Can We Work Together?

Dorian Miller

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science.

Chapel Hill
2009

Approved by:

Advisor: P. David Stotts

Reader: Mary C. Whitton

Reader: Stephen F. Weiss

Gary Bishop

Prasun Dewan

Abstract

Dorian Miller

Can We Work Together?

(Under the direction of P. David Stotts)

People have a versatility to adapt to various situations in order to communicate with each other regardless of a person's disability. We research separate computer interfaces to support remote synchronous collaboration in two situations. First, a deaf person collaborating with a hearing person uses a shared workspace with video conferencing, such as the Facetop system. Second, a blind person collaborating with a sighted person uses our loosely coupled custom shared workspace called Deep View. The design features of the respective interfaces accommodate the disability of a deaf person or a blind person and enable communication with a person without a disability. The interfaces expand the ways in which people with disabilities participate in a collaborative task to a level of detail not possible without our interfaces. The design features of our user interfaces provide alternative channels for the collaborators with disabilities to communicate ideas or coordinate actions that collaborators without disabilities would otherwise do verbally or visually.

We evaluate the interfaces through three user studies where collaborators complete full fledged tasks that require managing all aspects of communication to complete the task. Throughout the research we collaborated with members of the Deaf community and members of the blind community. We incorporated the feedback from members of these communities into the implementation of our interfaces. The members participated in our user studies to evaluate the interfaces.

To those
– with or without disabilities –
who may benefit from the researched technology

Acknowledgements

This dissertation is made possible through the teamwork of many people who helped me with resources or contacts that I did not have myself. The staff, students, and faculty of the Computer Science department provided an encouraging and stimulating environment for pursing graduate school and research. My contacts in the Deaf communities and blind communities were so generous to openly include me in their communities.

Some of the individuals I would like to thank who helped with the technical aspects of the dissertation:

- My dissertation committee – David Stotts, Mary Whitton, Stephen Weiss, Gary Bishop, and Prasun Dewn – who gave me the freedom to explore my interest in assistive technology and supported me in the process. Their expertise and experience were tremendously helpful, especially in group meetings.
- Rakesh Babu, Sina Bahram, Veli-Pekka Tätilä, Jay Macarty, Suzanne Balik, Larry Weiss, Philip Arnette, and Gary Bishop for helping to create the Deep View user interface by discussing or evaluating early versions of the interface.
- Karl Gyllstrom for completing the full usable experience of Facetop so user study participants could use seamlessly to focus on the collaborative task.
- Alex McLin and Covey Denton for an initial review of using Facetop for members of the Deaf Community. Alex also had the idea for the Facetop Tablet project.
- James Culp for implementing the Facetop Tablet application.
- Keith Lee and Jason Smith for contributing to piloting the user study involving Facetop.

Table of Contents

List of Tables

List of Figures

Chapter 1
Introduction

Coming together is a beginning.

Keeping together is progress.

Working together is success.

-- Henry Ford

The most important part of Henry Ford's quote above is that "…working together is success." A team effort benefits from each member's bringing a unique set of skills and knowledge to the table. Furthermore, a team member's personal background gives the person a unique perspective; for example, a person with a disability has learned strategies to accommodate for the disability. Helen Keller is an example of a person who despite deafness and blindness was, through teamwork, a successful author, mentor, and activist. She reflected on the value of working together by saying, "Alone we can do so little; together we can do so much." Keller enriched society with her insights on her situation, but also relied on others to learn, communicate, and, in general, accommodate her disabilities.

Through novel computer interfaces we design and evaluate we expand the possibilities for people with and without disabilities to collaborate remotely. We consider two pairs of collaborators. One pair is a deaf person working with a hearing person. The difficulty for the collaborators is to communicate fluently because we assume the hearing person does not know ASL and the pair cannot communicate verbally. The other pair is a blind person working with a sighted person. The collaborators' difficulty is accessing documents or other workspaces they are working on. Although we provide the

collaborators with different interfaces to accommodate the collaborators' disabilities, in both cases we research how the interfaces support the fundamentals of the collaborators' communication. The interfaces we research are a form of assistive technology, defined as:

> *"Any item, piece of equipment, or system, whether acquired commercially, modified, or customized, that is commonly used to increase, maintain, or improve functional capabilities of individuals with disabilities."*
> *[Section 508 of the Rehabilitation Act, (Section508)]*

Henry Ford's quote reflects on collaboration beyond individuals or teams working together. Ford's quote reflects society's efforts to cooperate with communities of disabilities to accommodate their needs, specifically related to making assistive technology available where needed. Society has made a "beginning" in "coming together" by passing laws and regulations, such as Section 508 of the Rehabilitation Act, to give companies and other organizations requirements to act on. These organizations are making "progress" by developing and implementing initiatives, such as the W3C Web Accessibility Initiative (WAI). Although the accessibility of web resources has greatly improved, inaccessible portions of resources prevent users from obtaining complete information. Researchers in assistive technology find that guidelines, such as WAI, provide basic access, but do not enable members of communities with disabilities to access the information as completely as users without disabilities. In our research we hope to reach the pinnacle of success by "working together" with communities with disabilities to enhance the available assistive technology, in particular, to facilitate the collaboration between people with and without disabilities. I refer to myself and my dissertation committee as "we" to reflect the collaborative effort.

1.1 Introduction to technology

In this dissertation we conduct three user studies to learn how collaborators use their respective interfaces to communicate when one participant is hard of hearing or visually impaired. We focus on the situation where collaborators are at different locations

(remote) and collaborators use a shared workspace supported by a computer system. Collaborators use the shared workspace to play games or create content, specifically node-link diagrams. Although the interfaces are different for a deaf person working with a hearing person and a blind person working with a sighted person, aspects of the interfaces support the fundamental concepts of communication that enable collaborators to communicate.

We engineered the computer interfaces we evaluate by integrating existing technologies. The collaborative interface for a deaf person working with a hearing person is adopted from existing technologies. The main part of the interface is the Facetop video conferencing system implemented in prior research at UNC Chapel Hill. Our contribution is to evaluate video conferencing technology with an unexplored combination of collaborators. Previous research assumes two hearing collaborators communicating verbally or two deaf collaborators communicating through sign language.

To research the collaboration of a blind person working with a sighted person, we engineered a custom shared workspace, called Deep View, to specifically access and edit node-link diagrams. With Deep View a sighted person uses a visual diagram application while the blind user examines and edits an audio representation of the same diagram. A collaborative interface with different representations of the interface is known as a loosely-coupled shared workspace. Our main contribution with Deep View is to expand the limited research on supporting collaboration between a blind person and a sighted person. A secondary contribution is to add a new accessible diagram interface to a list of existing research projects.

We evaluated the two interfaces in three user studies: one to evaluate the experience of the deaf person working with a hearing person and two studies to evaluate the experience of a blind person working with a sighted person. We support the thesis through evidence gathered from the three user studies where collaborators complete full-fledged tasks involving a shared workspace. Our findings are summarized in the thesis statement.

1.2 Thesis statement

> The collaborative interfaces we design and evaluate accommodate the disability of a deaf person or blind person to communicate in a remote synchronous collaboration with a person without a disability, thereby expanding the ways in which people with disabilities participate to a level of detail not possible without the design features. The design features of the user interfaces provide alternative channels for collaborators with disabilities to communicate ideas or coordinate actions that collaborators without disabilities would otherwise do verbally or visually.

We support the thesis through evidence gathered in three user studies where collaborators complete full-fledged tasks involving a shared workspace. The evidence includes evaluating the quality of the collaborators' produced artifact, analysis of the observations of the participants' interaction, low level artifacts of the task interaction, and responses to questionnaires and discussion with participants.

A deaf person working with a hearing person used a tightly coupled workspace:

- Deaf-hearing pairs rely on video to communicate by gesture compared to hearing-hearing pairs who do not use video.

- Deaf-hearing pairs rely on the telepointer to gesture at the workspace instead of gesturing through video as we originally hypothesized.

A blind person working with a sighted person used a loosely coupled workspace:

- A blind participant using the Deep View interface understands a simple node-link diagram almost as well as a sighted person using a visual diagram application.

- Although blind participants are slower reading or editing a diagram than a sighted participant, the blind participant's strategy of memorizing a simple node-link diagram enables the participants to discuss the diagram at a similar pace.

- Blind participants and sighted participants prefer to use the Deep View collaborative system to complete a diagram task rather than the currently available technique, where a sighted person edits the diagram and a blind person participates in the discussion.

- Semantic pointing gives collaborators symmetrical access to point at diagram nodes in their respective interfaces.

1.3 Scope of research

Our research incorporates three fields of Computer Science: Human Computer Interaction (HCI), Computer Supported Cooperative Work (CSCW), and Assistive Technology. At the most general level our research contributes to the HCI objectives of developing and evaluating user interfaces so that the design features of the interfaces present information most appropriately for the user to perceive and process. The specific interfaces we research are in the area of CSCW and we research how our design features support the communication of collaborators. Considering collaborators with disabilities and assistive technology broadens the HCI and CSCW research fields' perspective because conventional assumptions of communicating verbally or having visual content are not taken for granted. Following we describe our research relevance to the areas of assistive technology and CSCW.

1.3.1 Assistive technology

The technology used in our research and interfaces is classified as assistive technology because its purpose is to accommodate users' disabilities. A single technology, for example text-to-speech, can have a wide range of assistive applications. A screen reader for blind computer users makes the information in a visual GUI accessible to the user. On the other hand, text-to-speech can also help students with learning disabilities circumvent the difficulties of processing or focusing on printed text. Our research demonstrates how a range of interface design features are used to facilitate the communication of collaborators with and without disabilities.

An important design aspect of assistive technology is universal design. Ron Mace, an architect using a wheelchair, is the originator of the term and describes it as (Mace):

> *"Universal design is the design of products and environments to be usable by all people, to the greatest extent possible, without the need for adaptation or specialized design."*

A common example of universal design is automatic door openers or curb-cuts, which assist people with mobility impairments but are also useful for persons with

strollers or persons temporarily unable to use their hands to open a door. In computer technology, universal design improves, for example, the availability of assistive technology by taking advantage of economies of scale. The technology we apply in our research ranges in the degree of its universal applicability to users.

The video conferencing technology we use to facilitate communication between a deaf person and a hearing person has the most universal design. The video conferencing technology is the same as that used by collaborators without disabilities. In our study we investigate the impact of various configurations to the video conferencing setup. We learn about how the collaborators have a unique strategy of using the technology to communicate ideas about the collaborative task. The video conferencing technology is general and unmodified, and therefore collaborators can complete a wide range of tasks using existing collaborative applications.

The technology we use to facilitate a blind person collaborating with a sighted person has a less universal design. In our research we developed the Deep View application, an interface customized for the requirements of blind users to access diagrams. Although the interface is intended for blind users, the Deep View system is flexible enough to enable blind users and sighted users to transparently exchange diagrams. Deep View generates visual diagrams for sighted colleagues to use. Furthermore, we integrated Deep View into existing visual diagram applications so that blind users can access existing diagrams that sighted colleagues are using. When collaborating, the blind collaborator uses Deep View and the sighted collaborator uses the visual diagram application. Nevertheless, using a custom interface for the blind user limits the collaboration to diagram related tasks as opposed to the other group of collaborators who can complete a wide range of tasks with their video conferencing tools.

1.3.2 Computer Supported Cooperative Work

The research field of Computer Supported Cooperative Work (CSCW) investigates computer systems and interfaces to support multiple people working together. The CSCW field considers collaboration ranging from collaborators working synchronously to collaborators working asynchronously in order to complete a task. Our research is limited to two remote collaborators working synchronously on a task and this

is the focus of the following discussion. In this scenario there are several arrangements of the computer system to support the collaboration.

The collaborative tasks we research are based on the collaborators using a shared workspace; that is, remote collaborators working on different computers view a common application (shared workspace) and changes to the application state are reflected in each person's interface of the application. Shared workspaces are specific to a task and range from online multiplayer card games to collaborative word processors or spreadsheets, such as those provided by Google Docs™. The shared workspace also has a form of telepointer enabling collaborators to point at parts of the application.

The collaborators use a shared workspace with other video conferencing components to facilitate communication. Video conferencing enables the collaborators to see each other and converse verbally, if they are hearing. Furthermore, collaborators have chat messaging for communicating through written messages.

As for the synchronous collaboration we consider in our research, the collaborators work together more or less closely. Tightly coupled collaboration refers to collaborators working on the same portion of the task at the same time. The collaborators might also work loosely coupled, where they work on different portions of the task, for example, dividing the task between them.

The collaborators' interface design depends on the degree to which the collaborators are working together. Collaborators working tightly coupled use a tightly coupled interface where the interfaces of the collaborative applications are identical for both collaborators. An example of a widely used tightly coupled interface is desktop sharing applications, such as RealVNC (RealVNC). In comparison, collaborators working loosely coupled can use loosely coupled interfaces where the interfaces of the collaborative application can be different for each person. An example of a loosely coupled interface is Google Docs™, where collaborators writing a document can scroll to and concurrently edit different portions of the document.

1.4 Benefiting communities with disabilities

Our research contributes to the education and employment of people from the deaf and blind communities. We describe the US demographics of the deaf and blind communities and how our research can benefit people with disabilities. In our research we intend to design solutions for people with any degree of hearing or visual impairment. For simplicity of terminology, however, we will refer to blind persons and deaf persons.

Table 1-1 summarizes US demographic information about populations with hearing or visual impairments. While these statistics demonstrate trends within the respective communities, they are insufficient for comparing the Deaf and blind communities. For example, it is not conclusive that deafness is more prevalent than blindness. The statistics from each community come from different sources, collected at different times (mainly from 1990-1995). Also the definitions are inconsistent between communities; for example, there is a legal definition for blindness but not for deafness. Legally blind is defined as "central visual acuity of 20/200 or less in the better eye with the best possible correction, or a visual field of 20 degrees or less" (AFB 2005). Deafness does not have a legal definition but can be categorized in three ways: "deaf in both ears", "cannot hear and understand any speech", or "at best can hear and understand speech shouted into the better ear" (Holt, Hotto et al. 1994; Gallaudet 2005). From the statistics, the following three trends can be identified.

Table 1-1. Demographic statistics of deaf and blind communities.

Impaired community	Deaf/ hearing impaired (Holt, Hotto et al. 1994; Gallaudet 2005)	Blind/ visually impaired (AFB 2005)
Population	8.6% 20 million	4.2% 10 million
% of people with severe impairment	3% 0.6 million deaf	13% 1.3 million legally blind
% 65 and older	43%	50%
% Unemployed or not in work force	16.5% 18-44 years old 33.3% 45-64 years old	54% visually impaired (not legally blind) 32% legally blind
Graduated high school education (80% for general population)	16%	65%
Computer Users	Not available	1.5 million

First, the percentage of the overall US population with a severe impairment is small (less than 15%) compared to the population with a milder form of the same impairment. The implication for designing computer interfaces is that the people with mild impairments can still benefit from information delivered to their impaired senses, such as hearing a tone or perceiving if a light is on or off.

The second trend is that many people in the communities are not employed (33.3% deaf, 54% blind). Assistive technology, such as developed and evaluated in this dissertation, can provide tools to help them become active in the workforce and help those already employed. The tools assist people with disabilities in their work with colleagues without disabilities. In education, often a prerequisite for employment, these tools will enrich the learning environment by expanding accessible resources.

The third trend is that the prevalence of the two disabilities increases with age because they are often a result of aging. The majority of the people with a disability are older than 65. The collaborative interfaces will be beneficial regardless of age, although with different emphasis. For children with impairments, the collaborative interfaces can help them learn to work with and become familiar interacting with persons without impairments. For the older persons who have had experiences with sight or hearing, the tools will provide ways for the person to continue to work, participate, and interact with his colleagues.

1.5 Overview of dissertation

Our research is described in the following eight chapters. Chapters 2 and 3 describe the previously developed technology that relates to the design features we design and evaluate. Chapter 4 discusses knowledge from the field of psychology describing the fundamentals of communication between collaborators. We use the insights from the psychology field to design our collaborative interfaces. Chapter 5 documents the user study to explore collaboration between a deaf person and a hearing person. Chapters 6 and 8 describe the design of the Deep View interface (Chapter 6) and system (Chapter 8) that a blind person and a sighted person use to collaborate. Chapters 7 and 9 document the user studies to evaluate the interfaces described in Chapters 6 and 8 respectively. The conclusion in Chapter 10 ties together the knowledge we gain from studying the two kinds of collaborator pairs.

Chapter 2
Collaborative technology for deaf-hearing pairs

In our research of a deaf person collaborating with a hearing person we use existing collaborative computer interfaces. We focus on the collaborators using a shared workspace, which displays the task the collaborators are discussing and completing. Furthermore, collaborators use video conferencing to see each other and communicate through gestures, such as head nods and hand gestures. Our research contribution is to evaluate existing technologies in the unique situation where collaborators cannot communicate fluently verbally or through sign language.

In this Chapter we review previous research that assumes collaborators have a fluent means of communication. Specifically, we focus on video conferencing systems that overlay video of collaborators on a shared workspace. The main advantage is enabling a collaborator to point and gesture with his hand at the shared workspace as if the participants are face-to-face. Furthermore, the previous research investigates the behavior of the collaborators' eye gaze and how they observe each other. From a technical perspective, the previous research projects present a variety of techniques for integrating the image of the shared workspace and video of the collaborators.

First, we briefly describe how videoconferencing systems work. In videoconferencing systems, one person sees on a monitor a video image of the other person at the remote location. In the basic implementation of the system, each person's video image is captured by a camera, and the video image is transmitted to the other person's site, where it is displayed. Advances in the technology used to implement videoconferencing systems have made modern videoconferencing systems better and better: The quality of the video image has improved with better cameras and larger, brighter, more flexible displays. Initially video images were transmitted through

microwave radar systems or satellite networks. Nowadays, most video is transmitted through the Internet. Throughout, the engineering challenge has been the same – to transmit and display the video image with minimal latency. Video delay of even a second can interrupt the natural flow of face-to-face communication.

2.1 Basic video conferencing: "I can see you"

Videoconferencing technology has been implemented in many systems since the 1970s and research has evaluated the effectiveness of collaborators using the technology. In (Egido 1988) viewed videoconferencing systems as a failure because the significant broad impact promised by videoconferencing had not been realized. Instead, there were only niche applications, and only a few companies with the technology had been able to successfully apply it. Egido found that videoconferencing could not replace the face-to-face communication as vendors were claiming in their marketing of the technology. Today his suggestions for research to reduce the high costs and to find new and creative ways for videoconferencing to complement face-to-face communication have been somewhat realized.

Today it is common for people to use videoconferencing for business or pleasure. Video conferencing complements face-to-face communication because two individuals can have an impromptu meeting even when it is inconvenient to meet in person. A typical setup is readily available with a range of inexpensive software, such as Micorsoft's Net Meeting™ or Live Meeting™, and inexpensive webcams to capture the video. Affordable broadband speeds over cable, ISDN, or DSL are fast enough to stream good video.

