TeamRooms: Network Places for Collaboration

Mark Roseman and Saul Greenberg

Dept. of Computer Science, University of Calgary
Calgary, Alta, Canada T2N 1N4
Tel: +1-403-220-3532
E-mail: roseman,saul@cpsc.ucalgary.ca

ABSTRACT

Teams whose members are in close physical proximity often rely on team rooms to serve both as meeting places and repositories of the documents and artifacts that support their projects. TeamRooms is a groupware system that fills the role of a team room for groups whose members can work both co-located and at a distance. Facilities in TeamRooms allow team members to collaborate either in real-time or asynchronously, and to customize their shared electronic space with tools to suit their needs. Unlike many groupware systems, all TeamRooms documents and artifacts are fully persistent.

KEYWORDS: Groupware, shared electronic spaces, GroupKit

INTRODUCTION

Johansen [6] describes how team rooms have become an important tool used by business teams to organize their work. Team rooms provide a permanent shared space used by the team, which can serve as a meeting room, a work area, a place to store documents that are needed by the team's projects, and more generally, as a venue for communication within the group.

Traditional team rooms rely on the physical proximity of the team members and their easy access to the room. This access breaks down when workers are distributed, as is increasingly the case today. Trends towards more flexible organizations and telecommuting can result in teams whose members are at various times scattered over large geographic areas. For such teams, shared physical spaces are not an option.

We describe here a groupware system called TeamRooms that supports the team room concept for teams whose members work either co-located or at a distance. It combines aspects of both real-time and asynchronous groupware to provide the team with a "network place" serving as a locale for their collaborations. The system is highly customizable, allowing the team to bring different tools into their electronic room on the fly to suit their needs, just as they do with their physical meeting rooms.

Each room contains both generic communication tools (a chat tool and a backdrop acting as a shared whiteboard) and any number of *applets* needed to support the group's work. Typical applets would be diagramming tools, outliners, brainstorming tools, browsers for information such as web pages, notes to other team members, as well as more frivolous items such as card games. When team members are in a room at the same time, they see not only each other but also each other's actions, both through immediate changes in the room's artifacts and through mechanisms such as multiple telepointers. As with real rooms, all artifacts in the electronic room persist even when no one is in the room.

After describing the metaphor used in TeamRooms, this paper continues with an overview of its user interface features that support collaboration. We then describe some of the different applets we have constructed. We also examine briefly the system's architecture and implementation, and report on our early usage experiences with it.

MOTIVATION

Unlike many real-time groupware systems that support the metaphor of a *meeting*, TeamRooms is based on the metaphor of a *place*. Here we characterize the attributes of places that motivated our system design: they are long-lived, their contents are persistent, they provide a forum for people to communicate, and they can contain both general and special purpose tools.

Places are generally long-lived, and can play host to any number of events (such as meetings) held within them. Unlike meetings, places address the issue of what happens outside the real-time collaboration itself. Places organize work; for example, a group can choose a particular place to work on a given project. The group can use this place to meet, to work on tasks both individually and collectively, to store project artifacts, and to leave project information for others.

Unlike meeting-centered models where shared documents do not exist beyond the meeting (unless saved as files or exported to other tools), places are inherently persistent. Any object in the place continues to exist even when all users leave the place, and later visits to the place by one or more users will find the object unchanged.

Places provide a venue for communication. When several people inhabit a conventional space at the same time, communication between them is afforded. For example, they can see and talk to each other.

Places offer common tools for collaboration. For example, team rooms or meeting rooms provide sets of conversational props that are always ready-to-hand, including common tools such as whiteboards, flip charts, or overhead projectors.

Finally, groups can bring in special purpose tools to customize the place to suit their own particular needs. These will be as diverse as the group's project or task, and can include books, reports, documents, writing tools, ongoing status or progress reports, calculating or computing devices, task lists, and so on.

The challenge we have is to mimic these characteristics of physical team rooms in electronic ones. This is reflected in the following design requirements for TeamRooms:

- rooms must be long-lived, and usable by both individuals and groups
- rooms and their contents must be fully persistent
- rooms should provide interpersonal communication facilities
- generic collaboration tools should be automatically made available in each room
- special purpose tools can be added to any room

We then added the following considerations particular to electronic rooms:

- rooms must be accessible from anywhere on a network
- access to rooms must be limited to a designated group
- external computer and network information should be easily accessible from within the rooms

USER INTERFACE

Figure 1 illustrates the TeamRooms user interface, where the user (Mark) is in a room called "Mark Roseman's Room" with two other users (Carl and Saul). Also shown are six applets: a concept map tool, a sticky note, an image, a URL reference, an outliner, and a tetrominoes game.

