Designing and Examining PC to Palm Collaboration

Ivan Marsic, Allan Meng Krebs, Bogdan Dorohonceanu, and Marilyn Tremaine
Rutgers University, USA
{marsic, krebs, dbogdan}@caip.rutgers.edu, tremaine@acm.org

Abstract. One trend in day-to-day computing involves moving seamlessly from large powerful workstations to small handheld devices. A second trend is continuous collaboration with colleagues. Combining these trends requires solutions to both the problem of transferring large complex displays to smaller less capable devices and of ensuring that a viable collaboration takes place even when the collaborators are using vastly different tools and viewing screen environments that differ significantly in display richness. We briefly describe an architecture for managing displays across multiple platforms, which we call the Manifold framework. This architecture is incorporated into applications using our DISCIPLE collaboration system. We explore the use of Manifold by creating a 3-dimensional layout task that communicates with a 2-dimensional version of this task running on a Palm Pilot that is wirelessly connected to the Internet. In order to get measurable data on the collaboration problems and successes that users might encounter in this diverse communication tool arrangement, we ran two separate studies that captured performance time, user errors and transcripts of the communication exchanges between the two users. We found that interface problems with each environment affected task performance and that the different capabilities between the 3-D and 2-D environments created collaborative advantages rather than negatively affecting the collaboration.

Introduction

A key component for synchronous collaboration is real-time sharing and manipulation of information. Most collaborative applications provide synchronization
support, the so-called WYSIWIS (What You See Is What I See) technique (Stefik et al. 1987). WYSIWIS allows one user to point or discuss an object on the screen that they are confident is visible to their collaborator. Providing support for WYSIWIS with past systems was a difficult but not impossible task because the groupware used common platforms. Developing groupware applications that are interoperable across diverse environments is significantly more difficult and costly. However, with recent anytime-anywhere proliferation of computing technology, support for heterogeneity is inevitable. Mobile applications involving synchronous collaboration are emerging in many fields, e.g., business, healthcare, military services and transportation. The classic example is the provision of just-in-time assistance between a deskbound expert and a mobile fieldworker using a portable device. The mobile worker may work with blueprints while the expert is using a 3D CAD model to repair a vehicle or the plumbing in a building. Kraut et al. (1996), for example, show that fieldworkers make quicker and more accurate repairs when a remote expert is providing assistance.

The heterogeneity of computing platforms manifests itself in processor speed, memory, input devices, display capabilities, response time delays, continuity of message transmission and network bandwidth, with the last four accounting for the most prominent differences. This paper focuses on display differences since current developments indicate that they are likely to be the most variable of those cited. In addition, displays are often designed to support the mobility of users, e.g., displays mounted in eyeglasses\(^1\), OptiScape technology\(^2\), windshield-projected displays in cars, or foldable displays\(^3\). These displays are invariably more limited in size and quality than those on the desktop. Enlarging the screen real estate for mobile computers remains impractical because of the inconvenience of weight and size. Weight may also limit the compute power and bandwidth needed to display the graphics and animation that occurs in a desktop environment. Moreover, there is also a human information processing limitation that constrains the mobile user. A person in a private office can allocate vastly different cognitive resources to a display than a person in the field who needs to pay attention to the external environment, e.g., traffic (Kahneman 1973).

All the above factors result in the mobile user and the stationary user using different applications and working with semantically equivalent but different information. The problem at the system level is with synchronization of heterogeneous application states while at the user level it is enabling effective collaboration between users who have different views of the task.

\(^1\)http://www.microopticalcorp.com/
\(^2\)http://www.inviso.com/
\(^3\)http://www.ices.cmu.edu/design/
In this paper, we present a framework for representing and mapping information from one platform to a second one with different display and computing capabilities. However, this solution immediately creates two environments that will never achieve WYSIWIS status. Since the beginning of groupware software development, this WYSIWIS has been viewed as an essential capability in groupware systems. What effects will the differences between platforms have on collaboration when they persist throughout the collaboration? Certainly, shared information via computer displays is more than users worked with in prior non-co-located collaborative settings, e.g., when an employee working in the field telephones the main office for information, but the expectations of the interaction are likely to be different. The person with the Personal Data Assistant may expect more detailed help, and the person with the powerful workstation may expect faster response and quicker problem solutions.

