

Experiences with Workflow Management: Issues for the Next Generation

Kenneth R. Abbott
XSoft, A Division of Xerox
3400 Hillview Avenue
Palo Alto, CA 94303, USA
E-Mail: kabbott.osbu_north@xerox.com
Telephone: 1 415 813 7293

Sunil K. Sarin
XSoft, A Division of Xerox
Four Cambridge Center
Cambridge, MA 02142, USA
E-Mail: sarin@xait.xerox.com
Telephone: 1 617 499 4426

ABSTRACT

Workflow management is a technology that is considered strategically important by many businesses, and its market growth shows no signs of abating. It is, however, often viewed with skepticism by the research community, conjuring up visions of oppressed workers performing rigidly-defined tasks on an assembly line. Although the potential for abuse no doubt exists, workflow management can instead be used to help individuals manage their work and to provide a clear context for performing that work. A key challenge in the realization of this ideal is the reconciliation of workflow process models and software with the rich variety of activities and behaviors that comprise "real" work. Our experiences with the InConcert workflow management system are used as a basis for outlining several issues that will need to be addressed in meeting this challenge. This is intended as an invitation to CSCW researchers to influence this important technology in a constructive manner by drawing on research and experience.

KEYWORDS: Workflow, business process reengineering.

INTRODUCTION

Workflow software is currently a hot technology. Computer industry analysts tout workflow as the "technology of the 1990s" and predict that workflow will become part of all office applications in the next decade.

The need for workflow software has arisen out of the proliferation of desktop computing and networks. While the amount of computing power and intelligence available on the desktop has grown dramatically in the past 15 years, overall productivity of the office worker has scarcely been affected. As businesses cope with global recession and global competition, they have been forced to improve productivity. The greatest opportunities for dramatic improvements in productivity lie in improving and streamlining multiperson business processes.

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association of Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.

CSCW 94- 10/94 Chapel Hill, NC, USA
© 1994 ACM 0-89791-689-1/94/0010..\$3.50

Workflow software provides the infrastructure to design, execute, and manage business processes on a network. End-user productivity is improved by eliminating overhead time spent in "setting up" and "tearing down" (e.g., collecting and disseminating the information needed for performing tasks). Lag time spent in routing work from one person to another is reduced. Using workflow, managers are better able to monitor work in process, allocate resources, and receive ongoing feedback about status.

Businesses contemplating adoption of workflow technology take a broad, process-oriented perspective on the costs and benefits. Costs include not just hardware and software costs, but also people costs associated with training, operation, and administration of workflow-based systems. Benefits accrue to all the different types of workflow users—task performers, managers, process designers and administrators, and executives—all of whom participate in the business process. Notice that automation of individual tasks directly affects only one type of user (the task performer). Managers, process designers and administrators, and executives perceive workflow as an information technology (IT), not just an automation technology. As an information technology, workflow provides improved information feedback and tools to monitor, manage, and control business processes, with or without task automation. To get an accurate picture of the business impact of workflow, businesses must add up the costs and benefits across all the different beneficiaries (users) of workflow, and not focus solely on one type of user.

It is important to distinguish today's "workflow automation" (an unfortunate choice of words which has led to unnecessarily negative misperceptions of workflow) from the "task automation" that was the focus of early office automation work. The emphasis in workflow management is on using computers to help manage business processes that may be comprised of many individual tasks, not on using computers to automate the individual tasks. The latter may be applied selectively to some tasks, but such task automation is not a prerequisite for using and benefiting from workflow.

Such a process-oriented (as opposed to task-oriented) view of work allows businesses greater opportunities for efficiency and cost reduction. The current business process

reengineering (BPR) trend [7] is based on this observation. In many cases, just the act of analyzing business in terms of processes has led to breakthroughs in efficiency without automation. In the context of BPR, workflow management provides a means for enacting reengineered processes and for gathering live information about the actual performance of these processes. This information can be fed back into the BPR activity and used to drive further process redesign and improvement.