Companies like WebEx.com use videoconferencing technology to provide innovative services, such as technical support for their products. In a typical scenario a WebEx customer service representative can access the customer's computer through a computer sharing application. The representative and customer can discuss the issue on the phone, or if desired start a video conference to see each other while talking.

Research studies, such as by Olson and Olson (Olson and Olson 1997) verify that collaboration through videoconferencing can be comparable to face-to-face collaboration, i.e., given that the remote collaborators have high quality audio and video connections.

In Olson's study, 222 individuals were broken into 74 groups of three and given the task of drafting the requirements for a system. The task was considered realistic because it involves planning, creativity, decision-making and cognitive conflict. The groups completed the task in one of four situations.

1. Face-to-face on a whiteboard with pens
2. Face-to-face with shared editor
3. Remotely with a shared editor and with only a high quality audio connection
4. Remotely with a shared editor and a high quality audio and video connection

The empirical results related to the impact of the video are that remote groups using the video performed similarly to the face-to-face group that used the shared editor. The quality of the task products was similar. Also the satisfaction of the participants was comparable.

The subjective feedback of the participants clearly indicated the value of the video. Working remotely without video and with only an audio connection, participants had more difficultly communicating. Participants were less certain about how someone else reacted to their ideas and participants found it more difficult to resolve disagreements.

An artifact of collaborating using video is that the discussion involved more overhead of processing the information being conveyed compared to the face-to-face discussions. This suggests that the video did not convey necessary information that would be conveyed face-to-face. This would be an area for future research.

2.2 Video conferencing for the Deaf community

Video conferencing was a breakthrough technology for the Deaf community enabling it to communicate over long distances. The history and discussion here are a summary of anthropologists Keating and Muir's work (Keating and Mirus 2003). The Deaf community could not take advantage of the telephone, invented 1876, and corresponding infrastructure until the invention of the teletypewriter (TTY), invented 1965, by the deaf physicist Robert Weitbrecht. TTY transmits chat messages over

telephone lines. Later with video conferencing two deaf persons could converse by signing using the mentioned typical setup.

Video conferencing has impacted sign language. A signer is influenced by his/her body, the camera, and the computer environment. A signer will sign at the camera so the remote person can more easily recognize the signs in the 2D camera video image. The sign language is adapted so that the sign gestures are within the camera's field of view; for example, the sign for "baby", usually signed around the waist, is signed under the chin. The equivalent of screaming in text messaging (bold text) is conveyed by signing close to the camera, filling the video image. The vibrant deaf culture around video conferencing demonstrates how well the technology has been adopted and that it will continue to be very important to the Deaf community.

Telephone companies are using video conferencing technology to enable a hearing person to have a telephone conversation with a deaf person through an interpreter. In 1996 Sprint started the first video replay service (VRS). For, say, a hearing person to start the call, he/she dials a toll-free number to reach a signing interpreter. In turn, the signing interpreter establishes the videoconferencing session with the deaf person. Then the interpreter translates between the verbal and signing to enable a fluid conversation. This service is free to the public and funded by the National Exchange Carrier Association, which in turn collects funds from telecommunication companies.

For a successful signing conversation through videoconferencing the video has to have a certain quality. The standard H.323 (ITU 2006) videoconferencing protocol defined by the International Telecommunications Union (ITU) specifies the transmission of video and audio data over packet networks. The specification however accounts for a range in the video quality. Hellström (Hellström 1999) calculated the specifications for video involving signing and lip reading (a part of signing language). The frame rate should be 25-30 frames per second so that smooth motion of signs is recognizable. The image resolution to recognize fingers should be QCIF (Quarter Common Intermediate Format) (176 pixels per line x 144 lines). However, in order to recognize cues from eye gaze, the resolution should be CIF (352 pixels per line x 288 lines). Hellström reports that the end-to-end delay should be less than 400 ms for fluent conversing.

Research into improving videoconferencing for the Deaf community focuses on video compression algorithms that maintain details of sign language. The principle is to have higher resolution for a signer's head and arms and trade it off for lower resolution of the background surrounding the person. Muir's (Muir and Richardson 2002) work is an example of this.

2.3 Research improving integration of video and workspace

By 1990 videoconferencing systems and collaborative applications such as a shared whiteboard were extensively researched. At the time, the systems were typically used separately; people would look at the video to converse or talk while working on the shared application. Research in the 1990s developed systems and interfaces to provide a more natural working environment with video and a collaborative application. This is accomplished by integrating the workspace and the collaborators' interpersonal space . The research focused on how people work together, rather than previous research evaluating the final product of the collaboration (Tang and Minneman 1991).

In this section we review 11 research projects, mostly in chronological order, to show how the 11 systems are built on top of each other. In the earlier work, the computer systems to support the collaborative interfaces were so complex that the majority of the research was in building the systems rather than evaluating them. With advancements in technology, the same interfaces can be implemented much more simply. This makes it easier for more researchers to explore the technology. Also, researchers can spend more time innovating and evaluating the interfaces rather than designing and building the systems.

Finally, we review the aspect of how computer systems provide mutual eye gaze in communication. For collaborators, mutual eye gaze is an important cue of the other person's attention. Video systems that convey mutual eye gaze are complex to build because the camera has to be positioned in the same place as the user's display. We review several projects that address mutual eye gaze.

2.3.1 TeamWorkStation and VideoDraw

Developed around the same time, TeamWorkStation (Ishii 1990; Ishii, Kobayashi et al. 1993) and VideoDraw (Tang and Minneman 1991) are interfaces that let remote collaborators interact with each other and with the collaborative application similarly to being face-to-face and drawing on a piece of paper. A collaborator looking at the paper can see her hands and the hands of her collaborator. Through hand gestures two collaborators have an effective mechanism to coordinate actions and communicate.

The TeamWorkStation and VideoDraw accomplish a similar working environment for remote collaborators. Video of the collaborators' arms and hands is transparently overlaid on the collaborative workspace. When a user gestures, the camera above the hands captures the live video that the collaborator watches on her monitor.

For the VideoDraw research team, the VideoDraw interface was the final result. The starting point for the interdisciplinary research team (computer scientists, anthropologists and designers) was to identify what the important communication cues are for people working together, for example, on a diagram. In brief, they found that hand gestures and their timing were important for the collaborators to coordinate actions. Also gestures added meaning to the drawings; the final diagram was not completely understandable without knowing certain hand gestures made while drawing the diagram. Communication through gestures will be explained in more detail in Chapter 4. Through these insights into gesturing, the research team created VideoDraw to provide gesturing to remote collaborators. The final evaluation of the system was low key with several short informal trials and a few longer trials.

The TeamWorkStation interface also lets collaborators use hand gestures to communicate. The emphasis of the research project, however, is providing remote collaborators realistic work environments where it is cognitively easy to switch between different workspaces. The collaborators can easily switch from viewing papers on a physical desk to viewing a computer application on the monitor. To switch workspaces, one user adjusts the position of the camera to point at the desk or at the monitor. Through the overlay video, one person could see where the other person was pointing and it was a natural action for the pointer.

The research team evaluated TeamWorkStation for one year by using it in their office for everyday work. The researchers found that the flexibility to switch between workspaces suited the dynamics of their working style. Also working through the collaborative environment reduced the number of documents they copied and distributed.

The system implementation of VideoDraw and TeamWorkStation are complex. Several computers and hardware components are necessary to transmit the video images of each collaborator's hands and arms. The video image also has to be mixed with the image of the shared workspace to create an overlay effect. For TeamWorkStation, the users would view the interface on a monitor separate from their computer.

VideoDraw has an additional complication. In the setup, the user naturally gestures over the screen showing the collaborative application. The video camera, in capturing the hands, would also capture the computer screen. So the computer screen is masked from the camera by placing it orthogonally polarized slides over the screen and camera; this prevents feedback in the camera from the computer screen.

2.3.2 VideoArms

VideoArms (Tang and Minneman 1991) is a more recent project that revisits the concepts of the VideoDraw and TeamWorkStation interfaces. The implementation demonstrates how technical advances have simplified implementation. The hardware required is networked computers each connected to a camera. Any display can be used, even touch sensitive whiteboard-size displays on which one can draw directly into the collaborative application. In place of masking the screen image through polarization, the same functionality is accomplished with readily available image processing software packages.

The contribution of VideoArms is to provide new techniques for blending the image of arms and hands with the workspace. The objective is to avoid having the image of the arms overlap and block the content of the shared application. Some techniques to accomplish this are rendering an outline image of the hand/arm or making the video transparent.

2.3.3 Clearboard

The Clearboard (Ishii, Kobayashi et al. 1993) system is Ishii's next iteration of TeamWorkStation. ClearBoard's unique contribution is to enable collaborators to have mutual eye gaze while naturally interacting with a collaborator as if she were on the opposite side of a piece of glass. That is, looking at the computer screen, a person can see the shared drawing application, and the other person appears behind it. With ClearBoard, pointing and gesturing is very natural.

Achieving mutual eye gaze was the main objective of the project. The research team observed that users effectively used eye gaze. When two collaborators speak to each other they look each other in the eye. Also, when one person points a finger and moves it across the screen, the other collaborator would follow the motion with his/her eye gaze.

The main engineering achievement of ClearBoard is to provide users with mutual eye gaze. Eye gaze is perfectly preserved as would be when two people look at each other in a mirror. In fact, the video image of the collaborator is a mirror image of that person. The computer screen a user looks at is covered with a half mirror. The camera capturing the video image for the other collaborator captures the image of the user observing themselves in the mirror. The computer screen is at an angle so that the reflection returns to the camera and not the user. As the mirror is only half a mirror the user can still see the computer screen because light from the screen passes through the half-mirror. Orthogonal polarized slides are used on the screen and camera to mask the screen image. The same technique was used in VideoDraw.

Pointing and gesturing at the computer screen is a special case. The collaborator does not see a mirror image, but rather the same view as in VideoDraw and TeamWorkStation. When the user reaches over the screen, the user's arm occludes the half-mirror and the camera captures the same view of the arm as in VideoDraw and TeamWorkStation. Although the view of the collaborator is different than the mirror view, the interface is still very comprehensible.

Although the half mirror is very useful to achieve mutual eye gaze, it has certain limitations. Use of the polarized slides and half mirror reduces the brightness of the screen for the user. Also, the camera captures the reflection of the user's background, which clutters the collaborator's view.

The ClearBoard interface was evaluated with two tasks. In the first task a teacher instructs a student on how to play backgammon. The pair had frequent eye gaze exchanges, while the teacher was explaining the rules. While speaking to each other, they look each other in the eye. Also when the teacher pointed at something, he would occasionally glance at the students to check that he, too, was looking at the same thing. Eye exchanges were less frequent during playing a game, because the pair was engrossed in playing.

There were three situations for the participants in the backgammon study. The situations compared people working face-to-face, people with a glass plane vertically between them (a mockup of clearBoard), and finally people with the ClearBoard prototype. The researchers recorded that more eye gazes were exchanged with ClearBoard than face-to-face. ClearBoard made it easier to switch between looking at the other person and at the backgammon board.

In the second task, a pair of collaborators uses ClearBoard to effectively solve a river crossing puzzle. The river crossing puzzle is significant, because Huthcin and Herb found that usual eye gaze was importance for two people to solve the puzzle.

2.3.4 Hyper Mirror

Hyper Mirror (Morikawa and Maesako 1998) was developed independently from ClearBoard, but it was built with similar objectives. Similar to ClearBoard, collaborators can share eye gaze and naturally gesture as in face-to-face communication. With Hyper Mirror, however, the collaborative workspace is extended beyond the small area of the computer screen to the area and objects surround the collaborators. For example, one collaborator could show and maneuver an object that the other person could see and point at. Of course, the remote person could not touch the local object because they were remote. Also it was possible to imitate shaking hands through the interface.

The principle of Hyper Mirror is that collaborators work together by looking into a virtual mirror. To work together, the collaborators have to continue to watch the virtual mirror. The virtual video mirror is created by mixing video of both collaborators. One person sees the full video of himself and his background. The other person's video is

added to the first person's video through a process called chroma key. In chroma key a person's background is blue, and in video editing, the blue background is subtracted.

Each collaborator sits at least 2 meters away from the screen so that the camera next to the projected screen can capture his image. Watching the video, the collaborators can coordinate the gestures in the same frame of reference. In each collaborator's real space, however, they are waving in thin air. The setup enables eye gaze to be preserved because the offset between the screen and camera position is minimized by the user's distance from both. Although the video makes it natural to collaborate, the users' posture is not natural. The person gestures in front of themselves, however, he has to look over his shoulder to see the video of the other person.

The researchers of the Hyper Mirror system conducted the user study to determine the best video image of the participants. The two parameters of the video were: normal video image vs. mirrored video image; and showing only the remote collaborator vs. showing both collaborators. 25 participants completed the study. The task was to either observe the different situations or interact with the other person such as acting out to shake hands. The main metric of the user study was for the participants to evaluate their "reality of presence" of the remote person. As expected, the video used in Hyper Mirror (mirrored image showing both participants) gave participants the highest sense of presence.

Informally, the researchers gathered feedback from public demonstrations of Hyper Mirror. Passersby intuitively understood how Hyper Mirror works and started interacting with people at the remote location for fun, such as acting out patting a passerby at the remote site on the head.

2.3.5 Large screen video conferencing displays

Next we review a series of systems that, similar to Hyper Mirror, use wall-sized projected video of remote collaborators. These systems are different from Hyper Mirror because they do not provide a shared workspace. As the emphasis of this writing is on the integration of interpersonal space and workspace, these systems will be briefly described.

One of these systems is VideoWindow (Fish, Kraut et al. 1990). The interface works as if a glass window were placed through the middle of the room. The people on one side of the glass are in one location, and the people on the other side of the glass are at another location. VideoWindow shows the users the same view as when a user looks at the glass window. In VideoWindow, a glass window is really a projected video image of the remote site. VideoWindow and similar systems (Bly, Harrison et al. 1993; Benford, Greenhalgh et al. 1998; Jancke, Venolia et al. 2001) have been used to research how people interact informally. Similar to the public demonstrations of Hyper Mirror, it is easy for people to become engaged with people at the remote site through the projected video.

2.3.6 SharedView and its evaluation

So far in the reviewed systems the captured video of participants and/or workspaces was from a fixed perspective. In the next series of projects the video is a first-person view of one of the collaborators. A possible scenario is an operator working on, say machinery, in the field collaborating with a coworker at the office. The coworker sees a video image similar to the operator's view through the camera mounted on the operator's head. As the operator moves his head, the view changes accordingly.

An example of such a system is SharedView (Kuzuoka 1992). SharedView is a novel edition, however, in that the operator sees video of the coworker's gestures captured in the office on his see-through head mounted display. Although the system provided the coworker with a mechanism to point, it was difficult to use in practice. To point successfully, the operator would have to hold still until the coworker finished motioning, i.e., pointed at the desired object. This would enable the collaborators to have a common frame of reference. Instead, as the coworker motioned, the operator moved his head to follow the motion. During the operator's head motion, the coworker has to continuously adjust his pointing motion to reach the desired point.

Fussell et. al. conduct user studies documented in papers (Fussell, Kraut et al. 2000; Fussel, Setlock et al. 2003) to evaluate the value of having the operator's first-person view of the shared workspace. The researchers' backgrounds are in psychology and their emphasis is on supporting communication between remote collaborators. In

(Fussell, Kraut et al. 2000) the argument is that a shared visual workspace is beneficial to an operator and a coworker situation as described. The accompanying user study falls short of demonstrating this. In the study pairs of collaborators complete the task of assembling a bicycle under three conditions: face-to-face, through an audio connection, and through an audio/video connection. The video is from the operator's head mounted camera. Pairs working with the audio/video connection were not able to complete the task as well as the face-to-face pairs. The shortcomings are explained by three limitations in the video. First, the operator cannot see the boundaries of the camera's field of view and is uncertain if the coworker can see what he is referring to. Second the field of view is limited and does not provide the same cues as if the entire workspace were shown. Third, the collaborators can not see each other's gestures and the coworker cannot see the operator's face.

The second user study builds on the first by adding a scene camera, i.e., a fixed camera at a distance that captures the entire workspace. Participants in the user study this time assemble a complex robot in one of four situations: face-to-face, through an audio connection, with audio and a scene camera, and with audio, first person video and a scene camera. As before face-to-face pairs complete the task better and with smoother communication. Pairs working with only a scene camera did better than pairs working with a scene camera and a head mounted camera. This suggests that the scene camera adds valuable information. The researchers suggest that adding head mounted camera worsened the collaborators' performance, because the coworker paid more attention to the first-person video rather than the scene video. The implication for future system video systems is to give coworkers better instructions on how to utilize the video.

2.3.7 Office of the Future

The focus of the "Office of the Future" (Chen, Towles et al. 2000) project is to develop the next generation of videoconferencing systems. Sitting at a desk a collaborator sees a 3-D image of the remote collaborator in real-time on a projected screen. The 3-D image is created from the camera array placed around each person. The passive stereoscopic display consists of two projectors with orthogonal polarized filters. The user wears polarized glasses, so that each eye sees the image from the corresponding

projector. The viewer wears a head tracker and as the viewer moves his head, the perspective is adjusted accordingly. The prototype demonstrates the concepts, although the latency must be reduced and the quality of the 3-D reconstruction has to be improved.

A secondary objective of the projects is to integrate the workspace into the interpersonal space. A novel innovation is to make a virtual shared workspace using 3-D computer graphics. This way, the collaborators can equally access the shared workspace; unlike physical objects, where only the local collaborator can manipulate them. In a Office of the future demonstration, the task was to arrange furniture in a room; the collaborators used a wand (controlled by a tracker in the hand) to manipulate the furniture.

History might repeat itself and like the systems developed in the early 1990's, today's complex system implementations might become simpler to replicate because of advances in technology; then researchers could study these systems in more detail.

2.3.8 Facetop

Facetop is the main video conferencing system we use to evaluate collaboration between a deaf-hearing pair. Facetop was originally created to support pair programming (Stotts, Smith et al. 2004). Since then it is being adopted for use by deaf persons as part of this dissertation and other applications described in (Miller, Gyllstrom et al. 2007).

Facetop lets collaborators use the video to visually gesture, such as pointing a finger, at the shared workspace similar to as if the collaborators were face-to-face. To use Facetop, each person points the camera connected to their computer at himself; for example, by placing the camera on top of the computer monitor. Figure 2-1 is a screenshot of what each person sees; that is a video of himself and his collaborator semi-transparently overlaid on the computer desktop. The computer desktop shows a shared application, in this case a checkers game. The collaborators are playing checkers against the computer, and together are plotting a game strategy. The collaborators control the checkers game with a mouse cursor as if the video were not present.
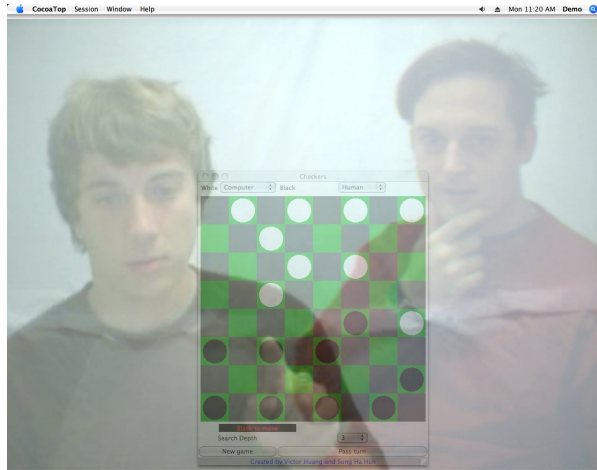
Figure 2-1. The collaborators are playing checkers against the computer using the Facetop video conferencing system.