We have developed our framework on top of our DISCIPLE groupware system (Wang et al. 1999) that supports the development of applications on multiple different platforms. The approach we take is data-centric in that the shared information is defined in a generic architecture that is inherited by display specific software on each of the platforms.

We demonstrate this framework by running a 3D application on an office workstation and its associated representation in 2D on the limited mobile platform, in this case a Palm Pilot. We then examine the effectiveness of the collaboration that takes place between colleagues in these decidedly different representations, first in an environment where the latencies and dropped message packets are controlled for and then in a real life environment where communication is between a mobile laptop and a Palm.

The paper is organized as follows. We first review related work in this area. Next, we overview the architecture of the DISCIPLE system and the Manifold framework. We describe several collaborative applications we built for heterogeneous collaboration using DISCIPLE and Manifold. Then we present the two studies that compare heterogeneous environment collaborations to homogeneous environment collaborations. Finally, we discuss our results and conclude the paper.

Background and Related Work

The need to allow conferees to collaborate on dissimilar terminals was recognized early on by D. Engelbart, the pioneer of computer-supported collaborative work (Engelbart 1984). Many collaborative systems and toolkits address these same issues. For example, Garfinkel et al. (1994) discuss the adaptation of the shared display (in
particular, color maps) to various display devices. If necessary, their system, SharedX, degrades the quality of the displayed images to match the capabilities of the display hardware. However, the authors assume that users have the same replicated application and are just applying device-specific rendering.

An early design for heterogeneous groupware is presented in (Karsenty et al. 1993), but it does not deal with platforms of significantly different computing and communication capabilities and it does not employ a model-view separation. Rendezvous (Hill et al. 1994), GroupKit (Roseman and Greenberg 1996) and several groupware toolkits thereafter use model-view separation so that developers can create distributed models and drive different views. However, no such implementation is reported, and in some cases (e.g., Hill et al. 1994) the situation is greatly simplified by using centralized groupware architecture.

In addition to developing an architecture for heterogeneous collaborative applications, we examine team performance characteristics for such applications. Although the WYSIWIS idealization recognizes that efficient reference to common objects depends on a common view of the work at hand, strict WYSIWIS was found to be too limiting and relaxed versions were proposed to accommodate personalized screen layouts (Stefik et al. 1987). In subsequent work (Tatar et al. 1991), problems were reported with non-WYSIWIS systems because manipulation and editing processes were private and only results were shared. This discontinuity of the interaction created ephemeral environment differences that affected collaboration.

Our essential principle for heterogeneous collaboration is that every user’s action is interactively and continuously reflected in other users’ workspaces, with a varying degree of accuracy or realism or through a qualitatively different visualization. We use this continuous update of user actions in both environments to maintain our shared reference between users although it is not known how users will interpret and verbally communicate this shared reference. Non-WYSIWIS is quite common in collaborative virtual environments (CVEs) (Hindmarsh et al. 1998, Steed et al. 1999) where collaborators navigate independently to accomplish their own goals. A recent study (Hindmarsh et al. 1998) suggests that users have difficulties in establishing mutual orientations in CVEs, but that having some common frame of reference, for example a 2D map added to the CVE, might alleviate this. Another study (Billinghurst et al. 1999) found that asymmetries in collaborative interfaces impair collaboration, but the impact is decreased if the asymmetries matched the role people played in the collaboration. In both of these studies, the differences between the collaborators were not in the capabilities of the platform but in the states that each of the collaborators personally created for themselves. Our studies focus on collaborations were one of the collaborators has distinctly less capability than the other.
System Architecture

The system architecture of our collaboration framework comprises three main layers: the basic communication infrastructure (DISCIPLE), the framework for developing collaborative applications for heterogeneous environments (Manifold), and the task-specific applications (Figure 1).

DISCIPLE Communication Infrastructure

The DISCIPLE communication infrastructure is based on a replicated architecture for groupware (Dewan 1999). The central part of DISCIPLE is conceptualized as the collaboration bus (cBus) (Wang et al. 1999). The bus achieves synchronous collaboration through real-time event delivery, token-based event ordering and concurrency control. Each user runs a copy of the collaboration client and each client contains a local copy of the applications that are the foci of the collaboration. The local clients (on the same local area network) are communicating using the cBus (reliable multicast), whereas the remote clients are connected to a gateway on the local network, using a point-to-point TCP/IP connection. The gateway forwards the messages from the remote client to the other clients using the cBus.

