

Activity Explorer: Activity-centric collaboration from research to product

W. Geyer
M. J. Muller
M. T. Moore
E. Wilcox
L.-T. Cheng
B. Brownholtz
C. Hill
D. R. Millen

Activity Explorer is the first product from IBM that supports the notion of activity-centric collaboration. This new collaboration paradigm organizes and integrates resources, tools, and people around the computational concept of a work activity, with the goal of increasing work quality and efficiency. In essence, activity-centric collaboration is an important and compelling example of contextual collaboration. Activity Explorer emerged from a multiyear research effort on activity-centric collaboration. This paper presents an overview of the most significant milestones of this research program and highlights the most interesting findings. The research behind Activity Explorer is based on many empirical studies, design explorations, and infrastructural engineering and technical simulations. We demonstrate how our research not only influenced product direction, but also the IBM vision for activity-centric collaboration.

INTRODUCTION

Activity Explorer (AE), a part of the IBM Workplace* product family,¹ introduced the concept of activity-centric collaboration to the market. The basic idea behind activity-centric collaboration is simple: Reorganize collaboration to reflect the work being done rather than the technologies that support the work. An activity can be defined as a logical unit of work that incorporates all the tools, people, and resources needed to get a job done. Just a few examples of activities are: preparing an executive meeting, planning a conference, closing a sale, planning and executing the conversion of multiple bank branches to new computing systems, and writing or responding to a request for proposal.

This paper provides an overview of the multiyear research behind AE and activity-centric collabora-

tion. It describes the journey from research prototype to product. We summarize the concepts, findings, and outcomes of this successful research-to-product experience and present some new results of follow-up research on alert management, partial sharing, and design explorations.

The idea for AE emerged from our research on new “instant collaboration” techniques in the context of our Reinventing E-mail project.²⁻⁴ The problem of e-mail overload⁵ was a major trigger for our work on

©Copyright 2006 by International Business Machines Corporation. Copying in printed form for private use is permitted without payment of royalty provided that (1) each reproduction is done without alteration and (2) the Journal reference and IBM copyright notice are included on the first page. The title and abstract, but no other portions, of this paper may be copied or distributed royalty free without further permission by computer-based and other information-service systems. Permission to republish any other portion of the paper must be obtained from the Editor. 0018-8670/06/\$5.00 © 2006 IBM

activity-centric collaboration. Research indicated that people use e-mail not only to communicate, but to manage various types of work activities.^{6,7} These types of work activities are not well supported by e-mail because—for activities that extend over long periods of time or over large numbers of participants—it rapidly becomes unmanageable.⁵ There is very little support for a structured and rich collaboration within e-mail. At the other extreme, structured shared workspaces provide better support for making sense of large bodies of messages and files. However, they are relatively labor-intensive to initiate and manage over time, which makes them cumbersome and ill-suited for use on small-scale or short-term collaborations. The AE implementation of activity-centric collaboration bridges this gap between informal ad hoc communications and highly structured workspaces, and can help people move their work activities out of the inbox, while still being able to maintain the lightweight character and flexibility of e-mail.

In the next section, we define activity-centric collaboration from a broader perspective and explain its benefits and values to business users. In the section “Activity Explorer,” we provide an overview of the concrete AE user interface by illustrating how AE can be used to collaborate in an activity, and we describe the underlying, initial design rationale of AE. In the section “Activities infrastructure research,” we describe the architectural model for AE and place it in the broader space of activity-centric collaboration applications. From a systems perspective, AE is only one instance of an activity-centric collaboration client. Using the same activities backend, different client user experiences can be built.

We deployed AE multiple times during our research over the past three years, which gave us the opportunity to collect valuable data. In the section “Empirical and design research,” we present the most intriguing results from our work, illustrating how our research influenced design and development. Field work, participatory analysis, usability evaluations, customer reviews, and design explorations deeply informed our research with the real work that we hoped to support. This helped us verify and refine our concepts and gain useful insight for future product versions.

Our project was one of several research explorations of the concept of activity management, which is

concerned with providing a rich and diverse set of resources to teams and individuals as they plan, execute, and reuse work. Our project has engaged in a dialogue with the Unified Activity Management (UAM) project, which has emphasized a task-oriented approach to activity management.^{8–11} Together, AE and UAM have deeply informed the next version of the IBM Notes* client and the IBM vision for activity-centric collaboration. In the section “Designing the future: Next-generation AE,” we discuss how AE concepts are evolving in the next product generation, currently under development. One of the goals of this project is to provide an extensible and flexible programming model that allows building customized user experiences. Another goal is to deliver a new, lightweight, and general-purpose activity-management client that supports the collaborative capabilities of AE ad hoc activities, but also incorporates the latest work on patterns, that is, templates for repetitive activities from the UAM research project.^{8–10}

ACTIVITY-CENTRIC COLLABORATION

Collaboration technologies can be arranged on a continuum of specificity¹² as indicated in *Figure 1*. At the left side of the spectrum are ad hoc communication tools that support informal, unstructured work. At the right side are technologies that support formal, structured business processes. To simplify the discussion, we categorize technologies on the continuum into three boxes with shared workspace systems in the center.

At the left of Figure 1, ad hoc collaboration systems such as e-mail and chat are lightweight and flexible and provide good dynamic support for short-term communication needs.^{13,14} Collaboration is usually managed and controlled by end users. This is certainly one of the reasons why these tools are so popular and why they are used for all sorts of collaborative activities.¹⁵ However, for those collaborations that extend over longer periods of time or involve a larger numbers of participants, these media quickly become unmanageable.⁵ Users are flooded with collaboration artifacts, and making sense of so many of them becomes more and more difficult.

In the middle of Figure 1, structured shared workspaces provide better support for making sense of large bodies of messages and files.^{16,17} They typically support sophisticated access control and

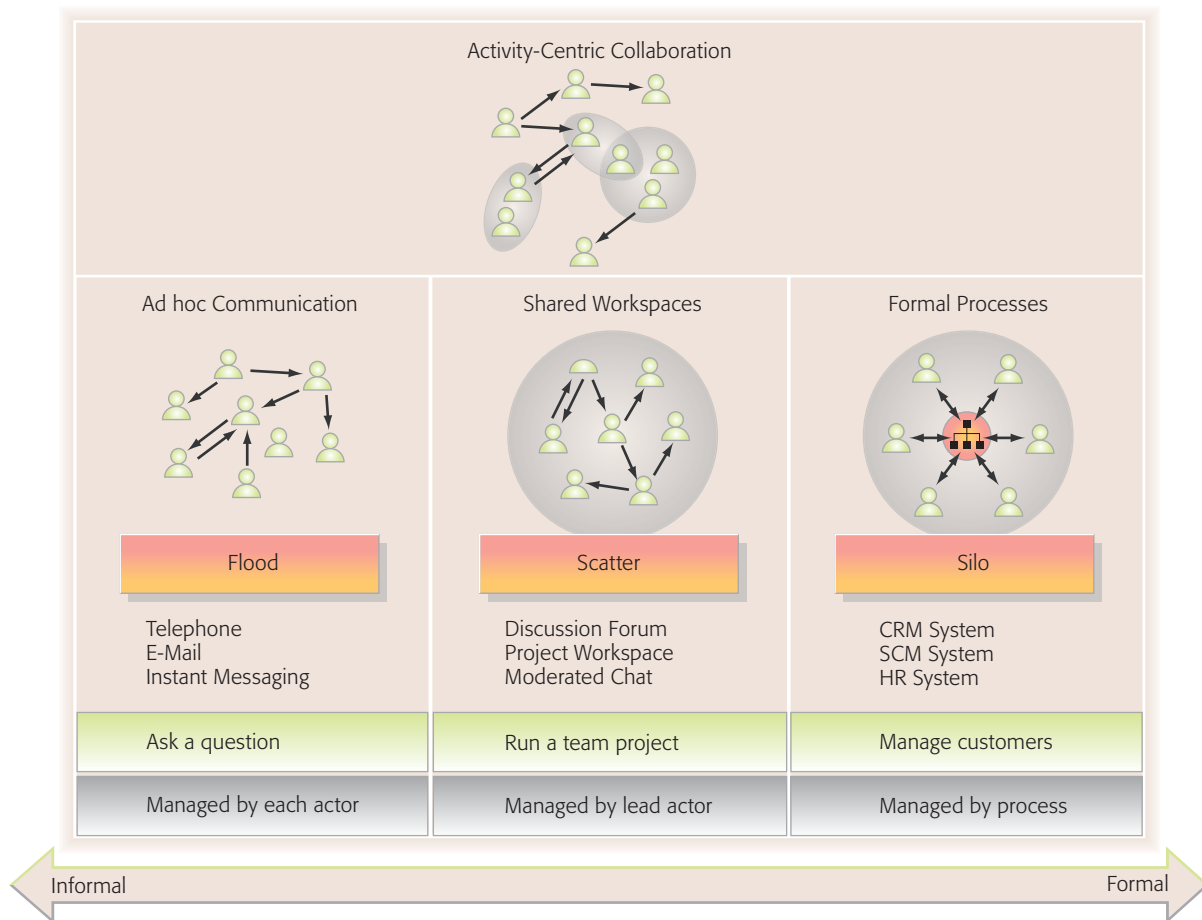


Figure 1

Collaboration technologies and activity-centric collaboration. Each domain is illustrated with a suggestive rather than exhaustive list of the technologies or types of systems that it entails.

CRM=Customer Relationship Management; SCM=Supply Chain Management; HR=Human Relations

role models, both of which help to manage large team projects. Many of these systems provide rich support for structured collaboration, such as document or task management. However, these environments are relatively labor intensive to initiate, which discourages people from using them for small-scale or short-term collaborations. They are more difficult to manage because access control is more complex. These systems also add to the problem of manually managing and monitoring an ever-increasing number of scattered online places (e.g., see Reference 18).

At the right of Figure 1, business process technologies, such as supply chain management (SCM) and customer relationship management (CRM) systems, are highly optimized to support repetitive business

tasks. The collaborative business process is fully represented, and a business process engine helps to execute the process. These systems achieve high efficiency; that is, they complete a high number of collaborative processes in a short time with little human interaction. However, because they are rigid, exceptions typically are not handled well.¹² Users have to retreat to ad hoc tools to resolve exceptions. This, however, can be quite difficult because business process applications are silos, meaning they are not well integrated with other collaboration technologies. Worse, users sometimes have to invent false entities (e.g., fictitious work items or fictitious workers) in order to achieve efficient execution of the human aspects of the work that are overspecified by a formal workflow model.¹⁹ These systems are not well-suited to handle processes that

differ slightly from execution to execution because the system needs to be reconfigured whenever a business process changes, which imposes high cost.

In reality, a single collaborative activity is often managed with multiple collaboration tools and technologies at different levels of formality. These can include e-mail, chat, wikis (a type of web site that allows users to easily add, remove, or otherwise edit all content), discussion databases, listservs, document management systems, workflow systems, and ERP (Enterprise Resource Planning) systems. This diversity means that people must monitor and participate in multiple shared venues, spreading their attention and their effort across multiple media. Even if they succeed at this context management task, they still face the difficulty of having to determine the scale of any new collaborative activity in order to select the best medium. A wrong determination results in having to move resources from one media environment to another, which can be difficult because of the many inconsistencies and incompatibilities among the various technologies. For example, a brief conversation that begins as a chat may need to be transferred to e-mail in order to more efficiently include a larger number of participants or make the communication more convenient for participants in different time zones. If the number of people or shared resources continues to increase, then e-mail may become a chaotic venue, and it will become necessary to transfer the resources again into a structured discussion space or a document management system.