The main reason a user sees a video of himself on the computer screen is to be able to make gestures relative to the shared application, such as pointing an index finger at an item in the shared application. The person making the gesture has to verify that his intended gesture is properly reproduced in the video image that the observer of the gesture sees. For the person making the pointing gesture, it is as natural as pointing an index finger at part of a physical mirror. The mirrored image of the index finger appears to point at the same location. Likewise, pointing at the computer monitor, the video image on the screen is the same as if the screen were a mirror. Like in a mirror, it appears as if the person is on the other side of the screen. Unlike a mirror where a person can touch it and see the mirror image, the person gesturing at the screen may not touch the screen. The person gesturing at the screen has to gesture some distance from the screen so that the gesture is present in the camera's field of view. The observer of the pointing gesture will see the pointing person's index finger overlap the item pointed at. As in Figure 2-1, one person is pointing to a checkers piece.

There are three additional advantages to a person seeing himself in the video. The first advantage is that registration of the camera user and computer are arbitrary. Regardless of the positioning, the user can adapt his gestures so that the gesture appears correctly in the video image. The second advantage is that participants of related videoconferencing systems have expressed a preference of seeing their own video so that they know how others see them (Sellen 1992).

24

The third advantage is most applicable to the deaf users. Overlapping the images of the video and shared application, the user can observe both images at the same time, unlike typical videoconferencing systems, where the user has to look at different parts of the screen. Also as the video image is larger, it is easier to see smaller details such as when one person waves to get the other person's attention. Although this feature could be useful to hearing users, they do not rely on it as much because the information conveyed in the video can be conveyed verbally.

The drawback of the semi-transparent video is the same as in our group meeting application. Visual details in the semi-transparent video can be washed out and make it difficult to recognize sign language. We will try to mitigate this issue in future work. The interface as is, however, could still be useful for collaboration between a deaf person and a hearing person. These collaborators would use more rough gestures, such as pointing, "thumbs up" or hand waving, which do not require as much detail to be understood.

There are a few requirements of the shared application and the collaborators' monitors. The application has to have the same relative position and size on both monitors so that when one person makes a pointing gesture, the elements pointed at is the same on both persons' monitors.

2.4 Mutual eye gaze

Mutual eye gaze is an important communication cue, however, designing systems to provide mutual gaze is difficult. For communication, mutual gaze is important for indicating where a person's attention is or for indicating when a listener is attending to a speaker (Vertegaal, Slagter et al. 2001). Implementing mutual eye gaze, for example in ClearBoard, required a half mirror and polarized filters. The following discussion is on two research projects that evaluate the necessity for mutual eye gaze and possible workarounds.

The first research system, called Hydra (Sellen 1992), was developed to provide videoconferencing participants with mutual eye gaze. A user of the Hydra system sits in front of an array of video conferencing units, each corresponding to a participant in the discussion. A unit has a small video screen, a camera above the screen, and a speaker to hear the corresponding participant's audio. When the user looks at a participant's screen,

the camera captures the necessary image for the pair to have mutual eye gaze. Other participants will see that the user's face is pointed away from them.

The Hydra system was evaluated in a user study, where a group of four participants discussed news topics. Topics were discussed in three situations: face to face, with Hydra, and with a conventional videoconferencing system called Picture-in-picture (PIP)(i.e. video image of all participants is displayed on a computer monitor).

Comparing face-to-face to the video mediated discussions, the characteristics were very similar; for example, on the metric in which participants took turns. With simultaneous speech however, there was more in the face-to-face condition.

The Hydra and PIP systems each had its advantages. Overall, the participants preferred Hydra. The spatial audio from the speakers made it easier to follow a speaker when several spoke simultaneously. Also, it was easier to attend to side discussions. One third of the user study participants liked that in PIP they could see themselves; they felt it was important to see themselves so that they know how other people see them. Also, one participant felt that the PIP video made her feel part of the group. The researchers came to the conclusion that video mediated discussion systems need not simulate face-to-face conditions entirely in order to be successful.

The second research project by Grayson (Grayson and Monk 2003) is an experiment to see how well an estimator can judge where a gazer is looking. Instead of being face-to-face, however, the estimator saw the gazer like in a typical videoconferencing system, where eye gaze is distorted. The camera capturing the gazer was on a 17 inch monitor that the gazers sat in front of. In the experiment the estimator had to guess where the gazer was looking on a horizontal line displayed on the screen. In one case, the camera was directly in front of the gazer; and in another case, the camera was horizontally offset. Despite the eye gaze distortion, the estimator accurately identified where the gazer was looking. The accuracy in the case where the camera was aligned with the gazer was 87%. The estimator was less accurate by 67% when the camera was at the offset position. This experiment suggests that with distorted eye gaze, it is still possible to judge where someone is looking. Further research should be done, however, to verify that this holds when the gazer is interacting more realistically, such as looking anywhere on the screen.

2.5 Balance of attention between workspace and collaborator

The computer interfaces reviewed in this section have provided collaborators views of their collaborators and the shared workspace. Olson's user study reports that video of the participants was useful. However, how do users divide their attention between looking at the shared workspace or at the collaborator's face? Observations from at least two studies show that the majority of the time is spent looking at the workspace. First, in ClearBoard, the backgammon teacher and student used eye gaze during instruction, but while playing they were so engaged that they did not exchange glances.

Second Graver's research (Gaver, Sellen et al. 1993) also shows that remote collaborators use the workspace view more than the view of the collaborator. In his experiment, too, people collaborate in order to lay out the furniture in a dollhouse. The remote person (away from the dollhouse) had one monitor on which he could switch between the view of the collaborator or three views of the shared workspace. On average, the remote person used a view of the collaborator for as little as 11% of the time. The view was used for lengthy negotiation and occasional glances to judge the collaborator's mood and level of engagement.

Although the observations show users' attention to be disproportionate between the collaborator and workspace, it demonstrates the value of viewing each. Most of the collaboration can be completed viewing the workspace and communicating verbally on how to interact and manipulate the shared workspace. Occasionally, however, the collaborators need to negotiate more abstract ideas, which they can communicate verbally and use cues from facial gestures.

2.6 Historical aside

In 1968 Douglas Englebart (Engelbart and English 1968) already tinkered with video conferencing concepts mentioned in the research after 1990. It was coincidence, however, that he overlaid the video of a collaborator on the workspace like ClearBoard, VideoDraw and other research projects mentioned here. It was an artifact of the technology; the technology did not allow placing the video in a separate opaque window like the now conventional video conferencing setup. To create the image he mixed the video signal of the computer output and of a video camera on a TV monitor. Engelbart's

prototype worked to demonstrate the idea but building a more comprehensive video conferencing system would have been much more difficult than with the technology of the research systems from the 1990s. Engelbart only briefly explored the idea mentioning in a paper that a video of the collaborator would be a useful feature.

2.7 Summary of video conferencing technology

Facetop innovates on research projects on integration of interpersonal space and workspace. The Facetop interface is most closely related to ClearBoard. Similar to ClearBoard, in Facetop people can see their collaborator as well as the shared workspace. Technical advances have made the Facetop implementation much simpler; it only requires a computer and video camera (webcam). The camera can not be positioned in such a way as to provide mutual eye gaze, but research into mutual eye gaze might provide a workaround. Also Video rendered in Facetop is similar to techniques used in the VideoArms project.

A novel contribution of the Facetop project is to apply the interface to support a deaf/hearing or deaf/deaf pair of collaborators. All other collaborative systems assume hearing collaborators and provide an audio connection for communication.

Here is a summary of interfaces reviewed. The systems are listed in chronological order according to the publication date of the corresponding paper:

Table 2-1. Chronological order of reviewed video conferencing systems

System	Year
Engelbart	1968
TeamWorkStation	1990
VideoDraw	1991
SharedView	1992
Hydra	1992
Clear Board	1993
Hyper Mirror	1998
Office of the Future	2000
VideoArms	2004
Facetop	2004

Chapter 3

Collaborative technology for blind-sighted pairs

Our research of a blind person collaborating with a sighted person contributes to two existing areas of assistive technology research. The first and main area is the actual collaboration between such pairs. We expand the research in this area by investigating a diagram editing task, which is more complex and general than a simple game investigated in previous research.

The secondary research area we investigate is accessible diagram interfaces. We require a custom accessible diagram interface so that a blind collaborator can access the node-link diagram in the shared workspace. Our contribution is to tailor our accessible diagram interface to accommodate the blind person accessing the diagram while discussing it with a sighted collaborator.

In this Chapter we review previous research related to our two research areas. Section 3.1 reviews an interface to support a blind person collaborating with a sighted person. Section 3.2 reviews accessible diagram interfaces.

3.1 Research on blind-sighted collaborators

We are aware of only one research project that explores computer interfaces to support collaboration of a blind-sighted pair. We review the project, which demonstrates the feasibly of such a collaboration. We also describe how our research expands knowledge in this area.

3.1.1 Previous research

Winberg's research (Winberg and Bowers 2004) introduced the CSCW community to the concept of collaboration between a blind person and a sighted person.

He demonstrates that it is possible for the pair to share a workspace and constructively complete a task. The user study task is to complete the Tower of Hanoi game (discs of different sizes are moved between three poles, according to constraints). The shared workspace is the game board. Each collaborator uses the most appropriate display: the sighted person uses a visual display and the blind person uses an audio display.

Winberg also developed the audio display for this project. The audio display is based on direct manipulation, which is how most common graphical user interfaces (GUI) work. In direct manipulation users can manipulate objects – represented visually in a GUI; for example, in the Tower of Hanoi game a user can click and drag the disc from one pole to another. Winberg's audio interface functions similarly. Instead of a visual representation, objects appear to emit sounds; for example, in Tower of Hanoi each disk and each pole has a unique sound. The disks and poles have the same spatial layout, as in the visual display. A blind person uses a mouse to explore the shared workspace displayed auditorily. As the mouse moves the person hears the objects close to the mouse position. Objects are manipulated like in the visual display; for instance, by listening for the objects, a user can select a disc and can drag a disc from one pole to another.

In the evaluation, a blind/sighted pair collaborated to solve the Tower of Hanoi game. Three pairs participated in the study. Each pair used one computer with a visual display (monitor) and an auditory display (headphones). The collaborators shared a telepointer that they had to coordinate in controlling. In theory, when the sighted person controlled the telepointer, the blind person could hear the objects it was passing over.

Overall, the collaborating pairs were able to successfully communicate and complete the task. The pairs' work was balanced, such that, after discussing the move, the blind person and the sighted person would take turns to interact with the interface and complete the move. Although the participants took turns to control the interface, sometimes the blind person would start to control the telepointer before the sighted person was done. The problem was quickly rectified: listening to the audio display, the blind person understood the problem and let the sighted person finish.

Occasionally, it would be complicated for the blind person when the sighted person made a series of changes to the game. When the blind person took control again, he/she would scan the entire game board to clarify the state of the game. The researchers'

observation is that the blind person integrates many sources of information, including listening/interacting with the audio display, talking with the sighted collaborator, and memory. Also, there is a delicate balance with a blind person listening to both the audio interface and the sighted collaborator. The researchers' suggestion for future work is to design interfaces such that they provide the functionality to assist the collaborators in the observed situations.

3.1.2 Our contributions

Our research in collaborative interfaces for a blind/sighted collaborating pair builds on the experience from the Tower of Hanoi interface. Like in the Tower of Hanoi interface, a sighted person uses a visual interface and a blind person uses an audio interface. The interfaces, however, are loosely coupled so that each person can navigate the application independently. Also, the interfaces are laid out differently to most appropriately accommodate how a user accesses the interface. The semantic pointing mechanism in our interface lets the collaborators shift focus to the same object. With this interface, the confusion observed in the Tower of Hanoi game should be minimized. Also our collaborative task of viewing and editing diagrams further demonstrates the range of collaborative applications a blind/sighted pair can participate in. Diagrams demonstrate a more complex, shared workspace, including detailed textual information that the collaborators will be able to work with.

The loosely coupled collaborative interface used in this research is based on CSCW research in flexibly coupled interfaces. Suite (Dewan and Choudhard 1991) and Rendezvous (Patterson, Hill et al. 1990) are two research projects that demonstrate this research. The original purpose was to accommodate flexibility in which collaborators work. Before flexibly coupled interfaces, collaborators' interfaces were identical, and they had to work on the same task. However, collaborators, such as coders or prose editors, would sometimes prefer to divide the task and work on separate parts. Furthermore, instead of sharing every edit, a person may want to make several edits before sharing with the collaborator.

The flexibility of the interfaces is based on the coupling values of the shared model (i.e. data) and display. Consider a card game where the players are remote and use

separate computers. In this case the coupled display shows the card table and the player's hand of cards. The table would be rotated appropriately for each player so that the table appears in front of the player. Also a player only sees his own hand of cards, because it is private information. When cards are played, they appear publicly on the table for everyone to see.

To illustrate coupling values in the shared model, consider the card game, where two people play together and share a hand of cards (maybe an expert is teaching a novice). The coupled value to consider is the card to be played. The team of players would want to show each other which card to play. When they have decided on a card, they can make the value public, i.e., place the card on the table. So the played card value would be private to the team before it is made public. This is an example of flexible coupling.

Pointing in flexibly coupled interfaces is done using a semantic pointer. When one user selects an object to be pointed at, the corresponding object in the other user's interface is highlighted.

In our collaborative diagram interface for the blind/sighted pair of collaborators, the display is flexibly coupled and the values in the shared model are tightly coupled. A sighted person uses a typical visual interface, whereby the blind person interactively navigates a diagram and listens to textual descriptions. At least in the initial research, we assume collaborators see identical models of the diagram, and therefore the values of the model are tightly coupled.

3.2 Diagram interfaces

To research collaborative interfaces for a blind/sighted pair we pick the application of viewing and editing node-link diagrams, such as UML (Unified Modeling Language) diagrams. Working with diagrams provides two benefits to the blind community. An accessible diagram interface provides blind users with access to the visual information in diagrams. Diagrams are significant for communicating ideas; for example, in software engineering the diagrams are used to document designs of systems. A blind programmer needs access to diagrams of a system to understand the system and to modify, implement, or test the system. Diagrams are also important for collaboration.

In designing the system, diagrams can be a main point of discussion. A blind person can participate in the discussion if he has access to and can interact with the diagram.

We review several accessible diagram interfaces. The interfaces provide access to diagrams ranging from pictures, maps, to node-link diagrams. We focus on how the interfaces present the information and how a blind user navigates a diagram. None of these systems were designed for collaboration between a sighted person and a blind person.

In our research we develop a single user accessible diagram interface based on concepts from the reviewed interfaces. We enhance the single user interface to enable collaboration between a blind person and a sighted person.

3.2.1 Comparison of existing accessible diagram interfaces

Deep View contributes to the research field of making diagrams accessible. Deep View's emphasis is on supporting a variety of node-link diagrams and aspects of navigating them. Another important issue is making existing visual diagrams in various formats available to blind users through the accessible diagram interface.

Deep View focuses on presenting the relationships between a diagram's elements. Several other projects take a similar approach specializing on diagrams in different domains. Similar to Deep View, the Eclipse plugin project (Smith, Cook et al. 2004) and TeDUB (Petrie, Schlieder et al. 2002) focus on representing UML diagrams (similar to an ERD). Also the Proof-checker (Stallmann, Balik et al. 2007) project makes finite automata accessible in a classroom setting. On the other hand, Kekule (Brown, Pettifer et al. 2004) presents the hierarchy of a chemical's proteins, acids and atoms. Deep View is generalized to handle a variety of node-link diagrams defined by advanced Deep View users. For example, the chemicals presented in Kekule could be modeled as a Deep View diagram with sub-diagrams.

Rather than present the relationships in a diagram, an alternative approach to making diagrams accessible is to present the spatial layout. The spatial layout conveys implicit relationships besides the explicit relationships expressed in a diagram's links. Several projects have focused on presenting the spatial layout of diagrams. The Audiograf project (Andrea 1996) presents node-link engineering diagrams, such as UML

diagrams. The Plumb (Cohen, Yu et al. 2005) and Bats (Parente and Bishop 2003) projects present the spatial layout specifically for geographic maps. Although a general map is not a node-link diagram, the Plumb project treats a map as a network where cities are nodes, and links connecting the cities represent routes.

These interfaces enable a blind user to explore the spatial layout of the diagram with a cursor that the user moves over the map. The interface reads or sounds information at or near the current cursor position. An innovation of these interfaces is to use commodity products to create cursors affordable to the blind community, i.e. without expensive specialized hardware. The projects leverage finger touch screens, pen touch screens on a laptop, trackballs, or a standard game-pad controller. An ideal accessible diagram interface for blind users would probably combine the presentation of relationship information and spatial layout to provide the most complete information.

A unique contribution of Deep View is to enhance techniques for navigating node-link diagrams. The hyper-linking feature helps navigate individual nodes and links. Deep View also provides the high-level queries for diagram characteristics (paths, cycles, and parallel paths), which are challenging for a blind user to identify. Without the queries a blind user would have to tediously sift through nodes and links to identify the characteristics. Identifying the high-level diagram characteristics can be difficult even if the spatial layout is displayed. A blind user still has to trace a path between several nodes in order to recognize the high-level characteristic.

Deep View addresses a separate issue which is for blind users to access existing visual diagrams stored in various formats. Deep View takes advantage of visual diagram applications which store the diagram model and allow it to be accessed programmatically. Unfortunately, many diagrams are not stored as such. Many diagrams stored electronically or on paper only maintain a diagram's visual artifacts, such as lines, shapes, colors, text, etc. Two projects DocExplorer (Ishihara, Takagi et al. 2006) and TeDUB (Petrie, Schlieder et al. 2002), for example, study algorithms to interpret a diagram's semantic information, such as identifying nodes, links and their relationships from the visual diagram. The DocExplorer project interprets diagrams in PowerPoint slides. The TeDUB project analyzes node-link diagrams stored as image files and in other formats.

Deep View is unique in providing a complete solution for a blind user to edit and visualize diagrams. In Deep View editing is possible through a series of dialog windows controlled by keyboard shortcuts. Deep View visualizes a diagram automatically with the GraphViz tool. Furthermore, the Deep View plugin enables the diagrams to be created in Rational Rose or Microsoft Visio. Few other projects address the issue of making diagram editing accessible. The IC2D (Kamel and Landay 2002) project provides an accessible diagram editor for diagrams, which are general pictures. In IC2D, similar to a visual diagram application, a blind user places and sets shapes, lines, colors, etc. Navigation is controlled through the keypad. An alternative to editing a diagram in the way Deep View enables can be found in the projects that automatically layout a visual diagram, such as GraphViz (Gansner and North 2000) and SugiBib (Eichelberger 2002). In these projects a diagram is defined in a text file. Although the text file is accessible and editable, a text editor would be cumbersome for a blind person to use because the editor does not provide a navigation mechanism specific to diagrams.