History

At least two technology trends have contributed to the development of workflow systems. Early work in office procedure systems such as SCOOP [16] and OfficeTalk-D [5] paralleled development of integrated office automation systems. This early research did not lead to any commercial products. One reason for this was the inflexibility of procedure definitions in dealing with "real" work (an argument that is sometimes unfairly applied against today's workflow systems), but this was not the only reason. Computers and networks were not as widely used 10 to 15 years ago as they are today; it was hard to apply a computer system to help manage work when very little of the work itself was done using a computer. This particular barrier clearly no longer exists.

On another track, development of imaging applications rapidly led to inclusion of routing and tracking capabilities for images. Developers and users of such routing applications rapidly realized that the routing capability was an important component of managing their business processes, independent of the data being routed. Many of the established workflow products today started as image management applications.

Generation	Major characteristics
First:	<ul style="list-style-type: none"> • application-specific • workflow capabilities expressed in particular applications (e.g., image, document management) • hardcoded process definitions • closed and proprietary
Second:	<ul style="list-style-type: none"> • factored application • workflow capabilities factored out from application domain • workflow as a separate application • limited selection of 3rd-party tools • process definitions tailorable through script language
Third (current):	<ul style="list-style-type: none"> • tailorable service • generic workflow services accessible to other applications through APIs • open, standards-based architecture • full integration of 3rd-party tools - tailorable through GUIs • proprietary workflow interfaces and interchange formats
Fourth (next):	<ul style="list-style-type: none"> • embedded enabler • workflow services fully integrated with other middleware services (email, desktop management, directory) • standardized interfaces and interchange formats • "workflow-enabled" applications • ubiquitous but invisible

Table 1. Generations of Workflow Technology

From these beginnings, workflow management systems may be considered to have evolved through three "generations" of technology. These are shown, along with the next or fourth generation that many project, in Table 1. We elaborate on the current and next generation in more detail below.

There are now well over a dozen workflow management systems available on the market. These can be characterized along the following dimensions, which serve as differentiators among the products. (We avoid mention or comparison of specific products, including ours, because we want to focus on technology issues rather than particular product features and functions.)

- **Design versus runtime.** Design-oriented workflow products emphasize the creation, modeling, analysis, and simulation of workflow processes. They typically implement a particular analysis/design methodology. Runtime workflow products are engines that provide generic routing and tracking services to applications.
- **Mail- versus database-driven.** Mail-driven workflow systems build on electronic mail as the basic underlying mechanism for routing and presenting work to users. Mail-driven workflow occupies the low- and mid-range of the workflow market; mail-based products are typically cheap and oriented toward informal processes and naive end-users. Work design and management of work in process is simplistic. However, mail-driven workflow achieves wide-area distribution by piggybacking on mail transport systems. Database-driven workflow systems use underlying database technology to drive the process. Routing and status information is stored in tables that are queried by clients. Database-driven workflow systems are in the mid- and high-end market segments. They have strong management and reporting capabilities, and robust operational characteristics in large scale deployment. Although they may be used in end-user applications, they require expert knowledge to set up and configure.
- **Document- versus process-oriented.** Document- (or data-) oriented workflow systems associate routing information with the particular data objects being worked on. Folder management and image routing systems fall into this category. They are good for handling manual, paper-based procedures electronically. Process-oriented workflow systems model the work process as a sequence of steps. Data objects are attached to steps in the process, but different objects can be routed at different steps in the process. Process-oriented systems can handle more complex workflows, but process analysis and definition is more taxing and may be less intuitive in simple cases.

An interesting parallel exists between well-established database management system (DBMS) technology and nascent workflow technology. Both are fundamental technologies which underlie the whole spectrum of business applications. They are related through the process/data duality which appears repeatedly in computer science:

DBMS corresponds to data; workflow corresponds to process. Over time, we expect workflow architecture and technology to mature and grow in a pattern similar to DBMS architecture and technology. That is, workflow will recapitulate similar phases of evolution: development of runtime engine capabilities; development and standardization of conceptual models and design methodologies; proliferation of application development tools; and canonization as an underlying commodity service.

The "next generation" of workflow, where workflow is simply another invisible capability that permeates all (or most) applications, is still a vision. This vision includes blurring many of the current distinctions among systems (e.g., achieving both the convenience and wide-area distribution of mail and the robust management and reporting capabilities of database systems). Many of the system architecture and infrastructure issues implied by this vision are beginning to be addressed by industry. For example, the Workflow Management Coalition, comprising over 50 vendors and users, is working towards standards for interoperability and interchange among heterogeneous workflow systems.