The technical goal of activity-centric collaboration is to bridge these gaps of rigidity and tool boundaries by horizontally integrating different collaboration tools and technologies through the concept of a work activity. The intent is not to provide yet another collaboration tool. It is to provide a technology that can organize collaboration so that it reflects the work being done, rather than the tools that support the work. In addition to organizational and work efficiency, activity-centric collaboration can improve organization and work quality by reducing the extra work needed to assemble relevant resources and by providing a single structure within which all records of an activity may be collectively located and, if necessary, collectively accessed by the original members of the activity or by others who need the information generated by that activity. The intent is for an activity represen-

tation to contain all the resources, tools, and people required to get the work done. The representation for an activity needs to be simple yet flexible enough to accommodate different levels of rigidity, which often are not known in advance.

We believe that activity-centric collaboration can deliver several benefits to business users:

- By integrating diverse tools, activity-centric collaboration can increase the integrity of information around critical business activities, leading to enhanced coordination among human and machine actors, reduced costs associated with information retrieval and analysis across tool boundaries, and improved ability to audit activity histories.
- By inserting collaborative tools into the context of process applications, activity-centric collaboration can enable richer modes of collaboration in these applications, thereby extending their management aegis beyond strongly formal situations.
- By inserting simple activity management into ad hoc collaboration environments, activity-centric collaboration can help users manage their everyday, human-driven communication stream with little or no increased effort.
- By making it easy to bring a variety of tools and best practices to a problem, activity-centric collaboration can empower users to collaborate more freely and better match their collaboration methods to the job.
- By enabling users to capture and disseminate best practices as activity patterns, activity-centric collaboration can enable a new collaborative approach to process improvement.

ACTIVITY EXPLORER

AE currently runs as a separate application within the IBM Workplace Client. This section describes AE in the IBM Workplace Client 2.6 (AE 2.6). Note that the nonproduct research prototype of AE has only a slightly different design and feature set. These differences will be noted in the text as pertinent.

User experience

In AE, an activity is modeled as a set of related, shared objects representing a task or project. The set of related objects is structured as a hierarchical thread, called an *activity thread*, representing the context of the task at hand. Users create new activity threads by creating root objects from any type of

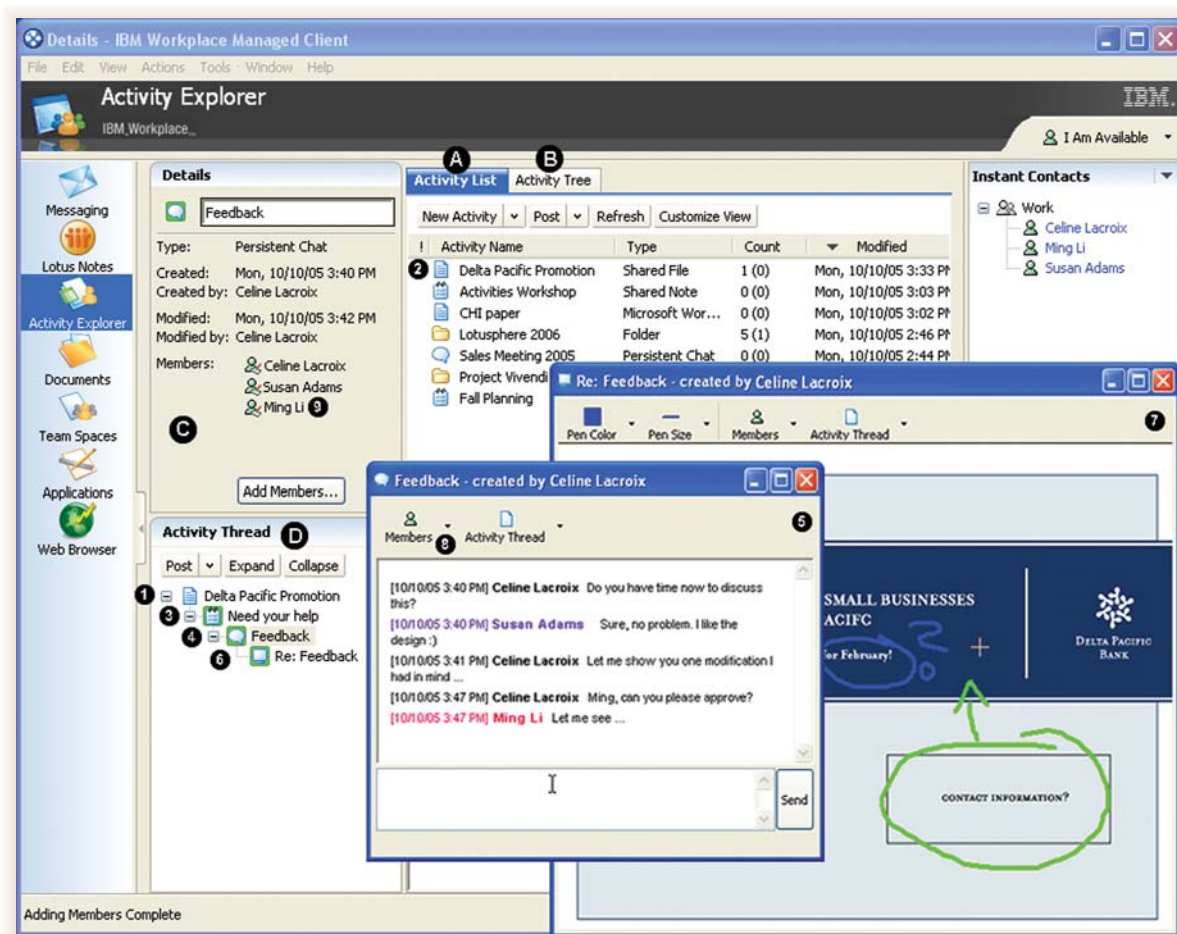


Figure 2
Activity Explorer in IBM Workplace 2.6

content or communication. Users add items to an activity thread by posting either a response or a resource addition to its parent object. Activity threads combine different types of objects, memberships, and alerts. AE initially supports the sharing of five object types: message, chat transcript, file, folder, and annotated screen capture. The AE research prototype also supports task objects.

Activity structure and membership are managed by several user interface components (*Figure 2*). The Activity List tab (A) shows a multicolumn inbox-like activity list. It supports multiple views on activities and can be sorted and filtered. New activities always bubble up on top of this list per default. The Activity Tree view (B) shows an overall tree structure of all user activities and can be used in a fashion much like Microsoft Windows** Explorer. Selecting a

shared object in the list view or tree view populates a read-only details pane (C). The Activity Thread view (D) maps a shared object as a node in a tree representing an entire activity thread. The Activity Thread and the Activity List and Tree views are synchronized by object selection. Additionally, users interact with objects or members displayed in these views through right-click context menus. Representative icons are highlighted green to cue users of shared object access and member presence.

The following scenario illustrates how shared objects, as building blocks for activities, can be used to collaborate in an activity that starts from a document. This scenario highlights only core features; for a more complete description of AE capabilities, see Reference 20. Figure 2 is a snapshot of an activity in progress, shown from the perspec-

tive of one of the actors (Celine). The activity thread is built dynamically as the actors collaborate.

Scenario

Celine is a designer. She works with Susan on a print promotion flyer for Delta Pacific Bank. Ming is their project manager. The first review meeting with Delta Pacific is approaching. Celine has crafted a draft of the flyer and would like Susan's feedback. Celine glances at her Instant Contacts for Susan's

■ Activity Explorer is the first product from IBM that supports the notion of activity-centric collaboration ■

name and sees that Susan is currently offline. From her desktop, Celine drags the draft image file onto Susan's name, starting a new activity thread named "Delta Pacific Promotion" (1). The file is now shared and shows up as a new activity in Celine's activity list (2). Celine right clicks on the file object to add a message asking Susan for her comments (3).

A few hours later, Susan returns to her desktop. In the system tray, Susan is alerted to the new activity by an alert message (whenever an object is changed—including the addition of a child object—all people who have access privileges on that object receive an alert message about the change). Clicking on the alert, she is taken to the activity thread. She opens the message and, while she is reading it, Celine can see that Susan is looking at the message because the shared object is lit green (3). Celine seizes the opportunity to expedite their progress; she right clicks on the initial message and adds a chat to this activity (4). A chat window pops up on Susan's desktop and they chat (5). Celine refers to a detail in the image file; for clarity she wants to show Susan what she would like changed. By right clicking on the chat object, Celine creates a shared snapshot object (6). A transparent window allows Celine to take a snapshot of any region of her screen. She freezes the transparent window over the draft image. The snapshot pops up on Susan's desktop. Celine and Susan discuss a few changes by annotating the image in real time as if it were a shared whiteboard (7).

Aware of the upcoming deadline, Celine wants Ming informed about the status. Within the chat, she

selects Invite to add Ming as a member (8). On his client, Ming receives a pop-up invitation to join the chat, and he accepts. Note that Ming is now a member of the chat and the shared snapshot only. He is not a member of the other objects in the activity (9). Ming approves the changes, and Celine begins to work on them.

This scenario demonstrates how AE helps people move seamlessly and effortlessly back and forth from private to public information and from asynchronous to synchronous real-time collaboration without having to manually create a shared workspace or set up a meeting. Collaboration starts with a single shared object and evolves into a multi-object activity structured in real time by the participants as they create and add new shared objects. An activity thread provides a persistent activity context that aggregates a mix of different object types. Alerts provide up-to-the-minute awareness of person-relevant changes, even if AE is not the topmost application on the user's computer screen.

Design principles

The initial AE design was motivated mainly by the desire to combine the lightweight and ad hoc characteristics of e-mail and the rich support for sharing and structure in shared workspace systems.

Object-centric sharing

A key design decision of our system was to allow sharing of resources in a fine-grained way. In our approach, the basic building block for collaborative activities is a *shared object*. Shared objects hold or point to one piece of persistent information, and they define a list of people who have access to the object and the underlying content.

Sharing on an object level was motivated by the difficulty of predicting the scale of a new activity when people start collaborating. Collaboration might be very short-term or instantaneous and involve only small amounts of data to be shared, few people, and few steps of interaction, for example, exchanging one or more files, setting up a meeting agenda with people, or jointly annotating a document. These activities might or might not become part of a larger collaborative work process. However, people usually do not create highly structured, shared workspaces to perform these tasks. Shared workspaces are place-based; that is,

users first have to create a place, assign access rights, and then add content to the place that will be shared. Our approach to shared objects is rather document-centric and very similar to Dourish's notion of place-less documents.²¹ In this approach, the document itself becomes the place, and sharing is a property or feature of the content.

Notifications and awareness

Our design deliberately blends asynchronous and synchronous types of collaboration. If other users are present when an object is accessed, they can work synchronously; if not, work is asynchronous. When new objects are created, users are automatically subscribed to change modifications of the object they are a member of; that is, when objects are modified, notifications are sent in real time to all members of the object informing them of the change. The primary technical purpose of notifications is to update the local state of a shared object. These state changes are used to update a user's view of this shared object and to notify users about current activity on this object through alert notices.

A user's presence on an object (i.e., being active on a document) is also part of the state of an object. Awareness about who is currently present on an object can serve as a trigger for opportunistic collaboration and can help keep collaborative activities moving forward.

Activity and conversational structure

In order to be able to add structure to users' collaborative activities, we allow them to combine and aggregate heterogeneous shared objects into structured collections as their collaboration proceeds. This collection of related but shared objects is what we have been calling an *activity thread*. It represents the context of a collaborative activity. We see this structure being defined by the ongoing conversation among users in an activity; that is, each object added to an existing object can be considered a response to the previous one. This approach is based on the assumption that the conversation around artifacts is a major driver for the execution of an activity.²² The threaded design, however, blends well with a more resource-oriented way of structuring an activity, for example, for the purpose of planning. In that regard, an activity thread can also be preinstantiated with objects to represent structure that gives people more guidance in executing the activity.

