Gutwin, C., Greenberg, S., Blum, R., Dyck, J. (2005) Supporting Informal Collaboration in Shared-Workspace Groupware. Report hci-tr-2005-01, Dept. Computer Science, U. Saskatchewan, Saskatchewan, Canada

Supporting Informal Collaboration in Shared-Workspace Groupware

Carl Gutwin, Saul Greenberg, Roger Blum, and Jeff Dyck
Department of Computer Science, University of Saskatchewan
Department of Computer Science, University of Calgary
HCI-TR-2005-01

Abstract. Shared-workspace groupware has not become common in the workplace, despite many positive results from the research lab. One reason for this lack of success is that most shared workspace systems are designed around the idea of planned, formal collaboration sessions – yet much of the collaboration that occurs in a co-located work group is informal and opportunistic. To support informal collaboration, groupware must be designed and built differently. We introduce the idea of *communityware*, in which groupware is organized around groups of people working independently, rather than shared applications, documents, or virtual places. Communityware provides support for three things that are fundamental to informal collaboration: awareness of others and their individual work, lightweight means for initiating interactions, and the ability to move into closely-coupled collaboration when necessary. We demonstrate two prototypes that illustrate the ideas behind communityware, and argue that this way of organizing groupware supports informal collaboration better than other existing approaches.

Introduction

A number of studies and projects have shown that shared workspaces are valuable for distributed collaboration (e.g., Whittaker et al., 1993; Tang 1991). Shared workspaces allow people to carry out joint work on tasks, simplify communication about artifacts, coordinate activity through visual means, and maintain awareness of others. However, even though shared-workspace groupware has been technically feasible for many years, there are few examples where this kind of groupware has succeeded in the real world. This is not because there is no market for real-time groupware, since both instant messaging (IM) and online multi-player games have proven to be extremely popular. The lack of success for shared workspaces leads Whittaker (2003) to ask:

> Why aren't shared workspaces used more? Despite the ubiquity of document-centric interaction in face-to-face settings, and the demonstrated success of shared workspaces in laboratory settings […], these have yet to become pervasive. (p. 16).

There are a number of possible reasons for the low adoption rate of shared workspaces. One that we explore in this paper concerns the way that collaboration is organized by the design of the groupware system: most shared workspaces are designed to support planned, formal collaboration sessions, but it appears that much of the shared work that happens in co-located work groups is informal, unplanned, and opportunistic. Informal collaboration arises through everyday encounters with other people in a work group: for example, one person walks past another's desk, notices that they are working on a particular task, and stops momentarily to ask a question, point something out in the workspace, or help out with the activity. Informal interaction is common in most co-located work groups, and is usually much more frequent than planned group work.

If the majority of collaborative work over task artifacts occurs informally, it seems crucial for shared-workspace groupware to support this style of interaction. However, informal collaboration is poorly supported in current systems. There are five basic requirements for successful informal collaboration, and no shared-workspace groupware systems currently support them all:

- *Individual work*: people carry out their own individual tasks in their own work settings, until such time as an opportunity presents itself;
- *Awareness of the community*: informal collaboration is based on a rich awareness of other people in the group and their work environments;
- *Lightweight initiation*: since informal interactions are discretionary, they will only occur if the costs of engaging in collaboration are low;
- *Negotiation of engagement*: people need to be able to negotiate the possibility of engaging in an informal collaborative episode;
- *Interaction with task artifacts*: when work involves objects, people need to be able to refer to and manipulate these objects in their collaboration.

In this paper we present a new approach to the design of shared-workspace systems that supports informal and opportunistic collaboration in distributed groups. We call this new approach community-based groupware, or *communityware*. Communityware has several features that enable it to provide support for informal collaboration:

- An overview representation of the community that shows who is in the group, who is available, and what individual tasks they are working on;
- Lightweight communication to allow easy initiation of interaction and simple discussion of tasks and artifacts;
- The ability to zoom into another person's workspace and interact with the task data using full groupware capabilities.

Community-based groupware attempts to recreate some of the information and capabilities that are available when people work in a co-located setting. People can 'walk past' others' workspaces in the community overview and notice what they are doing, can easily ask a question or point to an artifact, and can move into closely-coupled collaboration without starting up any new applications.

Gutwin, C., Greenberg, S., Blum, R., and Dyck, J. (2005)
Supporting Informal Collaboration in Shared Workspace Groupware.
HCI Technical Report 2005-01, The Interactions Lab, University of Saskatchewan, Saskatoon, Saskatchewan, Canada, March.

Communityware takes lessons from several other types of groupware that already support informal interaction to some degree, including IM, online games, MUDs, and object repositories. Although many of the ideas already exist, they have not previously been brought together; therefore, the communityware approach implies a fundamental change in the way shared workspaces are designed and built. The focus is much more on supporting a group of individuals; the techniques for supporting synchronous interaction that are the focus of current groupware are still needed, but are held in reserve until an opportunity arises. Our initial investigations of the concept suggest that it has enough value to be adopted and explored further by groupware designers and CSCW researchers. In the next sections, we review previous research that underlies the idea of communityware, then introduce the concept in more detail, and demonstrate two prototypes that we have built to explore the communityware design space.

Informal Collaboration

Real-time collaboration occurs in many forms and in many ways. People can arrange meetings for planned work, they can be assigned to a shared task and work together on a regular basis, or they can simply interact informally when an opportunity for shared work presents itself. This last type is *informal collaboration*: unplanned and opportunistic interactions that occur in an impromptu fashion in the workplace. Informal communication and collaboration have been looked at in several prior workplace studies, showing its characteristics and its value to co-located collaborators (Kraut et al., 1993; Bellotti and Bly, 1996; Whittaker et al., 1994). This research paints a picture of workplace collaboration that is very different from the assumptions that seem to underlie many current groupware tools. Eight findings from this work are directly relevant to the problem of supporting opportunistic collaboration in distributed settings.