In the discussion below, we first show how a user starts the system, and how they can create a new room or enter an existing room. The standard tools in every room (such as the shared whiteboard) are then described, including those used when meeting others in a room. In the next section, we move on to describe the applets, which can provide more specific support for a group's activities.

Starting Up

When users first start up, they connect to a central TeamRooms server that maintains a set of rooms for the group. To gain access, they are prompted for a user name and password. If they are among the work group permitted to use the server, they are then placed in a user-specified default room. Before detailing the contents of a room, we first describe how users can navigate to other rooms on the server, or can create new ones.

Entering other rooms. Figure 1a displays a list of all rooms available to the group. Users enter other rooms by selecting them from this list. Though primarily used for navigation, the list also displays which users are in each room. This can highlight existing meetings or sessions between several users, or it can act as an affordance for casual interaction.

Creating new rooms. Users create new rooms by choosing a command from the Rooms menu, specifying a name for a room. Currently the room's creator has no special priveleges, allowing others in the work group to enter, change or even delete the room. We plan to provide more fine-grained access control in the future.

Locating other users. Figure 1b displays a list of all users currently connected to the group's server, as well as the room each user is currently working in. This helps to quickly locate other people, and facilitates making contact with them. Their images are normally still images (e.g. that the user provides from a scanned-in picture). When a video camera is available, we replace the still image with a snapshot taken several times per minute. This provides useful information about whether other users are actually present and available for collaboration, while still using very modest bandwidth. These snapshots serve a function similar to those in Portholes [2].

User information. More information on users can be found by selecting them from the user list. This will display a "business card" window as shown in Figure 1c. Having