3.2.2 2D tactile display

A two-dimensional Braille display is the preferred interface in the visually impaired community. The Braille display would function similar to a computer monitor. Instead of using light to display a visual image, each pixel in the image corresponds to a point that is either level or raised to create haptic pixels. A user would perceive the displayed diagram as she/he runs a finger across the display. The haptic surface might be created on paper with relief or with electromagnetic pins, such as developed by the National Institute of Standards and Technology (NIST 2002).

3.2.3 IC2D

The IC2D project (Kamel and Landay 2002) is for blind persons to create 2D pictures with lines and shapes. Its contribution is the navigation technique with the number keys of the number key pad.

The screen is divided into nine regions, each corresponding to a number on the keyboard. Pressing a number lets the blind user zoom in on the corresponding region. The navigation is recursive, that is, the zoomed in region is again divided as before. With each key press, the user can narrow in on a specific part of the diagram.

To draw and examine the diagram the blind user navigates to a desired region. At the desired region he uses a keyboard command to listen to a listing of drawing items in the region. The user can also add to the drawing. The drawing element to add is selected from a menu. For example, to draw a line, the user navigates to the two ends of the line and gives the command to place a line. Of course, one could imagine that the final diagram can be printed on relief paper for the blind user to process by touch.

3.2.4 AudioGraf

The AudioGraf (Andrea 1996) system is an audio interface that presents technical node-link diagrams. A user examines and navigates the diagram by passing a pen over a touch panel and the system plays audio sounds for the diagram elements below or near the pen's position on the panel. This navigation enables the user to understand the spatial layout of a diagram. Another feature of the system is to provide sighted persons with a tool to author diagrams. User studies conducted as part of AudioGraf support the idea that it is possible to create interfaces for blind persons to effectively access node-link diagrams.

3.2.5 TeDUB

The emphasis of the TeDUB (Technical Drawing Understanding for the Blind) (Petrie, Schlieder et al. 2002) diagram interface is to import diagrams from different sources. In particular, the interface supports circuit diagrams, certain UML diagrams, and architectural floor plans. The imported diagrams can be exported from UML tools, bitmaps, and vector graphics. The UML tools export a standard XML format that TeDUB can parse and create the diagram from. For the other diagrams, researchers are developing automatic tools to recognize the nodes and links of the diagram that are displayed in the interface.

A blind user accesses the diagrams with the TeDUB Diagram Navigator. A screen reader reads the textual information displayed in the GUI. In the first two fields the user can select a diagram and the elements in the diagram. The other fields display the details of the selected diagram element.

3.2.6 DocExplorer

The main objective of DocExplorer (Ishihara, Takagi et al. 2006) is to automatically identify nodes and links in PowerPoint diagrams, which consist of simple drawing elements. Besides identifying the diagrams, DocExplorer provides an accessible interface for the diagram. Integrating the project into PowerPoint is practical to easily expand accessible content for the blind community because PowerPoint slides are a common medium for sharing diagrams.

3.2.7 BATS

The Bats (Blind Audio Tactile Map System) system (Parente and Bishop 2003) presents map information at the level of states or countries. The user passes a cursor over the map and the program speaks textual information or plays sounds corresponding to objects at or near the cursor's position. Textual information includes descriptions of landmarks, such as the name and population of a city. The user hears sounds appearing to come from surrounding landmarks, such as traffic from cities or flowing water from lakes, rivers, etc. The sound is spatial to help give the user a sense for the location of objects. For example, on a map of North Carolina, when the cursor is at the location of the arrow, the user hears car traffic coming from Durham and water splashing from nearby Jordan Lake.

The Bats research team evaluated several devices as cursors. A touch panel seemed the most intuitive for a user to touch and move around the screen; however, the user's arm got tired from having to hold it above the touch screen. The user might have an easier time with a Tablet PC, where the user can rest his arm. Another cursor device was a trackball. It was successful because the user could be comfortable with it and judge relative distances and directions by how he turned the ball.

Yet another device is a game controller, such as from a Playstation or Xbox. The concept is to use commodity devices that are readily available and inexpensive. The controller also had tactile feedback from a vibration unit. The tactile feedback enriched the information conveyed, for example, as the user moved across a boundary the user would detect it by feeling a vibration.

3.2.8 Plumb

The researchers of the PLUMB project (exPLoring graphs at UMB) (Cohen, Yu et al. 2005) intend to make diagrams accessible to students with visual impairments. The project focuses on navigating the spatial layout of network diagrams on a Tablet computer with a pen interface. The projects innovation is to design techniques for using the Tablet pen to navigate the diagram by tracing along links connecting nodes.

3.2.9 Kekule

The Kekule system (Brown, Pettifer et al. 2004) is an audio interface used to represent chemical compounds, which are a category of node-link diagrams. Atoms are nodes and bonds are links. The interface helps a user effectively navigate the diagram of a chemical compound through a hierarchy of levels of detail. Researchers argue that the different levels help the user comprehend the chemical. The user can start at the highest level, the entire compound, and progress into the lower levels that provide more detail, such as the amino acids, and the atoms that make up the amino acid.

Kekule is a java application with an audio and visual interface. The audio interface is controlled entirely by the program, which uses Java's text-to-speech to speak textual information. The visual interface gives sighted users feedback as to where the blind user is in the chemical; the current level of detail and the selected element are highlighted. The user controls the interface with keyboard commands.

3.2.10   Proof Checker

The Proof Checker project (Stallmann, Balik et al. 2007) was originally created to teach finite-automata to college students (see **Error! Reference source not found.**). The project is expanded to make the interface accessible to students with visual impairments. The accessible interface features keyboard shortcuts to access and edit details of the finite-automata, such as state values and input values on transitions.

3.2.11   GraphViz and SugiBib

GraphViz (Gansner and North 2000) and SugiBib (Eichelberger 2002) take a different approach to creating diagrams. The diagrams are defined in textual scripts and then the

visual representation is automatically generated. The original purpose of both projects is to research automatic layout of node-link diagrams. Graphviz is intended for general node-link diagrams. SugiBib specializes in software engineering UML diagrams.

3.3 Types of displays

Deep View complements several other projects addressing the accessibility of diagrams. A complete accessible diagram interface would incorporate many of the features of these projects including the systems mentioned above. The reviewed projects demonstrate two general approaches to making diagrams accessible; we call these the *one-to-one mapping* interface and *scripting* interface.

3.3.1 One-to-One Mapping

The first approach is a one-to-one mapping of the visual diagram to a haptic or audio interface. Examples are the tactile display, Bats, IC2D, and Audiograf. The interfaces convey the spatial layout of the diagrams. A cursor on the diagram presents the information of the diagram near that location. In practice, a user explores a diagram by moving a cursor over the diagram and the interface displays the elements below the cursor. The advantage of this method is that it applies to many different diagrams, such as pictures, geographic maps, or functional graphs.

Although the spatial information is very important, we do not convey it in our Deep View system. Instead Deep View conveys a diagram's elements and the relationships (links) between the elements.

3.3.2 Scripting

The second approach to creating and viewing diagrams is to script the diagram as in GraphViz and SugiBib. The script is a text document that describes a diagram's elements and properties. A tool automatically lays out the diagram described by the script. Although this method was not designed to be assistive technology, it is readily accessible to a user with a visual impairment because with a screen reader he can access the text in the script.

One advantage of a textual description over a visual diagram is that it is possible to explore more details. In a visual diagram, space limits the amount of detail because

more detail requires more space. In a textual description, the detail does not have to be limited.

3.4 Pointing

Researchers of the diagram interface have not focused on pointing for the purpose of collaboration; however, most interfaces could be modified to support pointing. Pointing for collaboration is similar to a semantic pointer. One person selects an item to point at and the corresponding item is highlighted in the other person's interface.

There are three requirements for the interface and application:

1. A blind user must be able to select items to be pointed at.
2. When the sighted person points at an item, the corresponding item in the blind person's interface has to be highlighted, i.e., brought to the forefront of the user's attention.
3. The items pointed at must be concrete, well defined, and distinguishable from other items. Items must correspond in a sighted person's and a blind person's interfaces.

Interfaces that have a dynamic display can support this. It is trivial for a blind user to select an item in the interface. It is more complicated when the sighted person points at an item. The item has to be brought to the blind person's attention. In general the interface can dynamically switch what it is presenting to display the item pointed at. Details to resolve would be to have a smooth transition to bring the item to the blind person's attention. Also the display would have to provide contextual information to help the blind person quickly identify which item is being pointed at.

Most reviewed diagram interfaces are dynamic. An exception is the relief paper, which is static. For a sighted person to point out something for a blind person, the sighted person would have to guide the blind person's hand to the desired item. If the sighted and blind persons are remote, this is not possible.

3.5 Accessibility implementation

The implementations of the diagram interfaces reflect a range of design choices made in the implementation. The interfaces have in common that, besides the computer, the cost of additional hardware is minimized. If the interface is entirely in software, then a basic computer is sufficient. As for additional hardware, TeDUB and BATS both explicitly state that they aim to use commodity devices, such as joy sticks and mouse controls, so that the devices are readily available and affordable.

Also the implementers have to choose a technique to present the audio display; for example, the computer speaks textual information and uses sounds to indicate information, such as a "bing" notifying the user of a new event. The two possibilities are: for an accessible application to manage the entire audio display internally or have the screen reader read the textual information presented by an application. Most applications, such as word processors, spreadsheets, browsers, etc., make use of the second technique. The applications were not originally designed to be accessible but with a screen reader the applications becomes accessible.

Diagram interfaces such as BATS, Kekule, Audiograf, Plumb, and Proof Checker customize the audio interface internally. Having the application control the audio interface provides implementers the flexibility to make novel innovations not possible with a screen reader. Bats, for example, uses spatial audio to give the user a sense of the displayed item's location. It is also useful to customize the visual display. In Bats the visual display is a graphical map and the information the user hears is not visually displayed on the map

The other technique for designing the audio display is to have the screen reader read out all information, such as with TeDUB and DocExplorer. The TeDUB interface consists of standard GUI widgets. The screen reader reads the textual information in the widgets. The blind user is already familiar with navigating the interface because it is similar to other applications controlled with a screen reader. Also for a sighted person it is easier to recognize how the interface is used because the blind person's actions are

reflected in updates to the widgets. Although the interface is visual, it does not present the information in a useful manner for a sighted person.

3.6 Summary

The Deep View interface is similar to the scripting approach. Like the scripting approach, our interface displays textual information about elements in the diagram. However, the text is not in a flat text file, where a user has to navigate the file by searching the text for the desired information. Our interface is hyperlinked, which makes navigation between linked objects easier. This navigation is similar to that of Kekule.

Table 3-1. Chronological order of reviewed systems for blind users

System	Year
Rendezvous	1990
Suite	1991
NIST	2002
IC2D	2002
Bats	2003
Tower of Hanoi	2004
TeDUB	2004
Kekule	2004
Plumb	2006
DocExplorer	2006
Proof Checker	2007

Chapter 4

Fundamentals of communication for collaboration

In the previous sections of the background we have reviewed many interfaces that support collaboration. Our emphasis was on how the interfaces integrate the workspace and interpersonal space to facilitate communication between the collaborators. In the case of the deaf/hearing pair, the collaborators could point and gesture at the shared workspace. In the case of the blind/sighted pair, the interface provided each user with a custom display of the shared workspace; i.e. a visual interface for sighted collaborators and an audio interface for blind collaborators. Throughout, we have only mentioned that these interfaces support communication between the collaborators. In this Chapter we will explain in detail what it means for collaborators to communicate. This will help us understand how the Facetop (for deaf/hearing pair) and Deep View (for blind/sighted pair) interfaces support successful communication.

We turn to psychologist Herbert Clark's thesis (Clark 1996) on using language to explain how people communicate with each other. Clark's comprehensive theory is based on his research in the past 30 years and other research spanning the last 100 years. The theory is particularly appropriate for this discussion for three reasons. First, the theory encompasses a large range of situations in which people communicate, including remote collaboration. Second, the theory includes not only verbal communication, but also visual communication, for example from gestures. This helps us explain the communication between the two pairs of collaborators we are researching. Third, the theory explains how the shared workspace is used to communicate.

Clark refers to communication by the term "using language." Two or more people use language to accomplish a primary goal. In Clark's explanation: to accomplish the goal, a speaker *initiates* an exchange that the addressee *identifies* (initiates and identifies

will be explained later in detail). Examples of language use are: a comedian entertains an audience, a teacher instructs students, and debaters try to convince each other of their point of view.

One of Clark's propositions for his work is that face-to-face conversation is the basic setting for language use and all other language use is a derivative of this. His reasoning is that face-to-face conversation is the most common, natural and easiest way for people to communicate. Children learn their first language through it

Communication between the deaf/hearing pair and blind/sighted pair use special techniques and procedures to accomplish the same face-to-face exchange as between people without disabilities. By seeing each other, the deaf/hearing can communicate visually with gestures. To converse with each other, they can write notes, or use a signing interpreter.

For the blind/sighted pair the basic collaboration is similar to a phone conversation. If the collaborators want to make use of the shared workspace they need a special technique. If the collaborators are side-by-side, they can make use of physical objects that both people can touch and refer to, i.e. point at. If the collaborators are remote, they can use a system like Deep View to access the same shared workspace through customized, sensory appropriate interfaces.

In CSCW research, several researchers have applied Clark's theory of language use to their situations. Clark intended his explanation to apply to a wide range of situations: from an informal discussion between friends to language use at very formal events, such as a wedding ceremony that is heavily scripted. The theory also covers writing, where the writer communicates with the reader. CSCW researchers have used Clark's explanation of using language to explain collaborators' communication in virtual reality, mobile computing, and videoconferencing settings. The research projects corresponding to each setting have been described in previous sections of the background.

4.1 Language use in computer science

We consider language use between people communicating as different from the usual use of language in computer science. The difference is in how context related to the

language is used. Language use in computer science is based on the product tradition (Clark 1996, pg. 56). This is a linguistic approach, where words and sentence structures are studied for their meanings. The sentences are considered independent of the context in which they might be spoken or written. A large basis for this work is Noam Chomsky's work on generative grammars. In Computer Science the product tradition has been very useful in defining programming languages to control computers. Instead of ignoring context, the language is fixed to an abstract model of a computer such as a Turing machine.

In this dissertation, the language use we consider is in the action tradition. It is based in the fields of psychology and social science. In this use of language, context and behavior of participants is the main information for interpreting the meaning of what people speak or write. The context carries a wide variety of information that clarifies spoken ambiguities and conveys unspoken messages.

4.2 Using language overview

Before describing how people use language to communicate, we give a summary of the subsections and their relevance to this dissertation. A note on terminology: Clark emphasizes that communication is an interaction of all people involved in the interaction. Therefore many of his terms use the word "joint".

Following is an overview of the subsections. The titles follow those that Clark uses:

1. **Joint action**: Joint action is Clark's term describing the overall interaction that two or more people participate in, hence the people are referred to as participants. They key aspect to a joint action is how the participants coordinate the content presented to each other and the delivery of the content.

2. **Common ground**: Participants' actions in a joint action is based on the context of their surroundings. Clark refers to the participants' awareness of the surrounds and each other as their common ground.

3. **Signaling through gesturing**: Besides communicating through conventional language (verbally or by signing) Clark describes how people communicate through gestures.

Clark's description of using language is particularly relevant to this dissertation. His general description of how people communicate also encompasses how a person without a disability would communicate with a person with a hearing or a visual impairment. It also encompasses computer mediated collaboration between remote collaborators. Joint actions and common ground are the foundation of communication. The foundation encompasses vast information that is redundant and does not require sight or hearing to obtain the information. The subsection on gesturing applies specifically to communication between a deaf/hearing pair because gesturing is one of the main mediums they can use. The discussion of gestures also highlights the kinds of information blind persons do not have access to.

4.2.1 Joint actions

In a joint action people come together to complete a greater common objective than they could accomplish separately. To complete the common objective, however, they rely on each other. As an example consider Alberta, a customer at a grocery store trying to find canned olives. Alberta can engage in a joint action with Benjamin, a clerk at the store available for assistance, to find canned olives. The joint action is very simple: Alberta approaches Benjamin and asks, "Can you please tell me where the canned olives are?" Benjamin completes the joint action by answering "aisle 6" and maybe taking her there.

This is a very simple joint action, and we will use it to illustrate the processes and premises to complete it. At the highest level of abstraction, Alberta and Benjamin (*participants* of the joint action) had to coordinate their actions. Coordinating their actions is known as the coordination problem. There are two ways the participants have to coordinate their actions. First they have to coordinate the *content* that is the intentions of the participants. Second, they have to coordinate the *processes,* such as taking turns speaking. These two ways are discussed in the next two subsections.

Coordinating content

In our example, Alberta's question is the content that coordinates the joint action.

The question is a cue for Alberta and Benjamin that sets the stage for the joint action they will complete together. Benjamin can use the question to coordinate his answer. Note that the cue leads to a unique solution -- in our case, Benjamin answers the question. More generally, Clark refers to the cue as a *coordination device.* Later we summarize the different forms coordination devices take.

A coordination device provides a solution to the coordination problem. The ideal solution follows the principal of joint salience (Clark 1996, pg. 67):

> **Principle of joint salience**: The ideal solution to a coordination problem among two or more agents is the solution that is most salient, prominent, or conspicuous with respect to their current common ground

In our example it is most salient for Alberta to direct her question at a store clerk, who would have the answer. It is part of Alberta's and Benjamin's common ground that Benjamin is a store clerk as is indicated by his uniform.

Alberta's question also provides Benjamin with the necessary information he needs to complete the joint action. That information is summarized in two corollaries of the joint salience principle: the solvability premise and the sufficiency premise. The solvability premise is (Clark 1996, pg. 68):

> **Solvability premise**: In a coordination problem set by one of the participants, all of the participants can assume that the first party:
> 1. Chose the problem,
> 2. Designated its form,
> 3. Has a particular solution in mind, and
> 4. Believes the participants can converge on that solution.

Related to the solvability premise, Benjamin can use the question to formulate his answer. He can assume that as Alberta asked the question the answer is intended for her and he can direct it to her.

The sufficiency premise is (Clark 1996, pg. 69):

Sufficiency premise: In a coordination problem set by one of its participants, the participants can assume that the first party has provided all the information they need (along with the rest of the common ground) for solving it.

Alberta's question provides all the information Benjamin needs to answer it. Specifically, he knows she is looking for the canned olives. Also, being in the given store, Benjamin can assume that Alberta was just looking for the olives in the store. So Benjamin can answer, "aisle six" and can possibly take Alberta there.

Examples of coordination devices

Coordination devices are the concrete mechanisms used to solve the coordination problem between two or more participants in a joint action. They convey the intentions of the participants and fulfill the principle of joint salience. Coordination devices can be communicated in a variety of ways, although most commonly they are expressed through language (verbally, signed, etc.).