1. Informal interactions are common in co-located groups. Informal communication and collaboration patterns are evident in many kinds of 'knowledge worker' communities, such as research labs (Kraut et al., 1993; Kristoffersen and Ljungberg, 1999), software teams (Herbsleb and Grinter, 1999; Muller et al., 2004), design firms (Bellotti and Bly, 1996), and telecom companies (Nardi et al., 2000). The open-plan office is perhaps the canonical setting for informal interactions:

> Most of the office spaces at QED are open plan, fostering a relaxed atmosphere and permitting easy access between co-workers for conversation and the discussion and demonstration of design ideas. Informal and frequent interactions seem to be critical to the way the organization conducts its work as a whole. (Bellotti and Bly, p. 210).

2. Informal communication is brief and frequent. Kraut and colleagues (1993) found that in a research lab setting, 88% of conversations were not previously arranged, 50% were unplanned (in that neither person knew the conversation was

going to occur), and that 21% were opportunistic (in that they were triggered by the sight of a person in the environment). The more spontaneous a conversation was, the shorter it was likely to be: planned meetings were generally longer than 30 minutes, and informal conversations were generally less than 10 minutes. Finally, informal communication was more likely with those people who are physically nearby; 87% of the conversations occurred on the same floor of the building where the initiator worked.

3. Informal interaction is grounded in awareness of the work environment. It is clear that informal collaboration arises from awareness of the people, the objects, and the activities going on in the work environment – that is, "keeping up-to-date with things going on, both on and off one's own project" (Bellotti and Bly, 1996, p. 213). For example, Kraut and colleagues (1993) discuss 'social browsing':

> …people walk down the halls, peering into open offices and public spaces as they go to the printer, copy machine, bathroom, or other ultimate destination […] This process of browsing the social environment while on other business provides people with a substantial amount of information about the world in which they live.

This ambulatory model was also mentioned by several other researchers; and as Bellotti and Bly (1996) state, people also maintain awareness as an end in itself:

> We sometimes saw people wandering around just to see what was going on, apparently with no other motive. Gus called this doing a 'walkabout.' In fact, useful information seemed to be obtainable *passively*, just by coming into close proximity to others. Conversations could be overheard and people seen working together at PCs, or on design models, or showing each other documents. (p. 213).

4. Informal collaboration can be triggered by people, objects, actions, or interactions. When informal interactions are opportunistic, they can arise from seeing a particular person, seeing an object in a person's workspace, seeing someone perform some type of action, or seeing or hearing an already-started interaction between other group members. The most common trigger was seeing another person (Kraut et al., 1993), but the other triggers are just as important for informal collaboration. Being able to see the details of a person's work environment is valuable in that it leads to task-based interactions rather than discussions. This is illustrated by an example from Bellotti and Bly (1996):

> Harry is sitting in an open workspace putting parts together for the Turbo breadboard model. Another QED employee, Ted, enters the space and overtly bends over Harry's shoulder to stare closely at what he is doing. […] Harry turns from what he is doing to look at Ted.
>
> Ted, though not a team member, is clearly interested in this project and has even helped out in the past. Seeing that Harry is working in a public workspace, he approaches and makes a show of looking at what Harry is doing. Harry, who has the option of ignoring this play, instead willingly offers information on what he is doing and why he's doing it. (p. 214)

Similarly, Whittaker and colleagues (1994) found that documents were involved in more than half of the informal interactions that occurred near a person's office. They found that these documents were often important to the

conversation, acting as cues and conversational props for the participants, and that people maintained a joint visual focus on the documents.

Finally, it is also common for interactions themselves to draw others into the collaboration. As Bellotti and Bly (1996) state:

> Being within earshot afforded an entry into relevant or interesting discussions, or enabled people to learn things which they might not otherwise have done. Visual and auditory accessibility clearly provided the awareness which facilitated or prompted spontaneous communication. In this way people sharing office space learned a great deal about one another's ongoing activities and were more likely to interact informally as a result. (p. 214).

5. Informal collaboration is often discretionary. Informal collaborations are flexible in two ways: they do not have to happen now, and they usually do not have to happen at all. In many work settings, the people in the group are only loosely coupled to one another (Pinelle and Gutwin, 2003), meaning that they (or their tasks) do not have strong dependencies that require them to collaborate. Since loosely-coupled groups have a common goal, however, there are many occasions where collaboration is possible but not absolutely required. This means that people must decide whether the potential benefits of working together are worth the effort of initiating and joining into a collaborative session. If the costs are too high, loosely-coupled workers can manage the task on their own.

6. Informal interactions are easy to initiate. One of the hallmarks of informal collaborations is that they do not require extensive or cumbersome processes to initiate. As Kraut and colleagues (1993) state, the "confluence of topic and availability [are] the minimum preconditions for a conversation to occur" (p. 10). In a co-located environment, people's presence is obvious, their availability is relatively easy to determine, and the shared work environment suggests topics for collaboration. In these situations, a collaborative session can be started simply by walking up to someone, pointing at an object, and asking a question.

In fact, if any more effort than this is needed, people will often not bother to interact. This is one reason given for the much higher frequency of interaction with people who are physically nearby; as one participant stated, "It's a real pain to be in different places. […] There's a lot of inertia even in just having to pick up the phone" (Bellotti and Bly, 1996, p. 214). People's willingness to expend effort on the collaboration is fairly low: if effort is required to determine the other person's presence or availability, or if one must wait while the other person finishes other work or completes another conversation, the initiator will oftentimes abandon the interaction or postpone it until later.