We intentionally decided not to require any particular object as a structural container for a collaborative activity. Each individual shared object can function as the parent object for an activity thread or a subthread within a thread. This design is intended to facilitate the creation of activities and derives from the fact that any piece of content can represent an activity, and it supports the evolutionary character of ad hoc activities as described in Reference 7. By extending AE, new object types can be added that semantically represent the activity. The folder object we included in our initial design comes close to being such a structural container. However, we do not force people to create a new activity with a particular object type.

Dynamic membership

Membership in a collaborative activity can be dynamic and heterogeneous. Activities often generate side activities that involve a different set of people, or they might require bringing in new people or excluding people from certain shared resources.⁷ Dynamic membership within an activity thread comes as a by-product of our object-centric design because each object has its own access control list. As collaboration proceeds, we allow users to include new members in—or to exclude old members from—selected shared resources in the thread. Membership can also be used to mix and match private and shared objects within an activity.

In many regards, our approach is similar to threads in e-mail or discussion databases, or *thrasks* (combination of e-mail communication with task-like coordination).²³ However, it is richer for a number of reasons:

- Objects are shared (unlike in e-mail or thrasks).
- Activity threads may contain different types of objects and are not restricted to messages only, as are e-mail and discussion databases.
- All objects are equal, unlike e-mail or discussion databases, in which attachments are subordinates contained in a message.
- Membership is dynamic and may differ within an activity thread from object to object (unlike in discussion databases), and membership can be redefined after an object has been created (unlike in e-mail).
- Objects support synchronous collaboration through built-in real-time notifications and pro-

vide rich awareness information about who is currently working on what.

Note that the representation of an activity in an activity thread is central to our design. The activity thread represents the context of an activity by

■ An activity can be defined as a logical unit of work that incorporates all the tools, people, and resources needed to get a job done ■

horizontally integrating heterogeneous content, the tools to modify the content, and the people who are part of the activity.

ACTIVITIES INFRASTRUCTURE RESEARCH

To realize the AE user experience, we designed and built a novel activities back-end service called the *instant collaboration* (IC) server, which was initially implemented in our research group. In our research, we investigated different architectural models, comparing the peer-to-peer model with centralized models.^{24–26} We studied different approaches to provide scalable activity services through simulation,²⁷ and we explored various data representations. It is beyond the scope of this paper to describe that work in detail. In the following sections we describe a few key infrastructural choices behind the IC server design. We also list the lessons learned from our initial implementation in the research laboratory. These lessons helped us select IC server concepts for the activities server development.

Activities data model

The IC server offers activity-related services based on the concept of generic shared objects (GSO).^{27,28} GSOs are persistent collaboration objects that can be used as building blocks to create new collaborative applications. The IC server manages a collection of GSOs and their relationships. Hierarchical GSO structures are aggregated through reference. The ability to aggregate GSOs allows us to model more complex collaboration structures, such as the activity threads in AE.

The activities service of the IC server is accessed through a client-side application programming

interface (API) that internally uses three basic primitives: *Request*, *Response*, and *Notification*: A client asks for some service by issuing a Request to the server. The server replies with a Response to inform the requesting client about the result of its request. Depending on the type of request, the server also sends out Notifications to other connected clients.

Each GSO holds one or more pieces of persistent information through a simple model that defines a set of fixed properties and a set of variable properties for content. Fixed properties include: unique ID, name, type, author, creation time, modifier, modification time, reader, last access time, member list, and member status pertinent to the GSO. The variable properties of a GSO describe its actual content. A GSO does not provide any means for semantically describing the content. Content is associated with a GSO by adding an arbitrary list of <name, value> pairs. The interpretation and use of the <name, value> pairs is left to client applications. The value field can be of various types, for example, string, integer, double, or Boolean. A GSO can also support binary content.

A GSO member-list property controls access to the GSO and represents a distribution list for broadcasting notifications about initial creation and subsequent modifications of the GSO. Also, the member list can be changed at runtime. In general, any change to the set of fixed properties (e.g., membership) or to the set of variable properties (content) of a GSO is not only stored in the underlying data store, but is also automatically broadcast to all members of that GSO by means of notifications. The default behavior is that every modification to a GSO is broadcast to all its members. In other words, whenever someone is added to the GSO member list, he or she is implicitly subscribed to all change events of that object.

In our AE research prototype, each shared object is represented by a GSO. Activity threads are represented by a hierarchical structure of GSOs. The different shared object types use GSO properties to manage their content. For example, consider the persistent chat object. Each chat message is stored as a property. To send a chat message, the AE client application adds a new variable property to the GSO representing another line in the chat and submits a request to the collaboration service. The IC server

receives the request, updates this change in its database of GSOs, and notifies all members of this GSO of the new variable property. The members of the chat receive this property change, and the chat message is added at the end of their transcript.

Data representation

We used various data representations of the GSO model described in the previous subsection. The first version of the IC server used a custom-built Java** Object Store layer on top of a relational database. Every Java object was registered in an object table, and the fields of the object were stored as `<name, value>` pairs in a separate properties table. This approach provided significant flexibility in developing and evolving the IC server. It also scaled sufficiently well for the various research deployments described in this paper.

However, we noticed that the server was slowing down proportionally with the number of members of a GSO. Membership was managed with object properties, which is not an efficient access control implementation. Another disadvantage of the `<name, value>` pair storage model is that Structured Query Language (SQL) queries (e.g., for search) cannot be used efficiently because the semantics of the data model are hidden in properties instead of being represented directly in the database tables.

The activities service for AE 2.6—which is the product version as distinct from the prototype research version—used an entirely different data representation based on the Collaborative Application Infrastructure (CAI) from the Workplace server.²⁹ CAI is an abstraction layer that features a collaboration model where Collaborative Domain Objects (CDOs)—similar to GSOs—can be aggregated through a collaborative context. A community service supports membership management through role-based access control.

Each shared object in AE 2.6 is mapped to a CDO and has its own associated community. The hierarchies of activity threads are represented through the containment feature of the collaborative context.

All of the aforementioned data representations add a layer of abstraction on top of the relational data store. While they offer more flexibility for schema changes, a direct representation of the activities data

model in a relational database will outperform those approaches and scale much better. To overcome performance bottlenecks, we implemented a proof-of-concept version of the IC server using a direct mapping of the GSO data model onto a relational data store. The resulting schema basically consists of a number of distinct tables for storing shared objects, membership, hierarchical relationships, and variable properties. This schema was the basis for the activities data model used in the next-generation work described in the section “Designing the future: Next-generation AE.” It includes several enhancements and optimizations. The schema also supports features, such as patterns, from the UAM research project,^{8–10} it provides tagging capabilities to better organize activities, it stores variable properties in a richer RDF-like (Resource Description Framework-like) format, and access control is applied on a per-activity basis to yield higher scalability.

Infrastructure beyond Activity Explorer

The IC server and its general purpose GSO model have been successfully used as a back end for real-time collaboration and communication applications other than AE. Jazz³⁰ enhances the Eclipse** integrated development environment with team collaboration for software developers and uses the GSOs on the IC server to provide persistent chat and online awareness of people and the files they are manipulating in the IDE. C+BSeen is a proof-of-concept instant-messaging client that uses the IC server to share rich awareness of users’ activities on their desktop environments.³¹ The Activity Spaces project³² promotes a new approach to shared workspaces by making them activity centered. It uses the IC server to provide enhanced awareness and selective subscription of team activities around software development tasks and their related artifacts.

AE 2.6 uses an API similar to the GSO API in the IC server. However, for simplicity AE 2.6 supports content only as a large binary object. There is a single variable property that allows storing a pointer to external content. Persistent real-time collaboration has been separated from the model. Only basic create, retrieve, update, and delete operations are being broadcast with notifications but not, for example, chat messages. In particular, the persistent chat object in the product uses a chat service external to the activities model, and the chat shared object simply stores a pointer to the external

content. Our simulation analyses²⁷ indicate that separating out real-time collaboration from the model implementation ultimately yields better performance and scalability. Note that this separation can be made completely transparent to the user of the GSO API.

Our GSO model does not make any assumptions about the semantics of the information stored. Storing pointers to external content is crucial for integrating activity objects from other sources and for integrating external real-time applications. We cannot assume that every new object type in activities will be implemented using the activities data store. The goal of our research prototype was to validate the basic activities concept using five predefined object types. However, to be successful in the market, the product needs to be open enough to integrate all sorts of object types in order to best represent real-world activities. While AE provided some basic extensibility functionality for new object types, the next generation, described in the section “Designing the future: Next-generation AE,” is taking this much further.

EMPIRICAL AND DESIGN RESEARCH

To validate the AE design, we collected data through field work, participatory analysis, usability evaluation, customer reviews, and design explorations. AE was deployed multiple times in research over the past three years. A large portion of our data stems from empirical studies of these deployments. The empirical and design research described in the following sections helped us verify and refine the activity concepts and gain useful insights for future product versions.

Empirical studies

The first major study of AE took place during the summer of 2003 when we provided it to a research community of 33 people (researchers, staff members, and interns and their mentors) for use in conducting summer research projects and administrative work. This trial deployment succeeded well beyond our expectations, and our initial success became the basis for the adoption of AE as a feature in Workplace.

Our initial research report was based on the first 100 days of use, during which the community created more than 1,400 objects organized into more than 200 activity threads.³³ This report combined analy-

ses of log data with ethnographic interviews. We continued our field trial for another two months. During this time, we substantively improved our data logging and analyses, based on a subset of questions that we could not answer with full confidence from the data set of the first 100 days. This second set of analyses was based on enhanced data logs (>2,300 objects).³⁴ In these analyses, we focused on whether the design principles of the preceding section had been supported in the design and in use.

During the summer of 2005, we conducted two additional studies. The first of these studies reexamined a problem that we had encountered during the 2003 field trial, namely, alert management. In the example scenario given previously, Susan’s attention was focused on the existence of a new activity by an alert message that appeared in her system tray. This feature seemed highly desirable to us in theory; however, in practice, our 2003 field trial members experienced “too much of a good thing,” as they were being showered with more alerts than they wanted. In our first 2005 study, we deployed a modified version of AE to a group of 34 users (researchers, interns, product designers, and developers) over a period of two months. We explored alert management by allowing users to rate each alert in terms of its desirability and modeled users’ preferences about receiving alerts.

During 2005, we also conducted a participatory analysis of activity management in which we invited people to use paper-and-pencil materials to show us how they would use the existing resources (object types). The use of paper-and-pencil materials allowed people to invent (on paper) additional object types that they would want to be able to use to describe their work or to collaborate with others on that work. For example, participants filled in slips of paper with the name and attributes of each activity component and then arranged those slips of paper into a hierarchical list, similar to the activity threads of AE. If a participant discovered the need for a new type of object (e.g., a *meeting* object), he or she could simply write “meeting” and (optionally) draw a meeting icon. For the purposes of the paper-and-pencil analysis, a meeting object had just been sufficiently invented to show how it would be used if it existed. Our team and the product team were thus informed of potential user needs and could conduct more systematic investigations to

determine whether the features should actually be implemented. We discovered additional object types and found that participants highlighted a small subset of objects as being of interest to other people—a pattern of behavior that we called *selective sharing*.³⁵

In the results subsections that follow, we draw on several of these studies to pursue emergent themes from our multiyear research program.

Diversity among activity threads

We had designed AE to be used by relatively small groups of collaborators for relatively brief periods using a handful of objects in each activity thread. We had hypothesized that small, ad hoc collaborations would continue to occur in chat and e-mail, and that large, formal collaborations would continue to occur in discussion databases. Indeed, in the data from the first 100 days, we found 110 activity threads (54 percent) that corresponded to this pattern (2–14 objects, 1–7 days duration, a small number of collaborators).

We were surprised by other activity threads. **Figure 3** shows the distribution of activity threads in terms of the number of objects in each thread, the number of members of each thread, and the duration of each thread. We eventually separated the data into three groups: small threads (one object), medium-sized threads (our expected range of 2–14 objects), and large threads (19 or more objects).