The most common solution of a coordination problem is a convention. Conventions are a community's solution to recurrent coordination problems. A convention has five properties:

1. a regularity r in behavior
2. partly arbitrary
3. that is common ground in a given community C
4. as a coordination device
5. for a recurrent coordination problem s.

Consider the western convention of shaking hands to solve the recurrent problem of greeting between participants. In the western community it is common ground for people to use it regularly.

Language (verbal or signing) also uses conventions for coordination. The conventions are intended to determine what a speaker means and an addressee

understands. Lexical and grammatical rules are the basic conventions for creating sentences to express ideas. There are conventions for how words and phrases are used; for example, it is conventional to say "bless you" when someone sneezes. There are conventions in perspective; for example, in America the first floor is level with the street and in Europe the first floor is one level above the street.

A language convention, however, is not sufficient to cover every case. Following are some cases and how they are covered:

- Ambiguity: resolve the meaning using the salience of the situation; for example, zero can have multiple meanings when used as "I met a zero" or "It's zero outside".
- Contextuality: Determining the meaning might require contextual information, for example, some familiarity with computer science terminology is necessary to make the following understandable, "Rushed to complete the test and get at least partial credit, I *core dumped* for the last question."
- Indexicality: Knowledge of the current discussion is necessary to identify the professor in the utterance, "That professor is my advisor".

Beyond language there are non-conventional ways of coordinating to account for specific instances. People can make an explicit agreement, for example, for a one time meeting. Another example is precedent; in writing, terms such as "let us call this…" clarify a definition for the rest of the writing.

External events can also be coordination devices that direct a conversation. For example, two people who are conversing witness an accident, and the accident is the coordination device for the people to switch the conversation to react to the accident. Clark refers to these events as perceptual salience.

Coordinating processes

Coordinating processes involves coordinating how participants deliver the content. The delivery includes the timing with which a participant speaks and gestures. It

also includes how the participants coordinate taking turns. For the participants, it is a matter of continuous coordination.

A discussion between participants is broken into phases that are alternating as participants take turns speaking. These phases have the follow characteristics: Phases can range from balanced where each participant contributes equally to unbalanced where one speaker dominates the amount of speaking. Phases can be periodic or aperiodic. It could be periodic, for example, when people have a casual conversation. It could be aperiodic, for example, a lecture where the presenter speaks and occasionally the audience asks a question. Phases are usually synchronized when only one person speaks at a time and the speakers do not overlap when speaking.

Overall, participants are very accurate in coordinating the phases. The timing participants use to take turns reflects their mental processes. The usual case is for a participant to follow another participant immediately. This suggests the second participant is able to immediately understand the first participant when he finishes. On the other hand, if it takes longer to respond it suggests that the respondent needs time to think about what was said. Then again, the respondent might interrupt the first speaker, suggesting the respondent already understands the point before the first speaker finishes.

4.2.2 Common ground

All interactions (joint actions) with people are set in the context of their surroundings. Context provides a wealth of information so that, for example, in a conversation people can get directly to the purpose of the discussion without much additional explanation. The starting point for a discussion is based on the context of the participants and their situation.

Common ground refers to the contextual information that participants of the joint action continue to have for an understandable exchange. Common ground is the contextual information all participants interpret similarly. Otherwise, there is confusion when one person makes a reference to something that another person has interpreted differently.

Another perspective on common ground is that it deals with the self-awareness of the participants. A participant must have a self-awareness of his surrounding context and awareness of how other people will interpret that context.

To illustrate common ground, consider the earlier joint action in the grocery store. Before Alberta and Benjamin even start conversing, they have a common ground about each other's roles as customer and clerk. They come to this common ground by observing the context of the store and making assumptions from similar past experiences. The common ground information is so rich that Alberta and Benjamin can complete the joint action in less than 15 words. In fact, coordination in a joint action relies on common ground; it is explicitly expressed in the principle of joint salience.

Definition of common ground

In most cases, common ground and context do not need much explanation because they are intuitively understood. We use it naturally all the time when interacting with others. A precise definition of common ground, however, will be helpful in analyzing how the interfaces we design support common ground. We can identify the common grounds that the interfaces support and can possibly identify other information to add.

To explain the definition of common ground, we will use in example of a joint action continuing in the grocery store. Once Alberta finds the canned olives, she purchases them at the checkout from the cashier. In the process, Alberta hands the cashier a $5 bill to pay for the canned olives (costing, say, $1.50). Alberta and the cashier have a lot of common ground, however, for this example we focus on the $5 bill. Alberta's action with the bill indicates to her and the cashier that this is how Alberta is paying for the canned olives; it is part of the convention of shopping.

The example illustrates the *shared basis* definition of common ground. The definition is (Clark 1996, pg. 94):

> **p** is a common ground for members of community **C** if and only if:

> 1. every member of **C** has information that basis **b** holds;

2. **b** indicates to every member of **C** that every member of **C** has information that **b** holds;

3. **b** indicated to members of **C** that **p**.

In the example, proposition **p** is that Alberta is paying for the canned olives. The community is Alberta and the cashier. The basis, **b**, is the $5 bill that Alberta is handing the cashier.

A derivative definition of common ground expresses the participants' mental representation of the situation, that is the participants' awareness of the situation. It eliminates the mention of the shared basis. This is the reflexive definition of common ground (Clark 1996, pg. 95):

> **p** is a common ground for members of **C** if and only if:
> (**i**) the members of **C** have information that **p** and that **i.**

In our example, Alberta and the cashier believe **p** and believe that the other person believes it too.

Of the two definitions, the shared basis definition is more fundamental than the reflexive definition. Besides the participants believing the same proposition, the belief has to be based on the same basis. If not, one participant's assumption of the other is based on incorrect information. This might lead to confusion between the participants as they continue to converse.

An issue with these definitions of common ground is that of containing self reference. The self reference in the case of Alberta is: Alberta is aware that she is handing the $5 bill and that she has this awareness. In traditional logic self-references are not allowed because they lead to paradoxes, such as Russell's Paradox and a liar's Paradox. Self-references, however, are part of certain logics.

Categories of common ground

Commons ground comes in many forms. In this subsection, we review three categories of common ground:

1. **Initial common ground**: Facts and assumptions that participants presumed to have of each other at the start of a joint activity.

2. **Current state of the joint activity**: The current state the participants assume the joint activity to be in.

3. **Joint activity specific**: Events and actions that have happened so far in the joint activity.

It is in the interest of the participants to establish the largest common ground possible. The more information that is available, the more they can refer to it efficiently using, for example, verbal shorthand. They can also explore the topic of the discussion in more depth. If the participants require more information to be common ground than they have, they have to establish it by taking time to discuss and explain it.

Initial Common Ground

Before participants even start a joint action, they already have a wealth of common ground they can use. The common ground comes from the setting of the joint action, the participants' roles, and the participants' affiliated communities.

Given a setting and the participants' roles, participants have the common ground to assume the possible goals of a joint action. For example in the grocery store, it is assumed that the customers purchase goods with the help of the clerks and cashiers. The joint actions the participants pick are related to the shopping topic.

The communities that participants belong to determine how the participants can interact. Communities can be nationality, profession, hobbies, language, disabilities, etc. Participants belong to multiple communities; however in a joint action, a person can use only information that participants understand of a shared community. If a participant is an outsider to the community, that person's information will be much more vague than an insider's information about the community. People are very quick and accurate at assessing other participants' affiliations at the start of an interaction.

A given community can provide the following information about its members:

- Human nature in the most general sense applies to all communities, such that we live on planet earth, experience gravity and other Newtonian forces, and have basic senses (although some senses might be disabled).

- Lexicons determine the kind of language used, such as dialects, jargon, or slang

- Cultural facts, norms, procedures: For example a national community shares common background (e.g. history), customs, and social roles

- Ineffable background information is that obtainable only by personally experiencing it. Deaf and Blind communities have many of these experiences. It is difficult for an outsider to these communities to gain the same experience. Although an outsider could simulate the experience for some time, it would be difficult to have a full appreciation of having the disability all the time.

People working together from communities with disabilities and communities without disabilities have to take advantage of multiple communities. Related to the abilities/disabilities community, people with and without disabilities are outsiders to each other's communities. This could possibly lead to misunderstanding, such as a hearing person making a gesture that would be considered impolite in the Deaf community.

The people have to take advantage of other communities they belong to. For example, two employees at a company working on a project can exploit these communities they belong to. They have detailed information about the culture of the company and background on the project.

Current state

A physical model is a powerful way to represent the common ground information related to the current state of a joint activity. Participants can view and manipulate the physical model. For collaborators working through computer mediation (the topic of this dissertation), a shared collaborative application is such a physical model.

To explain the features of a physical model, consider again that Alberta is purchasing a cart full of goods from the cashier. The main part of the physical model is a typical grocery store checkout counter with a conveyor belt that Alberta puts the goods on and another conveyor belt that the cashier places the rung-up items on. Also the cash

55

to pay for the goods is part of the physical model. Let's refer to the items to be purchased and cash as markers in the physical model.

Markers contribute information about the common ground in three ways. First a marker's position has meaning; items to be rung-up are on one conveyor belt, and afterwards on the other conveyor belt. Second, markers are easy to manipulate and thereby reflect a corresponding change in the common ground. Third, participants can simultaneously access the physical model either to make changes or direct their attention to a desired part.

Overall, the physical model is reliable and an effective memory aid. It is reliable because as markers are manipulated all participants can easily view the change. Also, if the participants are distracted from the joint action, when they return to the joint action, they can scan the physical model to remember what the current state of the joint action was.

Joint activity specific

Throughout a joint activity, new common ground is established or built on other common ground. A participant's actions perceived by other participants become part of the common ground; that is, everyone can refer back to it. This common ground, however, is specific to the current joint activity and future joint activities based on the current one. Consider, for example, participants designing a solution to a problem, the participants might have discussed three possible solutions. In the progress of the discussion the participants can easily refer back to them possibly by an agreed upon name or index.

For joint action participants with and without disabilities, they can lay out conventions at the start. A deaf person and hearing person can agree on basic gestures for the most often repeated actions. A blind person and a sighted person can agree on the state of the shared physical model (maybe the sighted person explains it to the blind person).

4.2.3 Signaling

In this section, we describe signaling, which Clark (Clark 1996, pg. 155) defines as the deliberate human act with which a speaker means something. When the speaker executes the behavior to present a signal, it is referred to as signaling. A deliberate human act is intentionally named because the action can take many forms: spoken, gestured, or auditory (intonation when speaking or simply making noises).

The many forms of deliberate actions are linguistic (relating to language), although they may not seem so. The most studied form in linguistics is speaking, that is using words, sentences, etc. However, gestures – such as pointing with a finger and saying "I want this one" – also contribute to an addressee's understanding of what a speaker says. In this example, pointing identifies the specific object that is wanted. Clark points out that using the finger to point is a method of signaling. This method of signaling in itself is non-linguistic.

In the following subsections we discuss three categories of signaling: describing-as, indication, and demonstrating. For now, narrow examples to distinguish between them: describing-as is signaling by speaking; indication is signaling by pointing with a finger; and demonstration is signaling by imitating something that is intended to be communicated. Each subsection begins with an abstract definition that encompasses all the possible forms signaling can take: spoken, gestured, and vocalized or sounded out. When a speaker uses the signals, they are usually composited and the speaker's presentation is accurately timed.

Signaling is significant to this dissertation because it involves communication techniques available to deaf or blind collaborators. On the other hand, the description identifies communication techniques that are not possible because of the disability. From the techniques available for communication, we can reason about the kind of conversations that are possible.

Describing-as

Describing-as is the method of signaling that includes speaking and writing. The definition of describing-as is conveying meaning by using objects that are meaningful because of rules assigned to them. In writing and speaking, words are objects with

definitions. The organization of the words in a sentence follows rules that convey the meaning of the sentence.

Describing-as can also be signaled through gestures and auditory sounds. Ekman and Friesen (Eckman and Friesen 1969) named these emblems. An example of an emblem that is a gesture is a nodding head; in this case, the rule is that this action means "yes". Emblems can stand alone, such that after nodding, it is not necessary to still say "yes".

Kendon (Kendon 1981) identified three ways in which emblems are used. First, emblems can be used for interpersonal control, such as waving hello or placing a finger in front of the mouth to signal "be quiet". Second, emblems can be for personal states, such as thumbs up for "I approve" or shrugging shoulders, for "I don't know". Third, emblems can be for evaluation, such as circling a finger around a person's ear to indicate "He is crazy."

It is also possible to have auditory emblems. For example, clapping indicates "I approve" and hissing indicates "I disapprove."

Indication

A speaker uses indication to identify in space and time something he is talking about. The indication is a kind of index for which Clark sets the following requirements:

1. Attention: The index is in the participants' joint focus of attention
2. Location: The index locates the object in space and in time.
3. Physical connection: The index locates by means of the physical connection with the object.
4. Description: The object is specified under a particular description.
5. Computability: The speaker presupposes that the addressees can work out 1-4 based on their current common ground.

To illustrate these requirements, consider someone pointing a finger at a book, while saying "I would like this book." The person's action of pointing draws the participant's attention to the location where the book is. Participants can identify which

book, possibly because the finger is closest to that one. The description in this case, is that the speaker would like the book. All this takes place within the participants' common ground.

More generally, indication is accomplished with an instrument and locative action. Examples of instruments are a person's fingers/hands/arms, eye gaze, or head. With these instruments, a person can locate something (a locative act) by pointing, moving, or nodding, respectively.

A person's voice can also be used as an indication. In this case, the instrument is the voice and speaking is the locative action. For example, if a lecturer asks, "who would like to volunteer." A person answering, "I would", indicates themselves. The person's voice meets the requirements for an index. It draws the other participant's attention, locates the person, and identifies the person.

Indication can also happen through a person's actions. Clark gives the example of a clerk asking a customer, "Can I help you?" The customer indicates what they need, for example, by placing the items he wants to purchase on the counter to be rung-up.

In all the ways indication is used, the temporal placement of the signal is crucial. Indication has to be executed with an understanding of what it corresponds to. A unique example demonstrating this is when an addressee is giving a speaker feedback while the speaker is speaking. The addressee can for example agree by nodding his head or say "yes" just after the speaker completes the part the feedback is intended for.

Demonstration

Using a demonstration to signal provides depicted information to the subject being discussed. People might choose to use a demonstration, because it is easier than expressing it using words. Examples include showing how someone walks, maybe with a limp; or mimicking how someone speaks. Another example of a demonstration is a person indicating, for example, the size of a box by gesturing in the air – one hand, at opposite corners of the box.

Iconic gestures are a specific kind of demonstration. For example, while explaining directions, a person might gesture left and right turns. In this case, the gesture is iconic for turning left or right. It is debated whether common gestures are necessary for

language or if they simply function as an aid for the speaker to find words; that is if a person momentarily cannot think of the word, the gesture will help jog your memory. Clark argues that these are essential, because the discussion would be less vivid without them; he suggests, explaining how to tie a bow tie without gestures would be more difficult.

Facial and vocal gestures can also be demonstrative of emotions. By making a face, a speaker conveys the emotion that corresponds to what they're saying; for example, a speaker can frown while telling a sad story. The intonation and the speaker's voice express the degree of his emotion; compare, for example, a casual "oh" to a surprised expression of "oh!" The expressed emotions can be independent of or dependent on the speaker's current feeling as in the first and second examples, respectively.

Impact on communication

A hearing or visual impairment makes it difficult to take advantage of all the techniques for signaling. The availability of the technique for a person with a hearing or visual impairment depends on whether the technique is visual or auditory, respectively.

Describing-as: Fortunately with describing-as, in some cases, the same meaning can be conveyed visually or orally; for example, nodding head "yes" or saying "yes".

Indication: Poses difficulties for people with either a hearing or visual impairment. For the most common form of indication – pointing with the gesture – sight is necessary to see the gesture and hearing is necessary to hear the corresponding description.

For a person with hearing impairments, there are few cases where the gesture and description can be communicated visually; for example, in playing checkers, a player can point at a checkers piece and demonstrate where to move it.

Demonstration: Deaf or blind persons can take advantage of certain kinds of demonstrations. A deaf person can easily use visual gestures that do not require an additional description as is the case with an indication. A blind person can make use of the demonstration in the speaker's voice.

4.3 Summary

In our research, we incorporate three concepts from the theory of using language in the design and evaluation of the collaborative computer interfaces. Features in the computer interfaces enable the collaborators to accommodate for one person's hearing or visual impairment. The first concept is the common ground that comes from a shared workspace. Given an accessible shared workspace, collaborators can independently confirm the current state of a collaborative task's status. The second concept is signaling; as collaborators discuss the collaborative task they will use the interface to reference (signal) elements in the shared workspace that collaborators without disabilities might reference both visually and verbally. The third concept is for the collaborators to find their own pattern to coordinate their actions of accessing the shared workspace and discussing the task. On the other hand, collaborators without disabilities might simultaneously access the shared workspace and discuss the task.

Chapter 5
Collaboration between a deaf person and a hearing person

We conducted a user study to evaluate computer interfaces designed to support remote synchronous collaboration between a deaf person and a hearing person. As would be typical in the deaf person's workplace, we consider the case where the hearing participants do not know ASL. We also assume that an ASL interpreter is not available because interpreters are not always available in the workplace. The difficulty for the collaborators is to converse fluently because the hearing person does not know ASL and it is not possible to converse verbally. The collaborators have the advantage of seeing each other and the shared workspace.

In our user study we compare the experience of a deaf person working with a hearing person (DH pair) to two people without disabilities (HH pair). We hypothesize the following:

> *DH pairs will rely on video conferencing to communicate as opposed to HH pairs who will make minimal use of the video.*

The DH pairs might benefit from using the video conferencing to express themselves through gestures similar to being face-to-face. We substantiate the hypothesis by analyzing the collaborators' experiences completing the tasks. We categorize and log gestures participants use from video taken during task performance. Furthermore for the checkers task, we collect low level information about checkers moves and participants' mouse activity. Participants also complete a post-experiment questionnaire.

5.1 User study design

Our user study is an introductory exploration to understanding how DH pairs communicate when collaborating remotely using a computer system to mediate the communication. Therefore we evaluate the collaborators experience in a variety of situations, which results in a factorial user study design. In summary, collaborators complete the task of playing checkers against the computer or creating a brainstorm diagram. We compare the experience of two hearing collaborators (who we refer to as a HH pair) to that of a deaf person and a hearing person collaborating (who we refer to as a DH pair). Participants use separate computers with a shared workspace and tools for communicating with three configurations, including video conferencing, a telepointer for pointing and gesturing at the workspace, and chat messaging. We make a within subjects comparison of the collaborators experience using the three configurations.