DISCIPLE provides features that handle heterogeneity at the communication and networking level, but not at the application logic and user interaction levels. For these levels, we take the approach of designing a framework for display representation and cross-domain transformation that then guide the development of applications on each platform. Below we briefly review this framework which we call the Manifold framework. A detailed description is presented in (Marsic 2001).

![Figure 1. The system architecture. Palmscape, cWorld and Flatscape are example applications described below.](image)
Manifold Application Framework

Manifold is scaleable and can therefore be used on a wide spectrum of platforms, ranging from handheld access devices to high-end graphics workstations. It supports application logic with varying complexities of behaviors and visualizations with varying degrees of realism and media richness.

To achieve this, Manifold defines a common data structure across the heterogeneous applications, an event mechanism to propagate changes to other clients, and two ways for data transformation to take place between the different representations of data.

The key concept of the common data structure is a Glyph, which represents all objects that have a geometry specification and may be drawn (Calder and Linton 1990). In our interpretation, a Glyph is essentially a container for a list of \(<property, value>\) pairs. Properties include color, dimensions, constraints on glyph manipulation, etc. Adding or removing properties and hierarchical data abstraction control the Glyph size.

All Glyphs in a document form the scene graph, itself a Glyph, which has a tree data structure. In specific applications that extend the Manifold framework the scene graph is populated with different vertices (Glyphs). Glyphs are divided into two groups. Leaf Glyphs represent individual graphic elements, such as images, geometric figures, text or formulas in spreadsheet cells. PolyGlyphs are containers for collections of Glyphs. They correspond to branch nodes and can have children. Example PolyGlyphs are group figures, paragraphs, or calendars. PolyGlyphs have all the functionality of the Glyphs and in addition can contain Glyphs or other PolyGlyphs.

Changes to Glyphs, including changes to the scene graph, are exchanged between the clients using objects called CommandEvents. CommandEvents comprise one or more primitive tree operations, such as add/delete vertex, and operations on the Glyph properties.

Another key feature of the Manifold framework is a structured way of transforming different representations of data. A need for transformation has been identified in two general cases. The first case is when two collaborative applications have different views of exactly the same data structure. This can, for example, be 2D or 3D graphics representations of the same data. This case is addressed by inserting a transformer layer between the model (the Glyphs) and the view in the Model-View-Controller design pattern (Gamma et al. 1995). Manifold supports this by having defined a transformation interface so that concrete transformers can be plugged in as needed, and also by the implementation of a transformer that can be configured via an XML document.
The second case occurs when a platform does not support the same data types and/or processing capabilities as the other clients. This is the case when using Palm Pilots as clients, since the Java implementation on Palms does not support float or double data types. The mathematical support is also very limited, which makes it difficult to handle position and orientation of graphical objects given in floats or doubles. Manifold defines another interface for transformers of CommandEvents to and from the remote clients. The interface enables plug-in of transformers in the proxy gateway supporting remote clients (see Figure 1).

Example Applications

Using the Manifold framework we developed three complex applications: a 2D graphics editor (Flatscape), a 3D virtual world (cWorld), and a 2D graphics editor for Palm Pilots (Palmscape).

Flatscape

Flatscape is an expandable 2D graphical editor with the typical functionality of such editors. New applications can be built upon it, and it has, among other things, been used to build on-line virtual collaborative laboratories for biology teaching (Subramanian and Marsic 2001). Flatscape is developed using Java2D. Two example applications of Flatscape are shown in Figure 2.

The telepointers in Flatscape are 2D arrows pointing away from the center of the document (room, map, etc.) towards the part of the document currently viewed by the user. This is different from the typical telepointer, e.g., that found in (Roseman and

Figure 2. Two sample applications built using Flatscape: a room floor plan showing also a two-dimensional telepointer (left) and a situation map for mission planning (right).
Greenberg 1996), which shows only position and not orientation. In our implementation the telepointer owner, rather than the recipient, has control over its visibility. The telepointers, thus, can be considered as primitive avatars.

**cWorld**

The cWorld application (Figure 3) enables synchronous, multi-user building of collaborative virtual environments (CVEs) using Java3D. cWorld does not require special hardware and can be operated using the keyboard and a mouse although it also supports the use of the Magellan SPACE Mouse. This device provides the six-degrees of freedom movement used in navigating 3D spaces.