7. Engaging in an informal interaction is a several-step process. Despite the simplicity of initiating informal collaboration, there is still a definite process to it: as Kristoffersen and Ljungberg (1999) state, "establishing interaction usually involves more interaction" (p. 83). The process has primarily to do with determining another person's availability, and there are a number of unspoken rules governing how people go about this. People use both verbal checks and visual information to judge another person's receptivity:

> First, the mere sight of Baker served as a stimulus that jogged Able's memory that he had something to say to him. Second, the 'hello' served as a channel checking routine, indicating that Baker's attention was free and that he was available for conversation. (Kraut et al., p. 10)
>
> The visual channel was used to verify the opportunity for conversation. That is, by looking at a potential target of conversation, the initiator can often interpret the target's locus of attention and infer whether and when he or she is available for conversation. (p. 12)

In addition, people try to make their intentions public to the potential target, without a heavyweight verbal exchange. People may stand at the periphery of another person's work area to signal their interest in an interaction, or may make a tentative conversational opening (e.g., 'Ted…?'). The target's response to these initial moves (e.g., whether they orient themselves to the other person, or take their conversational turn) often determines whether the interaction will proceed. This is illustrated in the QED example above, where Harry "*has the option of ignoring [Ted's] play*" (Bellotti and Bly, 1996, p. 214).

Informal Collaboration and Real-Time Groupware

Different forms of groupware imply, through their design assumptions, different kinds of collaborative interaction. Here we consider four different types of groupware, and how the different design approaches support informal collaboration. We first look at the organizing principles visible in shared-workspace systems, and then look at three other types of systems: instant messengers, object repositories, and collaborative virtual environments.

Shared-Workspace Systems and Session Management

Shared workspace systems allow multiple users to interact with task artifacts in real time from different locations. A main architectural question in the design of these systems is session management – that is, how collaborators are to be joined together in the shared collaborative session. Edwards (1994) divides session management techniques into two main groups: explicit and implicit. In explicit sessions, participants must intentionally connect a client to other clients or a server, either by accepting an invitation sent out by another person, or by joining a conference using a 'yellow-pages' interface (or by manually entering connection information such as the server address and port). Explicit session management is the oldest, and still most common means of getting a group into a collaborative session.

Implicit session management, by contrast, does not require that people carry out some action to join into collaboration. Instead, the system monitors people's environments and activities, and connects people automatically when it infers that they are already engaged in joint work. Three kinds of implicit session management have been considered, based on three different aspects of the work environment: artifacts, activities, and places. In *artifact-based* collaboration,

documents or shared objects 'know' when two people are both using the artifact, and so implicitly join them into a shared session. *Activity-based* session management is similar; when it is recognized that two people are working on the same task, an implicit connection is created between them. In *place-based* groupware, people log into a system that organizes tools and artifacts into persistent locations such as rooms or worlds.

These different styles provide different types of support for informal and opportunistic collaboration. Explicit session management provides perhaps the least support: there is no awareness of others, their activities, or their availability for contact before the session is begun, and people must carry out a series of heavyweight steps in order to get into the session.

Implicit session management is an improvement, in that the connections are made transparently (and thus effortlessly); however, it is still the system that decides whether a collaborative session should be initiated, rather than the participants themselves. Furthermore, it is impossible for a person to join into collaboration with another person based purely on the target's activities – implicit session managers assumes that their job is only to make connections between people who are already working on the same objects or activities.

Place-based groupware is worth additional mention here, because it has certain features that are valuable for supporting informal collaboration: places persist over time, and so could represent people over time as well; and places do not constrain the type of activity that participants undertake. TeamRooms (Roseman and Greenberg 1996) is a good example of a place-based system that supports some elements of informal collaboration: it allows people to work on individual tasks, simplifies connection and communication tasks, and provides full groupware capabilities in its tools. However, the design of TeamRooms is still strongly oriented towards a temporal session model (for example, when a person logged out of the TeamRooms system, they disappeared from view entirely). Similarly, there is not extensive support for awareness of others, particularly when they are in other rooms.

In summary, none of these different approaches to session management completely supports the type of informal and opportunistic collaboration that happens in work rooms and organizations. The main reasons are that either it is too difficult to determine who is around and what they are doing (so to be able to perceive the things that trigger the opportunity), or it is too hard to initiate a session (so to act on that trigger). As Bellotti and Bly (1996) state,

> Distributed collaborators cannot do anything like a 'walkabout' to survey current work at a remote site […] Furthermore, their technology only affords explicit communication rather than the kinds of implicit communication available through co-presence and mutual awareness […] There is no casual or lightweight entrypoint into discussion about current work; nothing to prompt the passing remark or enquiry. (p. 214)

Awareness Servers and Instant Messaging (IM)

Awareness systems provide information about people's presence, activities, and availability. They can be based on a variety of environmental information, but the most common information sources have been video, audio, and on-line status. Systems for providing persistent awareness of others have existed for many years: for example, Cruiser (Root, 1988) and Portholes (Dourish and Bly, 1992) were early video-based systems, Telefreek (Cockburn and Greenberg, 1993) showed login status, and the Active Badge Locator (Want et al., 1992) showed location in a building. More recent systems provide several different types of information at once, with the goal of allowing observers to better determine a person's true availability (rather than just their presence) (Begole et al., 2004).

Instant messaging systems are the most successful version of awareness servers, and possibly the most successful real-time groupware application ever. IM systems have several features that support aspects of informal interaction:

- They are organized around communities of people: IM systems organize people into buddy lists, rather than adopting an object or place metaphor; people remain members of the community regardless of their login status.
- They provide persistent awareness of that community: people who are offline are still represented, but shown as unavailable.
- They represent availability: IM systems have a number of different representations for people that help others determine their availability for contact; some of these are automatically assigned (e.g., idle time, login status) and others are explicitly set (e.g., busy, do not disturb).
- They provide an easy transition between awareness and communication: it is easy to send a message to a person that appears to be available.
- They allow negotiation of initiation: text chat allows people to discuss availability in a lightweight fashion. As stated by Nardi et al., (2000):

> Many IM conversations […] took the form of preambles where initiators attempted to determine the preparedness of recipients for IM interaction. Often people would send simple instant messages like, 'Suzi?' to see if someone was available for an IM exchange. If the recipient responded, an 'attentional contract' was established in which both parties explicitly agreed that the communication could proceed. (p. 83).