5.1.1 Independent variables

Figure 5-1 shows the three interface conditions we evaluate in the user study. The conditions are explained in more detail below.
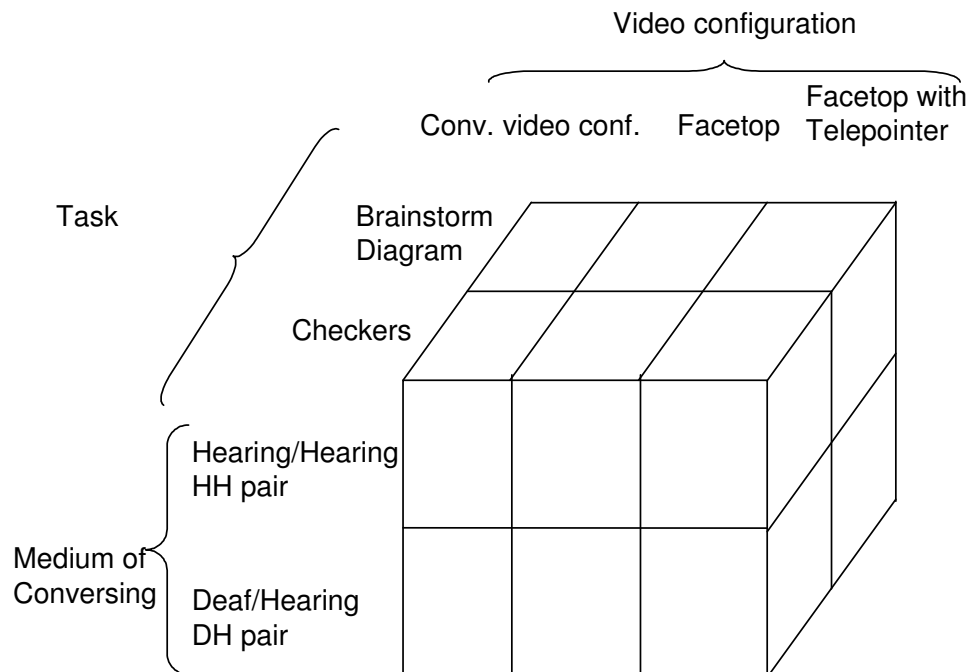
Figure 5-1. Independent variables of user study

Video conferencing and telepointer configuration

In the study there are three video conferencing configurations in conjunction with a telepointer, with which participants point and gesture at a shared workspace. In all cases, the video is high resolution and with minimal latency so that the video is detailed and responsive allowing participants to easily recognize the other participant's gestures.

The first configuration is a conventional video conferencing setup. As shown in Figure 5-2 (a), one participant's screen shows a 3-by-4 inch video of each collaborator. A participant sees himself to verify the camera is setup properly. The video is on the same screen next to the shared workspace application. This video conferencing configuration is always used with a telepointer.

a)   b)
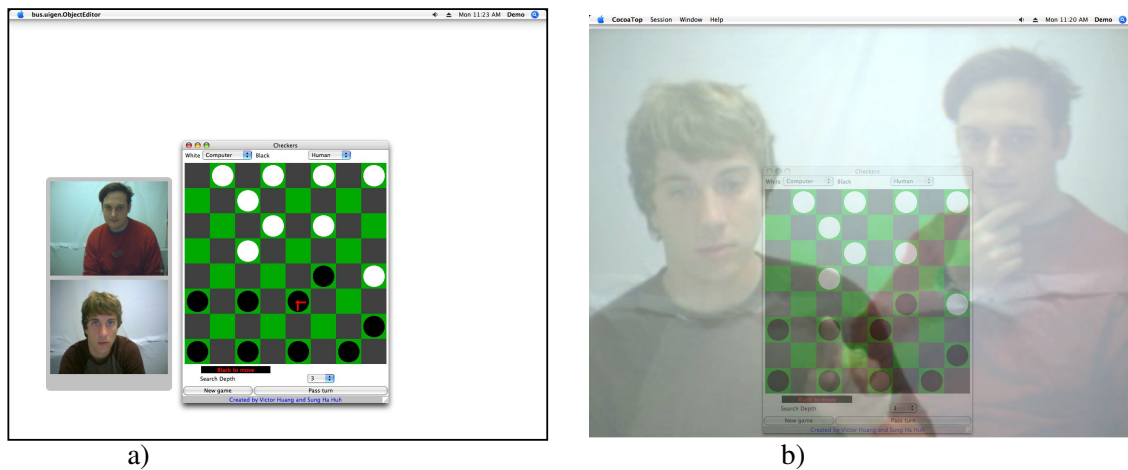
Figure 5-2. a) Conventional video conferencing configuration b) Facetop video configuration

The second video conferencing configuration is the Facetop (Stotts, Smith et al. 2004) setup, shown in Figure 5-2 (b) In Facetop, video of participants is transparently overlaid on the entire desktop. Participants can easily distinguish the participants' video from the shared workspace. Through the Facetop video (unlike the conventional video), participants can gesture and point at the shared workspace similar to how they would when face-to-face. A participant sees his own video for simple self registration of the camera and to identify where his finger gesture points. The latency of the video is low enough for a participant to easily gesture at the workspace. This is important because a

significant lag in the video would cause a participant to readjust as the video catches up with the participant's motion.

In one study situation participants use the Facetop video with a telepointer and in another study condition participants use the Facetop video without a telepointer. Without the telepointer, participants point with their fingers at the workspace through the overlaid video, similar to being face-to-face. With the telepointer, participants have a choice between pointing with the telepointer or through the video.

Checkers and diagram tasks

In the user study pairs of participants complete one of two tasks. One task is playing checkers against the computer. The other task is completing a brainstorming diagram. The tasks encourage collaboration because completing the tasks requires the participants to exchange ideas and make decisions. We chose the tasks to represent different kinds of interaction.

In the checkers task, a pair of participants play checkers against the opponent controlled by the computer. On each turn, the pair decides their checkers move. The pair must evaluate their pieces and the opponent's (computer's) pieces. Participants mainly exchange ideas pointing at the checkers board. Conversing is helpful but not necessary. The pointing interaction limits the ideas participants' can express and does not reflect a more general collaboration.

In the diagram task a pair of participants group words into categories. At the start, participants are given a topic (hobbies, family, food) and given a moment for each to independently brainstorm five words. Then the participants share their words and decide how to group the words into categories. The visual diagram the pair creates is a tree structure with the main topic as the root node. The initial words might be categories in the tree or leaf items.

Unlike the checkers task, the diagram task represents a more realistic collaborative task. The negotiation is an open ended creative process. Participants have to converse in order to group words into categories. Participants have the flexibility to choose how to communicate; such as editing the shared whiteboard or by exchanging chat messages.

The checkers task is more suitable than the diagram task for conducting a controlled experiment. By making checkers moves, participants are rapidly repeating the interaction of deciding on a move. We have instrumented the checkers application to gather low level information about participants' checkers moves and mouse activity.

5.1.2 Measurements

In the user study we make the following measurements:

1. Questionnaire and interview participants: Gather the perspectives and experiences of the participants completing the user study. Although the data is subjective, the participants' perspective is the most significant because people similar to them will ultimately use the interfaces.

2. Video, Audio, chat recording of sessions: Used to make observations about the collaborators' interaction. The interactions will be coded and tallied.

3. Logging events in shared application: The monitored activities can be tallied as another metric to categorize the interaction. Analyzing these low level measurements has to be done with care as there might be several possible explanations for a given result.

5.1.3 Participants

The study involved 26 participants – six HH pairs and seven DH pairs. Seven participants (in each of the seven DH pairs) are deaf or hard of hearing and do not understand spoken language. The hearing participants in the DH pairs are familiar with the deaf community and have some understanding of the situation of the deaf participant. Hearing participants who know ASL were asked to refrain from using it in order to simulate the scenario we are evaluating.

For this study the experimenter did not know ASL to communicate with the deaf participants, however, we prepared two videos to convey the informed consent participants completed and instructions of the tasks. One video is a closed captioned Quicktime video created with the Magpie closed captioning project. The second video is an ASL video we created. The video included a screen capture of Facetop when

explaining the shared workspace applications. Using Facetop, it was beneficial that the ASL interpreter could sign and point at the application. The remaining discussions between the deaf participant and the experimenter were mediated by someone who could sign or simply writing written or chat messages.

5.1.4 Computer setup

Two participants complete a user study session. Figure 5-3 shows the setup of the user study.
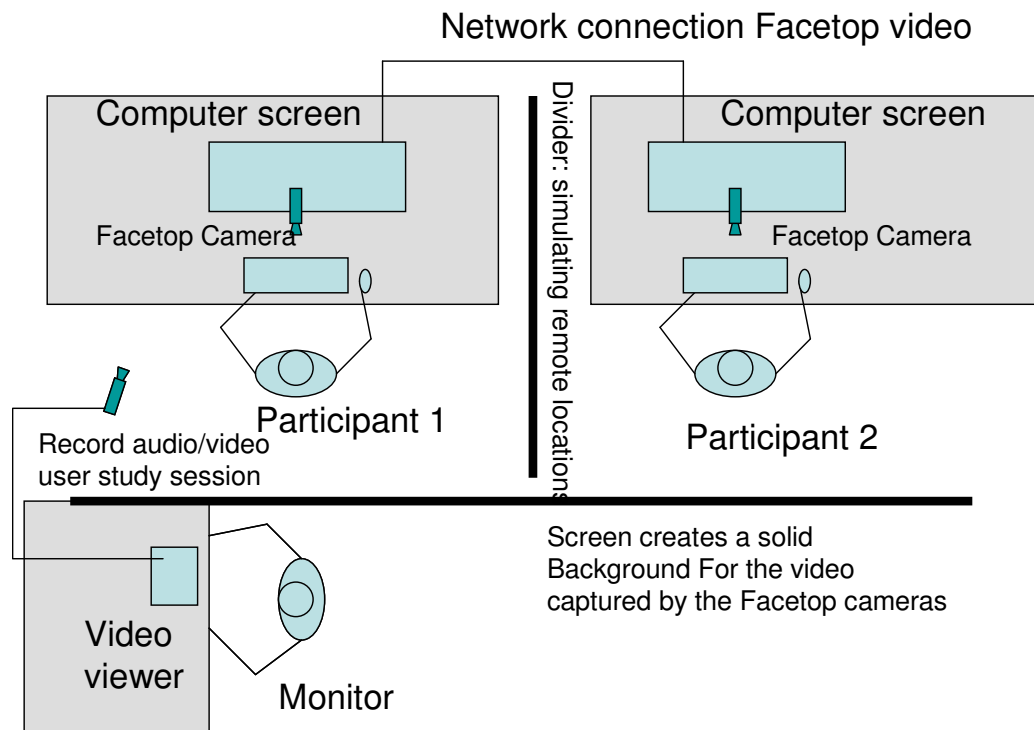
Figure 5-3. Facetop user study setup of workspace with two participants and user study experimenter.

The room is set up to simulate the participants being remote. There is a divider between them so that they cannot turn to each other as in face-to-face communication. Being in the same room the audio quality is ideal for a HH pair; unlike when the audio is transmitted and the signal is delayed or distorted. The participants still wear a headset with microphone. Nothing is heard in the headset but the microphone is used to capture the audio recorded on the video tape of the session.

Also there is a white screen background behind the participants. The background helps make the video image of the collaborator easier to view and understand. Without the background the video would be cluttered with an image of the room. The screen also happens to separate the user study experimenter from the participants; the participants will be less aware of and less influenced by the experimenter's presence. The experimenter observes the live video image on a video display connected to the camera recording the session.

5.2 Results

The results compare the HH pairs and DH pairs collaborative experiences. This includes evaluating the quality of collaborators completed artifact. We also identify the collaborators' strategy when using the shared workspace and video to communicate.

5.2.1 Sessions quality of tasks and interactive communication

We consider a session successful based on two factors. The first factor is whether the participants have an engaging session exchanging ideas. The purpose of the study is to evaluate the ability of the computer interfaces to support communication. The second factor is the quality of completing the task. The communication is meaningful if participants can deliver good quality results.

One might assume the HH pairs would outperform the DH pairs because HH pairs can easily exchange ideas in conversation. Overall the HH and DH pairs were able to complete the tasks. By some measures, HH pairs performed better than the DH pairs.

Overall all pairs, with the exception of one DH pair, were able to successfully communicate and exchange ideas. In the post-experiment questionnaire, pairs of HH and DH participants self report a successful collaboration. Participants report enjoying the sessions. They also felt each person in the pair contributed equally to completing the task. The experimenter's subjective observation is that participants made smooth progress with engaging communication. Participants seemed to have a good understanding of the task, the other person's actions, and effective communication. HH pairs conversed verbally whereby DH conversed with chat messaging.

Session quality details related to each task follow. Evaluating and comparing task quality of the DH and HH pairs is difficult to quantify. We use available measures to approximate the session quality.

Checkers game quality

For checkers, we use logged measurements from the checkers game to evaluate the task quality. One measure of success is counting the number of obtained checkers Kings. Obtaining a king means the participants have maneuvered their piece past the computer pieces. Throughout the game HH pairs were able to obtain twice as many kings as DH pairs (1.5 kings for DH vs. 3 kings for HH as shown in Figure 5-4). It is statistically significant by *p<0.0562.* In two games played by DH, no Kings were obtained, however, the same groups obtained several kings in other games. By this measure the HH pairs performed better than the DH pairs.
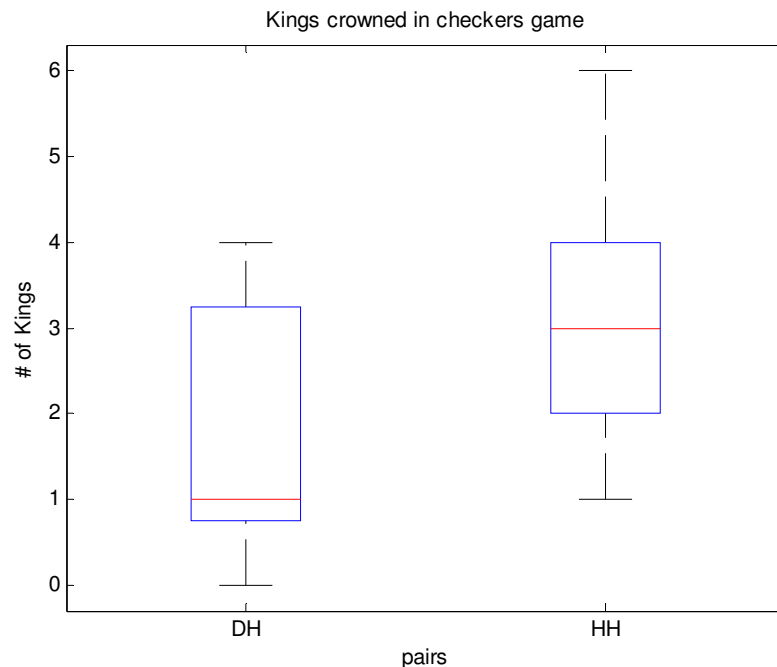
Figure 5-4. Kings crowned in checkers games (N= 9 DH games, 9 HH games)

Another measure is total number of moves shown in Figure 5-5 are grouped by HH and DH sessions. HH pairs had 40% more moves than DH pairs. Games averaged 39

moves and 55 moves for DH and HH pairs respectively. The difference is statistically significant by *p<0.0670.* More moves might suggest the participants were able to avoid being taken by the computer. The longest games by HH pairs taking 81 and 92 moves, however, does not necessarily indicate a better checkers game. In these cases, many of the moves were made in the final game stages, where the participants were moving their kings to evade the computer. The evasion prolonged the game but did not result in an advantage.
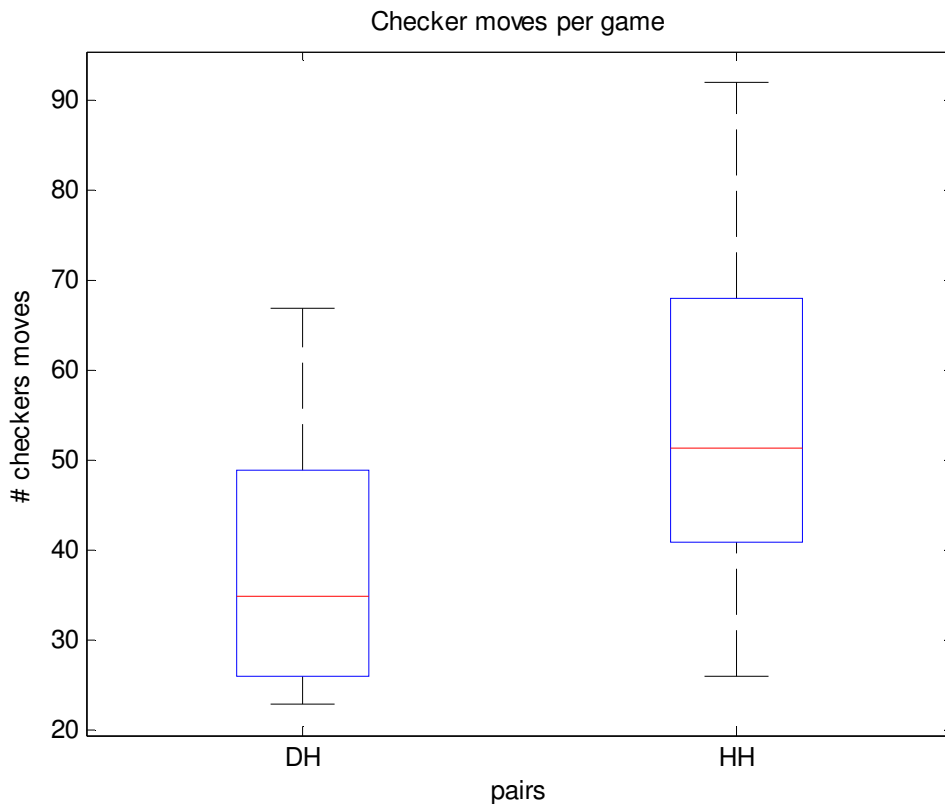
Figure 5-5. Number of checkers moves per game (N= 9 DH games, 9 HH games)

It seems the task quality is similar for DH and HH pairs. Defeating the computer in checkers was a matter of having a fortunate situation – luck. Of 21 checkers games, one HH pair and one DH pair won games against the computer. By subjective observation it is not obvious that one pair was better than the other. Overall, participants were able to progress to the final stage of a checkers game where the participants and the

computer each have one or more Kings. At this stage it is difficult to pin down kings and it is a matter of luck if all the computer's kings can be taken.

Communication was smooth and consistent for both HH and DH pairs.