Each user has a unique 3D telepointer, shown in Figure 3 (left). These devices function as primitive avatars and appear at the discretion of the user. The telepointer is drawn at the position and orientation of the user’s line of sight.

**Palmscape**

The Palmscape application (Figure 4) is developed using J2ME CLDC 1.0 (Java 2 Micro Edition – Connected, Limited Device Configuration). It is a simpler version of the Flatscape editor, but still with most of the same basic functionality (e.g., create, delete, move and rotate objects) and user input is via the Palm stylus and buttons. Unlike Flatscape, it does not support multiple documents. In the present version telepointers are not implemented.

Both Flatscape and cWorld run on desktop workstations, though Flatscape can run on much less powerful PC’s than cWorld. In the first experiment reported below, where only Flatscape and cWorld were used, the heterogeneity was simulated both by
using different display applications and by restricting the window size of the Flatscape environment to the screen size of a typical Windows CE handheld computer (320×240 pixels). In the second experiment we used cWorld together with Palmscape, and heterogeneity was present both in data representation (2D vs. 3D) and in the computing platform (3D workstation vs. Palm Pilot).

User Studies of Heterogeneous Collaboration

Flatscape, cWorld and Palmscape demonstrate that Manifold supports heterogeneous collaborative environments. Although we show that we can construct these environments, we do not know how effective they will be for collaboration. It is readily assumed that a common view is necessary for effective communication, but others have found that non-WYSIWIS systems do not impair collaboration and may even facilitate it (Stefik et al. 1987). Obviously this depends on how different the shared views are. Our 2D vs. 3D displays represent a common future difference expected in office-field communication.

We study the effect of this difference. In the first study, we have collaborators share information between cWorld and Flatscape using desktop workstations and a LAN. In the second study, we employ a more realistic environment using cWorld and
Palmscape. The first study allows us to control for the effects of wireless response time and dropped packets. In this study we primarily look at display differences where cWorld gives the user a richer and more encompassing view of the room plus detailed feedback on the success of the user’s task.

Each worldview requires unique user interactions and views. In the 2D environments, only a part of the scene is visible requiring the user to scroll. This is done either with the mouse in Flatscape or with buttons on the Palm V. Jones et al. (1999) show that scrolling increases task performance time. The 2D space also leaves out useful information such as the orientation (front vs. back) of the objects and their aesthetic placement. In the 3D view, aligning objects with walls and corners of rooms is more difficult because distance and alignment are harder to estimate. On the other hand, 3D users have more cues available for correct orientation and aesthetic placement of objects.

Navigation in the 2D space is done by scrolling a top-down view of the world. In contrast, the 3D user navigates with 6 degrees of freedom controlled by Magellan SPACE mouse movements. 3D users have no external frame of reference (top, bottom, left, right) and must communicate about their position and viewpoint using common features in the space. The user also sees the space from different locations in the room so that there is no sense of a left wall or a front wall.

Each view of the environment therefore gives users certain advantages for creating and placing objects on their display. However, their different views of the world are likely to lead to misunderstandings in the communication that transpires. We expect these misunderstandings to be recognized (because of the observed changes on each user’s display) and repaired through the verbal channel. These repairs are likely to be quick and considered a natural part of the collaboration. However, they are also likely to increase the amount of time required to perform the collaboration task. If the interaction between environments is too complex it will require both collaborators to exchange longer and more detailed verbiage as they strive to establish a common ground. In this case, we can expect task completion times and number of communication repairs to be longer and larger for the communications in which the platforms are heterogeneous. We examine whether this is true in a study that compares collaboration across the 2D→3D and 3D→2D heterogeneous environment to collaboration in both the 2D→2D and 3D→3D homogeneous environments. The arrow in the above collaborations indicates who is giving and who is receiving directions. In the 2D→3D collaboration, the user of Flatscape is asking the user of cWorld to place objects.
Overview of Study 1

In the LAN-based study we collected two types of data: the number of communication repairs that occurred in a shared room arrangement task and the amount of time needed by a collaborative team to perform each room object arrangement.