IM systems clearly do already support informal interaction (Nardi et al., 2000). Their awareness displays provide triggers for some types of opportunistic and informal interactions; however, IM is completely disconnected from work activities, and so cannot support informal collaboration about tasks.

Shared Object Repositories

Shared object repositories are on-line containers for artifacts that are used by a group of people. They provide both storage and a visual representation, so that each group member can see the objects and obtain them for local work. The

repositories can contain a variety of content: for example, messages, pictures, and news clippings (e.g., Greenberg and Rounding 2001), files and folders (e.g., Muller et al., 2004), or source code (e.g., CVS, www.cvs.org).

ActivityExplorer (Muller et al., 2004) shows how an object repository can support informal collaboration. The system provides IM-style communication and persistent awareness of others, shows who is editing each object in the repository, provides an alerting service to indicate when particular objects are changed, and allows information about activities to be overlaid on the repository, providing an implicit grouping of objects and people into tasks. Although there are no real-time collaboration facilities for the shared objects, people can send screen snapshots of their current work to a collaborator, and draw overtop of them as in a shared whiteboard. ActivityExplorer provides at least some support for each of the informal collaboration requirements mentioned above. However, information in this system is at a fairly coarse granularity: there is no way to get detailed information about a person's activities, and no way to join them in closely-coupled work over the shared objects.

Collaborative Virtual Environments (CVEs)

Collaborative virtual environments provide a virtual world presented in either graphics or text, in which multiple people can interact and carry out tasks. People are represented as avatars, and can move around and explore the space. CVEs have been explored extensively by CSCW researchers (e.g., Benford et al, 1995), but the most successful CVEs are built for on-line games: some are transient (e.g., those used for most first-person shooter games), and others persist indefinitely (e.g., the worlds for MUDs and games like EverQuest). The transient worlds generally use explicit session management (that is, players explicitly join a game from a list of servers), but the persistent worlds have several characteristics that are interesting from the perspective of supporting informal collaboration.

- *Persistent avatars*. When people log out of games like EverQuest, their in-game representation (i.e., their avatar) does not disappear. Instead, their avatar goes to 'sleep,' providing others with awareness information that the player is around, but not currently available.
- *Rich visual identity.* Considerable information can be gathered about a person from their avatar's visual appearance: for example, an EverQuest player's character type, clan, and possessions are all represented on the avatar's body, and their player name and status is shown as a floating label above the character's head.
- *Simple transitions between awareness, communication, and work*. The activities needed to move into collaboration are all tightly integrated with the game's real-world metaphor, and thus it is easy to move between awareness-gathering and communication, and between communication and shared activity in the game. For example, finding someone to join you in a task

involves walking up to them in the game, typing text messages, and if they agree to join you, simply starting the task.

Communityware

Communityware is a new way of designing and building groupware, with the goal of better supporting informal collaboration. It organizes groupware around the idea of a community of individuals, and there is no requirement that anyone works together until an opportunity is seen to do so. The goals of communityware are to enable both impromptu and opportunistic collaboration: things like asking someone to come over and look at your workspace for a quick consultation, or starting a discussion after noticing another person's work during a walkabout.

Community-based groupware has three fundamental concepts that provide support for informal collaboration: a representation of community that provides ongoing awareness of people's presence, their availability, and their work activities; mechanisms for lightweight initiation of interactions; and the ability to move into more closely-coupled interaction over work artifacts when necessary. We next consider the requirements for communityware in each of these areas.

Representation of the community

As described above, there are four types of information that could trigger informal interactions: information about the people in the community, about their individual workspaces and artifacts, about their activities and actions in their workspace, and about their interactions with each other. Below we consider what is needed to support these kinds of triggers.

- *Information about people*. Simply seeing another person can remind the observer that they want to interact. To support this kind of trigger, a communityware system must allow people to determine who is in the community (membership), who those people are (identity), who is around now (presence), and who is available for possible interaction (availability).
- *Information about workspaces and artifacts*. Noticing a particular artifact, document, or tool in another person's workspace can lead to an interaction; therefore, the system should provide a representation of people's individual workspaces. The representation could be literal or abstract, but to enable gradual engagement (discussed below), these representations should be reduced in size or fidelity.
- *Information about activities*. Noticing what actions a person carries out in their workspace can also trigger interactions. Communityware should represent two types of information: changes to artifacts in a person's workspace, and major actions by the person themselves (such as moving across the room to look at another person's workspace).

- *Information about interactions*. Ongoing interaction between others in the community can lead to the observer's participation in that collaboration. Two mechanisms that can be supported in communityware are showing that people are interacting (e.g., through proximity of avatars), and making discussions available to be overheard.

How should this awareness information be presented? There are many possible ways, and different solutions may work well for different communities. It is possible that the community representation could be textual or graphical, literal or symbolic, persistent on screen or transient, and oriented more towards browsing or searching. The examples we describe below, however, are both based on a graphical, literal, persistent, and browseable representation that we call a community overview. We believe that this representation provides a good balance in the various design requirements for communityware; however, different approaches may be more appropriate depending on the community.

Lightweight Initiation of Interaction

Establishing a collaborative interaction involves several issues for both the initiator and the target. The initiator must decide that it is worth making an opening, must be able to assess the other person's availability, and must be able to make their interests known. The target must be able to notice the initiator's interests, and must be able to act out their next move, either moving the interaction forward, delaying it, or preventing it entirely. In general, communityware must support gradual engagement, where people can negotiate the entry into a collaborative interaction.

