Electronic Enterprise Engineering - An Outline of an Architecture

Michael Bieber, Michael Bartolacci, Jerry Fjermestad, Franz Kurfess, Qianhong Liu,
Marvin Nakayama, Ajaz Rana, Wilhelm Rossak, Richard Scherl, Murat Tanik,
Jason Wang, Raymond Yeh, Peter Ng, Richard Sweeney
New Jersey Institute of Technology
Newark, NJ 07102, U.S.A.
rossak@cis.njit.edu

Fabio Vitali
University of Bologna
Bologna, Italy

Abstract

In this paper we put forth a vision for organizations to fully embrace computer support. We propose a business-process oriented architecture for Electronic Enterprise Engineering (EEE) that will enable enterprises to manage and evolve all technological and organizational processes effectively; integrate and manage all enterprise information electronically; and empower knowledge workers at all levels with broad decision support capabilities. Our goal is for the EEE architecture to empower an enterprise to make the best use of its informational assets to operate effectively in this new era of electronic commerce. As part of this project we are developing a standard-based, customizable, integrated tool set called the Support Environment for Enterprise Engineering (SEEE). This paper presents the current SEEE architecture and shows how it supports the three EEE goals.

1. SEEE: An architecture for EEE

The drive towards electronic commerce is pushing companies to move more and more of their operations on-line.Organizations must deal with clients and customers on-line, handle telecommuting and workgroups distributed across the country or world-wide, deal with government agencies and other organizations with whom they have relations on-line, and so on. Certainly many companies have moved some mission-critical portions of their operations on-line, and many have made great strides towards workflow management and on-line communication. Yet even the most advanced companies are neither as fully integrated as they could be, nor we argue, as they should be. Furthermore, most companies are looking for ways to operate more efficiently and effectively.

In this paper we put forth a vision for organizations to fully embrace computer support. We propose a process-oriented infrastructure for Electronic Enterprise Engineering (EEE) that will enable companies to:

1. manage and evolve all technological and organizational processes effectively;
2. integrate and manage all enterprise information electronically; and
3. empower knowledge workers at all levels with broad decision support capabilities.

Figure 1 presents a high-level conceptual overview of our proposed Support Environment for Enterprise Engineering (SEEE) architecture. In practice, components can be distributed across networks of different machines and platforms. To all extents practical, SEEE will employ existing software and integrate in existing enterprise applications. SEEE, however, will require much innovation.

The *process manager* guides, executes and analyzes the enterprise's processes. The *view builder* constructs integrated interfaces tailored to user tasks and preferences. The *hypermedia engine* manages sophisticated navigation. It also provides users with direct access to meta-level hypermedia relationships among and across