Diagram quality

Before discussing the results of the diagram task, Figure 5-6 is an example of a complete brainstorming diagram. The main topic is family, the root of the tree. The leaf tree items are the words the participants chose. As in the example, most diagrams have 3 levels and occasionally four. To complete the diagram, participants chose the categories for the words, visually layout the diagram and drew the connecting lines. Coloring the terms was optional; in the example the participants went the extra steps to color the categories green.
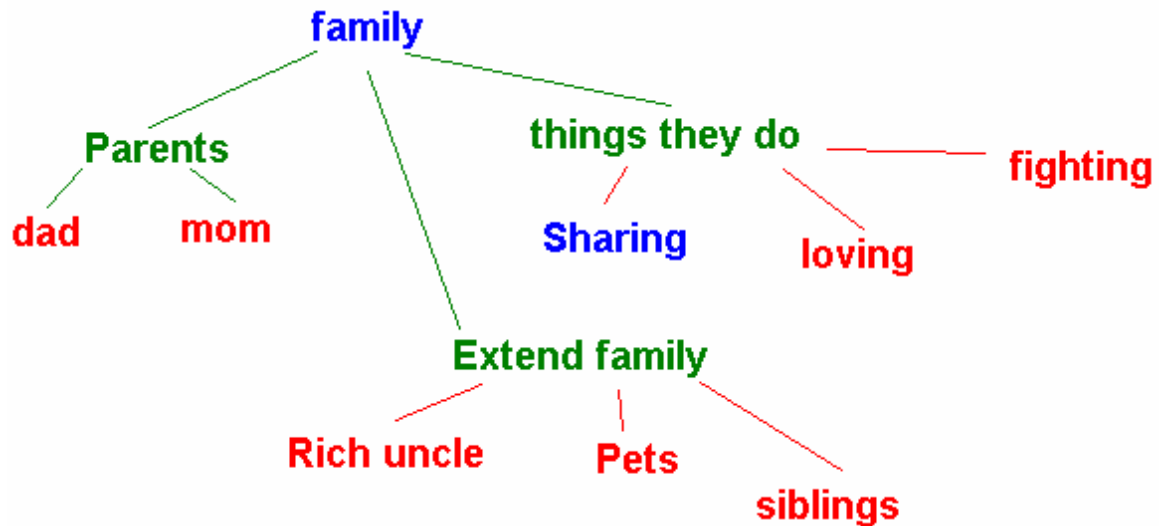
Figure 5-6. Brainstorm diagram with the main topic hobbies.

The results of the diagram task were inconsistent. The three HH pairs completed the task successfully. In the four DH sessions only one pair completed the task successfully as instructed. Figure 5-6 was created by the successful DH pair. Two other pairs completed the tasks, however, relied on communicating in ASL instead chat messaging as requested by the experimenter. Although use of ASL is not part of this study, the experience gives insights about the Facetop interface. The fourth pair communicated via chat messaging, however, they were not able to complete the task

71

successfully. Therefore the discussion of the results focuses on the one successful DH diagram session. Some observations about the diagram sessions follow.

One DH pair was not able to complete the diagram task successfully. The pair created brainstorming diagrams, however, they did not negotiate the categories. Instead, each person completely separate portions independently and simply agreed to the final result. It seemed the participants were hesitant to start a discussion and find a strategy. In contrast, other pairs communicated more often as they discussed one category and progressed to another. It is for further research to investigate improve the social environment to foster more interaction.

Compared to the HH pairs, the DH pairs took significantly longer to complete the diagram task (about 2 times longer). The time to complete a diagram is taken from the time participants complete entering their words into the share workspace until the diagram editing is complete. HH pairs were able to complete the task faster because of the ease of communicating in a familiar medium (verbally). Figure 5-7 shows the distribution of times for HH and DH pairs to complete the diagrams (9 and 12 diagrams for HH and DH groups respectively).
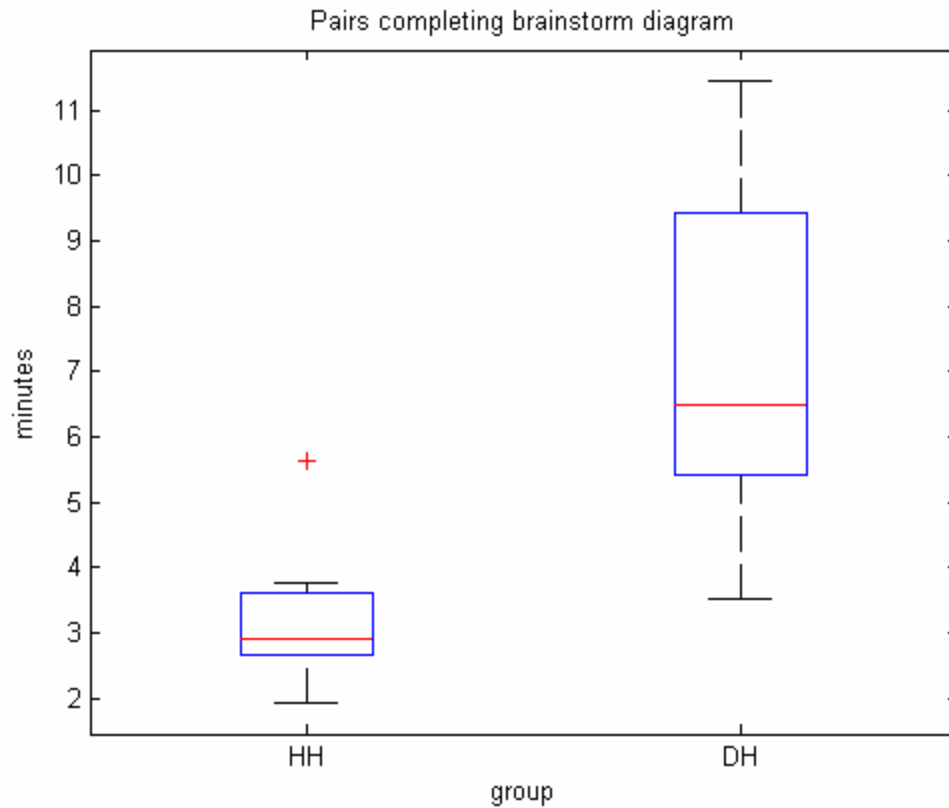
Figure 5-7. Time to complete brainstorming diagrams (N= 9 HH diagrams, 12 DH diagrams)

5.2.2 Conversing and workspace usage

In this section we summarize pairs' overall interaction strategy of conversing, verbally or via chat, and using the shared workspace (checkers board or whiteboard for diagrams). Although HH and DH pairs communicated differently, their overall strategy is similar. Our observations apply to the checkers and diagram tasks. Other tasks would illustrate different forms of interaction. Besides conversing via chat, DH pairs communicated through gesturing in the video. We report those results in the following section.

We divide Communication in the session into three categories.

- Discussing ideas verbally or through chat;
- Coordinating actions, such as deciding a checkers move; and
- Discussing ideas specific to the task, which was completed through the workspace.

73

Checkers

In the checkers task, the main communication is in deciding the next checkers move. HH and DH pairs mainly suggested moves visually through pointing at the checkers board (i.e. shared workspace). Participants point at a piece and gesture where to move the piece. Then participants have to agree on the suggested move or evaluate an alternative move. HH and DH pairs also used a similar divided up the task in a similar way. One participant was designated to control the checkers games (i.e. clicking on the board to move pieces). Both participants were involved in deciding the move.

Although conversing differed for DH and HH pairs, the time per move was comparable. Figure 5-8 is a box plot of time to complete moves grouped by HH and DH pairs. Most moves were completed in less than 10 seconds. In these short moves, participants agree to a suggested move. HH pairs are more likely to have longer sessions lasting as long as 30 seconds.
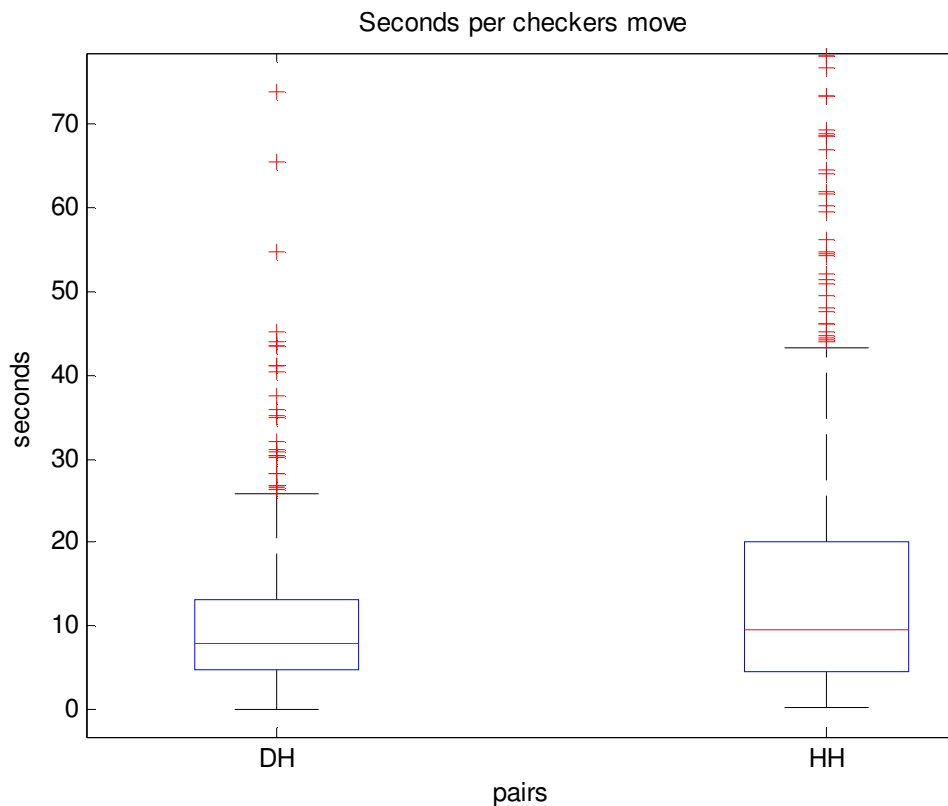
Figure 5-8. Seconds per checkers move by DH and HH pairs (N= 9 DH games, 9 HH games).

Conversing (verbally or via chat) ranged from simple to more elaborate exchanges. Simple verbal utterances annotated the visual pointing of the move, such as "move this piece to here". The visual information was sufficient for DH pairs to convey the same move information. After one participant suggests a move, the other participant agrees or disagrees to it. HH pairs verbally communicated their decision – either to agreed or disagreed. As will be explained in more detail, DH pairs communicated their decision visually by nodding or otherwise gesturing through the video.

More elaborate conversing involved deliberating about checkers moves. HH pairs would discuss and try to anticipate the computer's next one or two moves. They would discuss different possibilities. They would also give reasons for deciding a move.

DH pairs did not deliberate to the same level of detail as HH pairs. The pairs could deliberate on several options. Within one turn, participants might consider several moves. If one participant turns down the other participant's suggestions, the participants consider an alternative until they agree. Discussions via chat were rare. Chats were used occasionally for special cases, for example when the computer made a double jump. The participants used chat to clarify the event to each other. Other chats related to task related details about checkers.

Diagram

In the diagram task participants have to first decide on categories and then decide which category to add a word to. HH pairs can easily discuss categories verbally. DH use chat and the workspace to exchange these ideas. Some categories are discussed with a chat exchange. Alternatively, a participant makes a suggestion by directly editing the workspace. The other person indicates his/her agreement with the suggested category. The HH pairs' deliberation of categories, however, is more in depth than the DH pairs. While discussing categories the HH participants sometimes elaborate on their personal associations with the word. The association might change the final category.

The workspace is also useful for dividing the task among participants, where participants work independently on separate parts of the workspace. In some cases, after a HH pair verbally agreed on the categories, they decide to work separately on drawing different categories.

The DH pairs divided the task differently than the HH pairs. DH participants worked on different parts of the same category. The participants divided the tasks of placing the categories, associated words, and drawing connecting lines. As one person edited the diagram, the other person could work on a different part. Rarely did participants edit the same part but when they did, the participants found the accident amusing. This suggests that the workspace helped participants be aware of the other person's actions. Participants made edits to the diagram to avoid an overlap with the other person.

The DH pairs that communicated well and less well contrast how the chat messaging was used while using the shared workspace. The communicative DH pair had a good conversational flow switching between chat and the workspace. On the other hand, the less communicative DH pair was not as fluid with chat. In a few cases one person wrote a message that the other person did not notice until shortly after. One possibility to improve the communication would have been for one person to attract the other participant's attention by gesturing in the video.

5.2.3 Gestures

DH relied heavily on gestures for making decisions. The gestures were conventional, including nodding head, shaking head, or shrugging shoulder suggesting agreement, disagreement, or indecision respectively. The gestures we report in the results were used frequently and consistently in all DH checkers sessions and the one successful DH diagram session. In comparison, HH pairs did not rely on gestures and communicated similar ideas verbally. The following discussion of gestures applies to the DH pairs only and not to the HH pairs.

Positive acknowledgments

Participants used the positive acknowledgement most. For example one participant nods his head to agree with the other participant. As expected in a typical collaboration making fluent progress, a participant suggests reasonable ideas that the other participant agrees to. Likewise HH pairs usually agreed to initial suggestions.

76

A head nod gesture was the most frequently used to communicate a positive acknowledgement. Acknowledgement gestures occasionally included an OK hand gesture or thumbs up gesture. If the hearing person in the DH pair knew ASL, he/she sometimes used the ASL "yes" gesture, a raised bobbing fist.

Acknowledgment gesture in checkers task

Figure 5-9 shows the frequency of positive acknowledgements for the three DH checkers sessions and the three games per session. The number of acknowledgements is normalized by calculating gestures per checkers move to account for the games ranging from 24 to 67 moves. On average, there is half a gesture per move (or one gesture every two moves. Gestures per move varied widely for the game where DH pairs used the Facetop interface. In one case it is coincidence and the other two anomalies will be explained later.
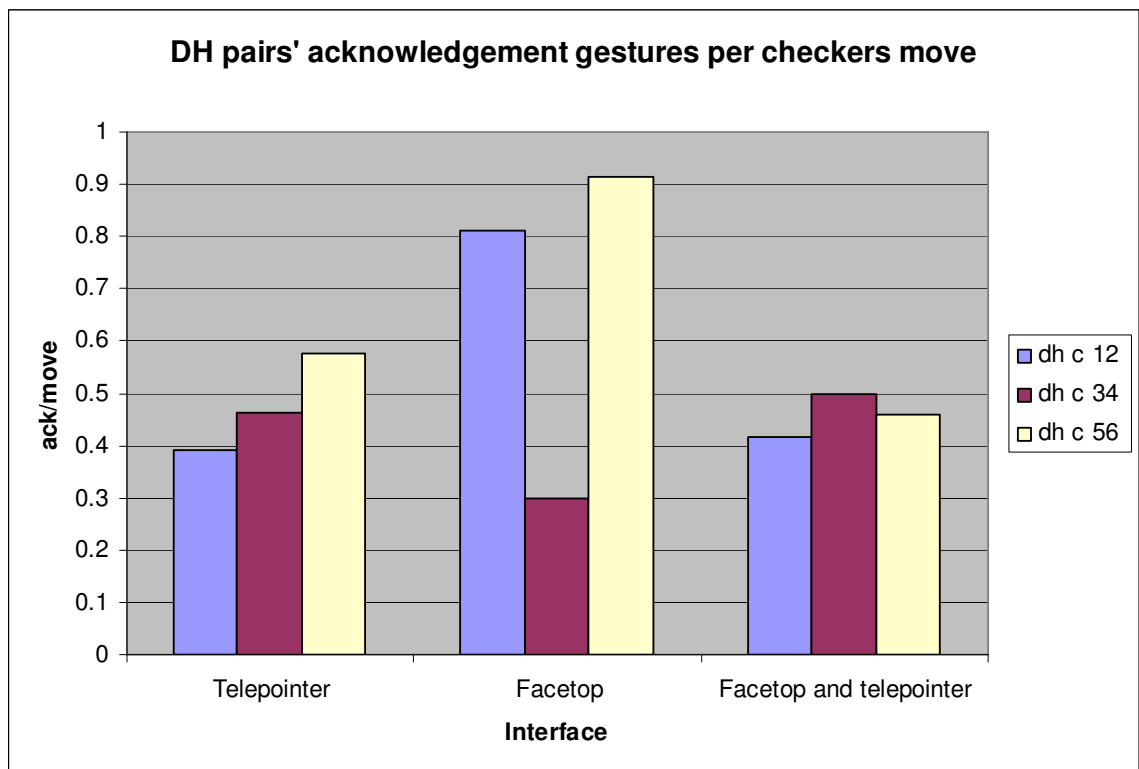
Figure 5-9. DH pairs' acknowledgement gestures per checkers move (N= 9 games)

In checkers, an initial estimate for the average number of acknowledge gestures per move would be one. That is the players agree on a move and execute it. Some moves would have no gestures or more gestures. No gestures are needed when participants agree on a series of moves at once, such as moving a piece forward to make it a king. Other more complicated moves require more deliberation where participants consider several moves. In this case, participants might agree to a suggested move but then the participants change their mind.

Participants can also agree on a move without explicitly gesturing. HH pairs used the same exchanges verbally. In one checkers scenario, one person controlled the checkers application to make moves. This participant can agree to the other participant's suggestion by simply making the move. On the other hand, one participant in a DH pair playing checkers had a tendency to nod after a move was completed indicating his affirmation of a move.

Participants used acknowledge gestures equally. This reflects the balance of the collaboration. Acknowledgements mainly correspond to the other person's suggestions. In checkers 8 of 9 sessions have 35%-50% distribution of gestures between participants. The Facetop dh-C-12 session had a balance of 1:10. This brings us to the anomaly in Figure 5-10.
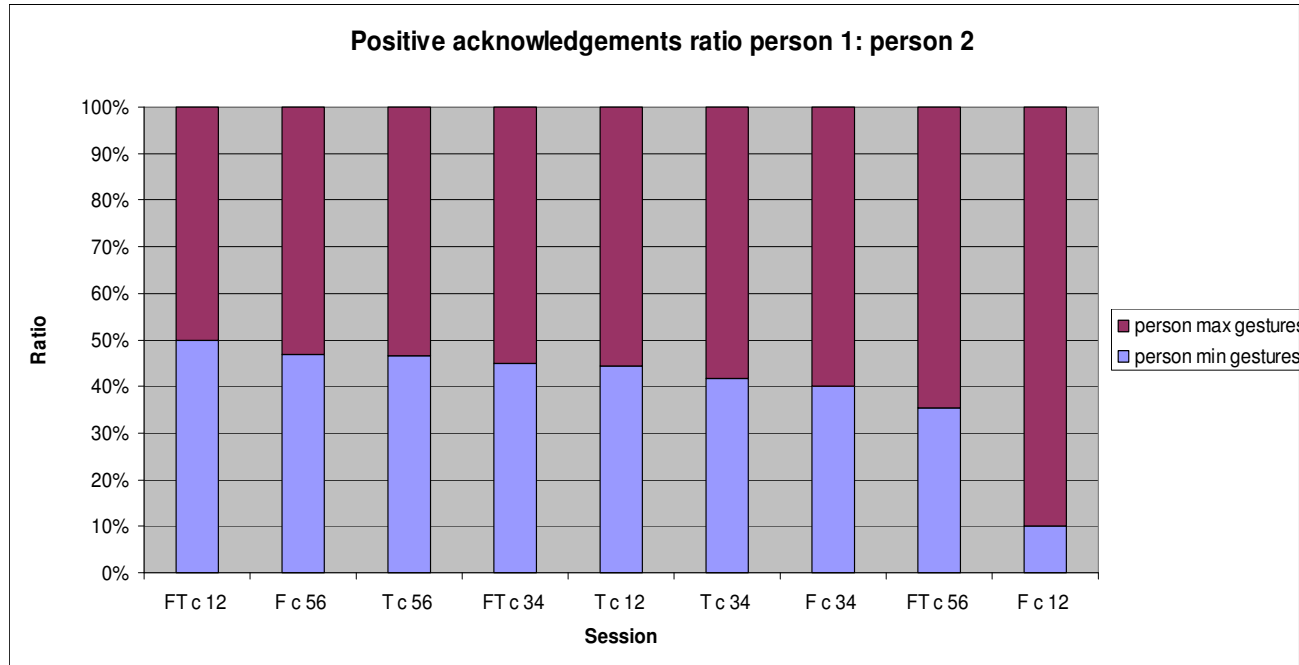
Figure 5-10. Portion of positive acknowledgments by each person in the pair (N = 9 games) (T= telepointer, F= facetop, FT= facetop and telepointer)

The Facetop dh-C-12 session was unbalanced collaboration between the participants. The person controlling the checkers game suggested most of the moves. The other person agreed to them thereby making the majority of acknowledgment gestures. Also the number of acknowledgment gestures (30) corresponds to slightly less than the number of checkers moves (37).