- *Investigating triggers*. When initiators see something that could lead to collaboration, they need to be able to get more information about that trigger. Therefore, communityware should allow people to get more detail about workspaces and activities: for example, by increasing the size or the fidelity of the other person's workspace representation in an overview.
- *Noticing interest.* An initiator's interest (such as enlarging a workspace representation) should be noticeable to the target; this will give them the first indication that another person is interested in their work. One problem for overview-based communityware is that people are likely to be looking at their own work rather than at the community overview; therefore, extra work may have to be done to make people's interest noticeable (e.g., by integrating visual indicators into the individual applications as well as the community browser, or projecting the community representation on the wall behind the user's screen).
- *Determining availability*. Initiators need to be able to determine the availability of someone who they wish to interact with. Three important aspects suggested by prior work are indications of presence (whether the person is logged in and at their computer), a way for people to explicitly

indicate unavailability (e.g., through a 'do not disturb' flag), and an indication that people are already in conversation with someone else (although as suggested, this can also be an opening to join the interaction).

- *Making the opening*. Communityware should definitely support verbal openings such as the text-based 'preambles' observed in instant messaging (Nardi et al., 2000). In addition, a community overview could show openings through avatar position: a person's avatar could move to the periphery of another's workspace as an implicit request for attention.
- *Lightweight communication*. IM research suggests that text messages can be lightweight enough to support informal communication. Communication should be integrated as much as possible with other visual aspects of the community overview; for example, using transparent fading text as done in many first-person shooter games.

Moving to Closely-Coupled Interaction over Task Artifacts

Once people have established an interaction, the final phase of informal collaboration involves actually getting down to work. In some cases, this will involve collaboration in full shared-workspace systems; however, full groupware will not be required in all instances. Many informal interactions will finish successfully in a short time: for example, a brief question will be answered, a concern will prove to have an easy explanation, or a comment will be accepted without need for further discussion. These examples suggest that there should be a range of interaction techniques available to collaborators, from lightweight techniques for brief interactions, to full groupware when people decide to work in a more closely coupled fashion. This leads us to three principles:

- *Support pointing first*. Many small interactions do not require all of the power of a full-scale groupware system. The most important interaction technique that occurs in short collaboration is the ability to point (Whittaker et al., 1993), and so this capability should be supported even in the workspace overview.
- *Gradual increase in power for the observer*. As the observer's representation of the target's workspace becomes larger or more detailed, additional abilities to manipulate the artifacts should be added. Non-destructive operations should be made available first.
- *Lightweight transition to full groupware*. When the collaborators decide that the observer should become a full participant, it should be simple to move from the workspace representation in the community overview to the full groupware system. No additional connection or session management information should be required, and there should not be a long delay in starting the application or receiving the data. This may require that data structures from each person's workspace must be cached on each machine, in preparation for a possible collaboration.

In the next section we illustrate the idea of communityware with two examples: one based on the idea of collaboration transparency, and one using groupware that is fully collaboration-aware. These examples are not intended to demonstrate complete communityware systems, but are presented to show how the concept can be implemented in two fairly different ways.

Example 1: Multi-VNC

Our first prototype of a communityware system used UltraVNC, an open-source screen-sharing application (ultravnc.sourceforge.net). Each person in a group of six people in our lab ran a VNC server and five separate instances of the client, with remote screens scaled to 1/8 size (see Figure 1). This arrangement of windows was put on a second monitor on each person's computer; the person's normal work was carried out on the primary monitor (and this was the screen that was distributed to the group). This prototype did not require any additional coding besides what was already available in the software; only the organization of the system was novel, since VNC is normally used for one-to-one sharing rather than for overviews of several collaborators' screens. Verbal communication was through VNC's own point-to-point messaging system.

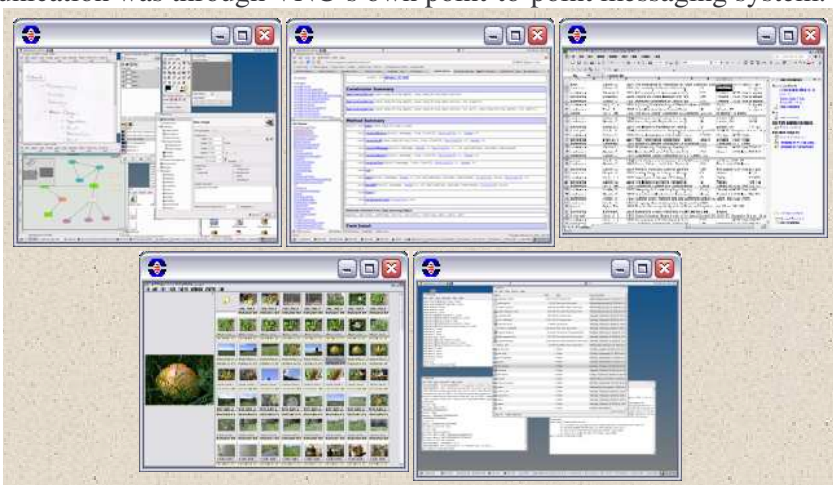

Figure 1. Multi-VNC, showing five collaborator's screens scaled to fit onto a secondary screen. The user's main screen is not shown.

Even this simple setup showed that some of the communityware principles are valuable. Multi-VNC allows group members to see each others' work in general (what applications are running, but not data in those applications), allows people to send text messages to one another, allows pointing from the overview to the original screen, and allows people to zoom in to a full share with the other person's system. There were two clear disadvantages, however. First, all the connections, scaling, and zooming had to be done manually, with commands issued through the VNC control dialogs. This was difficult and time-consuming; it is clear that this is not a viable alternative for a real work community without a

wrapper application that simplifies connections and provides shortcuts for common functions like zooming in to another person's workspace. Second, Multi-VNC provides no representation of people. This makes it difficult to remember which workspace belonged to whom, and difficult to determine who was active or away from their computer. Furthermore, it was impossible to see who was looking at your screen, and whether other people in the group were involved in collaboration. Again, however, it is possible that a wrapper application could provide embodiments that would convey this information.