The unanticipated uses made some AE activity threads into simple chat vehicles (e.g., Can you introduce me to ...). Out of 203 activity threads, a total of 71 (35 percent) contained a single chat object (with an average of 18.92 turns per chat, a median of 7 turns, and a range of 1–222 turns). Thus, despite the fact that these 71 threads contained a single object, the single chat object contained evidence of extended collaboration (i.e., two or more chat turns).

Other single-item activity threads were composed of a message object (24 threads, 12 percent), a file object (12 threads, 6 percent), a folder (3 threads, 1 percent), a task (1 thread, <1 percent), and a shared screen (1 thread, <1 percent). There is not space in this paper to analyze these nonchat objects in greater detail; briefly, we hypothesize that these were either failed collaborations in which the



Figure 3
Frequency distributions of activity threads in 100-day sample, by thread length, number of members, and duration

intended collaboration partner never responded, collaborations that started or continued outside AE, or one-way, one-time information exchanges.

The kind of work supported in the medium-sized activity threads was varied. For example, one activity thread that was composed mostly of file objects contained three drafts of presentation slides with accompanying messages. Another example was a four-person discussion, spanning a period of eight days, about a technical topic.

An example of a large-sized activity thread was the multiperson planning of a summer intern project. This thread included chats about the work, specification documents related to individual IBM products, presentation slides about related topics, and messages that contained how-to pointers for the intern.

The unanticipated uses also converted some AE activity threads into four-month community resources and sites for detailed, extensive development of project contents, such as writing research papers for conferences. Among the longest threads were two that were directly related to the AE project, the *Alpha testing* and *Informal usability inspection* threads. The *Pilot feedback*, *Photobook*, and *Intern tips and tricks* threads were examples of interns' reinvention of AE for their own community purposes. The *AJW*, *Eddie*, and *Planning* threads were intern projects that generated large, partially archival sets of materials. The *Group 2003*, *Momail*, and *User study* threads were researcher activities toward conference papers. The *Jazz* thread was a research exploration of collaborative software development environments.

Our prototype was used much more broadly than we expected. How did this happen? The student interns in our group in 2003 took over AE and made it their home environment.³⁶ Interns' self-reports during the ethnographic interviews were consistent with the server log data. One intern remarked, "I never have used less e-mail." Other interns told us about their ongoing use of AE as a reliable, always-on medium:

"I kept it on all the time to coordinate with [my mentor].

"It's sometimes interesting to see what people are reading about.

"I liked the address book—the people list and their photos." (The address book was a spontaneous intern project, actively promoted by one of the interns. It contained digital photographs and biographies of the interns, a kind of who's who database used by both interns and others.)

Several interns said that they had never loaded the company-standard instant messaging (IM) client, because they felt that AE gave them all of the workplace chat capability that they needed. Led by the interns' innovations, multiple groups of researchers also began to use AE in new ways. The result was that the users arguably reinvented AE through use (e.g., see References 37–39). (Some of the surprising results were clearly due to interns' activities. However, most of the surprising results also involved one or more research staff members, and fully half of the longest, most surprising threads were primarily full-time staff collaborations.)

Thus, our first finding is that people used AE in ways that we did not anticipate. We had thought of AE as a kind of niche solution, providing support for collaborations that fell between chats and e-mails at one extreme and discussion databases at the other extreme (i.e., our medium-sized threads of Figure 3). However, users showed us that AE could also be used easily for highly informal, very brief collaborations (the small-sized threads) and well-structured, large, and long-lasting collaborations (the large-sized threads).

Heterogeneous threads and diverse objects

We had designed AE to support people who needed to collect diverse resources in the right combinations and in the right structures for their use. We performed a simple test of this hypothesis by noting how many threads contained more than one type of object. Small threads were not counted in this analysis because each, by definition, contained only a single object. We, therefore, analyzed the 91 medium and large threads to determine the number of types of objects in each thread. The AE prototype supported six object types, so the range of our dependent measure was 1 to 6.

Figure 4 shows that, for threads of length > 1, 81 percent of the threads contained at least two types of objects, and 46 percent contained at least three types of objects. Thus, when offered the opportunity to combine diverse types of objects in a single activity,

people overwhelmingly took advantage of that capability. The popularity of certain object types depends on the type of activity. Overall, the most popular objects in our 2003 data were messages, chats, files, and tasks. Our 2005 participatory analysis data with office workers confirmed the popularity of messages and files, but not chats. That study also highlights the importance of tasks and meetings. The next subsection sheds more light on object usage within certain activity communication patterns.

The participatory analysis work in 2005 supports the use of diverse object types. In our adaptation of the CARD method for participatory analysis,⁴⁰ people used paper-and-pencil materials to indicate what they thought was the ideal way of structuring their work. People were free to invent new types of objects and did so, for a total of eight new object types. If we restrict our analysis to the six object types in the AE prototype, then of the 24 threads collected in this study, 92 percent contained more than two types of objects, and 75 percent contained more than three types of objects. If we consider all 14 object types that people included (the six AE prototype object types plus the eight invented object types), then of the 24 threads, 100 percent contained two or more types of objects, and 75 percent contained five or more types of objects. Thus, the pencil-and-paper results are strongly corroborative of the logged activity threads from the AE prototype.

Users' comments from the ethnographic interviews support these quantitative results. One user said, "We would store chats with the objects that they were about." Another user reported, "In the [development folder], messages and chats on a related topic or subtopic appeared together with screen-shares . . . useful especially in [user interface] development, showing updates Screen-shares usually followed a message, 'I've updated this and that.'"

Until AE, most systems provided separate databases for different types of objects, for example, e-mail in a mail system, chats in an instant-messaging system and documents in a content management system. Our findings suggest that to support real work activities, we need a system that supports a variety of different object types as first-class objects.

Patterns of object usage

(This section is based in part on an earlier work, "Patterns of Media Use in an Activity-Centric

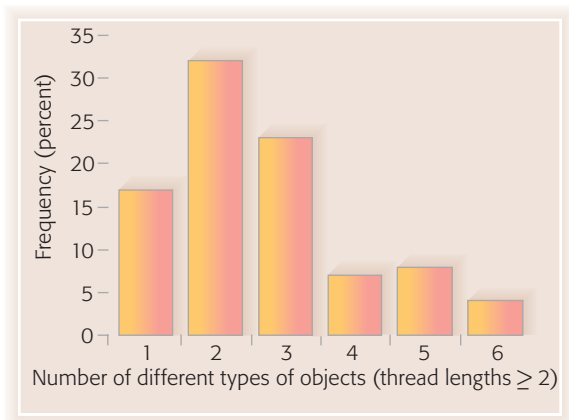


Figure 4
Number of different types of objects per thread, for threads of length two or greater

Collaborative Environment," *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems*, Portland, OR (2005) © ACM, 2005. Reprinted with permission.)

In the preceding section, we concluded that an activity-centric collaboration system needs to integrate diverse object types in a single, integrated representation. We then asked, "What patterns (or genres) of usage of combinations of types of objects emerge?"

We performed a K-means cluster analysis of the object types used in each of the activity threads of the 2003 extended data set (five months of usage data).³⁴ We normalized the thread data by computing the percent of each activity type for each activity thread and then clustered the threads by object types. The resulting cluster solution, shown in **Table 1**, shows the average percent of item type for each of four easily interpretable clusters.

The largest cluster (49 percent of activity threads) consisted mostly of message objects. These kinds of activity threads were conceptually similar to most e-mail or discussion group threads, and so we labeled this cluster *communication*. We named a second cluster (20 percent of activity threads) *mixed*, because it contained both synchronous contributions (chat and screen-sharing transcripts) and asynchronous contributions (files and messages). These *mixed* threads may illustrate one of the strengths of AE: the ability for the activity to

Table 1 Clusters of activity threads, with percentage of each item type occurring in each cluster

(Percentage)	Clusters			
	Communication	Mixed	Coordination	Archival
Chat	3	43	5	3
File	5	15	8	61
Folder	3	2	9	4
Message	85	22	29	29
Screen	3	16	3	3
Task	0	1	47	0
Number of clusters	74	30	12	34
Clusters	49	20	8	23
Notes	• Similar to e-mail threads • Relatively brief	Begin informally, then add structure	Manage a project with tasks and messages	Store files and messages

begin with one type of object—typically an informal, synchronous object—and then change and grow into a longer conversation with different types of objects. The third kind of activity pattern (8 percent of activity threads) contained more task objects than the other patterns did. It was supplemented primarily with file objects and was usually used for project management. We called it *coordination*. The final activity pattern, *archival*, was used to collect and share files and messages about them among group members (23 percent of activity threads). Archival threads tended to be long; many of the threads in the large-sized thread category of Figure 3 were archival. Examples of this kind of activity thread were a 55-item preparation of a conference paper submission and other reports and presentations.

These patterns of use suggest that there may be genres of activities that benefit from specialized support. The communication pattern suggests a genre of activities that provide integrated support for diverse communication acts that include e-mail, chat, and perhaps also e-meetings. The coordination pattern suggests an enhancement to project management systems, adding the activity flexibility and informality to the formal and analytical powers of project management environments. The archival pattern suggests a range of record-keeping actions that may be required for business, financial, or legal reasons. Specialized support for certain use patterns could also come in the form of predefined templates for activity threads. If users know in advance what kind of work activity they are planning, templates can provide helpful guidance in executing an

activity. Our research in the UAM project⁹ covers more structured activities, including patterns and templates.

Alerts: Benefit and problem

We turn now from the relatively structural analyses of objects, threads, and patterns to a second major attribute of AE, namely the dynamic alerts that are generated when users create, modify, and access an object.

AE provided several awareness features. Icons were provided for both users and objects. When users were running AE, the icon that represented them would change to green to indicate their presence in AE. When an object was accessed, the icon for that object would change to green. Thus, it was possible to see people enter and leave AE (see Reference 41 for an earlier project that showed people's presence in a compartmentalized shared space), and to see objects being used and then falling out of use.

Whenever any user accessed (viewed) an object, that access was sent as a notification to all members of that object and produced an alert message in the system tray. In some cases, this was a useful feature. One participant reported, "I like being able to know who is looking at something right now. It's cool to see that an object is green."

Another commented, "[my mentor] is my code source . . . I had three persistent chats with [my mentor]. It's faster, simpler in [AE]. I can see his little green penguin." (The icons for people, which indicated their online status by turning green, were called "penguins" by the interns.)

However, we had not anticipated that there would be large threads with many members. People were sometimes overwhelmed by a nearly constant stream of messages from these large shared activities. We had responses like the following:

“I turn alerts on for a couple days max after I post something, and then I turn them off.

“Wanted to be informed only for new postings.

“I would like to select what I would be informed about”

These problems with alerts eventually led to the design of new features that helped people control the nature and volume of the alerts they received. Our 2005 field study used a modified version of AE, with the goals of gaining a better understanding of the alert problem and devising a solution. We instrumented AE alert bubbles with thumbs-up and thumbs-down buttons, allowing users to rate the usefulness of an alert. Over a period of two months we collected 6,248 alert ratings. During those two months we had also removed any preferences for controlling alerts, that is, users were unable to turn them off. Interestingly, out of the 6,248 ratings, 51 percent were rated thumbs-up, indicating that many alerts were considered useful. However, the 49 percent that were not rated thumbs-up validated the interview data from the field trial (i.e., that there were too many alerts) and indicated an opportunity to save users from approximately 50 percent of the interruptions.

A deeper data analysis showed that users almost always rated alerts positive when the following occurred:

- The alert action was adding a new member to a shared object.
- The alert action was creating a new resource (as opposed to modifying or viewing a resource).
- The number of subscribers to a shared object was four or less.

On the other hand, users almost always rated alerts negative in these situations:

- The user had received a similar alert in the last two minutes.

- The user had received four or more alerts in the last minute.
- The number of subscribers to the shared object was greater than seven.