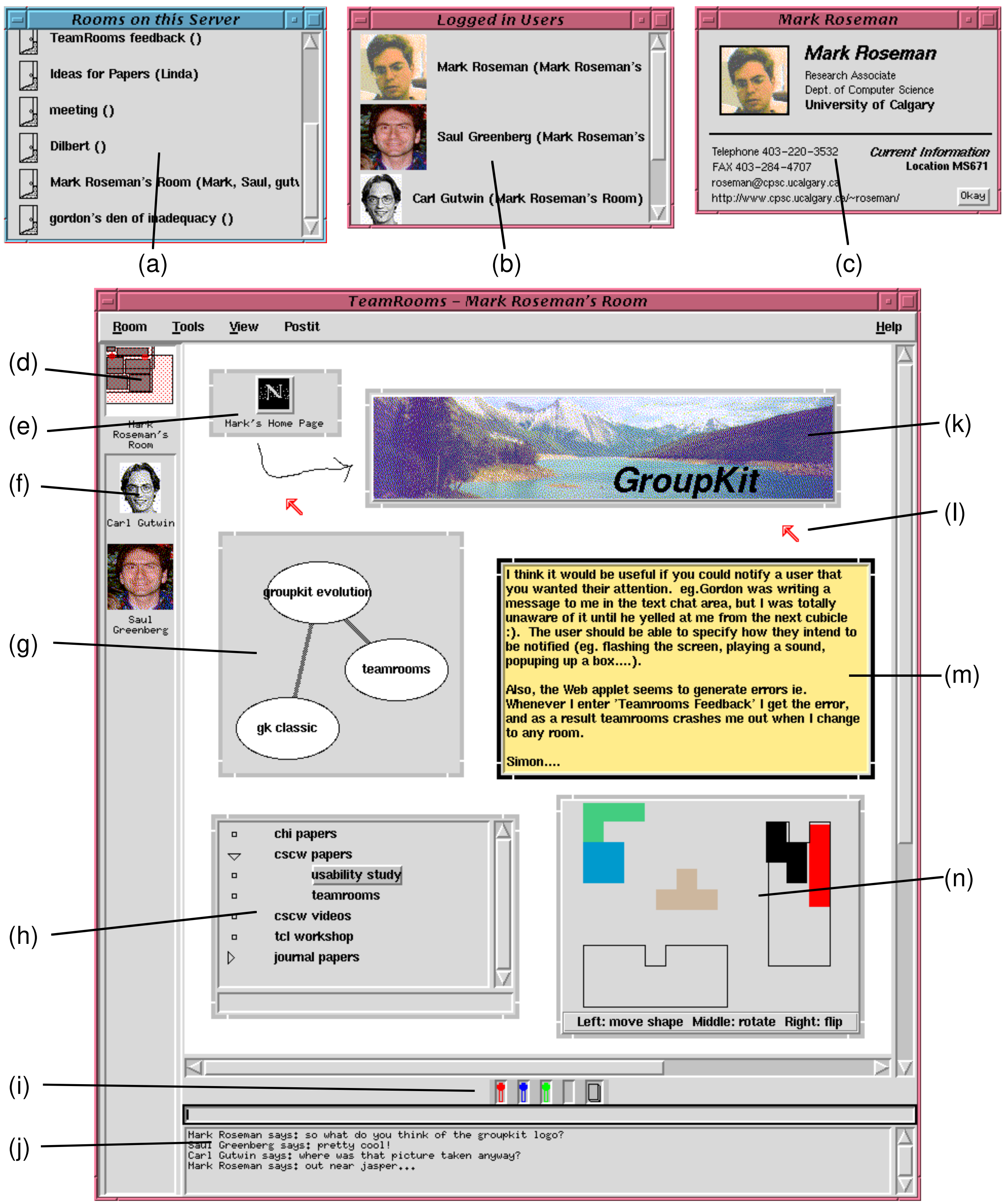

Figure 1. TeamRooms user interface, showing (a) available rooms; (b) connected users; (c) business card; (d) radar overview of room; (e) URL reference applet; (f) users in room; (g) concept map applet; (h) outliner applet; (i) whiteboard pens; (j) text chat area; (k) image applet; (l) telepointer; (m) postit applet; (n) tetrominoes applet.

information such as phone numbers or current location available can assist in setting up phone calls in parallel with TeamRooms.

Standard Tools in Every Room

Standard features are available in every room. Though each room in TeamRooms can be customized to suit the specific needs of the team and their task, certain generic facilities are provided that we think are useful for a wide range of groups and activities.

Chat tool. A simple text-based chat tool (Figure 1j) is provided that allows users to type messages to each other when they are in the same room. Though we would hardly expect a lengthy meeting to be carried out in TeamRooms without the benefit of an external audio channel (e.g. a conference call), we realize that text can be a useful medium for short or sporadic interactions, or where it is impractical to provide an audio connection for reasons of cost or logistics. While audio or video conferencing can provide an excellent complement to TeamRooms, we chose not to make such facilities necessary to use the system.

Shared whiteboard. A second facility is a shared whiteboard, which occupies the "walls" of each room. Users select different colored pens or the eraser from the pen tray (Figure 1i), and can produce freehand drawings. The colors have no special significance, though group members could of course decide to use different colors to identify themselves. Shared whiteboards are common in groupware, allowing users for example to design new artifacts and annotate existing ones [3].

Awareness Features

TeamRooms also provides several facilities for maintaining awareness of other team members in the room, both a general awareness of who is around, and a more fine-grained awareness of others' actions in the workspace [5].

Room users. TeamRooms displays a list of users in the current room, as shown along in Figure 1f. As before, the display is either of image stills, or video snapshots updated several times per minute. Though we have not implemented it, we could easily see these images replaced by a video-conferencing system when appropriate facilities and bandwidth are in place. The system can also show idle times for each user, to indicate the likelihood they are present and attending to the room.

Telepointers. Within the workspace itself, telepointers, one for each user, communicate gestures to provide a fine-grained sense of awareness of the actions of other users (Figure 1l).

Room overview radar. Because the room can be larger than will fit on a single display, a radar view (Figure 1d) provides a miniature overview of the entire room. The radar shows the locations of all applets in the room, the position of each user's viewport into the room, and miniature telepointers to show the location of their mouse cursor. As users move around the room and manipulate applets, the radar tracks their actions.

APPLETS

While the facilities described previously provide fairly generic support for collaboration, applets in TeamRooms are designed for more specific needs of the group. Each applet is in fact a special-purpose groupware application, which team members can include in their rooms as needed.

For example, a room used to manage a particular software project may have applets for task lists, bug reporting forms, a pointer to an online version of the project's specification and so on. In contrast, a "coffee room" might have applets for a card game, online comics, or electronic postcards left in the room by travelling colleagues.