Example 2: Walkabout

Walkabout is a communityware system for collaboration-aware groupware; the name comes from the information-gathering tours observed in co-located groups. Walkabout provides a more complete implementation of the requirements discussed above for supporting opportunistic collaboration, but it is limited in that it only supports groupware systems that conform to its protocols. Below we describe how Walkabout implements support for the three main communityware requirements, and then briefly outline its design and implementation.

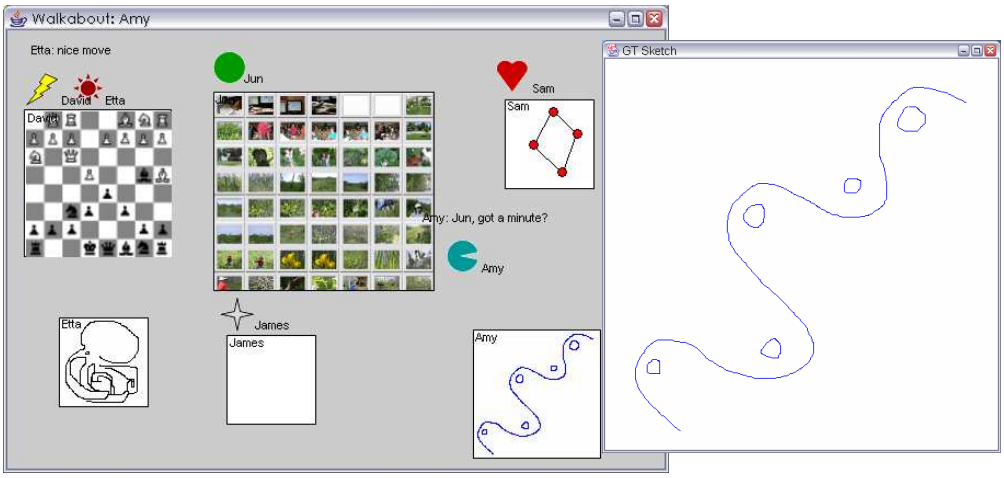

Figure 2. Walkabout community overview (Amy's view). In this scene, Etta has joined David's chess game, James is off-line (his avatar is grey), and Amy has just enlarged Jun's workspace representation and has started a conversation.

Support for Awareness

Walkabout's basic metaphor is an open 2D space in which each member of the community and their workspace is represented. People are represented with customizable avatars, and workspaces with miniatures. The space is organized by the participants: people can move their own workspace as they like, but avatars move to indicate interest and interaction with others. Workspace miniatures are updated regularly by the user programs; avatars show information about the

person and about their interactions with others. Table 1 summarizes the ways that Walkabout supports awareness information.

Support for Initiation of Interaction

Walkabout implements basic features for many of the requirements discussed above. As shown in Table 2, Walkabout allows users to investigate the contents of others' workspaces, provides lightweight text messaging, and uses avatar location and appearance to show availability and interest in a person's work.

Support for Closely-Coupled Collaboration

Walkabout supports pointing from the overview into the client application; this requires that the client subscribe to a communication channel as described below. Participants can move to full groupware with a simple double-click; at this point a groupware client is created in a new window, and the participants can continue as if they had started a shared-workspace session through traditional means.

Information about people	
Membership	Each member of the community has a persistent avatar that is shown regardless of connection status.
Identity	Name is shown beneath the avatar, and avatars can be customized (colour, shape, photograph, various decorations).
Presence	On-line and idle information are shown in the avatar with transparency.
Availability	Users can add a "do not disturb' sign to their avatar; when people are in conversation, their avatars and text messages move close together.
Information about workspaces and artifacts	
Workspace	A miniature version of each person's current workspace is drawn in the community browser.
Artifacts	All artifacts are duplicated in the workspace miniature
Information about activities	
Actions	Workspace artifacts show changes; avatars indicate major actions through position and movement.
Information about interactions	
Proximity	The avatars of people who are working together are shown side-by-side.
Overhearing	All conversations are shown in the community browser, near the participants' avatars.

Table 1. Summary of Walkabout support for maintaining awareness

Investigating triggers	People can enlarge a workspace miniature by clicking on it; when enlarged, the miniature is updated more frequently
Noticing others' interest	When an initiator enlarges a target's workspace, the initiator's avatar moves towards the target workspace
Determining availability	Avatars show online status, idle time, 'do not disturb' flags, and show when people are in conversation
Making an opening	Initiator's avatar moves to periphery of workspace; when initiator types, the message appears in the overview
Lightweight communication	Initiator points to target avatar or workspace and begins typing; respondent moves to overview and begins typing

Table 2. Summary of Walkabout support for initiation of interactions

Pointing	Pointing into miniature moves a telepointer in the client application (if client subscribes to this channel)
Moving to full groupware	Double-clicking the miniature spawns a full client joined to the target's application

Table 3. Summary of Walkabout support for closer interaction

Implementation and Architecture of Walkabout

Walkabout is itself a distributed groupware system independent of any of the application programs represented in the overview; however, it communicates with these programs and sends them information and events based on people's actions inside the community representation. Walkabout is written in Java using the GT toolkit (web address); this toolkit provides abstractions for text and object communication that simplify the design of Walkabout. Groupware applications that appear in Walkabout must also be built using the toolkit, and must conform to a small set of Walkabout protocols. There are several parts to the Walkabout system, which we describe here in brief:

- When a user program starts, it connect to the Walkabout system at a well-known address (this configuration information would be set up once and then stored in a settings file). A Walkabout client showing the community browser is also started on the user's machine.
- User programs provide information about the person (currently keyed by name) so that the Walkabout server can match the person's avatar to their workspace in the community overview.
- User programs send information about their workspaces to the Walkabout server, either as a thumbnail image, or as a Java object that knows how to draw itself in a scaled fashion. The Walkabout server distributes these to each community overview client for display.
- Actions in the community overview (e.g., avatar movement and text chat) are distributed to all overview clients for display.
- Walkabout publishes the text-chat messages and overview telepointer locations on two specific *channels* (a messaging abstraction available in the toolkit); user programs can subscribe to these channels to implement visualizations for the 'preamble' text messages discussed above and overview-to-application telepointers.
- We assume a replicated-groupware model, so all user programs are already resident on each person's machine; therefore, when a user zooms fully into a workspace overview, the correct groupware application can be invoked.

As mentioned above, Walkabout is still an early prototype; there are still a number of interface issues and architectural approaches to be investigated. Nevertheless, even in our limited experience with the system, it is clear that communityware makes it much easier to see what people in the group are doing, and makes it easy to start conversations with them. The community overview provides a strongly different experience for users and groups; rather than thinking about how to start up a session, it is easy to just join in to an application that you see running in the overview. Similarly, it is easy to see where there is interest from other members of the group, and to stay aware of others' interaction – something that is very difficult to do in other groupware systems.

Discussion

In this section we discuss issues raised by the communityware approach. We consider potential risks to privacy, and then discuss whether the approach can be more successful than shared-workspace systems have been in the past.

Privacy Concerns

There are two potential privacy issues with the communityware approach: first, that people will not want to make their individual workspaces available to others in the group (control over confidentiality); and second, that doing so will increase people's ability to intrude (control over solitude) (Boyle and Greenberg, 2005). These concerns were seen in our initial experiences with Multi-VNC.

There are a variety of techniques for protecting both forms of privacy: for example, workspace overviews could show only low-resolution representations, or could provide a few seconds' warning to the target before the observer was allowed to see the workspace. Similarly, controls could be placed on people's abilities to interrupt one another. However, these features do not always work, and might add so much effort that they reduce the informal collaboration that is the goal of the system in the first place.

It would be best if the system gave people enough information to deal with privacy issues using social protocols. People manage to do this in real-world co-located groups: that is, people's workspace are visually available in an open-plan office, and others are able to intrude upon one's solitude with informal interactions – and yet, this is not seen as a major privacy problem. The key is that in a co-located setting, there are clear physical and social constraints that help people protect information (e.g., by making sure nobody can see it) and prevent interruptions (e.g., by moving to an office with a door). It is possible that in a communityware environment, we will be unable to provide the degree of richness and subtlety that allows people to manage these issues in the real world. However, communityware is based on the idea of supporting some of the subtle interaction and awareness that happens in the real world, and so it seems at least possible that the right mechanisms and information sources can be found and represented. Part of our future work will be to investigate whether 'open-plan privacy' can in fact be managed in a distributed system such as Walkabout.

Can Communityware Succeed in the Real World?

The most important question about the communityware approach is whether this new way of thinking about groupware has any more chance of success than the shared-workspace systems that preceded it. There are arguments both for and against, but the one compelling argument in favour of communityware, as explicated throughout this paper, is that it better matches the way that people

really collaborate than does traditional shared-workspace groupware. This advantage is critical: a system that reflects real organization of work has at least a chance to compete, whereas systems based on false assumptions likely have none.

Whether communityware can change the groupware world is yet to be seen, however, because on the negative side, there are several potential problems for the approach. First, a variety of criticisms are possible about the usability and design of the prototypes: for example, the community overview may be too cluttered or too distracting, the privacy problems mentioned above might not be solvable, the system could drain too much screen space or computing power, or the interaction techniques in the overview could be too unwieldy for people to use. All of these problems are potentially real; however, they are not likely to be critical for all groups, and they can all (probably) be solved with the application of usability engineering and the various forms of Moore's Law. They are important problems but not the most critical ones – and there are two serious potential problems that should be considered first, problems that have plagued shared workspace groupware throughout its short history:

The "it's not Microsoft Word" problem. New groupware systems or collaboration infrastructures may solve major problems in distributed collaboration, but if they require applications to conform to a specific protocol or standard, they are likely doomed to failure. Individuals want to use their existing commercial tools, and these are not going to conform to others' standards.

This problem is serious, but there are responses to it. First the communityware approach also allows for collaboration-transparent solutions, as shown above with the Multi-VNC prototype. This kind of solution would allow people to use whatever applications they wished; although collaboration transparency is not as rich as collaboration-aware systems, it is possible that it is good enough for most informal collaboration. Second, there are a variety of niche markets where collaborators in a community all use a single system (e.g., developers might all use the same IDE such as IBM's Eclipse); in these settings, a communityware approach could be taken within a single application. Third, and in the longer term, the trend towards supporting distributed and networked systems at the infrastructure level is increasing, and it is not too hopeful to imagine that soon, all applications can be built as collaboration-aware groupware as a matter of course. The particular groupware standard that will win out in this environment is yet to be determined, but our limited examples of communityware suggest that the approach could be valuable regardless of the infrastructure used.

Is low-effort low enough? Many groupware systems are introduced and used for a time, but few manage to sustain interest over a longer term. In situations such as those described above, where people are autonomous and where collaboration is discretionary, there is a real question about whether a communityware system can provide enough value to keep people starting the system up day after day. Even though communityware is based on the idea of

reducing initiation effort to a minimum, it is possible that even the small effort needed to use it will present too much of a barrier.