These findings confirm one of our original hypotheses that users do not want to know everything about large activity threads with a large number of members, but would like to be more informed about smaller activity threads with few members. They also suggest that the type of action is an important attribute in weighing an alert. For example, alerts about the creation of shared objects are much more important than alerts about users viewing objects. The latter was one of the most controversial alert types in AE. When users viewed objects, the alerts triggered other users to view those objects as well because they thought that something interesting was going on. This flocking behavior led to alert explosions on users' desktops. Finally, this data also suggests that users can only digest a certain number of alerts in a given time, indicating the need for new ways of aggregating alerts or presenting them at the right time.

Encouraged by this data, we designed and built a new alert management system that uses a combination of collaborative filtering and rule-based filtering to cope with the problem. We designed a new collaborative filtering algorithm that achieves an overall accuracy of 73.61 percent of all alerts being correctly identified as either positive (show) or negative (don't show). This line of research is still ongoing. Together with rule-based filtering, we expect to reduce most of the alert noise by being able to present only alerts that matter to the end user.

Membership

The alert management problem discussed in the preceding subsection arises from the sharing of objects among multiple people or members of the object. We now focus on results that specifically inform our thinking about membership.

We designed AE with the feature of heterogeneous membership within an activity thread; that is, each shared object has its own access control list. This concept allows users to create both private resources in an activity and subthreads—for example, a side chat with one or two people within a larger thread—with different membership. We thought this was an

important feature to better represent real-world activities, and previous research indicated that membership in e-mail work activities was highly dynamic.⁷

The data from the first 100 days was insufficiently detailed to test our prediction that people would use dynamic heterogeneous membership in activity threads. Anecdotes among researchers (but not among interns) suggested that people used this capability. Hence, we analyzed our second and richer data set of five months of use, for which we had finer-grained data in enhanced server logs. This data showed that 28.6 percent of all activity threads were not uniform in membership. We also noticed that about 6 percent of all shared objects managed in AE were private, that is, not shared with other users.

This finding suggests that the ability of scoping membership within an activity is important. Our participatory analysis in 2005 confirms this.³⁵ At four sites, we were able to collect data from multiple people who participated in the same activity ($n = 6, 3, 2,$ and 8 for the four sites). These four sites, in total, included references to 477 objects, of which only 114 (24 percent) were mentioned by at least one other participant at that site (on a site-by-site basis, the percent of objects mentioned by at least two participants varied from 9 percent to 45 percent). Thus in the aggregate, 76 percent of the objects were mentioned by only one person. This pattern of selective sharing of shared activities suggests that much shared work takes the form of private actions in support of shared goals, with only a subset of the objects in a shared activity actually being shared. Interestingly, this result contradicts the number of 6 percent of private objects in AE during the field trial. We have considered three hypotheses to explain these findings:

- Private work in support of activities did happen outside AE, and AE was not well enough designed to support more private actions within an activity thread.
- Users provided access to most objects as a courtesy to one another, but certain objects were, in fact, used by only one user.
- The methodology of the participatory analysis (individual interviews of multiple members on a team) may have tended to underestimate the extent of shared knowledge of objects.

New postings in AE always inherited the membership list of their parent object. This feature is useful in many cases but also fosters the use of AE as a tool for sharing.

Like many features, fine-grained control of membership has both positive and negative aspects. The positive aspects include the ability to create sub-threads with limited membership and enhanced privacy and the ability to remove distractions from the view of people who do not need to see them. The negative aspects include additional complexity—in the ways that people have to think about objects, in the user interface, and in the data structures. As discussed in the next section, some of that complexity became a problem during the usability tests of the AE product.

Usability analysis

In addition to our empirical studies, the product development team conducted two usability studies in 2005 in order to better inform the design and usability of the AE product. The version of AE that was tested was functionally similar to the research prototype described earlier in that it supported hierarchical threads composed of up to five types of objects (task objects were not supported in the product), with membership that could be controlled at the level of each object if desired, and with alerting capabilities from each object to its members. However, this functionality and user interface were embedded in the large Workplace Client that also included general chat and e-mail capability, with the result that the overall user experience was somewhat more complex than that of AE by itself. Both of the usability studies were 90-minute test sessions within which a user was asked to perform approximately ten collaborative tasks that were considered core to AE; for example, sharing a file, posting a response, and initiating a real-time screen-sharing session. The testers could all be generally considered business users whose primary job does not involve the development of software, and none of the users was familiar with AE before the test.

The studies produced several findings. Many of the core values of AE were validated. Users were impressed with the capabilities of AE, and in particular found the ad hoc sharing features valuable. A number of critical usability problems were also identified. The five-panel user interface of AE appeared overly complex and somewhat con-

fusing to these first-time users, and some of the application terminology was initially unfamiliar. By the end of the 90-minute session, however, most users thought that the tool was valuable. Users also expressed a desire for better integration of AE with their usual business tools. The results of these studies helped drive improvements in AE 2.6, but it was not possible to address many of the broader usability findings, such as integration with other applications, without more extensive restructuring.

Customer feedback

AE has been presented to customers and business partners as both a research concept and a part of the IBM Workplace family of products. Several pilot deployments are currently underway with customers.

Many people reported that the notion of activity-centric work is a substantial part of their everyday work practice. They liked the capabilities in AE that support this notion, such as aggregating related items and seamlessly moving back and forth between different modes of collaboration. While the design philosophies of object-centric sharing and conversational activity structure seem to resonate very well, people had some concerns about our initial user interface design, the management aspect of activities, customization of activities, and integration with other productivity tools.

Our user interface, for example, shows activities as tree-like structures. A disadvantage of the tree-like approach is that this structure is forced on all users, including newcomers who might not understand how to navigate this structure. This becomes a problem as an activity thread becomes very large. We also observed these difficulties during our two internal deployments. We envision that there are better ways of presenting and navigating that information. More conversational interfaces that are structured by time could help with understanding the history of an activity. People-centered views could help focus on who you want to collaborate with and could display shared objects that relate to a certain user or a group of users. We are actively exploring different ways of summarizing or rolling up an activity so that its overall contents and significance can be understood at a glance.

Another major issue was the management of activities. People reported that they were engaged in

numerous activities every day. They were concerned that the user interface could easily become cluttered and that it would be hard to find information in the AE tree and list views. Adding search and better

■ Activity-centric collaboration promises to bring work and team productivity to the next level ■

ways of organizing activities were required to address that problem. In particular, people were asking for an archiving mechanism that would help them remove completed activities from the tree, but still have access to the information for later perusal.

The most common AE feedback that we received from customers was that they wanted to be able to tailor activities to specific enterprise problems. AE needed to interact seamlessly with their existing business applications and processes and not represent an additional destination that users were forced to understand, use, and manage. Customers expressed the strong need for the ability to exploit activity-centric collaboration in the context of their existing application deployments. For example, customers who already deployed web-browser-based solutions or Lotus Notes*-based solutions expected the new capabilities to be easy to apply to work conducted in those environments. More broadly, there was a need to link from activities to existing tools and applications, and there was a complementary need to be able to place an activity within an existing tool or document context.

Customers also expressed strong interest in wanting to reuse activity patterns as described by our UAM research.⁸⁻¹¹ Activity patterns allow users and businesses to create reusable activity templates that can be shared so that individual contributors can capture their business processes in reusable form and organizations can preserve and communicate their best-practice patterns. This is a compelling feature for customers whose employees perform work that is too varied to be reduced to formalized workflow instances in a cost-effective way.

Design explorations

In parallel to our empirical work, our team also conducted a series of design-based exercises to

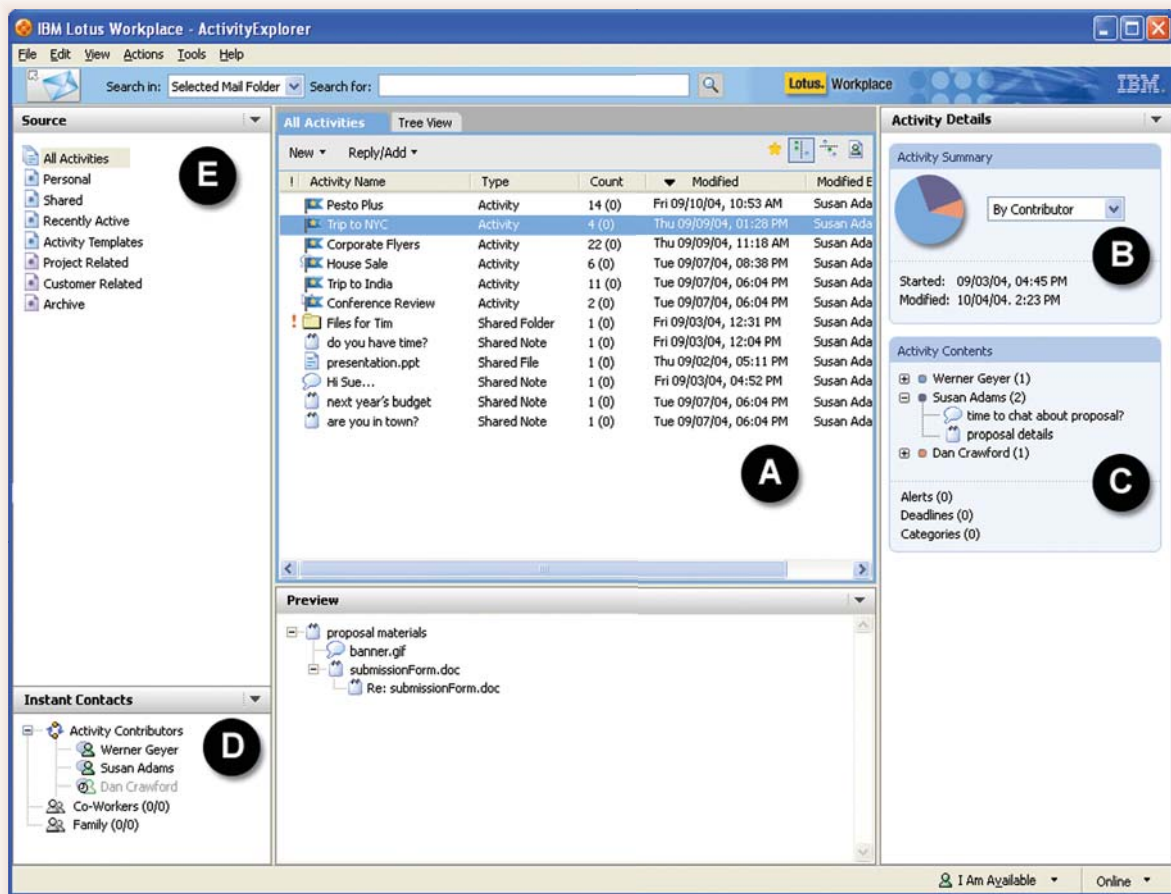


Figure 5
A conceptual design showing a view where users can elevate threads to activities

further explore activity-driven work concepts. The future design concepts were focused in two main areas:

- Breaking away from the strict post-and-reply hierarchy for creating threads by providing alternate visualizations for activities that could help users better understand and navigate the activity structure.
- Integrating the concept of activities with many diverse collaborative applications and tools and moving away from AE as a stand-alone activity management application.

For example, we explored a version of activity management that takes place more immediately through instant messaging,⁴² and we investigated whether an activity may be attached to (as part of) structures in other applications, such as business

controls. In these ways, we expanded the scope of current Lotus-brand IBM products and helped to find ways of moving the IBM activity-centric collaboration strategy forward into new domains.

A complete description of the exploratory design work is beyond the scope of this paper. **Figure 5** is an example that shows a design iteration in which a user is able to directly indicate when a thread of shared objects has grown into an activity. The threshold for determining this is left to the users' discretion. It was hypothesized that users could then allow unimportant or trivial threads to emerge and fade away, while elevating more vital threads to true activity status with an activity object as the root object.