0-8186-7889-5/97 $10.00

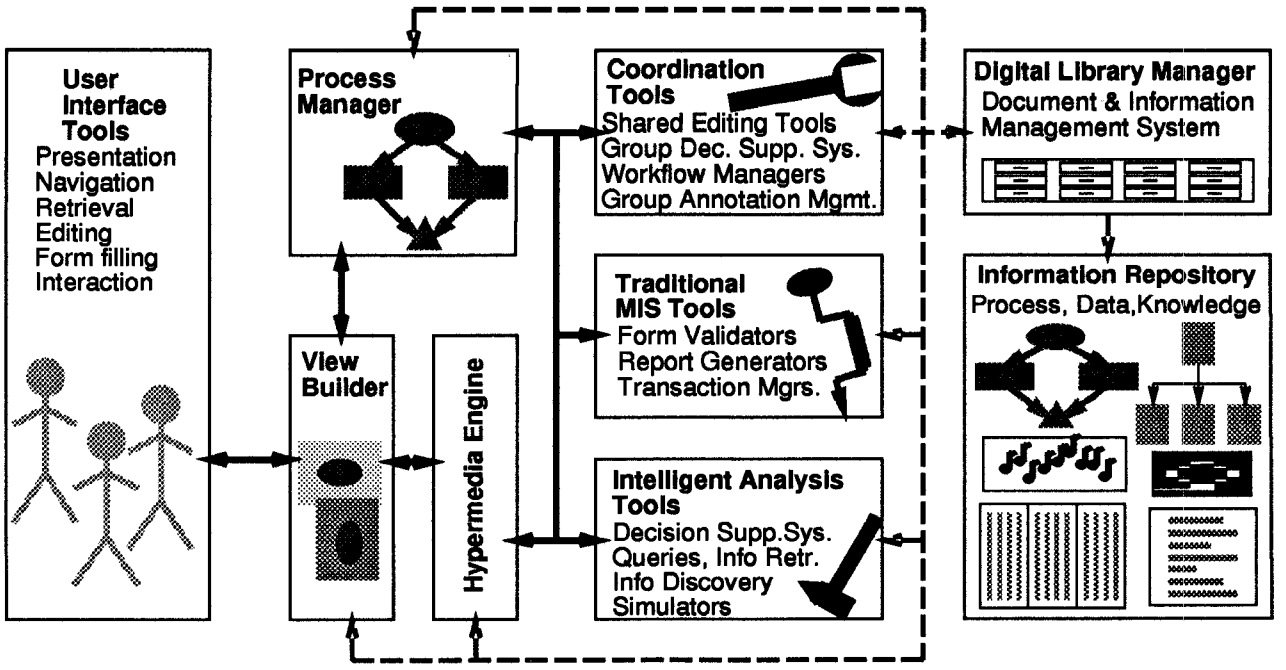

Figure 1. Basic Architecture of the Support Environment for Enterprise Engineering (SEEE).

all processes, tools and information. *Intelligent analysis tools* provide a comprehensive and innovative set of decision analysis features. *Traditional MIS tools* refer to most of an enterprise's "legacy systems" and other standard applications. *Coordination tools* provide processes and other tools with a rich collaboration support. The *digital library manager* provides sophisticated document management and digital library features, extending these where appropriate to the entire electronic information base. The *repository* maintains the system's data, including its metaknowledge and processes. Users interact with SEEE through the *user interface tools*. These couple familiar environments such as spreadsheets and word processors with sophisticated navigation techniques and World Wide Web access, so users can telecommute and otherwise work remotely.

In the following sections we describe the components of the SEEE architecture in some more depths and discuss many of the research issues each encompasses. We conclude with a summary and some observations.

2. Process manager

A process consists of a structure and a context. The structure defines the relationships among its activities—sequence, branch logic, and business rules. Its context consists of resources—people, machines, time, facilities, etc.—which are applied to the process, interelement coordination and interprocess communication.

Thus, the process manager architecture consists of three interdependent models: an *activity model* that captures process structure, a *coordination model* and an *infrastructure model* that represents process context [29]. Coordination deals with managing the intangible aspects of the organization in terms of personnel roles, communication, and culture. The infrastructure model contains all horizontal functions and assets in an organization such as data, security, and interoperability.

Process engineering requires technological support that falls into three distinct phases, each with different requirements: process capture and definition, process analysis and simulation, and process implementation and delivery (usually referred to as process enactment).

2.1. Process capture and definition:

To do good modeling, we need a modeling language representing the following aspects:

- Functional: what activities are performed with which products (activity model)
- Behavioral: when activities are performed (coordination model)
- Organizational: where and who performs activities (infrastructure model)

- Informational: information entities manipulated by the process (infrastructure model).

Designers will express processes in this modeling language, which the process manager will employ for analysis support and enactment.

The following observation supports such a modeling language. We can consider three dimensions of design components: data, function and control abstractions [24]. The infrastructure corresponds to data abstraction, the activity model corresponds to function abstraction, and the coordination model corresponds to control abstraction. The advantage of this correspondence would be that the mathematical machinery available in one model would be applicable equally in the other. For example, mathematical and algorithmic techniques presented in [2, 13, 19, 24] can also be used in the study of processes. The relationship between design abstractions and process models further should be exploited to take advantage of the integrated and continuous application of process improvements during design [6, 10]. In the context of concurrent engineering there are promising attempts to achieve product life cycle and process integration.