We believe that, in addition to addressing system-level issues, a parallel development is needed to achieve a closer and more natural match between the models and capabilities of workflow management systems and the way people work, collectively and individually. While management rarely needs to be convinced (in our experience) of the need to adopt workflow technology, the ultimate success of the technology will require acceptance by the individuals whose work will be managed using this technology. This latter evolution is the focus of this paper.

Related Work

Although InConcert has some differences with other workflow products on the market (such as [11]), this paper is not about such differences. Most of the successes and problems we describe are common across systems; our objective is to highlight user issues that affect the future of workflow technology in general.

Although there seems to be little mention of workflow in the CSCW research literature (perhaps because of the negative connotations that the term seems to carry), there is considerable overlap between many of the issues we raise and those addressed by recent research into "collaborative work processes" (such as [2, 14]). A contribution that we wish to make in this paper is to reinforce such research based on actual experiences with deploying a commercial workflow system. In addition, while the issues & challenges we describe make reference to related research in particular cases, there are many issues and directions we present that we believe have not been emphasized in other research.

Organization of This Document

The next section briefly summarizes the capabilities of one particular "current generation" workflow management system, InConcert, and our experiences with it. The section that follows (Usability Issues & Challenges) then elaborates on some issues and challenges that will need to be addressed in the next generation of workflow technology.

THE INCONCERT WORKFLOW MANAGEMENT SYSTEM

In terms of the above characterization, InConcert is a runtime database- and process-oriented workflow management system. This section briefly summarizes the design principles of InConcert and how they are reflected in system features, and then reviews our experiences and lessons learned in deploying InConcert in a variety of customer applications.

Design Principles

Explicit process representation. This is the most basic capability of any workflow management system, and in InConcert is reflected in a process model (described in more detail in [13]) that supports hierarchical decomposition of tasks, sequencing dependencies among tasks, and assignment of tasks to roles that are placeholders for users who will perform tasks.

Process enactment support. An active process instance can be created from a process definition or template. Tasks in a process instance go through the following life-cycle: Waiting (not yet ready because at least one preceding task is not yet complete), Ready (no longer waiting for any preceding task), In Process (an assigned user has taken responsibility for the task), and Completed (the given user has declared that the task is done, allowing subsequent tasks to become ready).

Help individuals keep track of their work. Individuals use the InConcert task organizer application (one of several GUI applications) to see the tasks assigned to them. These can be organized using a variety of criteria, such as priority, deadline, or the capacity/role in which the user was assigned the task. The user may select a task and ask for further information (such as to view the process to get an idea of why they are doing this task and for whom), or begin working (see below) on it.

Provide easy access to the content of the work. Each task has associated with it a set of documents, where a document is any kind of abstract object that needs to be manipulated by a task. This eliminates the burden (in most cases) of the user having to track down the information necessary to perform a task. When a user "opens" a task icon to work on that task, the documents in that task appear as icons within the task context; further, each document icon can in turn be opened to invoke the appropriate application (e.g., word processor or spreadsheet) for the given document. (The InConcert architecture allows arbitrary third-party applications to be associated with the documents in a task.)

Model the flow of work, not documents. Documents do not "flow" between users or tasks in InConcert. Rather, a process provides a shared information space which includes any number of documents that are needed within the process. This supports processes that need to manipulate more than one document, and also allows multiple tasks to access the same document in parallel.

Support status monitoring and reporting. Management can get current information about the progress of or history of events in a specific process, or of tasks that are overdue, etc. These are provided both through GUI applications and the reporting capabilities of the underlying relational DBMS.

Selective automation. The default assumption in InConcert is that a task will be performed by a human being, who will exercise whatever judgment is needed to achieve the desired objective. For cases where no such judgment is needed, InConcert allows a task to be assigned to a program that will execute it when it becomes ready. InConcert also provides a "trigger" mechanism [10] that supports automatic notification (via electronic mail) as well as invocation of customer-supplied "agent" applications on the occurrence of a variety of events such as a task being completed or becoming overdue.