As shown in Figure 1, each applet is embedded in its own frame within the room, in a similar fashion to OpenDoc or OLE components [9]. At any time, users may select new applets from the Tools menu, choosing from a list of available groupware applications. Users can move, resize, and delete applets. All such changes are immediately visible to all users in the room.

Though applets are full groupware applications, we prefer those with simple, direct manipulation interfaces. This reflects our belief that complex interfaces in groupware can impede group interaction. However, more complex applets can choose to add commands to the global menubar. When the applet has the focus, its menu commands are available.

TeamRooms also keeps a complete version history for each applet. Versions of the applet's contents are automatically saved when the last user in a room leaves. Users can browse and retrieve earlier versions, which creates a new instance of the applet in the room. This allows users to compare the two versions, or to review earlier stages in the project.

TeamRooms supports any of the types of applications which could be constructed in GroupKit [12], including text editors, drawing tools, card games, meeting tools, and so on. Several of the applets currently included in TeamRooms are described next. Most are fully group-aware, allowing shared views, immediate updates of fine-grained actions, and simultaneous editing. Applets are constructed using the full GroupKit API, as discussed later.

PostIt

One useful applet is the ubiquitous sticky note (Figure 1m). Once created, text in the notes can be changed at any time by any user in the room, and changes are immediately seen on other users' displays. As with their paper counterparts, these notes are used to leave messages for people, to serve as reminders, and to comment on other artifacts in the rooms.

Outliner

The groupware outline tool (Figure 1h) allows team members to hierarchically organize a set of notes or ideas. For example, one use was to keep track of ideas for papers, grouping possible papers under upcoming conferences or journals, and adding comments on the viability of each paper. Each item, which consists of one or more lines of text, can be placed (using a drag and drop technique) within the hierarchy of ideas, and portions can be collapsed and expanded. All actions are immediately visible to all users.

Concept Map

The concept map applet (Figure 1g) provides another way to organize information, this time as a graph. Nodes in the graph can represent ideas, while edges allow different ideas to be related. Users in the room can of course simultaneously add and edit nodes and edges.

Games

The tetrominoes puzzle (Figure 1n) is one of several "non-productivity" applets in TeamRooms. The idea is to have several users simultaneously manipulate the colored pieces to fit them within the outlines. In a similar vein, TeamRooms could provide applets such as a card table (no rules necessary, pick your own game with others in the room) or a chess board.

Image Tool

Conventional rooms often contain graphical images, either work related or for decoration. We added an applet which could display an image in a room (Figure 1k). Rather than storing the image on the TeamRooms server, we instead store a reference to that image, in the form of a URL. The image applet uses the reference to retrieve the image file from an HTTP server, and then to display it. Clicking on the image brings up a dialog to change the URL used to fetch the image. Common uses were for personal pictures, as well as pointers to popular daily comic strips available on the Web, such as the "Dilbert of the Day."

Database

A very simple database applet is provided (not shown). Users can define a set of fields (giving each field a name and choosing a type for the field), and then manipulate records. While this is not a replacement for a full database, it is sufficient to hold small collections of information, such as for a personal address book or managing tasks for a small project.

File Transfer

Another applet (not shown) allow external files to be placed in rooms. Users can upload the file into the room, where it appears as an icon and filename. Other users can later download the file from the room to their own file system. Newer versions of files can be uploaded, while TeamRooms' version mechanism allows access to the older versions.

External URL Reference

To import external information into TeamRooms, we developed an applet that contains a pointer to a WWW page. Clicking on the icon sends the URL associated with the applet to the user's web browser, while clicking on the label allows changing the URL (Figure 1e). This is not a shared view application, for while the URL is shared, the action of invoking the browser is strictly local.

GroupWeb

As a final example of including outside information, we adapted our GroupWeb browser [4] to TeamRooms (not shown in Figure). GroupWeb couples a simple Web browser with groupware features such as telepointers and page synchronization. While the external URL reference applet is convenient for leaving pointers for other team members to look at on their own, GroupWeb is useful to retrieve pages into a shared view, and discuss information with others in real-time, such as in a presentation or meeting.

Other Possibilities

Our major effort so far has been on the TeamRooms infrastructure and its associated generic collaboration facilities. However, we believe the system's strength is its ability to complement those facilities with specific applications meeting the group's needs. The applets presented are only the beginning. Other possibilities include decision support tools, drawing tools, version control for external files, front-ends to external databases, calendars, and other information systems.