These theoretical formulations constitute a basis to develop a variety of process design and simulation tools, such as ProSLCSE developed by ISSI at the University of Texas at Austin [12]. Developing process simulation tools for environmental processes would be a natural extension.

2.2. Process analysis support

Process analysis can answer, e.g., the following questions:

- How many resources were needed to be allocated to a specified activity or a specified project? Are the resources enough to perform other activities of the project?
- What will the cost be for a particular activity in a specific project?
- Was an activity completed on time? How much over-time has been expended? How much extra expenditures are still needed to complete the activity?
- Where is the longest queue? What is the current length and wait time of a specific queue?

Process analysis also can take advantage of simulation to perform quantitative trade-off analysis of cost, schedule, and resource risks associated with a process, as well as to provide visualization of the process in execution. This capability of visualization and interactive debugging provides the critical understanding of the process not otherwise obtainable from static analysis alone. Some of the metrics for analysis during process simulation include wait time of tasks for human resources, tasks with longest waiting time, missed milestones, tasks that missed their latest start or stop dates, number of interactions per person, and cost (per person, per activity, cumulative, etc.). While performed by SEEE's process manager, we view process analysis support as one of our intelligent analysis tools (see §5).

2.3. Process enactment support

Process enactment means execution of the designed process in a real world application. Enactment guides people assigned to activities in the process as to when and which activities should be performed, and which data and products should be consumed, referenced, and produced. Thus, enactment provides process driven coordination among individuals working within a process. It provides coordination across functional boundaries and freeing members up from overhead such as trying to figure out which activities are to be performed, can and cannot be performed, why and when, inputs required to perform an activity, and results to be produced, etc. The process manager captures all this information mentioned automatically so that an individual used can focus on the specific task rather than on figuring out what to do next. The architecture of a process manager is typically a client/server model with a local client *process performer* and a *process controller* server. They communicate via some middleware (CORBA, OLE, DEC, etc.) through network. The process controller basically is a state machine and has the authority to change project status according to which of the state a process is in (planned, ready, active, suspended, completed, canceled).

3. User Interface tools and the view builder

Providing an interface to the SEEE system is a particularly complex task. The SEEE system stresses two paradigms, which influence the interface:

1. *Tool integration*: the user may activate and use several different tools in order to perform a task, yet it is important that, as far as possible, these tools are seamlessly integrated and possibly even disappear behind a common interface that deals with them.

2. *Process-based flow*: the user need only be concerned with high-level tasks, which might involve several tools and several steps. The system must manage the details of these steps for the user (e.g., activating the proper tool at the right moment and feeding it the correct information generated in previous steps), and provide feedback on the current step and its role in the general task execution.

Several further issues need to be considered:

1. *Every functionality is multi-user*: not only must the system manage consistency and access, but also, in many cases, reciprocal awareness of the users.

2. *Every functionality is distributed*: besides graceful management of network problems, this encompasses handling different people, with different rules and habits.

3. *Every functionality is composite*: that is, functionalities rarely require only one tool, but most often the interaction and sequence of multiple services from different tools.

4. *Users are not necessarily novices*: the system must leverage the existing knowledge of the user, and allow him or her to use the tools to which he or she is most accustomed, in terms of hardware, operating system and end-user tools.

Therefore integrating access to different services, tools and data collections, and the number of combinations of different functionalities that the environment will need to provide makes it impossible to plan, create and hard-wire a fixed structure of interfaces. Such a system calls for the dynamic generation of the interface based on the user, his or her tools, the requested service, the context in which a service is requested, and the task the user performs.

Of the five types of user interface style listed in [23] (menu selection, form filling, command languages, natural language and direct manipulation), only form filling, menu selection and direct manipulation are appropriate in our context. Both HTML forms and Ms Excel dialogs already allow the easy specification of form filling interfaces, and of course HTML anchors can be used as menus. Basically most end-user commercial applications allow some form of management of these two types of interfaces, and actually WWW browsers have the widest flexibility. A general model for direct manipulation objects, though, is still lacking, although Java applets may provide the solution. Furthermore, CGI applications already create context-dependent forms for HTML documents, and complete applications have been created in which the HTML documents required for the interactions are not stored anywhere on an HTTP server, but created on the fly depending on the context.