Process mutability. InConcert allows a process to be modified during enactment, by allowing tasks or dependencies to be added or removed, roles reassigned, and so on. (These modifications can be accomplished by direct manipulation through a graphical user interface.) This is intended to allow the individual(s) responsible for a process to react to unanticipated events and exceptions. Thus, a process definition serves as a guide that suggests the desired or normal flow of tasks for performing the necessary work, rather than a straitjacket from which the users cannot deviate.

Experiences and Lessons Learned

InConcert is being successfully applied in a wide range of applications, with a wide range of process management policies:

- Electronic information publication and delivery
- Service order processing
- Loan processing
- Laboratory equipment purchasing
- Logistics support
- Computer Aided Software Engineering (CASE)
- Software problem reporting
- Employee performance-appraisal processing

One important lesson learned from deploying InConcert in the above environments is that a few simple things nicely executed can go a long way, and many of the benefits advertised in the Introduction are in fact achieved. Processes move faster (than before) because less time is wasted while important forms and documents are buried in piles on people's desks. Additionally, managers are able to quickly ascertain the status of one or more processes. A significant

benefit quoted by one customer was not having to run around to many co-workers' desks trying to find the information needed to perform their task.

An open and extensible architecture [10] has made it easy to adapt InConcert to many different environments. InConcert makes use of industry-standard components as much as possible, such as a relational DBMS (from any of several specific vendors) for storing workflow data. InConcert offers an Application Programming Interface (API) that can be used to build custom workflow applications; our own GUI applications use this same API. The API is expressed in terms of operations on abstract object types. The object model is extensible, and the ability to define custom object subclasses and attributes facilitates the development of workflow applications that are tailored to the customer's unique requirements. These benefits are in many ways independent of the particular workflow model.

Not all customer feedback is uniformly enthusiastic, of course. In some cases, automation of an existing process as a "pilot" exercise, with no attempt at process improvement, turned out to be too cautious and conservative: Users' negative feelings about the existing process were exacerbated when they were presented with on-line electronic versions of the paper forms that they already disliked. The seemingly mundane issues of installation and administration, which arise with any software product, become more visible and critical with a "glue" technology like workflow that needs to work with other applications and system software.

A different class of comments from customers has to do with applying and matching the workflow process model to the way work is actually performed. Although we explicitly designed in the flexibility to modify a process, users had difficulties translating what they wanted to do (e.g., "delegate" a task to another user) into low-level operations on the process structure; it therefore seems necessary to provide higher-level user commands that package the more common kinds of process modifications. This is one of several facets of the more general usability problem, which is discussed in the next section.

USABILITY ISSUES & CHALLENGES

The value of process definitions in themselves, even if they are only approximations of "real" work, has already been established in the business community. A major challenge for workflow management systems, we believe, is to be able to make use of process descriptions during enactment of a process. This requires that the runtime workflow "engine" be capable of accommodating the variety of behaviors that occur when performing work. Some of these issues, and directions for addressing them, are described below.

Integrating Procedural and Non-procedural Work

It is useful to distinguish between the procedural and non-procedural content of a work process. Procedural content refers to the structured aspects of the process. Non-

procedural content corresponds to unchoreographed interactions between people. The mix of procedural and non-procedural content depends on the process, the business, and the type of application. Some business processes, e.g., benefit claim processing, are high in procedural content. Others, like legal case management, are high in non-procedural content.

An explicit representation of a process, like a PERT chart, is useful for representing procedural content. Explicit representation makes it possible to analyze, simulate, manage, and administer the business process.

Non-procedural behaviors typically represent actions taken when handling "exceptions" to the structured process representation. Some examples of exceptions that arise frequently and must be handled are:

- reassigning work from one person (who is unexpectedly unable or unavailable to complete the work) to another person;
- overriding the process when a required approval has not been received and is impeding progress;
- sending work back to someone who sent it on without completing it;
- negotiating a new deadline when a task is overdue.

It is not possible to anticipate all possible exceptions in a process; even if it were, representing them and their responses explicitly would make the process description lose its value as it would become hopelessly unwieldy and hard to understand.