IMPLEMENTATION

TeamRooms is implemented using an extended version of our groupware toolkit GroupKit [12]. This section first briefly describes TeamRooms' system architecture. For readers familiar with GroupKit, we describe how it was adapted to provide new features needed in TeamRooms.

System Architecture

TeamRooms consists of three separate components: the server, the user client (already described), and the administration client. A single server process runs on some central machine, and any number of clients (user or administration) can connect over a network to that process.

The server acts as a communication hub, so that for clients to communicate with each other, they connect to the server. A database of registered users is maintained, and clients must supply a valid username and password to connect. The server also maintains a persistence repository, which stores information about the rooms on the server and their contents. When clients enter rooms, information about the room is retrieved from the repository. Each server can hold any number of rooms.

The administration client is used by whoever maintains the server. It can be used to create and delete accounts, change access permissions. For example, the administrator can specify that a particular user cannot create new rooms, or can give administrative priveleges to other users. It also provides tools to help manage the persistence respository, for example, specifying how long to keep old versions of rooms before deleting them.

Servers currently run on Unix machines. Both client programs currently run either on Macintosh, Windows 95, or various Unix systems under X11. Clients on different platforms do interoperate of course. We designed the system to not require very high bandwidth network connections, so it is even reasonable for users to use TeamRooms over 14.4 modem connections or long distance Internet connections.

GroupKit Extensions

GroupKit provides an Application Programming Interface (API) based on the Tcl language [10], that makes it easy to construct real-time groupware applications. GroupKit provides constructs such as message passing, shared data structures, and high-level groupware widgets to programmers. We wanted to preserve the API in TeamRooms, to move existing GroupKit applications into TeamRooms and rapidly create new ones. This necessitated a number of fundamental changes in GroupKit's internals.

There are a number of important differences between the TeamRooms application and programs built under our GroupKit toolkit. TeamRooms runs on Unix, Macintosh or Windows, while GroupKit applications run only under Unix. TeamRooms provides an integrated environment, unlike GroupKit which spawns a separate process running in a separate window for every conference (the equivalent of an applet in TeamRooms) resulting in a much more loosely coupled environment. GroupKit also provides for different session management strategies (again as a separate process), where TeamRooms bundles its policy within the application itself. TeamRooms supports both real-time and asynchronous collaboration, while GroupKit focuses almost exclusively on real-time groupware.

While original GroupKit applications ran in a peer-to-peer network arrangement, for TeamRooms we added support for a more traditional client-server arrangement, where messages directed at a peer are routed through a central server. This was necessary because of the greater amount of persistent data and a need to support less robust workstations connected over slower networks. Control remains in the clients however, with the server acting mainly as a message bus. The server has an understanding of GroupKit constructs, but no knowledge about the specific applications that make up TeamRooms. The central server was also important in supporting user authentication (i.e. logging in), which was more important for TeamRooms than in the more open arrangement we were using with GroupKit.

GroupKit has supported a primitive persistence mechanism, where the state of a groupware application could be saved and then recalled at a later time. We extended this mechanism to provide a more robust and richer persistence repository that would store an arbitrary number of versions of the application's state on the TeamRooms server, allowing users to browse the entire history of a groupware document.

By far the biggest challenge was supporting multiple groupware tools within the same application. GroupKit was originally designed with one groupware tool per process (and per window). We changed this to allow several Tcl interpreters within each process, each with its own groupware tool. Each interpreter behaves exactly as if it were running in its own standard GroupKit application. In TeamRooms, we have seperate interpreters for the main application, the overall room window (e.g. containing the whiteboard and chat area), and for each applet. Communication facilities (e.g. socket connections) are shared, so that messages sent from an applet interpreter are routed through the application's main interpreter. This is illustrated in Figure 2.