The study was a within-subjects design. Collaborative 3-person teams were assigned to each of four conditions, 2D→2D, 2D→3D, 3D→2D and 3D→3D communication. We expected the heterogeneous collaborations to experience more communication repair statements, and to take more time than the homogeneous collaborations. We expected increased communication repair to increase the overall time for heterogeneous tasks. In addition, we expected an increase in time taken to process a collaborator’s request because of the need to mentally map it to the different environment. We expected fewer repairs in the homogeneous tasks because synchronized views give the collaborators equivalent information.

Subjects

We used 24 subjects (6 females and 18 males) from the university undergraduate and graduate population. All of the subjects had high levels of experience with computers and video games. The skill with video games was required in our study in order to reduce the amount of training time for subjects using the 3D environment. All participants indicated they were comfortable using a computer and mouse and eight had considerable experience with 3D virtual environments. We assigned our subjects to groups of three.

Procedure

The study required each team to perform four office furniture arrangement tasks. In each task, a total of 9 pieces of furniture (bookcases, file cabinets and desks) had to be placed at specified positions in the room. Each team member was given a sheet of paper with a top-down view of the placement of 3 of the objects. They could not, however, place these objects themselves and needed to direct other team members in their placement. Color indicated the mover of the objects and the owner of each telepointer. For each of the tasks, we assigned a display configuration for each of the team members so that they were either in a 2D or 3D environment. This assignment is shown in Table 1.
Table 1. Assignment of display environments to subjects.

<table>
<thead>
<tr>
<th>Task</th>
<th>User1 (red)</th>
<th>User2 (green)</th>
<th>User3 (blue)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>2D</td>
<td>2D</td>
<td>2D</td>
</tr>
<tr>
<td>Task 2</td>
<td>2D</td>
<td>2D</td>
<td>3D</td>
</tr>
<tr>
<td>Task 3</td>
<td>3D</td>
<td>3D</td>
<td>2D</td>
</tr>
<tr>
<td>Task 4</td>
<td>3D</td>
<td>3D</td>
<td>3D</td>
</tr>
</tbody>
</table>

We used eight 3-member teams. Team members were placed in different locations in a large lab and connected by microphones and headsets. Participants used Windows NT workstations connected to Ethernet. Workstations were equipped with both a normal PC mouse and a Magellan SPACE mouse. Before the study, subjects had 15 minutes training with cWorld using the Magellan SPACE mouse, and 10 minutes training with Flatscape.

Color differentiated among the three users’ objects and telepointers. Red was assigned to user1, green to user2 and blue to user3. In Figures 2 and 3, user1 is showing his telepointer, user2 has just added a file cabinet and user3 has placed a desk in the scene. Each user used both Flatscape and cWorld twice.

We collected data via three observers who recorded the start and stop time for each object placement request. For each request, the observers counted the number of communication repairs, e.g., “no, the desk is facing the wall, rotate it 180 degrees so it faces outward into the room.” Table 2 gives an example of the type of data collected.

Table 2. Example of data collected in Study 1

<table>
<thead>
<tr>
<th>Object type</th>
<th>Start time</th>
<th>End time</th>
<th>No. of repairs</th>
<th>Request from</th>
<th>Perform by</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red desk</td>
<td>9:04:32</td>
<td>9:06:20</td>
<td>4</td>
<td>2D</td>
<td>3D</td>
</tr>
<tr>
<td>Green file</td>
<td>9:12:02</td>
<td>9:13:21</td>
<td>1</td>
<td>3D</td>
<td>2D</td>
</tr>
</tbody>
</table>

Results

We did not find the expected differences between heterogeneous and homogeneous environments. Table 3 presents the descriptive statistics of our measures.
Table 3. Average time and number of communication repairs per object placement for 2D→2D, heterogeneous and 3D→3D collaboration. The time is presented in seconds.

<table>
<thead>
<tr>
<th></th>
<th>2D→2D</th>
<th>2D→3D</th>
<th>3D→2D</th>
<th>3D→3D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean time (sec) / std. dev.</td>
<td>79 / 119</td>
<td>121 / 99</td>
<td>79 / 128</td>
<td>110 / 111</td>
</tr>
<tr>
<td>Mean no. of repairs / std. dev.</td>
<td>2.02 / 1.93</td>
<td>3.53 / 2.71</td>
<td>1.26 / 1.15</td>
<td>1.91 / 2.71</td>
</tr>
</tbody>
</table>

As can be seen from Table 3, we found no large differences in either the task performance times or the mean number of repairs. The largest differences are in the 2D-to-3D collaborations, where the individual viewing a Flatscape display is giving object placement instructions to a person in cWorld. A correlation of the object placement times and the repair counts (Pearson $r = 0.16$, n.s.) indicated that there was little relationship between the number of repair statements and the time it took to perform the task.