4. Hypermedia engine

Hypermedia supplements process engineering and EEE's application environment with linking and a variety of sophisticated navigation techniques. Ultimately, hypermedia increases comprehension by giving users access to information within a very rich context [25]. As alluded to in §3, the view builder could present the entire process within a hypermedia graphical overview diagram of its subtasks. The user could select any step to execute it or to find out additional information about it. Useful links about any subtask include a description, instructions, documentation, comments and examples. Within any task display, every object could have links available. Users could annotate any object in the system with a comment that the general public could access. Objects in other EEE component systems would have analogous supplemental information available.

In general, the engine could provide the following additional support to process subtasks: related documents and documentation, information about each person and resource associated, information about its role in the entire process, all tasks affected by and which affect this task, who is responsible or owns this task, detailed information about any component within the task, any subtasks of the current subtask, the previous and next subtasks in the process, and similar tasks in other processes.

Hypermedia brings a rich set of structuring and navigational functionality to applications, including guided tours, recommended paths, annotation, information overviews and sophisticated backtracking techniques [4, 18].

The hypermedia engine will automate hypermedia linking and navigation to the greatest extent possible. Our current hypermedia engine prototypes automatically infer the hypermedia links in decision support domains [3] and database domains [26]. [3] presents the prototype's internal structure in more detail. We generate hypermedia support based on the internal structure of application. In EEE's case this would be the internal representation mentioned in §2. Much research, however, remains. Our current engine serves only single applications and single users. While it automatically infers links and relatively simple forms of navigation and annotation, we still must determine how to determine overviews and paths within an inferred in-

formation space on the fly, as well as support the more sophisticated navigation a full-fledged application environment requires.

5. Intelligent analysis tools

"Intelligent Analysis" in the context of enterprise engineering is the ability of a firm to make key business decisions with cogent and current information. Decisions such as whether to manufacture a new line of products or whether to construct a new production facility must be supported with budgetary, marketing, environmental, design, and a host of other types of information. Often the amount of information involved could be overwhelming for managers and key decision makers. Provided within the EEE framework is the ability to analyze similar past decisions and also the ability to simulate results from future decisions. The intelligent analysis tools supports such functionality within the SEEE architecture.

Intelligent analysis tools cover the suite of applications found on an analyst's or knowledge worker's desktop and within executive information systems. These cover a wide variety of decision support systems supporting, e.g., data mining and knowledge discovery, enterpise simulation, and logical analysis. In this paper we will highlight only one such tool: the logical analyzer.

5.1. Logical analysis

Logical analysis plays a crucial role in confirming the robustness of the enterprise's processes. SEEE employs a logical model of the processes that take place in a particular enterprise—the preconditions of the processes and the effects of the processes. This logical model forms the basis for an automated reasoning tool that facilitates the management of an organization by keeping track of the various requirements that need to be met, proposing new ways of meeting requirements, determining whether or not a particular sequence of actions will satisfy certain requirements.

Logical representations of processes and actions (be they actual physical actions or the execution of a piece of software) can be used to represent and reason about the pre-requisites and effects of the processes and actions. Additionally, the action representation can be used to support business process reengineering. In this case, the repository contains organizational processes and the reasoning engine will be used to ensure both the correct fit of the different actions and to ensure that needed requirements are met at various stages of a proposed sequence of actions.

Our formalism for representing the processes is the situation calculus. This is a first-order language designed to represent changing worlds in which all changes are the result of named actions. The situation calculus provides convenient formalism for representing various actions and their effects; and also for reasoning with such a representation. We use a version of the situation calculus with a representation for actions that affect the knowledge of an agent [22] and a representation for complex actions [14].

Our reasoning method is based on a form of regression that reduces reasoning about future situations to reasoning about the initial situation. The result is a method for answering the question of what is true in the situation resulting from the execution of a particular sequence of actions. A model logic theorem prover is then used to determine the truth of the regressed expression in the initial situation. Further extensions involve the integration of reasoning about time, handling multiple agents and their interaction, and the addition of other epistemic states such as intention.