To achieve the OpenDoc/OLE style window embedding, we used a similar technique. Each subinterpreter believes it

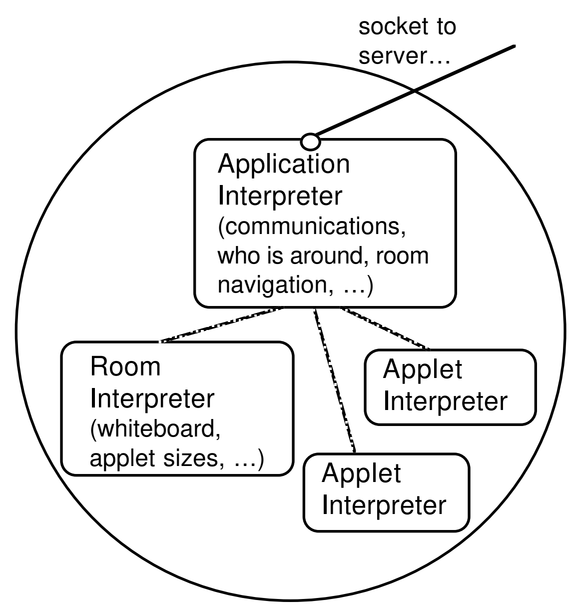

Figure 2. Structure of TeamRooms client, showing use of multiple interpreters.

"owns" the entire window hierarchy, while in reality its "top" window is mapped onto one portion of the window hierarchy in the main application. This was also used to support menu sharing, so that applets could add items to the main menu when they had the focus.

Despite these major changes in GroupKit, existing conferences required only minimal changes to move from standalone GroupKit applications to embedded TeamRooms applets. Applets are programmed using the full set of groupware constructs provided by GroupKit, including remote procedure calls, events, shared data structures, and groupware widgets [12]. An in-depth discussion of some of the architectural details, particularly the issues surrounding the use of multiple interpreters and window embedding, is provided elsewhere [11].

USAGE EXPERIENCES

Though TeamRooms is still under development, various early releases have been in use for several months by colleagues and "strangers" both at our site and from several others around the world. This section describes some of the preliminary experiences with the system.

First, TeamRooms crosses the synchronous/asynchronous barrier. Most of the the "interactions" we saw with the system were asynchronous. There were many occurrences of leaving information for others, whether the general community or a particular user. For example, one room devoted to feedback on the system itself was used heavily to leave notes about bugs or desired features. Responses usually took the form of other notes or addendums to the original. A second example is a room used for brainstorming ideas for new projects; users would browse the existing ideas stored in an outliner, and occassionally add new ideas of their own or comment on existing ones. While we think the amount of asynchronous interaction was increased by different time zones (the most frequent interactions were between Calgary and New Zealand), we did expect a relatively large number of non-real time interactions. As users browsed rooms, it became clear however that they needed some indication of which rooms had changed since they last visited, and what was new in them.

Second, TeamRooms greatly simplifies the setup of real-time collaborations. When real-time interactions did occur, they seemed basically the same as interactions we had observed in other real-time groupware systems. What differed dramatically was the ease of establishing the collaboration, a laborious process in most systems. In many systems, connections with individual participants must be made, or meetings must be created, stocked with groupware tools, and external information imported — for every new meeting. The chat tool and whiteboard that are part of every room were used frequently and heavily. Because there was no need to specifically "import" many objects (i.e. they were already in the room) it was easy and natural to use them as conversational props in the interaction.

Next, we found the tight integration in TeamRooms, where all applets are embedded in a single window, to be very effective. We had previously developed a rooms-based system in GroupKit where conferences (applets) ran in separate processes in their own window, along with other single-user applications. We found the integration and containment in TeamRooms greatly simplified the system. The applets felt like they were "in" the room, and not something that ran alongside it. Window management is greatly simplified, and users can rely on the positioning of applets (for example, placing a postit note or graphical annotation beside an uploaded file), without worrying about other users moving windows on their own screens to disrupt the layout. Gesturing is seamless across applets and the entire room.

While we realize that some power users may feel constrained by the tight integration, our experiences suggest very strongly that this model will be appropriate for the vast majority of users. Note also that the TeamRooms model does not preclude external applets running in their own windows, in the same manner that Web browsers do not preclude helper applications. In fact, the Tcl/Tk environment TeamRooms is based on has long supported a notion of *hypertools*, which extend the notion of helper applications by allowing external programs to communicate with each other [10].

Fourth, TeamRooms supports casual interaction. A phenomenon we observed was the practice of "hanging around", where users would connect to the server, bring up their window showing what other users are connected, and then carry on with other work. Using the "users" window they could see when others connected to the server and where they were working. If they wanted to initiate a real-time interaction, they could easily enter the same room as the other user. Because entering and leaving rooms is very lightweight, this encouraged a number of "quick chats" that would be cumbersome in groupware systems involving more complex setup for interactions.