A challenge for workflow systems, then, is to combine and manage both procedural and non-procedural interactions appropriately. "Conversation-based" models (e.g., [6]) are appropriate for non-procedural interactions but are often overkill for structured processes where agreements to perform work do not need to be negotiated every time. It is therefore necessary for users to be able to switch back and forth gracefully between these two modes of interaction. An interesting example of this is the work of [1] which allows a user in the WoORKS workflow system to "escape" to the UTUCS mail system (e.g., when it is necessary to open a negotiation or to request information). The authors of this work also note that they were able to considerably simplify process descriptions by removing many "requests for information" (which weren't always necessary) and treating them as exceptions.

When confronted with an exceptional situation, users should be able to obtain help in determining how to proceed. A knowledge-based "planning" system, which includes explicit representation of the goal(s) of a process, may be useful in suggesting revisions to the process plan, although work to date in this area (e.g., [3]) does not seem to have been applied during the runtime enactment of a process by multiple participants.

Support for External Activities and Meetings

Until workflow systems become truly ubiquitous, users of such systems will always need to access users and resources that live "outside" the domain of the particular workflow system. Interoperation between electronic mail and workflow (in both directions) is one kind of support that is needed, as is interoperation between disparate workflow systems.

Meetings are another common work activity that are usually performed outside a workflow system. Meetings are typically generators of project and action plans, and a workflow management system can capture and coordinate the execution of such plans. In one pilot project, InConcert was integrated with meeting facilitation software. Using these facilitation tools, meeting participants would generate action items for individuals, and these action items were then synthesized into a workflow process that was used to coordinate work that needed to be done between meetings.

The above approach, which coordinates between-meeting activities, does not treat the meetings themselves as part of a workflow process. More often, meetings represent significant milestones in a process, and need to be represented in the process and monitored. However, runtime support for holding meetings in the context of a workflow process is currently weak. Typically a meeting must be modeled as a "proxy" task that is assigned to one user who is responsible for recording the results of the meeting (i.e., minutes, task assignments, decisions as to what task(s) to perform next) and marking the proxy task complete. This is similar to (and no better than) the representation of other kinds of external work (e.g., making a phone call) in a workflow process.

Improvements in support for meetings is both desirable and possible. The workflow system should be able to associate additional participants so that the proxy task can appear in all participants' task lists. Knowledge of multiple participants would also make it possible to make use of a calendar application in scheduling the meeting.

Further extension would allow different outputs of the meeting to be associated with different contributors rather than with the one user who is reporting the results. In the long run, the workflow management system should be able to capture enough information to support invocation of a real-time "conference" [4, 8, 12] in which all (or many) of the participants are on-line.

Mechanisms versus Policies

A mechanism is a generic capability or feature, whereas a policy is a prescribed use of the features. Policies for various kinds of activities vary from one industry and application domain to another. For example, in some business processes, reassignment of work always involves a manager. In other, more time-critical, processes, work may be automatically reassigned (e.g., if nobody has acted on a customer call within a prescribed time period). In some application domains, there is value in being able to ensure

that a process is adhered to precisely and that only an authorized manager can modify it; in others, it is essential that the people involved in the process be able to modify it on a case-by-case basis. A flexible system, one which does not impose a specific policy, should make it possible to package functions differently for different kinds of users (e.g., managers versus end-users—a given person need not be always restricted to being one "kind" of user), so that each different user interface supports the set of functions that conform to the local policy for that kind of user activity.

When attempting to provide flexible mechanisms that are "policy-neutral," there is an unavoidable tradeoff between generality and ease of use. InConcert's trigger model, for example, provides a powerful and general mechanism for responding to events (including the passage of time), but currently requires some sophistication and programming on the part of the customer to realize a coherent policy. This needs to be remedied by providing sample policies that use the mechanisms in specific stylized ways (analogous to "styles" in word processing systems), which a customer can choose from and adapt as desired.

Access control is another kind of mechanism for specifying and enforcing policies; this is discussed in the next subsection.

Authorization and Access Control

Although it is important to provide the flexibility to adapt process structure in a variety of ways (such as the exception examples described above), it is also often important to have control over who will be able to make such changes. We addressed this issue in InConcert by providing an access control model based on objects and privileges. For example, modifications to the structure of a process can be performed only by users who possess the Update privilege on the given process object. The ability to grant and revoke privileges is itself controlled using the Manage privilege.

Such an object-oriented access control model has an established history in operating systems and database management systems, which is why we chose to use it. However, it becomes apparent what the limits to this model are when we try to support collaborative work processes.