Because of this low correlation, we felt comfortable in separating out the number of communication repairs and the object placement times in our multivariate data collection and running separate Student’s $t$-tests that compared heterogeneous to homogeneous collaboration environments. Our results are shown in Table 4.

Table 4. A comparison of communication repair and task performance time for homogeneous versus heterogeneous collaboration environments. Times are in seconds.

<table>
<thead>
<tr>
<th></th>
<th>Heterogeneous 3D→2D &amp; 2D→3D</th>
<th>Homogeneous 3D→3D &amp; 2D→2D</th>
<th>Student’s $t$ test results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean time (sec) / std. dev.</td>
<td>99 / 116</td>
<td>94 / 116</td>
<td>$t = 0.337, p = 0.37$</td>
</tr>
<tr>
<td>Mean # repairs / std. dev.</td>
<td>2.37 / 2.35</td>
<td>1.97 / 2.34</td>
<td>$t = 1.337, p = 0.18$</td>
</tr>
<tr>
<td>No. of objects</td>
<td>188</td>
<td>92</td>
<td></td>
</tr>
</tbody>
</table>

The two $t$-tests show that we were unable to find significant differences between the heterogeneous and homogeneous conditions.
Discussion

We expected the homogeneous collaboration to show better performance times and fewer repairs than the heterogeneous collaboration. We believe we did not find these differences because of two additional factors; (i) the difficulty of placing objects in the 3D environment with the Magellan SPACE mouse and (ii) the advantage of viewing objects in the 3D environment. The object placement times in the 2D→2D collaboration (Table 3) were, on average, 31 seconds faster than the 3D→3D collaboration. The mean number of repairs for these tasks was approximately equal. The implication is that the object alignment task in 3D is more difficult (our users mentioned this numerous times).

We examine this explanation further by looking at the 2D→3D and 3D→2D tasks. In the 2D→3D task, the person giving directions was in the 2D environment and the person performing the object placement was in the 3D environment. This was reversed in the 3D→2D task. If the difficulty were with the 3D environment, we should find that the average object placement time is longer in the 2D→3D case. The average time difference was significantly longer ($t = 5.149$, $p < 0.001$). Table 5 displays the results of this post hoc comparison.

Table 5. A comparison of the communication repair and task performance time between 3D→2D and 2D→3D object placement requests. Times are in seconds.

<table>
<thead>
<tr>
<th></th>
<th>3D→2D</th>
<th>2D→3D</th>
<th>Student’s $t$ test results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean time (sec) / std. dev.</td>
<td>79 / 128</td>
<td>121 / 99</td>
<td>$t = 5.149$, $p &lt; 0.001$</td>
</tr>
<tr>
<td>Mean # repairs / std. dev.</td>
<td>1.26 / 1.15</td>
<td>3.53 / 2.71</td>
<td>$t = 1.746$, $p = 0.086$</td>
</tr>
<tr>
<td>No. of objects</td>
<td>47</td>
<td>45</td>
<td></td>
</tr>
</tbody>
</table>

We also note that the mean number of repairs is considerably smaller for the 3D→2D than the 2D→3D object placement. This result is tending towards significance ($t = 1.746$, $p = 0.086$). We interpret this effect to arise for two reasons. The 3D view allows the team member to give better directions to the 2D member and the 2D application is easier to perform manipulations in. It is also possible that the effect we are measuring is only the result of the 3D application being more difficult to work in because of a lack of skill in using the 3D SPACE mouse. If this is so, then we should see no difference between the 3D→2D setup and the 2D→2D setup. Table 6 gives a post hoc comparison of these values.
Table 6. A comparison of the communication repair and task performance time between 3D→2D and 2D→2D object placement requests.