Figure 5 shows those activity objects as blue flag objects in one list, together with regular activity

threads (A). The details pane shows a graphical summary (B) and a categorized content summary (C) of a selected activity thread, including a list of all the contributors of the thread. Instant Contacts (D) show a contextualized view of all the members of the activity. The Source pane on the left (E) helps users organize activities through categorization.

The primary use of our design explorations was to motivate and set the vision for future versions of AE. They aimed to illustrate how an activity could be incorporated as a supporting platform for collaboration. For example, the exploration illustrated in Figure 5 contains early concepts of features such as activity object, summary view, and aggregated member lists; these features have become important aspects of the design of the next generation of AE, as described in the section “Designing the future: Next-generation AE.”

Outcomes

The empirical work has been crucial in several ways. First, the 2003 experiences showed that the first version of AE could be used for real work and that it could provide a valuable resource for doing that real work. While the quantitative and qualitative data collected during our study confirmed that we were on the right track with many of our initial design principles, we were pleased to see how readily our users made this new environment their habitat. This was unexpected, in particular in the presence of other tools, such as e-mail or IM (instant messaging). During the summer of 2003, several research projects were coordinated entirely within AE.

The experiences in the summer of 2003 were also important for showing us the power of the concept. As noted previously, we had envisioned AE as a niche solution between two well-established product families—informal tools, such as e-mail and chat, and formal tools, such as discussion databases. The spontaneous use of AE within each of those domains showed us that AE had broad applicability and that it could, in many cases, replace more specialized, single-medium tools such as chat and discussion databases. On this basis, AE became a very strong candidate for technology transfer into the Workplace product, where it is currently an important component.

Additional empirical work helped us mitigate weaknesses in the original concept and in our initial

implementation. The alert management work of 2005 proved the concept that we could model individual user alert preferences and ease the burden of the AE notification and awareness features. This second deployment again showed how readily people adopted AE to do real work despite the limitation of only six types of objects and the fact that AE was not integrated with users’ existing e-mail or IM tools. The participatory analysis work of 2005 helped us identify potential new types of objects that could be included within AE, suggesting some of the new extensibility features of the next-generation product. The participatory analysis also helped us discover the principle of selective sharing of shared objects, thus showing how to make AE more usable for groups of heterogeneous users who are collaborating on a shared project but are from different organizations and have different perspectives.

Thus, the empirical work was a crucial part of validating the concept of AE much earlier than occurs for many research prototypes, streamlining the path from research to product. Further empirical work provided early warnings of problems and suggested solutions for those problems that could be understood and adopted in a timely way by the product team.

The usability work of the product team was a natural follow-on to the issues that our field study had highlighted. In effect, the field study showed successes as well as areas for improvement. The usability testing continued this examination, achieving significant advances over what we could accomplish with the research prototype.

The customer comments helped keep us focused on real issues in real workplaces. They helped us to evaluate the utility of many features and to identify crucial integration needs and potential opportunities for new features and functionality, many of which have been pursued by the UAM project.¹¹

Finally, the design explorations provided one way of conceptually testing both proposed improvements and new concepts and opportunities that resulted from the field trial, the usability testing, and the customer feedback sessions. Several features that are now important aspects of the product first appeared in the design explorations. We continue to

use design explorations on the product team to consider how AE may grow in the future.

DESIGNING THE FUTURE: NEXT-GENERATION AE

In planning the next major release of AE, we studied the usage data from our internal research deployments and considered the results from usability studies and the feedback from customers and business partners. A major redesign effort was undertaken to make the core concepts of AE more accessible to users, to improve the first-time experience of the product, to include more features that customers had requested, and to incorporate new research from the UAM¹¹ and dogear⁴³ projects. A complete description of the next-generation work is outside the scope of this paper. Our goal in this section is to provide a glimpse into the future by outlining those aspects that were informed by the research described in this paper.

In order to simplify the user experience and improve users' first-time experience, our new design significantly reduces the number of interface elements in the new release, making it easier for users to manage their activities in a simple activity dashboard. In addition, activity capabilities are, for the first time, being linked to tools with which users are already familiar. Through activity extensions, users are able to add content to activities directly from their mail client, web browser, and office editors without having to switch to a different user interface destination. A side panel control in the next version of the Notes client and in the Workplace Managed Client will allow users to access activities in any context.

The need for scalability and a short response time in large-scale use was a significant pressure on the new design. To meet this demand, a modified data model was adopted in which a distinct notion of *activity* was introduced, where membership was controlled. Based on our research, the new design allowed any activity to have many entries of heterogeneous objects structured as a thread, which inherited their membership from the activity. However, from our research we knew that we needed to support sufficient privacy capabilities to encourage successful collaboration. The new model was therefore extended to provide finer-grained access control without sacrificing performance. This was done by allowing for private entries and by allowing users to

further break down work by including subactivities that had separate membership controls.

To meet the extensibility requirements identified by our research, we are learning from the way web-based applications have evolved into reusable network services. In our new design, activities will be exposed as a service through a simple, open API that allows customers to extend the activities user interface, integrate activities into their existing business systems, or write an entirely new user interface for interacting with the activity service. This approach makes it easy for customers to use our activities service and develop their own specific implementations as needed.

The new design also incorporates activity templates to support reusable activity patterns.¹¹ Templates offer another way to tailor activities to a particular business environment. They also allow users to reuse and share their activities so that individual contributors can capture their personal business processes and share them without having to become experts in traditional workflow applications.

To make activities more broadly accessible, we introduced a web-based dashboard user interface that can be accessed through an ordinary web browser. The dashboard shows a summary of all the activities in which a user is currently participating, similar to the Activity List in AE. The use of web development techniques such as Asynchronous JavaScript^{**} and XML (AJAX) makes a very rich experience for end users without requiring that special software be installed on each of their computers (a key requirement we learned from our customers).

Once we had shifted the primary user interface to the web browser, we considered the challenge posed by our research findings on notification requirements: How best could users be kept aware as activities progress? This need is being met by providing alternative ways to access activity data. Every activity has Atom Publishing Format⁴⁴ and Really Simple Syndication⁴⁵ (RSS) feeds that can be accessed in any third-party feed reader, making it easy for users to keep up with their activities without having to repeatedly visit their activity dashboard. A real-time alerting channel is also being implemented.

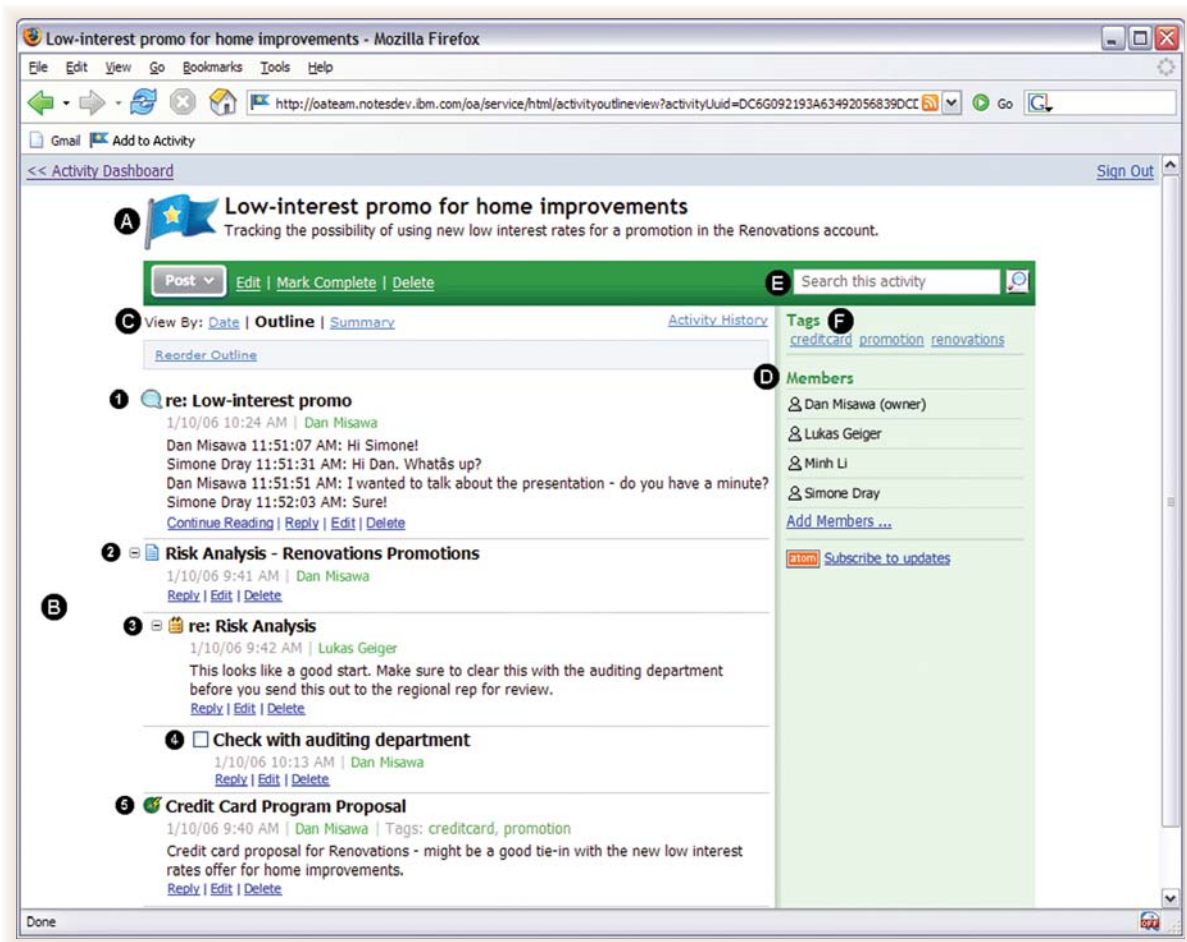


Figure 6
Preview of the new Web-based activity outline view

Figure 6 shows a preview of one of the user interfaces for an individual activity. The activity title and description is presented on top of interface (A). The activity thread is displayed as an expandable outline (B) with a more legible interface than the original activity thread in AE 2.6. Similar to AE, the activity consists of heterogeneous objects, including familiar types such as chat (1), file (2), message or comment (3), and task objects (4), but also new types, such as Web links (5), pointers to Lotus Notes documents (not shown), and meeting objects (not shown). Users can switch the view of the activity (C) to see the hierarchical outline of the activity (shown), a date-ordered view of all the entries in the activity, or a summary view that collects all the similar entries in the activity into a single list. The membership of the activity is always displayed on the right-hand side (D). The new design also

introduces search (E) and shared tagging, a feature that allows users to organize their work by keyword as well as by activity (F). This work was strongly influenced by work on social bookmarking done by the IBM Research Division.⁴³

Future releases of AE will enable a wide array of custom solutions based on the activity service. Further improvements will be made in personal activity management, with particular emphasis on improved interruption management, task prioritization, and deadline management and tighter integration with a user's mail, calendar, and contacts tools. Ultimately, activities will support extranet collaboration, allowing individuals or businesses to collaborate in activities across organizational boundaries.

RELATED WORK

Our developing concept of activity-centric collaboration began from the research literature on collaborative task management and shared objects, including studies on e-mail and activities, collaborative improvements to e-mail, and shared workspace systems. There is a long tradition of collaborative task management in e-mail, stretching back at least as far as the failed experiment of The Coordinator.⁴⁶ In Reference 47, Suchman argues that users rejected The Coordinator because of its underlying assumption that all communication could be forced into the strict, instrumental model of Searles' Speech Acts theory.⁴⁸ The more modest goal of supplementing e-mail with task management (rather than replacing e-mail with task management, as in The Coordinator) was not yet tested.