In modeling the work processes within an organization, it is not only necessary to represent the various actions carried out within an organization but also the reasons for carrying out these actions. The greater the expressivity of the formalism (capturing the intentions and motivations behind the activity, the requirements) the more flexible the method will be in reengineering the work process. Yet this greater expressivity will also cost us in terms of efficiency of reasoning and put us in the realm of less well understood formalisms. A useful middle point will have to be found.

6. Traditional MIS tools

It is vital that the EEE environment integrate the everyday existing applications and data that the enterprise currently uses. The enterprise will not abandon its legacy systems [1] indeed many of the processes EEE supports will employ them.

We see two basic approaches to integrating an existing system: redesigning just its interface or rebuilding it from scratch. Rebuilding entirely, while a very costly effort [1], gives an organization the opportunity to take advantage of years of experience with the domain to include missing features, as well as to comply fully to all EEE requirements. In this case developers must consider EEE's requirements throughout its standard systems analysis and design.

Less costly and more likely, enterprise developers will choose only to replace the existing system's interface. The developer must determine how to pass information between the SEEE environment and the

existing system's functionality. If it was coded in a modular fashion that clearly separated the interface from the computational aspects [4], or if it has an API or batch mode interface, then the developer basically has to match each object, attribute and operation with the appropriate internal call. EEE will need to map each of these to SEEE's internal representation (see §2) to ensure integration with the process manager and the hypermedia engine. Any other situation will prove much more difficult.

7. Coordination tools

Collaboration between individuals is intrinsic to the enactment of an enterprise's processes. That is, the effectuation of any given process most often requires cooperation among individuals (a group - ad-hoc or established). Additionally, at any given time, individuals are involved with more than one process and hence are, temporary, or long term, members of more than one work group. These groups continuously and simultaneously engage in activities that not only contribute to the effectuation of the enterprise's processes but to the well being of its members and to the continued functioning (maintenance) of the group itself [11].

The coordination component of EEE provides support for collaborative action in defining, analyzing, and enacting enterprise processes. Depending upon the nature of activities involved, support for collaborative action would include technological features that (i) augment human information processing limitations and deficiencies, (ii) facilitate the interactive communication process among group members, (iii) and help a group manage or regulate the group process in a systematic manner [9, 8, 21].

Currently, Group Support Systems (GSS) are being used to hold meetings, make decisions, or support regular work in both synchronous (same time, same place; and same time, different place); and asynchronous (different time, different place) modes [5]. Present day GSSs, commercial and non-commercial, differ in terms of the nature of tasks and processes each best suits. For example, systems originally designed for supporting face-to-face processes are not readily suited for supporting an asynchronous non-face-to-face group process. This is due to the fact that media of communication permitted (or imposed) by various GSS configurations differ in terms of the richness of information they are capable of conveying [7]. The face-to-face meeting offers the richest medium of communication, whereas text based (non face-to-face) communication is considered to be the least rich medium. Audio and video modalities lie between the two extremes.

The systems designed to support same time, same place group processes rely on face-to-face conversation for resolution of ambiguities and equivocality reduction, and hence do not generally include audio and video communication support. In a non-face-to-face asynchronous group process a rich medium of communication is crucial for certain tasks. The successful use of a GSS depends upon defining a fit between the information richness requirements of tasks, and the GSS configurations that are suited for conveying the needed media richness [16, 20].

Despite the fact that most present day GSSs include sophisticated decision, analysis, and work-flow support tools, they are not integrated with the enterprise's information repositories. We envision the EEE infrastructure to include group support tools that would provide a flexible interface with EEE's document and information management system, information repository, and analysis tools. Further, GSS tools would provide a multi-mode (synchronous and asynchronous) communication support environment. In this environment, the activities related to process definition, analysis, and enactment could all be done in a collaborative manner. EEE's process definition and capture structure (§2.1) may also be used to define the suitability of various GSS tools for supporting collaborative work.

8. Information repository

The repository and management of the enterprise information play a major and central role in integrating business processes and coordinating toolsets to be utilized effectively by workgroups.