Finally, TeamRooms supports both group and individual work. We looked at the types of things people did with rooms. As expected, one group of rooms was created to hold information on different projects (e.g. TeamRooms bugs) or for different activities (e.g. a room to hold meetings). Some rooms were also used strictly for single-user tasks, such as a project manager who found some of the TeamRooms applets useful to track tasks on a project — noone else used this room. We also found that users created personal rooms for themselves, analogous to the way people create individual WWW home pages. Besides being used by their creators to hold pictures etc., others used these rooms to leave messages for the room's creator. Though it was evidenced in other rooms as well, there was a definite sense of "property rights" in these personal rooms, where users who created rooms or objects felt they owned and controlled them and were surprised when they were changed or deleted by others.

We are very encouraged by these early experiences. Users have been quick to grasp the system's workings, and have been able to share work with each other in what seem very natural ways.

RELATED WORK

This section describes how TeamRooms relates to existing groupware systems. We first differentiate between the metaphor of place used in TeamRooms and other metaphors found in groupware systems. We then look at how the place metaphor has been used in different ways, and describe two place-based systems, CommonPoint and wOrlds, that bear similarities to TeamRooms.

Metaphors for Groupware

A common metaphor in conventional real-time groupware systems is "groupware as meeting" or "session". Systems based on this metaphor support distributed teams within single meetings, but provide limited persistence or support for asynchronous work.

Other systems rely on "groupware as process." Some real-time systems such as group decision support systems structure stages in the group's overall work processes. However, most support only a very restricted set of unstructured group activities. Similarly, workflow systems also structure asynchronous activity.

In contrast, the "groupware as place" metaphor captures both synchronous and asynchronous collaboration, is highly persistent, encapsulates both structured and unstructured work through its applications, and also takes into account both individual and group work.

Places for Session Management

Several systems have used "place" or "room" as a metaphor for session management. Session managers are programs that run other groupware tools, letting users create and join groupware sessions. Place-based session managers provide gathering points for collaborators; when several people are at the same point, they are connected by other tools. For example, the CAVECAT media space [8] used the idea of walking into the same room as another user to initiate a audio/video connection. The DIVA Virtual Office Environment [13] and GroupKit's Rooms session manager [12], use rooms to gather and organize people and their documents. Users in these rooms launch external groupware editing tools, which run in their own windows, separate from the room itself. This is a different approach from TeamRooms, which integrates the tools with the meeting place.

Places for Low-Fidelity Interaction

The simplest place-based systems are chat rooms, where users connect to a room and can chat to others in the room with typed messages. Recent chat rooms found on online services have added rudimentary graphics, but the interaction is still entirely text-based.

Another popular class of place-based systems are Multi-User Dungeons (MUDs). Born of text-based adventure games, traditional MUDs are text-based systems where users connect to a central server. Once connected, they enter any number of different rooms, chat with other users in those rooms, and type commands to create and modify objects in the rooms. Though primarily used socially, MUDs have been used in limited ways to support collaborative work. Several recent systems have augmented MUDs with non-textual tools, such as the Jupiter project [1] which added MBONE audio and video conferencing and shared whiteboards.

Places for High-Fidelity Interaction

wOrlds. The wOrlds system [14] is based around a number of "locales", which are similar to rooms in TeamRooms. Each locale collects both users and tools, which can include embedded tools like in TeamRooms as well as links to external tools running in their own window. Audio and video conferencing is also directly supported. TeamRooms is very similar to wOrlds, but requires more modest hardware, and does not directly support audio or video (though it could be extended to do so). The wOrlds system also relies very heavily on external single-user tools, which makes the environment feel less integrated than in TeamRooms where most tools are fully multi-user and embeded in the room.

CommonPoint. Probably the most sophisticated place-based system that we are aware of with strong similarities to TeamRooms is the CommonPoint desktop from Taligent (described in [9]). The desktop uses a "People, Places, and Things" metaphor to provide network places where people can gather and work with shared documents. Based on Taligent's very sophisticated object frameworks technology, CommonPoint provides a rich, fluid and highly customizable environment for collaboration. CommonPoint however requires radical changes, as it completely replaces conventional operating systems and their desktop metaphors. Recent strategic changes at Taligent have left it unclear how CommonPoint might evolve and integrate with traditional operating systems. In contrast, TeamRooms fits well in today's environments and operating systems.