<table>
<thead>
<tr>
<th></th>
<th>3D→2D</th>
<th>2D→2D</th>
<th>Student’s $t$ test results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean time (sec) / std. dev.</td>
<td>79 / 128</td>
<td>79 / 119</td>
<td>$t = 0.00, p = 0.43$</td>
</tr>
<tr>
<td>Mean # repairs / std. dev.</td>
<td>1.26 / 1.15</td>
<td>2.02 / 1.93</td>
<td>$t = 2.899, p &lt; 0.001$</td>
</tr>
<tr>
<td>No. of objects</td>
<td>47</td>
<td>96</td>
<td></td>
</tr>
</tbody>
</table>

Times to perform the task are equivalent but the number of repairs is significantly larger in the 2D→2D environment. Because we can find no difference in performance times but do find it in conversational repairs taking place, our explanation that the SPACE mouse difficulty alone caused differences would appear to be wrong. In fact, this post hoc analysis of the data suggests that the 3D-to-2D collaboration improves the task performance. This bears some similarity to work carried out by Steed et al. (1999) who found that leaders more readily emerged in a group task from the more immersive VR environment.

We also observed throughout the study that users readily adapted to the differences in their environments. For example, 2D users soon realized that phrases such as “left wall” made little sense to their 3D counterpart and adjusted their conversation accordingly. Furthermore, most teams did realize that it is easier to correctly align objects in 2D, and therefore asked a 2D user to perform the alignment. There were also occasions of transference between the 2D and 3D tasks. One user, for example, looked for other users’ telepointers on the floor when first entering the 3D environment.

We also found unexpected collaboration to support our findings. When we had a 2D→3D collaboration between two users, if the supposedly uninvolved user had a 2D environment, this person would sometimes help the person using the SPACE mouse with the object placement. When we had a 2D→2D object placement task, the “uninvolved” person with a 3D display would comment on the 2D placement, thus helping the task. In short, the collaborators recognized the limitations of their environments and changed the rules of our experiment in order to collaborate more effectively.

Because the $t$-tests we ran in Study 1 were a post hoc examination of the data upon not being able to find differences between the heterogeneous and homogeneous conditions in our study, we re-ran this study with hypotheses prepared from our post hoc examination, that is, that 3D→2D collaboration would be better than 2D→3D collaboration and also better than 2D→2D collaboration. We also reran the study to
examine our hypotheses in a realistic environment with true network latencies and reduced bandwidth and computing capability considerations. To control for the collaboration adaptation we viewed in our three person studies, we limited our collaborators to two individuals.

Overview of Study 2

We modified our study to compare cWorld and Palmscape. The differences introduced by moving to a Palm Pilot included the need to use a stylus and the much longer times needed to update the screen displays because of the memory and processor limitations of the Palm. Additionally, event arrival latencies were expected to be highly variable depending on the interference encountered in the cellular transmission. Because these latencies would make the 3D→3D communication extremely different from the 2D→2D communication, we moved cWorld, to a powerful laptop connected by a wireless modem to cellular transmission. Instead of running teams of three collaborators, we used 6 teams of two collaborators each, all male graduate students.

As before, the subjects had experience with computers and 3D environments. One of the collaborators was placed in the physical room that matched the virtual office being organized, creating a more realistic problem for the team. The collaborating teams were two floors apart and used cordless phone headsets for verbal communication. As before, we trained the subjects on the use of both environments, in this case training them until their performance time for each trial was approximately equal. This took in average 25 placements of objects in 3D and in average 13 placements of objects in 2D.

We captured the same dependent variables as in Study 1. Subjects placed five objects per task rather than nine objects and completion times were captured for each task rather than for individual object placement because of the difficulty of obtaining an accurate measure of the individual placement times in the mobile environment. We found no significant correlation between repairs and performance time as in Study 1 (Pearson $r = 0.29$, n.s.) so we again assumed independence of our repair and time variables and ran separate Student’s $t$ tests to examine our hypotheses. Tables 7, 8 and 9 illustrate our results.

Despite latencies of 2–15 seconds and the awkwardness of moving objects with the Palm V buttons and stylus, we found the same results as before, that is, that the display differences advantaged the collaboration in the 3D→2D versus 2D→3D collaboration. As before, we were unable to uncover performance and error differences between the heterogeneous and homogeneous environments. In the
comparison of 3D→2D and 2D→2D collaborations, both time and repair error data favored the 3D→2D setup, but one-tail t-tests did not find significant differences.