Enterprise information in the information repository includes process representations, documents, organizational data, hypermedia links, annotations and navigational constructs, coordination information, and knowledge manipulated by the intelligent analysis tools.

Our proposed digital library manager will provide an intelligent, generic information management system to identify, collect, distribute and analyze repository contents automatically within a multi-user, distributed, cooperative environment. We call it a ***digital library*** manager to highlight the sophisticated role it plays in analyzing and retrieving information, akin to the role envisioned in much of today's digital library research [27]. It applies these functions to all repository contents where appropriate, not only traditional documents.

The digital library management system provides the following functionalities:

- It supports storing, classifying, categorizing, retrieving and reproducing processes, software tools, knowledge, multimedia documents, and others.
- It supports extracting, browsing, retrieving and synthesizing information from a variety of documents, processes and software tools, etc.
- It supports knowledge and information discovery from the existing information in the repository.

We intend to grow the digital library manager from our TEXPROS document management system project [15, 17, 28, 30]. The current TEXPROS prototype is a filing-and-retrieval-oriented office document processing system, which supports storing, classifying, categorizing, retrieving and reproducing documents, as well as extracting, browsing, retrieving and synthesizing information from a variety of documents.

The following examples, based on a traditional library environment, illustrate several features of the digital library manager:

- A knowledge-based, customizable document classification handler that exploits both spatial and textual analysis to identify the type of documents, such as, different types of articles, user request forms, or invoice form, etc.
- Information extraction mechanism to extract the synopsis or the most significant information from documents, which is pertinent to each subprocess. For instance, the citation of a paper is extracted from the user's library request form and sent to the subprocess of finding the source library, while the user's personnel information is extracted from the same form and sent to the subprocess of accounting and auditing.
- An agent-based, predicate-driven filing architecture supporting document filing and reorganization. For instance, in the subprocess of finding the source library, the system needs to find the prospective suppliers. Traditionally, the the library staff handles this manually based upon their perception and experience. By employing our filing architecture, this can be uploaded automatically.
- An intelligent retrieval system to assist in improving the quality of the interlibrary loan process. For instance, the information about the requested articles, including which libraries have the articles, is stored in the system. When requesting these articles again, the source can be located immediately.

9. Closing observations

The concept of EEE resembles the Computer Integrated Manufacturing (CIM) initiatives of the mid-1980s. While widely viewed as having failed, CIM has evolved into many different concepts, one of which is business process reengineering. EEE will succeed where CIM failed, in that the EEE concept and its three goals of full integration, process engineering and broad decision support address the "big picture" and simultaneously link it to the underlying architecture. EEE provides an overall strategy for evolving in the age of electronic commerce. For those enterprises that do so quickly, the EEE infrastructure should help achieve competitive advantage.

NJIT's CIS Department has embraced the EEE approach as a leading research and development activity. We plan, in cooperation with other departments and partners from industry, to further refine and test the proposed SEEE architecture and to specify it in a (formal) architecture language. We are also working on problems related to business process management and modeling. Existing tools and environments are tested for their suitability to provide the architecture with the necessary underlying infrastructure. Case-studies and tests at sites run by the EEE consortium's industrial partners have and will form an integral element of our efforts.

References

[1] Keith Bennett. Legacy systems: Coping with success. *IEEE Software*, 28(1):19–23, January 1995.

[2] I. Bhandari and H. A. Simon. Models for test selection. *IEEE Transactions on Systems, Man, and Cybernetics*, 25(10):1349–1364, 1995.

[3] M. Bieber. On integrating hypermedia into decision support and other information systems. *Decision Support Systems*, 14:251–267, 1995.

[4] Michael Bieber and Charles Kacmar. Designing hypertext support for computational applications. *Communications of the ACM*, 38(8):99–107, 1995.

[5] R. P. Bostrom, T. W. Watson, and S. T. Kinney. *Computer Augmented Teamwork: A Guided Tour.* Van Nostrand Reinhold, 1992.

[6] M. R. Cagan. The HP softbench environment: An architecture for a new generation of software tools. *Hewlett-Packard Journal*, pages 36–47, June 1990.