Since The Coordinator, research on e-mail suggests that e-mail is a "place" where collaboration emerges.^{4,6,7,18,49} For example, Ducheneaut et al.⁷ describe how informal e-mail activities may evolve to more formal task-oriented agreements and actions. Bellotti et al.²³ introduce the notion of a *thrask* as a means for combining e-mail communication with task-like coordination; thrasks provide an integration of diverse object types, similar to our activity threads. Along the same lines, Kaptelinin⁵⁰ presents a system to organize resources into collections that support higher-level activities (see also References 51–53 for the need to combine heterogeneous resources related to a single topic).

However, much of this research tradition focused on the storing of messages (e.g., in a typical e-mail box) of an individual user. Unlike the individualistic perspectives in Bellotti et al.²³ and Kaptelinin,⁵⁰ we provide for shared objects. In this way, our approach is more similar to that of the persistent chats of Babble⁴¹ and the broader set of communication media of the Haystack project.⁵³ However, we go beyond the Haystack sharing of communication and coordination objects in that we include files and tasks, and we provide a means to structure the relationships among the shared objects.

There are several collaborative systems that implement replicated shared objects and collaborative building blocks similar to our prototype. For example, Microsoft Groove⁵⁴ features a large suite of collaborative tools and e-mail integration in a dedicated workspace. In contrast, our approach

focuses not on tools (as in Groove), but rather on shared artifacts and their relationships. In the narrower domain of e-mail, we note similarities to the shared folders and shared collections in Microsoft Outlook⁵⁵ and Lotus Notes.⁵⁶ Collaboration in these shared e-mail solutions is entirely asynchronous (e.g., users cannot work simultaneously on a whiteboard in real time) with no real-time presence awareness.

Finally, Bardram's activity-based computing depends on a deep understanding of medical and hospital practices, where the central artifact is an electronic patient record.⁵⁷ Tasks exist and are individually or collaboratively managed in terms of their relationship to that patient record. In this context, Bardram and colleagues assume that all information associated with the patient record should be visible to all staff members who are concerned with that patient. By contrast, our approach is not tailored to the needs of a particular setting. As a result, we do not assume the existence of a central artifact. We provide the means for establishing a primary artifact, if needed, as the root of an activity thread, but we also provide the means to begin an activity without a central artifact. Also in contrast to Bardram, we do not assume that all members of an activity should have access to all objects within the thread. Rather, we provide the means for selective sharing of objects as users deem appropriate.

CONCLUSION

IBM is striving toward the vision of activity-centric collaboration by introducing the next generation of contextual collaboration technologies. Activity-centric collaboration promises to bring work and team productivity to the next level. Our research and development functions are working hand in hand to make this vision a reality. AE is the first milestone in this joint effort, emerging from a multiyear research effort on activity-centric collaboration that not only influenced product direction, but also the IBM vision for activity-centric collaboration.

This paper described the most significant milestones of this research program and highlighted the most interesting findings. We introduced the vision of activity-centric collaboration and presented AE in IBM Workplace 2.6 as the first IBM activity-centric product in this area. The research behind AE has been conducted by a multidisciplinary team of

designers, ethnographers, computer scientists, and developers with close involvement from product groups. Our primary research instruments were empirical studies, design explorations, usability analyses, and systems engineering, including technical simulations.

In our research on the back end for AE, we investigated alternative architectures and data representations for activities. The lessons learned from this engineering research helped us select server concepts for the activities server product and prepared us to create the first set of user interfaces that were suitable for field testing.

Our empirical work has been crucial in several ways. First, it showed that AE could be used for real work and that it could provide a valuable resource for doing that real work, validating many of our original activities design ideas. Second, it helped us to mitigate weaknesses in the original concept and initial implementation. From a product perspective, this experience helped remove many uncertainties and paved the way for a successful technology transfer from research to development. Customer feedback and usability testing further helped to refine the concepts and our implementation of those concepts. Our design explorations expanded the scope of activity-centric collaboration beyond AE, helping to move the IBM activity-centric collaboration strategy forward into new domains.

Future activity-centric collaboration products from IBM will demonstrate a wide array of custom solutions based on a new, open, and extensible activity service. Our plan is for the core concepts of AE to become more understandable to users, the first-time use of the product to improve, integration with users' productivity tools to improve, and new features that customers have requested to be included. New research initiatives in the areas of personal activity management, activity-centric meeting management, and attention management are tightly aligned with current activity product directions.

ACKNOWLEDGMENTS

We thank interns Sophia Liu, Suzanne O. Minassian (now of IBM Software Group), Shilad Sen, Andrew J. Witt, Roberto Silveira Silva Filho, and Michael Wu for their participation in this research. Researchers Tom Moran and John C. Tang contributed deep

discussions and provocative exchanges as we compared and contrasted AE concepts with UAM concepts. Developers Kyungae Lim, Latoya Sankey, Bill Masek, and Eric Portner first adapted AE to product level. Architects Miguel Estrada, Sami Shalabi, Carl Kriger, Joe Russo, and all the Activities team members made major contributions to the product form of AE; we thank them for their collaborative spirit as their criticisms led all of us toward a more open and flexible activities model. Executives Michael Rhodin, Ambuj Goyal, and Distinguished Engineers Carol Jones and Scott Prager pushed us into new and uncomfortable areas as they helped us see how our research could enhance their IBM products.

We thank designers Sandra Kogan, Andy Schirmer, and Majie Zeller for their unfailingly patient and always constructive criticism of our designs and implementations as we moved from research concepts into usable prototypes. We thank our research community, the Collaborative User Experience group of IBM Research in Cambridge, and the Pesto team at IBM Research in Haifa, particularly Vova Soroka, Kushal Dave (now at Google), Jonathan Feinberg, Edward Bortnikov, and Ido Guy, for their invaluable help with AE development, and Irene Greif, Bernard Kerr, and John Patterson, who helped us turn our weak points into strengths.

*Trademark, service mark, or registered trademark of International Business Machines Corporation.

**Trademark, service mark, or registered trademark of Microsoft Corporation, Sun Microsystems, Inc., or Eclipse Foundation, Inc. in the United States, other countries, or both.

CITED REFERENCES

1. "IBM Workplace: Simply a Better Way," IBM Software, <http://www-142.ibm.com/software/workplace/products/product5.nsf/wdocs/workplaceoverview>.
2. B. Kerr and E. Wilcox, "Designing Remail: Reinventing the Email Client Through Innovation and Integration," *Proceedings of ACM Conference on Human Factors in Computing Systems*, Vienna, Austria (2004), pp. 837-852.
3. M. Wattenberg, S. L. Rohall, D. Gruen, and B. Kerr, "Email Research: Targeting the Enterprise," *Human-Computer Interaction* **20**, No. 1/2, 139-162 (2005).
4. D. Gruen, S. L. Rohall, S. Minassian, B. J. Kerr, P. Moody, B. Stachel, M. Wattenberg, and E. Wilcox, "Lessons from the ReMail Prototypes," *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, Chicago, IL (2004), pp. 152-161.
5. S. Whittaker and C. Sidner, "Email Overload: Exploring Personal Information Management of Email," *Proceedings of the ACM Conference on Human Factors in Computing*

- Systems*, Vancouver, British Columbia, Canada (1996), pp. 276–283.
6. W. E. Mackay, “More Than Just a Communication System: Diversity in the Use of Electronic Mail,” *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, Portland, Oregon (1988), pp. 344–353.
7. N. Ducheneaut and V. Bellotti, *A Study of Email Work Activities in Three Organizations*, Working Paper, Computer Science Laboratory, Palo Alto Research Center, Palo Alto, CA (2006).
8. T. P. Moran, “Unified Activity Management: Explicitly Representing Activity in Work-Support Systems,” Position paper at the *9th European Conference on Computer-Supported Cooperative Work*, “Workshop on Activity: From a Theoretical to a Computational Construct,” Paris, France (2005), <http://www.daimi.au.dk/~bardram/ecscw2005/papers/moran.pdf>.
9. B. L. Harrison, A. Cozzi, and T. P. Moran, “Roles and Relationships for Unified Activity Management System,” *Proceedings of the ACM SIGGROUP Conference on Supporting Group Work*, Sanibel Island, FL (2005), pp. 236–245.
10. T. P. Moran, “Activity: Analysis, Design, and Management,” <http://projectsfinal.interaction-ivrea.it/2004-2005/SYMPOSIUM%202005/communication%20material/presentations/Moran/ivrea-slides.ppt>.
11. T. P. Moran, A. Cozzi, and S. P. Farrell, “Unified Activity Management: Supporting People in e-Business,” *Communications of the ACM* **48**, No. 12, 67–70 (2005).
12. A. Bernstein, “How Can Cooperative Work Tools Support Dynamic Group Processes? Bridging the Specificity Frontier,” *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, Philadelphia, PA (2000), pp. 279–288.
13. M. J. Muller, M. E. Raven, S. Kogan, D. R. Millen, and K. Carey, “Introducing Chat into Business Organizations: Toward an Instant Messaging Maturity Model,” *Proceedings of the ACM SIGGROUP Conference on Supporting Group Work*, Sanibel Island, FL (2003), pp. 50–57.
14. E. Isaacs, A. Walendowski, S. Whittaker, D. J. Schiano, and C. Kamm, “The Character, Functions, and Styles of Instant Messaging in the Workplace,” *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, New Orleans, LA (2002), pp. 11–20.
15. M. Turek, “Instant Messaging: Time for IT to Pay Attention,” *Collaboration Loop* (January 6, 2004), <http://www.collaborationloop.com/features/instant-messaging-time-for-it-to-pay-attention-2.htm>.
16. IBM Lotus QuickPlace, IBM Software, <http://www-142.ibm.com/software/sw-lotus/products/product3.nsf/wdocs/ltwhome>.
17. Microsoft SharePoint Products and Technologies, Microsoft Corporation, <http://www.microsoft.com/sharepoint/default.aspx>.
18. U. Schultz and B. Vendenbosch, “Information Overload in a Groupware Environment: Now You See It, Now You Don’t,” *Journal of Organizational Computing and Electronic Commerce* **8**, No. 2, 127–148 (1998).
19. P. Dourish, “Process Descriptions as Organizational Accounting Devices: The Dual Use of Workflow Technologies,” *Proceedings of the ACM SIGGROUP Conference on Supporting Group Work*, Boulder, CO (2001), pp. 52–60.
20. A. O’Neal, “Discovering Activity Explorer in the IBM Workplace Managed Client” (July 26, 2005), <http://www-128.ibm.com/developerworks/lotus/library/ae/>.
21. P. Dourish, W. K. Edwards, A. Lamarca, and M. Salisbury, “Presto: An Experimental Architecture for Fluid Interactive Document Spaces,” *ACM Transactions on Computer-Human Interaction* **6**, No. 2, 133–161 (1999).
22. T. Takeda and D. Suthers, “Online Workspaces for Annotation and Discussion of Documents,” *Proceedings of the International Conference on Computers in Education*, Auckland, New Zealand (2002), p. 1294.
23. V. Bellotti, N. Ducheneaut, M. Howard, and I. Smith, “Taking Email to Task: The Design and Evaluation of a Task Management Centered Email Tool,” *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Ft. Lauderdale, FL (2003), pp. 345–352.
24. W. Geyer, J. Vogel, L.-T. Cheng, and M. Muller, “Supporting Activity-Centric Collaboration through Peer-to-Peer Shared Objects,” *Proceedings of the ACM SIGGROUP Conference on Supporting Group Work*, Sanibel Island, FL (2003), pp. 115–124.
25. W. Geyer and L.-T. Cheng, “Facilitating Emerging Collaboration through Light-weight Information Sharing,” *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, New Orleans, LA (2002), pp. 221–230.
26. J. Vogel, W. Geyer, L.-T. Cheng, and M. Muller, “Consistency Control for Synchronous and Asynchronous Collaboration Based on Shared Objects and Activities,” *Computer Supported Cooperative Work* **13**, No. 5/6, 573–602 (2004).
27. R. S. Silva Filho, W. Geyer, B. Brownholtz, I. Guy, D. F. Redmiles, and D. R. Millen, *Architectural Trade-Offs for Collaboration Services Supporting Contextual Collaboration*, Research Report RC-23756, IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598 (2005).
28. W. Geyer, J. Vogel, L.-T. Cheng, and M. Muller, “Personal Peer-to-Peer Collaboration Based on Shared Objects,” in *Peer to Peer Computing: The Evolution of a Disruptive Technology*, R. Subramanian and B. Goodman, Editors, IDEA Group Publishing, Munich, Germany (2004).
29. M. Estrada and S. Shalabi, “Details of Lotus Workplace Application Architecture,” [http://web6.notes.net/ldd/sandbox.nsf/ByDateNJ/590DD1EBF5D51EFB85256E51006D5FA8/\\$FILE/ad306.pdf?OpenElement](http://web6.notes.net/ldd/sandbox.nsf/ByDateNJ/590DD1EBF5D51EFB85256E51006D5FA8/$FILE/ad306.pdf?OpenElement).
30. S. Hupfer, L.-T. Cheng, S. Ross, and J. Patterson, “Introducing Collaboration into an Application Development Environment,” *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, Chicago, IL (2004), pp. 21–24.
31. P. Moody and J. Feinberg, *C+B Seen Project*, IBM Watson Research Center, <http://domino.watson.ibm.com/cambridge/research.nsf/99751d8eb5a20c1f852568db004efc90/d23c1156a11a7ac385256e270054bf21?OpenDocument>.
32. S. Hupfer, L.-T. Cheng, S. Ross, and J. Patterson, “Reinventing Team Spaces for a Collaborative Development Environment,” Position paper at the *ACM CHI Conference on Human Factors in Computing Systems*, “Beyond Threaded Conversation” Workshop, Portland, OR (2005), <http://domino.watson.ibm.com/cambridge/research.nsf/c9ef590d6d00291a85257141004a5c19/d28dfc8ea2dee52a85256f87007304c5?OpenDocument>.
33. M. J. Muller, W. Geyer, B. Brownholtz, E. Wilcox, and D. R. Millen, “One Hundred Days in an Activity-Centric