Table 7. A comparison of communication repair and task performance time for homogeneous vs. heterogeneous collaboration environments. Times are in seconds.

<table>
<thead>
<tr>
<th></th>
<th>Heterogeneous</th>
<th>Homogeneous</th>
<th>Student’s t test results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean time (sec) / std. dev.</td>
<td>457 / 145</td>
<td>575 / 225</td>
<td>t = 1.08, p = 0.14</td>
</tr>
<tr>
<td>Mean # repairs / std. dev.</td>
<td>7.00 / 5.33</td>
<td>6.92 / 4.38</td>
<td>t = 0.03, p = 0.96</td>
</tr>
<tr>
<td>No. of tasks</td>
<td>12</td>
<td>12</td>
<td></td>
</tr>
</tbody>
</table>

Table 8. A comparison of the communication repair and task performance time between 3D→2D and 2D→3D object placement requests.

<table>
<thead>
<tr>
<th></th>
<th>3D→2D</th>
<th>2D→3D</th>
<th>Student’s t test results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean time (sec) / std. dev.</td>
<td>382 / 90</td>
<td>531 / 158</td>
<td>t = 2.0, p &lt; 0.05</td>
</tr>
<tr>
<td>Mean # repairs / std. dev.</td>
<td>4.00 / 2.45</td>
<td>10 / 5.90</td>
<td>t = 2.3, p &lt; 0.05</td>
</tr>
<tr>
<td>No. of tasks</td>
<td>6</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

Table 9. A comparison of the communication repair and task performance time between 3D→2D and 2D→2D object placement requests.

<table>
<thead>
<tr>
<th></th>
<th>3D→2D</th>
<th>2D→2D</th>
<th>Student’s t test results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean time (sec) / std. dev.</td>
<td>382 / 90</td>
<td>479 / 117</td>
<td>t = 1.6, p = 0.07</td>
</tr>
<tr>
<td>Mean # repairs / std. dev.</td>
<td>4.00 / 2.45</td>
<td>7.5 / 5.36</td>
<td>t = 1.46, p = 0.09</td>
</tr>
<tr>
<td>No. of objects</td>
<td>6</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

Note that these results do not suggest that heterogeneous collaboration is better than homogeneous collaboration, nor do they in any way indicate that the WYSIWIS concept is not a good way to run collaborations. The richer environment presented in
the cWorld setting was also more difficult to manipulate adding to a user’s performance time and making both 2D→3D and 3D→3D tasks take longer. However, this rich environment helped the person doing the manipulation of the furniture objects in Palmscape. Thus, the 3D→2D and 2D→2D tasks both took less time to perform than any manipulation done in 3D. In contrast, the number of verbal communication repairs increases significantly when the person giving the directions is using the Palmscape environment. In all the studies we find the use of verbal communication to ground the task and repair misunderstandings. In our instantiations of collaboration differences, this verbal communication helps users understand their differences.

We also find that these differences, by giving collaborators different views of their environment, help them perform their tasks better. CWorld helped its user give better directions on object placement and Palmscape supported an easier to use input device. This suggests that when building collaborative environments that are heterogeneous, designers must be aware of the tasks the platforms will be used for and the roles each user will play in the collaboration. If not, we could end up with situations where a user is being instructed by a Palmscape user to place objects in cWorld rather than the more efficient role reversal of Palmscape object manipulation and cWorld direction giving.

Conclusion

Heterogeneous sharing in synchronous collaboration is important with the proliferation of diverse computing environments, particularly wearable computers and handheld devices. This work presents a data-centric design for synchronous collaboration of users with heterogeneous computing platforms. Our approach allows clients with different capabilities to share different subsets of data in order to conserve communication bandwidth. We have presented an instantiation of this approach in a furniture arrangement application that presents collaborators views of the task consistent with their platform capabilities. We examined users collaborating in the furniture layout task with different platforms in order to measure what effect the device differences would have on carrying out the collaboration. We found that instead of being hindered by the platform differences, the reduced display user (2D) coupled with advice from the information rich display user (3D) improved performance. These results are encouraging as they suggest that display, input device and bandwidth differences need not be a deficit in the collaboration and that they can, under the right circumstances be an advantage. However, we need to carefully state
that this work just touches a tiny part of large differences that can occur in heterogeneous setups.

References