[7] R. L. Daft and R. Lengel. Organizational information requirements, media richness and structural design. *Mangement Science*, 32(5):554–571, 1986.

[8] A. R. Dennis, J. F. George, L. M. Jessup, J. F. Nunamaker, and D. R. Vogel. Information technology to support electronic meetings. *Mangement Information Systems Quarterly*, 12:591–624, 1987.

[9] G. L. DeSanctis and R. B. Gallupe. A foundation for the study of group decision support systems. *Mangement Science*, 33:589–609, 1987.

[10] S. J. Gelman, F. M. Lax, and J. F. Maranzano. Competing in large-scale software development. *AT&T Technical Journal*, 72(6):2–11, 1992.

[11] L. R. Hoffman. *The Group Problem Solving Process: Studies of Valance Model.* Praeger, 1979.

[12] ISSI. ProSLCSE process toolset. Technical report, September 1994.

[13] A. Kusiak. Dependency analysis in constraint negotiation. *IEEE Transactions on Systems, Man, and Cybernetics*, 25(9), 1995.

[14] Hector Levesque, Ray Reiter, Yves Lespérance, Fangzhen Lin, and Richard Scherl. GOLOG: a logic programming language for dynamic domains. *Journal of Logic Programming*, (forthcoming).

[15] Q.H. Liu and P.A. Ng. A Browser of Supporting Vague Query Processing in an Office Document System. *Journal of Systems Integration*, 5(1), 1995.

[16] J. E. McGrath and A. B. Hollingshead. *Groups Interacting with Technology.* Sage Publications, 1994.

[17] F.S. Mhlanga, Z. Zhu, J.T.L. Wang, and P.A. Ng. A New Approach to Modeling Personal Office Documents. *Data and Knowledge Engineering*, 17:127–158, 1995.

[18] Jakob Nielsen. *Multimedia and Hypertext: The Internet and Beyond.* AP Professional, 1995.

[19] C. V. Ramamoorthy. A structural theory of machine diagnosis. In *Proceedings of the IEEE Spring Joint Computing Conference*, pages 743–756, 1967.

[20] A. R. Rana, M. Turoff, and R. Czech. Inquiring systems' valiadtion approaches as determinants of the media richness for technology supported group tasks. In *Proceedings of the America's Conference on Information Systems*, 1996.

[21] A. R. Rana, M. Turoff, and S. R. Hiltz. Task and technology interaction (tti): A theory of technological support for group tasks. *(under review).*

[22] Richard B. Scherl and Hector J. Levesque. The frame problem and knowledge producing actions. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 689–695, 1993.

[23] B. Shneiderman. *Designing the User Interface.* Addison-Wesley, 1992.

[24] M. M. Tanik and E. S. Chan. *Fundamentals of Computing for Software Engineers.* Van Nostrand Reinhold, 1991.

[25] Manfred Thuering, Joerg Hannemann, and Joerg Haake. Hypermedia and cognition: Designing for comprehension. *Communications of the ACM*, 38(8):57–69, 1995.

[26] Jiangling Wan. *Integrating Hypertext into Information Systems through Dynamic Linking.* PhD thesis, New Jersey Institute of Technology, CIS Department, Newark, NJ 07102, 1996.

[27] J. T. L. Wang and C. Chang. Fast retrieval of electronic messages that contain mistyped words or spelling errors. *IEEE Transactions on Systems, Man and Cybernetics*, 1996 (forthcoming).

[28] J.T.L. Wang, F.S. Mhlanga, Q.H. Liu, W.C. Shang, and P.A. Ng. *The Impact of CASE Technology on Software Processes*, chapter Database Support for Software Documentation: The TEXPROS Project, pages 103–185. World Scientific Publishing, Singapore, 1994.

[29] R. T. Yeh, S. N. Delcambre, and M. M. Tanik. Cosmos: An architecture for improving process capability maturity. In *3rd IEEE International Conference on Systems Integration*, pages 1166–1175, August 15-19 1994.

[30] Zhijian Zhu, Qianhong Liu, James McHugh, and Peter Ng. A Predicate-Driven Document Filing System. *Journal of Systems Integration*, 6(3), 1996.