Consider a workflow process in which a document is to be updated by the user performing a particular task. The document may have an independent existence, outside of any process, and may have associated access control information (e.g., Update privileges) specifying which users can modify it. What happens if the manager/owner of a process assigns a given task to a user who does not possess the necessary privileges to update the given document? The assigned user would encounter a runtime error when they try to perform the task. This, of course, is undesirable. At the very minimum, the system should be able to warn the manager of this inconsistency so that it can be fixed (by

reassigning the task, or by arranging to give the chosen user the necessary privilege) before problems occur.

The workflow system could be further improved by generalizing the access control model. Typically the "principals" or "subjects" to whom privileges can be granted are users, groups of users, and "the public." If this were expanded to allow privileges to be granted to a "role" in a process, then the process manager (in the above example) could temporarily delegate the necessary document privileges to the user who had been assigned (via the role) to the task in question. Such privileges would be applicable only in the context of the given process and its tasks, and the manager would not have to worry about whether he had made permanent changes to the access control list of the document that he would need to revoke later.

Expressive Process Structures

Although task hierarchies and dependency networks are a useful abstractions for process description, they are not always sufficient to represent the flow of work as it is actually performed. A simple example is that it is often necessary (under time pressure, say) to start work on a given task before completing all of its preconditions (i.e., predecessor tasks completed).

Similarly, although hierarchical task decomposition makes it easy to view progress at different levels of detail, top-down recursive decomposition is not always the most natural way to construct the hierarchical structure. Users (process designers) frequently need to lay out a detailed network of low-level tasks before they start aggregating groups of tasks (thus building the hierarchy bottom-up).

Defining synchronization dependencies across hierarchy and process boundaries is also frequently necessary. Such needs often arise dynamically, when a user in one process realizes that he or she needs some input from a task in a different process. Or, a user responsible for a given task in a process may decide to start a separate "subprocess" with its own shared role and information space (as opposed to subtasks within the same process), and to have the original task wait for completion of the new process. The "obligation" mechanism proposed for ConversationBuilder [2] is an example approach of how to meet this need.

Evolutionary Process Development

When a process needs to be initiated to achieve some goal, not all the details of how the goal will be achieved will necessarily be known in advance. This is especially true in application areas where most processes are not repetitive. The following are examples of capabilities needed:

Delayed binding. It should be possible to separate the "what" of the process (its goal) from the "how" (the procedure or plan), and to delay the selection of the latter. It should also be possible to switch to a different plan if the one originally assigned is determined to be inappropriate. This flexibility should be available at all levels of decom-

position of a process (i.e., allowing the individual responsible for a given task to select the procedure they deem appropriate). A similar view is taken in the Regatta system [15].

Finding process definitions. Users should be able to easily browse through a "library" of available process (and process component) definitions to find the one appropriate to what they are doing. InConcert currently supports a simple query mechanism, based on names and other process attributes, to assist the user in doing this. An example of a more general and powerful mechanism is the process inheritance hierarchy proposed by [9], in which increasingly specialized processes are defined as incremental refinements to more general ones.

Process evolution and learning. It should be possible to build the process definition "library" or hierarchy based on experience in executing processes. On completion of a process (or before that, if necessary) it should be possible to review how it was performed and to abstract information from it in order to "save" a process definition, or components thereof, in the library.

Change control. We have observed that, as businesses developed their processes, they needed to make use of many of the version- and configuration-control concepts that are now common for managing documents and other data. It should be possible to apply process modifications to process definitions as well as to individual process instances. Process (and process component) definitions should be version-managed, and it should be clear from which version of a process definition a particular process instance was derived. For process definitions that are composed from process library components, it should be possible to control the propagation of component changes to the composite process definition.

CONCLUSION

Driven by global competition and global recession, businesses are adopting technologies that improve productivity in modern computer-based office environments. Through techniques such as business process reengineering, businesses are adopting and implementing a broad process-oriented approach to improved business efficiency, rather than focusing on task automation. Workflow technology, derived from early business productivity applications including integrated office systems and image management, is evolving into a generic underlying technology for business applications. In this paper, we have identified some of the key trends in current workflow systems, and extrapolated that the evolution of future architecture and standards will follow a path similar to the evolution of database technology. Based on our experience with InConcert, a commercially-available third-generation workflow product, we have elaborated a number of issues (such as integrating non-procedural work and support for dynamic process evolution) encountered in actually applying a current workflow system in business applications. We believe these issues are not limited in

scope to any particular workflow management system, and offer them as an opportunity for research to drive further improvements in this important technology area.

