 14–30.
25. Rone, K.Y., MacDonald, R.B. and Houston, A.G. Technology development: A partnership that makes sense. *Comput. Res. News 4,* 3 (May 1992).
26. Salmons, J. and Babitsky, T. International OOP Directory, 1992. *SIGS Publications,* 1992, 4–13.
27. Taylor, D. *Object-Oriented Information Systems: Planning and Implementation.* Wiley, N.Y., 1992.
28. Walker, I. Requirements of an object-oriented design method. *IEE Softw. Eng. J.* (Mar. 1992), 102–113.
29. Waterman, D.A. *A Guide to Expert Systems.* Addison Wesley, 1986.
30. Zmud, R. Diffusion of modern software practices: Influence of centralization and formalization. *Management Sci. 28,* 12 (Dec. 1982), 1421–1431.

CR Categories and Subject Descriptors: D.2.1 [**Software**]: Software Engineering — *requirements / specifications;* D.2.10 [**Software**]: Software Engineering—*design;* I.6.0 [**Computing Methodologies**]: Simulation and Modeling—*general;* I.6.3 [**Computing Methodologies**]: Simulation and Modeling—*applications;* K.6.3 [**Computing Milieux**]: Management of Computing and Information Systems—*software management;* K.6.4 [**Computing Milieux**]: Management of Computing and Information Systems—*system management*

General Terms: Design, Methodology

Additional Key Words and Phrases: Analysis, Modeling

About the Authors:

TIMOTHY D. KORSON is an assistant professor of computer science at Clemson University and Executive Director of the Consortium for the Management of Emerging Software Technologies (COMSOFT). Current research interests, publications and consulting activities focus on object-oriented technology, reuse, and managing emerging software technologies. **Author's Present Address:** Clemson University, Department of Computer Science, Clemson University, Clemson, S.C. 29634; email: korson@cs.clemson.edu

VIJAY K. VAISHNAVI is professor of computer information systems at Georgia State University and research director of COMSOFT. Current research interests include emerging software technologies, object-oriented knowledge modeling, generic intelligent tutoring systems, and efficient data and file structure. **Author's Present Address:** Georgia State University, Department of Computer Information Systems, College of Business Administration, University Plaza, Atlanta, GA 30303-3083; email: cisvkv@gsuvm1.gsu.edu

This work was sponsored by AT&T research grant #TS90-79z and a research grant from the Research Council of the College of Business Administration at Georgia State University.