- Collaboration Environment Based on Shared Objects," *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems*, Vienna, Austria (2004), <http://domino.watson.ibm.com/cambridge/research.nsf/c9ef590d6d00291a85257141004a5c19/a81b30656f7d5c9a85256e510056b66e?OpenDocument>.
34. D. R. Millen, M. J. Muller, W. Geyer, E. Wilcox, and B. Brownholtz, "Patterns of Media Use in an Activity-Centric Collaborative Environment," *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems*, Portland, OR (2005), <http://domino.watson.ibm.com/cambridge/research.nsf/c9ef590d6d00291a85257141004a5c19/732e715f09be20a785256ffe006b1bcf?OpenDocument>.
35. M. J. Muller and M. Wu, "How Much Is Shared in a Shared Activity?" *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems*, Montreal, Canada (2006), <http://domino.research.ibm.com/cambridge/research.nsf/c9ef590d6d00291a85257141004a5c19/704d09e4d75a3e21852570a6005ce356?OpenDocument>.
36. M. J. Muller, S. O. Minassian, W. Geyer, D. R. Millen, B. Brownholtz, and E. Wilcox, "Studying Appropriation in Activity-Centric Collaboration," *Position paper at the 9th European Conference on Computer-Supported Cooperative Work*, Workshop on "Supporting Articulation Work: Approaches for the Reflective User," Paris, France (2005), <http://domino.research.ibm.com/cambridge/research.nsf/c9ef590d6d00291a85257141004a5c19/c6ef991f8945fc68852570af0070bac1?OpenDocument>.
37. A. I. Antón and C. Potts, "Functional Paleontology: System Evolution as the User Sees It," *Proceedings of the 23rd IEEE International Conference on Software Engineering*, Toronto, Canada (2001), pp. 421–430.
38. T. K. Bikson and J. D. Eveland, "Groupware Implementation: Reinvention in the Sociotechnical Frame," *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, Boston, MA (1996), pp. 428–437.
39. M. J. Muller and D. M. Gruen, "Working Together Inside an Emailbox," *Proceedings of the 9th European Conference on Computer-Supported Cooperative Work*, Paris, France (2005), <http://domino.research.ibm.com/cambridge/research.nsf/c9ef590d6d00291a85257141004a5c19/75b1309ed1cfff27852570a5006c18f4?OpenDocument>.
40. M. J. Muller, "Layered Participatory Analysis: New Developments in the CARD Technique," *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems*, Seattle, WA (2001), pp. 90–97.
41. T. D. Erickson and W. A. Kellogg, "Social Translucence: An Approach to Designing Systems That Support Social Processes," *ACM Transactions on Computer-Human Interaction* 7, No. 1, 59–83 (2000).
42. W. Geyer, A. J. Witt, E. Wilcox, M. Muller, B. Kerr, B. Brownholtz, and D. R. Millen, "Chat Spaces," *Proceedings of the ACM Conference on Designing Interactive Systems*, Cambridge, MA (2004), <http://domino.watson.ibm.com/cambridge/research.nsf/c9ef590d6d00291a85257141004a5c19/2dce4bfacdf3cc9f85256f0d008164f8?OpenDocument>.
43. D. R. Millen, J. Feinberg, and B. Kerr, "dogear: Social Bookmarking in the Enterprise," *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, Montreal, Canada (2006), pp. 111–120.
44. Atom Publishing Format and Protocol, <http://www.ietf.org/html.charters/atompub-charter.html>.
45. D. Winer, *RSS 2.0 Specification*, Berkman Center for Internet & Society at Harvard Law School, <http://blogs.law.harvard.edu/tech/rss>.
46. F. Flores, M. Graves, B. Hartfield, and T. Winograd, "Computer Systems and the Design of Organizational Interaction," *ACM Transactions on Information Systems* 6, No. 2, 153–172 (1988).
47. L. Suchman, "Do Categories Have Politics? The Language/Action Perspective Reconsidered," *Proceedings of The ACM Conference on Computer Supported Cooperative Work*, Chapel Hill, NC (1994), pp. 177–190.
48. J. R. Searle, *Speech Acts: An Essay in the Philosophy of Language*, Cambridge University Press, Cambridge, England (1969).
49. N. Ducheneaut and V. Bellotti, "E-mail as Habitat: An Exploration of Embedded Personal Information Management," *ACM Interactions* 8, No. 5, 30–38 (2001).
50. V. Kaptelinin, "UMEA: Translating Interaction Histories into Project Contexts," *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, Ft. Lauderdale, FL (<http://portal.acm.org/toc/cfm?id=642611&type=proceeding&coll=GUIDE&dl=GUIDE&CFID=76655168&CFTOKEN=508264672003>), pp. 353–360.
51. R. Boardman, "Workspaces That Work: Towards Unified Personal Information Management," *Proceedings of the 16th British HCI Conference, People and Computers XVI*, London, England (2002), pp. 216–217.
52. E. F. Churchill, J. Trevor, S. Bly, L. Nelson, and D. Cubranic, "Anchored Conversations: Chatting in the Context of a Document," *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, The Hague, The Netherlands (2000), pp. 454–461.
53. D. Quan and D. R. Karger, "Haystack: Metadata-Enabled Information Management," *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology*, Vancouver, Canada (2003), <http://haystack.lcs.mit.edu/publications.html>.
54. Groove Networks, <http://www.groove.net/home/index.cfm>.
55. Microsoft Outlook, Microsoft Corporation, <http://www.microsoft.com/outlook/>.
56. IBM Lotus Notes, IBM Corporation, <http://www-142.ibm.com/software/sw-lotus/products/product4.nsf/wdocs/noteshomepage>.
57. J. E. Bardram and C. Bossen, "A Web of Coordinative Artifacts: Collaborative Work at a Hospital Ward," *Proceedings of the International ACM SIGGROUP Conference on Supporting Group Work*, Sanibel Island, FL (2005), pp. 168–176.

Accepted for publication April 12, 2006.

Published online October 16, 2006.

Werner Geyer

IBM Research Division, Thomas J. Watson Research Center, 1 Rogers Street, Cambridge, Massachusetts 02142 (werner.geyer@us.ibm.com). Dr. Geyer is a research staff member in the Collaborative User Experiences Group. He is currently investigating the problem of activity attention management. He has been working on Activity Explorer and activity-centric collaboration for more than four years. He has an M.S. degree in information technology and a Ph.D. degree in computer science, both from the University of Mannheim,

Germany. Dr. Geyer is a member of the Association for Computing Machinery.

Michael J. Muller

IBM Research Division, Thomas J. Watson Research Center, 1 Rogers Street, Cambridge, Massachusetts 02142 (michael_muller@us.ibm.com). Dr. Muller is a research staff member in the Collaborative User Experiences Group. He has a Ph.D. degree in cognitive psychology from Rutgers University. As a member of Computer Professionals for Social Responsibility and the Association for Computing Machinery, Dr. Muller is an internationally recognized expert in participatory design.

Martin Thomas Moore

IBM Software Group, 5 Technology Park Drive, Westford, Massachusetts 01886 (martmoor@us.ibm.com). Mr. Moore is a product designer for the IBM Workplace™ family of software. He is the lead designer for the development team that is adapting Activity Explorer for IBM Workplace. He holds a B.S. degree in graphic design from North Carolina State University.

Eric Wilcox

IBM Research Division, Thomas J. Watson Research Center, 1 Rogers Street, Cambridge, Massachusetts 02142 (eric_wilcox@us.ibm.com). Mr. Wilcox is a design researcher in the Collaborative User Experience Group. Mr. Wilcox earned B.S. degrees in computer science and mathematics from Oregon State University and an M.S. degree in Human Computer Interaction from Carnegie Mellon University. Mr. Wilcox currently chairs the AIGA Boston Experience Design Community of Interest.

Li-Te Cheng

IBM Research Division, Thomas J. Watson Research Center, 1 Rogers Street, Cambridge, Massachusetts 02142 (li-te_cheng@us.ibm.com). Dr. Cheng is a research staff member in the Collaborative User Experiences Group. He is currently investigating how visualizations can provide further insight in software development and foster collaboration among developers. He has a Ph.D. degree in electrical engineering from Memorial University of Newfoundland and M.S. and B.S. degrees in systems design engineering from the University of Waterloo. Dr. Cheng is a member of the Institute of Electrical and Electronics Engineers and the Association for Computing Machinery.

Beth Brownholtz

IBM Research Division, Thomas J. Watson Research Center, 1 Rogers Street, Cambridge, Massachusetts 02142 (beth_brownholtz@us.ibm.com). Ms. Brownholtz is a senior software engineer in the Collaborative User Experiences Group. She was a developer on the Activity Explorer research prototype and on the team that brought AE technology into IBM Workplace. Ms. Brownholz holds a B.S. degree in computer science from Rutgers University.

Charles Hill

IBM Software Group, 5 Technology Park Drive, Westford, Massachusetts 01886 (charlesh@us.ibm.com). Mr. Hill is a Senior Technical Staff Member and a user-experience designer working on collaboration and portal software. He has a Ph.D. degree in mechanical engineering from Imperial College, London, and an M.Des.RCA in industrial design engineering from the Royal College of Art, London. He was the user-experience design lead for Lotus QuickPlace®. Mr. Hill's current focus is on social software and activity-centric computing.

David R. Millen

IBM Research Division, Thomas J. Watson Research Center, 1 Rogers Street, Cambridge, Massachusetts 02142 (david_r_millen@us.ibm.com). Dr. Millen is a research manager in the Collaborative User Experiences Group. Through field studies and development of new social software applications, his group studies the social and technological implications of online communities and large-scale collaboration. Dr. Millen has a Ph.D. degree in cognitive psychology from Rutgers University. ■