ACKNOWLEDGMENTS

We would like to thank Dennis McCarthy for his suggestions and comments on several drafts of this paper. Of our many other colleagues who have worked to make InConcert a reality, we would like to thank Tom Dwyer, Lisa Hebert, and Bill Shelley for their reports of customer experiences. And we would like to thank our customers who took the bold steps that made this paper possible.

REFERENCES

1. Agostini, A., De Michelis, G., Grasso, M.A., and Patriarca, S. Reengineering a business process with an innovative workflow management system: A case study. In *Proc. ACM-SIGOIS Conference on Organizational Computing Systems (COOCS'93)* (November 1993), pp. 154-165.
2. Bogia, D.P., Tolone, W.J., Kaplan, S.M., and Tribouille, S.M. de la. Supporting dynamic interdependencies among collaborative activities. In *Proc. ACM-SIGOIS Conference on Organizational Computing Systems (COOCS'93)* (November 1993), pp. 108-118.
3. Croft, W.B., and Lefkowitz, L.S. Using a planner to support office work. In *Proc. ACM-SIGOIS Conference on Office Information Systems* (March 1988), pp. 55-62.
4. Crowley, T., Milazzo, P., Baker, E. Forsdick, H., and Tomlinson, R. MMConf: An infrastructure for building shared multimedia applications. In *Proc. Conference on Computer-Supported Cooperative Work (CSCW'90)* (October 1990), pp. 329-342.
5. Ellis, C.A., and Bernal, M. Officetalk-D: An experimental office information system. In *Proc. ACM-SIGOA Conference on Office Information Systems* (June 1982), pp. 131-140.
6. Flores, F., Graves, M., Hartfield, B., and Winograd, T. Computer systems and the design of organizational interaction. *ACM Trans. Office Information Systems* 6, 2 (April 1988), 153-172.
7. Hammer, M., and Champy, J. *Reengineering the Corporation*. HarperCollins Publishers, 1993.
8. Lantz, K. An experiment in integrated multimedia conferencing. In *Proc. Conference on Computer-Supported Cooperative Work (CSCW'86)* (December 1986), pp. 267-275.
9. Malone, T.W., Crowston, K., Lee, J., and Pentland, B. Tools for inventing organizations: Toward a handbook of organizational processes. In *Proc. 2nd IEEE Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET-ICE)* (April 1993), pp. 72-82.
10. McCarthy, D.R., and Sarin, S.K. Workflow and transactions in InConcert. *IEEE Bulletin on Data Engineering* 16, 2 (June 1993), 53-56.
11. Medina-Mora, R., Winograd, T., Flores, R., and Flores, F. The action workflow approach to workflow

- management technology. In *Proc. Conference on Computer-Supported Cooperative Work (CSCW'92)* (November 1992), pp. 281-288.
12. Roseman, M., and Greenberg, S. GroupKit: A groupware toolkit for building real-time conferencing applications. In *Proc. Conference on Computer-Supported Cooperative Work (CSCW'92)* (November 1992), pp. 43-50.
 13. Sarin, S.K., Abbott, K.R., and McCarthy, D.R. A process model and system for supporting collaborative work. In *Proc. ACM-SIGOIS Conference on Organizational Computing Systems (COOCS'91)* (November 1991), pp. 213-224.
 14. Singh, B., and Rein, G.L. Role Interaction Nets (RINs): A Process Description Formalism. Microelectronics and Computer Technology Corporation, Austin, Texas. MCC Technical Report CT-083-92, July 1992.
 15. Swenson, K. Visual support for reengineering work processes. In *Proc. ACM-SIGOIS Conference on Organizational Computing Systems (COOCS'93)* (November 1993), pp. 130-141.
 16. Zisman, M.D. *Representation, Specification, and Automation of Office Procedures*. Ph.D. dissertation, Wharton School, Univ. Pennsylvania (1977).