Our response to this problem is that the effort that a group is willing to put into collaboration is always going to be variable, and that for some groups, any amount of effort will be too much. However, there should be many groups for whom some effort is acceptable, and it is possible that communityware will gain a foothold with these communities. Instant messaging presents a good example of how this adoption can work: IM is not effortless, but for many communities, it has provided enough value that it continues to be used.

Future Work

We plan to carry out future research in three directions. First, we plan to study in more detail the ways that co-located work groups engage in informal collaboration. We hope to provide more details on issues such as how often task artifacts are the trigger for (and are involved in) informal collaborative sessions; how large a fraction of the total collaboration that occurs in a work group begins informally; and what degree of synchrony really occurs in informal collaborative interaction. One of the aims of these studies will be to try and determine the limits to a collaboration transparency solution like Multi-VNC. Second, we plan to extend our two prototypes: we will add several features to Multi-VNC to allow representations of people, text communication, and easier setup for a dedicated group; we will also enhance the Walkabout prototype to allow greater individualization of avatars, improved representations of workspaces, and more subtle representations of text messages. Third, we plan to deploy one of our prototypes in a distributed group for a longer-term evaluation of the concept.

Conclusions

Shared-workspace groupware has been much less successful than other types of real-time groupware, particularly on-line games and instant message systems. We propose that one reason for this lack of success is that current shared workspaces are not designed to support unplanned and informal collaboration, a type of interaction that is ubiquitous in co-located work settings. Previous work suggests several requirements for informal interaction: awareness of others, their work, and their availability; the ability to negotiate the initiation of an interaction; the ability to interact in a lightweight fashion; and the ability to move into more focused work when needed. We presented a new approach to the design of groupware called communityware, which supports these requirements. We demonstrated two examples built to illustrate the idea of communityware, one for collaboration transparency and one for collaboration-aware systems. Although much more remains to be done, it is clear that communityware provides a level of

support for informal collaboration that does not exist in any other groupware system. We believe that the approach will eventually help the adoption of shared-workspace systems, because it uses a more realistic model of collaboration, and better supports the ways that people work in the real world.

References

Begole, J., Rosson, M., and Shaffer, C., Flexible Collaboration Transparency: Supporting Worker Independence in Replicated Application-Sharing Systems, *ACM ToCHI*, 1999, 6,2, 95-132.

Begole, J., Matsakis, N., and Tang, J., Lilsys: Inferring Unavailability Using Sensors, *Proc. ACM CSCW 2004*, 511 – 514.

Benford, S., Bowers, J., Fahlen, L., Greenhalgh, C., Snowdon, D., User Embodiment in Collaborative Virtual Environments, *Proc. ACM CHI 1995*, 242-249.

Bellotti, V., and Bly, S., Walking Away from the Desktop Computer: Distributed Collaboration and Mobility in a Product Design Team, *Proc. ACM CSCW 1996*, 209-218.

Boyle, M., and Greenberg, S., The Language of Privacy: Learning from Video Media Space Analysis and Design, in press, *ACM ToCHI*, 2005.

Cockburn, A., and Greenberg, S., Making Contact: Getting the Group Communicating with Groupware, *Proc. ACM COCS 1993*, 31-41.

Dourish, P., and Bly, S., Portholes: Supporting Awareness in a Distributed Work Group, *Proc. ACM CHI 1992*, 541-547.

Edwards, K., Session Management for Collaborative Applications, *ACM CSCW 1994*, 323-330.

Erickson, T., Smith, D., Kellogg, W., Laff, M., Richards, J., and Bradner, E., Socially Translucent Systems: Social Proxies, Persistent Conversation, and the Design of "Babble", *Proc. ACM CHI 1999*, 72-79.

Greenberg, S., and Rounding, M., The Notification Collage: Posting Information to Public and Personal Displays, *Proc. ACM CHI 2001*, 514-521.

Herbsleb, J., and Grinter, R., Splitting the Organization and Integrating the Code: Conway's Law Revisited, *Proc. IEEE ICSE 1999*, 85-95.

Kraut, R., Fish, R., Root, R., and Chalfonte, B., Informal Communication in Organizations: Form, Function, and Technology, in R. Baecker, *Readings in Groupware and Computer Supported Cooperative Work*, Morgan Kaufmann, 1993.

Kristoffersen, S., and Ljungberg, F., An Empirical Study of How People Establish Interaction: Implications for CSCW Session Management Models, *Proc. ACM CHI 1999*, 1-8.

Muller, M., Geyer, W., Brownholtz, B., Wilcox, E., and Millen, D., One-Hundred Days in an Activity-Centric Collaboration Environment, *Proc. ACM CHI 2004*, 375-382.

Nardi, B., Whittaker, S., and Bradner, E., Interaction and Outeraction: Instant Messaging in Action, *Proc. ACM CSCW 2000*, 79-88.

Root, R., Design of a Multi-Media Vehicle for Social Browsing, *Proc. ACM CSCW 1988*, 25-38.

Roseman, M., and Greenberg, S., TeamRooms: Network Places for Collaboration, *Proc. ACM CSCW 1996*, 325-333.

Tang, J., Findings from Observational Studies of Collaborative Work, *IJMMS*, 1991, 34, 2, 143-160.

Want, R., Hopper, A., Falcao, V., and Gibbons, J., The Active Badge Location System, *ACM ToIS*, 1992, 10, 1, 91-102.

Whittaker, S., Frohlich, D., and Daly-Jones, O., Informal Workplace Communication: What is It Like and How Might We Support It?, *Proc. ACM CHI 1994*, 131-137.

Whittaker, S., Things to Talk About When Talking About Things, *JHCI*, 2003, 18, 1/2, 149-170.

Whittaker, S., Geelhoed, E., and Robinson, E., Shared Workspaces: How Do They Work and When Are They Useful?, *IJMMS*, 1993, 39, 5, 813-842.