Toolkits for Building Places

CBE [7] is a software architecture that can be used to construct place-based systems. Their programmer interface in many ways parallels the internals of TeamRooms, except that TeamRooms tends to leverage off existing GroupKit primitives such as shared data structures. However TeamRooms, being an application rather than a toolkit, does not export much of this API to external programs or applets.

CONCLUSIONS

TeamRooms combines the rich applications and interfaces found in today's real-time groupware applications with a persistent work context equally suitable for asynchronous collaboration. The resulting system provides the electronic equivalent of a team room for groups that are either co-located or distributed. By providing both generic communication facilities as well as very specific tools, rooms can be tailored to fit the unique needs of the work group.

In our early usage experiences, we have found the "groupware as places" model supported by TeamRooms to be very suitable for our primarily asynchronous working styles, as well as providing a seamless transition towards real-time collaboration by providing a place to make contact with others. We have found that the system does afford many of the same behaviors seen when teams share a physical space.

Though many of its ideas have been drawn from other groupware systems, TeamRooms' flexibility and tight integration of features makes it a sophisticated yet easy to use environment for supporting real-time and asynchronous collaborations.

ACKNOWLEDGEMENTS

Thanks go to our users, in particular Gordon Paynter and Simon Gianoutsos of University of Waikato, who braved early versions of the system, and offered many useful suggestions and improvements. The code for applet window embedding is based on portions of Steve Ball's SurfIt! web browser, while the GroupWeb internals are based on code from Stephen Uhler. Thanks also to the Tcl/Tk group at Sun Microsystems Labs. Carl Gutwin provided valuable feedback on both the system and on drafts of this paper. The financial support provided by Intel Corporation and NSERC is gratefully appreciated.

More information about TeamRooms, including software availability, related projects in groupware, publications, and mailing lists, is being made available via the World Wide Web at:

http://www.cpsc.ucalgary.ca/projects/grouplab/teamrooms/

REFERENCES

1. Curtis, P. and Nichols, D. MUDs Grow Up: Social Virtual Reality in the Real World. In *Proc. of the Third International Conference on Cyberspace*. 1993.

2. Dourish, P. and Bly, S. Portholes: Supporting Awareness in a Distributed Work Group. In *Proc. of ACM CHI.* 1992.

3. Greenberg, S., Hayne, S. and Rada, R. Groupware for Real-Time Drawing. McGraw-Hill. 1995.

4. Greenberg, S. and Roseman, M. GroupWeb: A Web Browser as Real-Time Groupware. In *ACM CHI '96 Conference Companion.* 1996.

5. Gutwin, C., Greenberg, S. and Roseman, M. Supporting awareness of others in groupware (short paper suite). In *ACM CHI '96 Conference Companion.* 1996.

6. Johansen, R., Sibbet, D., Benson, S., Martin, A., Mittman, R. & Saffo, P. *Leading Business Teams*. Addison-Wesley. 1991.

7. Lee, J., Prakash, A., Jaeger, T., and Wu G. Supporting multi-user, multi-applet workspaces in CBE. In *Proc. of ACM CSCW.* 1996.

8. Mantei, M., Baecker, R., Sellen, A., Buxton, W., Milligan, T. and Wellman, B. Experiences in the use of a media space. In *Proc. of ACM CHI.* 1991.

9. Orfali, R., Harkey, D. and Edwards, J. *The Essential Distributed Objects Survival Guide.* John Wiley and Sons. 1996.

10. Ousterhout, J. Tcl and the Tk Toolkit. Addison-Wesley. 1994.

11. Roseman, M. Managing Complexity in TeamRooms, a Tcl-Based Internet Groupware Application. *Proc. of the 1996 Tcl/Tk Workshop*.

12. Roseman, M. and Greenberg, S. Building Real Time Groupware with GroupKit, a Groupware Toolkit. *ACM TOCHI.* March, 1996.

13. Sohlenkamp, M. and Chwelos, G. Integrating Communication, Cooperation and Awareness: The DIVA Virtual Office Environment. In *Proc. of ACM CSCW.* 1994.

14. Tolone, W., Kaplan, S. and Fitzpatrick, G. Specifying Dynamic Support for Collaborative Work within wOrlds. In *Proc. of ACM COOCS.* 1995.