The Facetop dh-C-34 session had the anomaly of the lowest gestures per move (0.29 or one gesture every 3.3 moves). This is the result of the longest DH checkers game (67 checkers moves). In the final stage of the game the participants made a series of rapid moves to avoid the opponent's pieces. Participants understood the strategy and did not have to confirm the individual moves.

Acknowledgement gestures in diagram task

For this section, we only consider the one DH pair that successfully completed the task without conversing with ASL. Like DH pairs playing checkers, the DH pair completing the diagrams also communicated with positive acknowledgment gestures. Participants used the acknowledgement gestures infrequently; it depended on the

situation. As explained, participants otherwise communicated by editing the workspace or through chat messages.

As Figure 5-11 shows, acknowledgment gestures were used about once per minute. The ratio of gestures between participants varied. In the part with the telepointer interface only one participant used gestures. On the other hand, participants used almost an equal number of gestures for the Facetop interface.
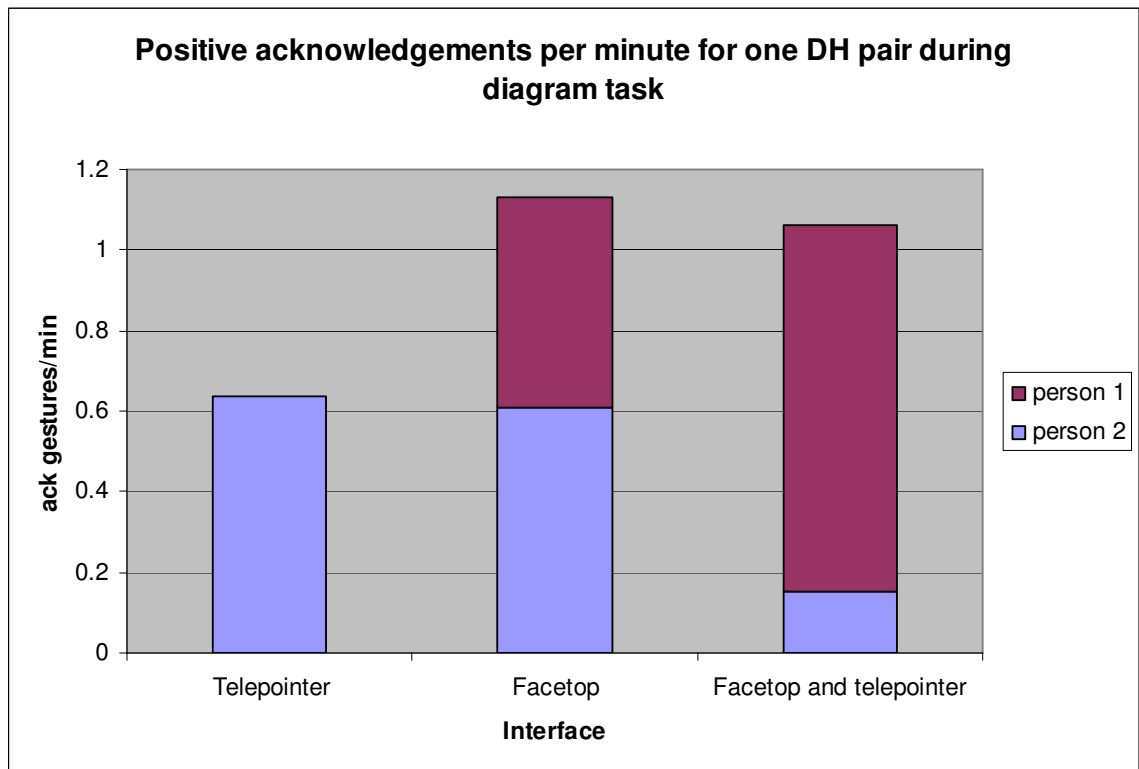
Figure 5-11. Positive acknowledgement used by one DH pair in diagram task

Acknowledgements were gestured in two situations. In the first situation participants gestured agreement to the other participant's edits to the shared workspace. In the second situation participants nodded in response to the other person's chat messages. Of course, participants would also write acknowledgments in chat messages. Figure 5-12 shows the acknowledgement chat messages were 10-30% to all chat messages.
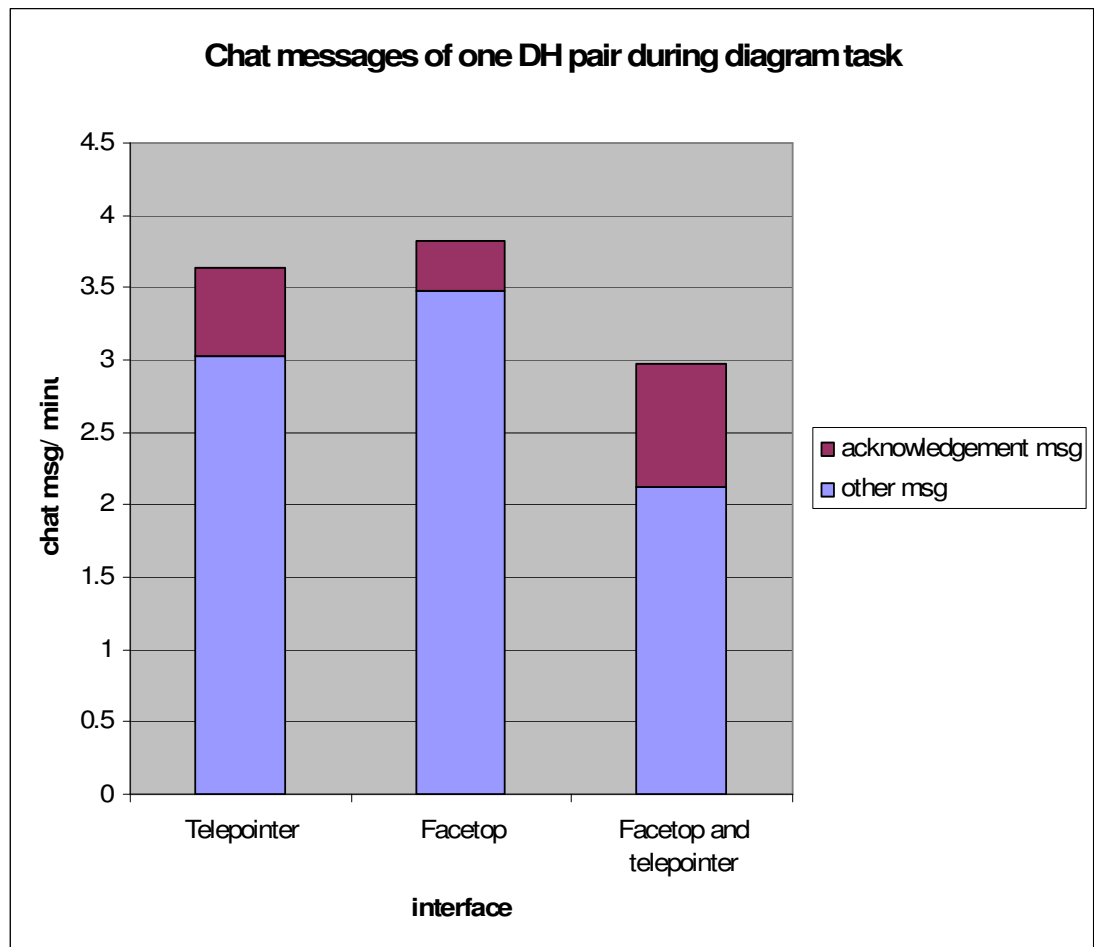
Figure 5-12. Chat messages in diagram session by one DH pair (N= 3 diagrams)

Limitation to positive acknowledgement

A limitation of evaluating a one-sided positive acknowledgment is detecting if one participant perceives the other participant's gesture, for example, in checkers before making a checkers move. Perceived or not, the first participant's response is the same: the checkers move is made. The checker move is also consistent with the second participant as he/she would not necessarily notice if the gesture was overlooked.

A further suspicion that gestures might be overlooked comes from HH pairs responses. In the post-experiment discussion, they report observing the workspace but not the video. One participant commented on using Facetop that his focus was on the other participant's pointing finger and he zoned out the video of the other person's head and

torso. The pointing finger was the essential information he needed to understand his collaborator.

In another case, a HH pair completing the diagram task was particularly animated with gesturing while conversing. In the post-experiment discussion, the participants commented on appreciating the opportunity to express themselves through gestures. Questioned further, however, the participants could not recall particular gestures the other person made. Participants seem to have used the gestures subconsciously.

The following section, however, provides evidence that DH pairs perceived each others gestures in some cases.

Responses to gestures (yes, no, don't know)

Besides the simple positive acknowledgment DH participants used other forms of acknowledgements. In these situations one participant reacts to the other person's gesture providing evidence that participants perceive each other's gestures. These gestures apply to DH pairs completing the checkers and diagram tasks.

DH pairs consistently used three gestures related to decision making. In short, the gestures are a double positive acknowledgment, negative acknowledgment (suggesting disagreement), and shrug of shoulders (suggesting indecision). The gestures were used much less frequently than simple acknowledgements. These gestures were used as the situation arose and depends on the state of the task. These three types of gestures were used slightly less frequently than simple acknowledgments. Figure 5-13 and Figure 5-14 shows the frequency of these gestures. The results for the checkers game in Figure 5-13 are normalized by the number of checkers moves per game. On the other hand, the results for the diagram task in Figure 5-14 are normalized by the time to complete the task.
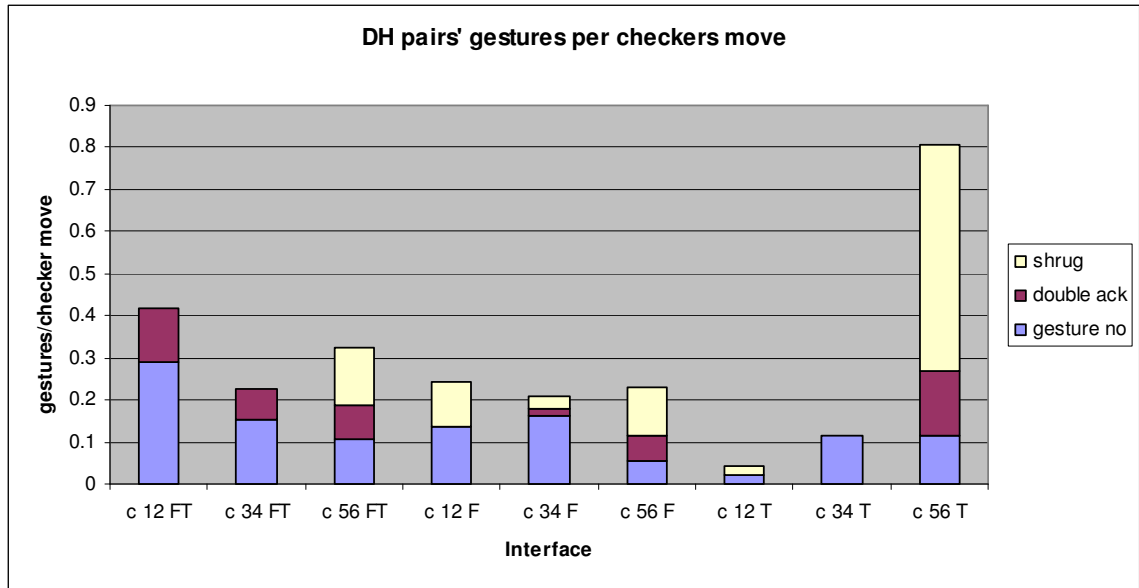
Figure 5-13. Non-positive acknowledgement gestures used by DH pairs in checkers task (N = 9 games) (T= telepointer, F= facetop, FT= facetop and telepointer)
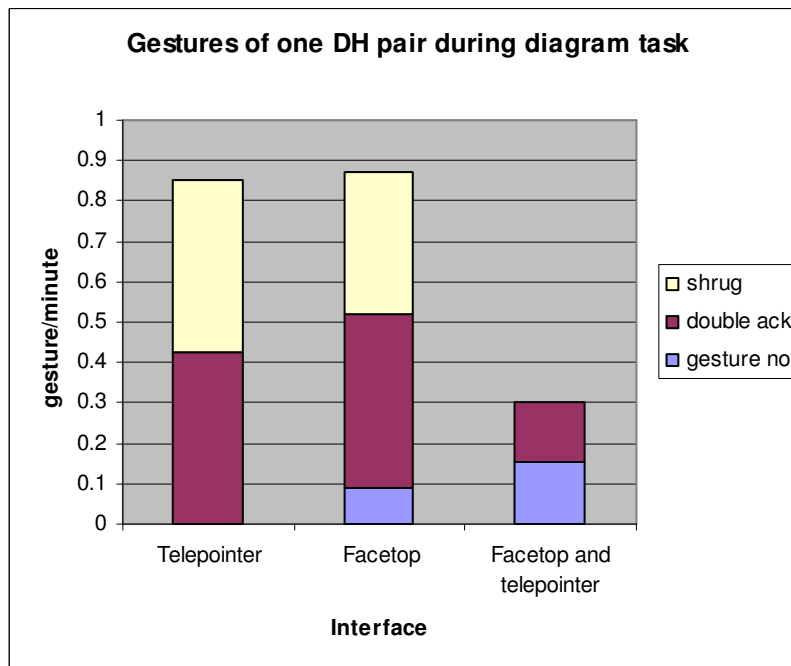
Figure 5-14. Non-positive acknowledgement gestures used by one DH pair in diagram task

The first gesture we refer to as a double acknowledgment. One participant makes a positive acknowledgment, recognizing the positive acknowledgment of the other person. HH participants used a similar acknowledgment verbally, such as in this exchange between participant A and B:

A: Should we move this piece here? [pointing at the move]

B: Yes

A: Ok, [A makes the checkers move]

Figure 5-13 and Figure 5-14 show the minimum double acknowledgments observed by the experimenter. Some double acknowledgements might be overlooked because the exchange between the participants was subtle or too temporally spaced apart to notice.

The second acknowledgement is a negative acknowledgement, for example gestured with a shaking of the head. Occasionally a participant gestured it by raising both hands with palms facing the other participant. The participant whose idea was turned down has to concede the idea and consider another idea. So unlike a positive acknowledgement, the observer of the gesture cannot continue with the suggested move.

The third gesture indicates indecision; for example, gestured by shrugging shoulders. When shrugging, a person raises his shoulders and sometimes raises hands palms facing upward. Shrugs were used in two situations. In the first situation, participants are separately thinking about the task and there is a longer wait. One person will initiate progressing the discussion by shrugging and suggesting: "I do not have a suggestion. What do you say?" The other person might respond with a suggestion. In the second situation, one participant suggests a choice between two options. The other participant responds with indecision, indicating, that he agrees with either option. The other person can respond by making the choice.

5.2.4 Other video observations

The following section describes observations of using the Facetop video. We compare pointing through the video via finger and using a telepointer. Facetop is intended for pointing at the workspace, which is well accomplished with a telepointer.

The advantage of the Facetop video, however is enabling participants to gesture with arms and hands like in a face-to-face discussion. Finally, we comment on issues designing the Facetop video.

Pointing, i.e. making references

One method participants used to reference elements in the shared workspace was pointing. The checkers task depended heavily on pointing in order to reference checkers pieces and board locations. The diagram task depended less on pointing, but pointing was used occasionally.

Our study gives insights into pointing through a computer interface. In the three parts of a session, participants are given three options for a pointing mechanism: using only a telepointer, using only the video to gesture and point at the workspace, or both pointing mechanisms. For this section, the main data is collected through observations in the recorded video, where it is possible to recognize how participants gesture at the workspace. Also for evaluating the checkers task, we consider the participants' logged mouse activity. Mouse activity reflects two uses of the mouse (1) pointing activity and (2) activity to control the checkers game.

Checkers

Participants rely on pointing mechanisms regardless whether it is pointing with a telepointer or gesturing through the video. Participants pointed at checkers pieces and board fields for two reasons. In the first case a participant flicked the pointing device repeatedly between two board locations to suggest moving a checkers piece between the indicated board fields. In the second case both participants would point at the same piece as a confirmation of the suggested checkers move.

Participants preferred pointing with the telepointer rather than pointing through video. In the session part where participants could choose the pointing mechanism, most participants chose to use the telepointer (as observed in the post-experiment video analysis). Also in the post-experiment questionnaire, a majority of participants report that it is faster, easier, and more accurate to point with a telepointer. Some participants report that their arms became fatigued from holding out their arm to point. Supporting the arm,

for example, on a desk, helps reduce the fatigue. A few participants used the video to point but it might have been their curiosity to explore the new Facetop interface.

Pointing through the video, however, can be useful as an alternative to a telepointer. In one situation a participant pointed through the video because the mouse for the telepointer was out of reach. He had shifted his position for comfort and leaned back in his chair.

In the checkers task, we compare pointing through video to an ideal telepointer, with independent pointers always in pointer mode. We suspect an actual telepointer would be less desirable; for example, if participants have to share a single telepointer, as is the case with desktop sharing software. Also participants in the diagram task commented that switching the mode of the mouse was inconvenient. Participants have to switch modes between using the mouse to point or using the mouse to edit the diagram.

Mouse activity

Our measurements of the checkers game give us insights into how participants used the mouse to control the checkers application. The relevant measurements are logs of mouse activity and observations from the recorded video.

Comparing the mouse activity in the telepointer interface to the Facetop interface we can estimate how much the mouse is used as a telepointer to point and how much to control the application. The logged mouse activity does not distinguish these cases because there is only one mode for the mouse cursor. It is obvious that overall mouse activity is reduced when the telepointer is not used in the Facetop interface.

We use an ANOVA analysis on the mouse activity per second to compare the mouse activity between the two interfaces and HH and DH pairs. The mouse activity per second is the total for both participants.

Figure 5-15 shows a box plot with the distribution of mouse activity grouped by the HH-DH pairs and interface. Mouse activity is reduced by 55 and 54 percent for HH and DH pairs, respectively, from the telepointer interface to the Facetop interface. The difference in mouse activity is statistically significant *p<0.0008* for DH and HH pairs.
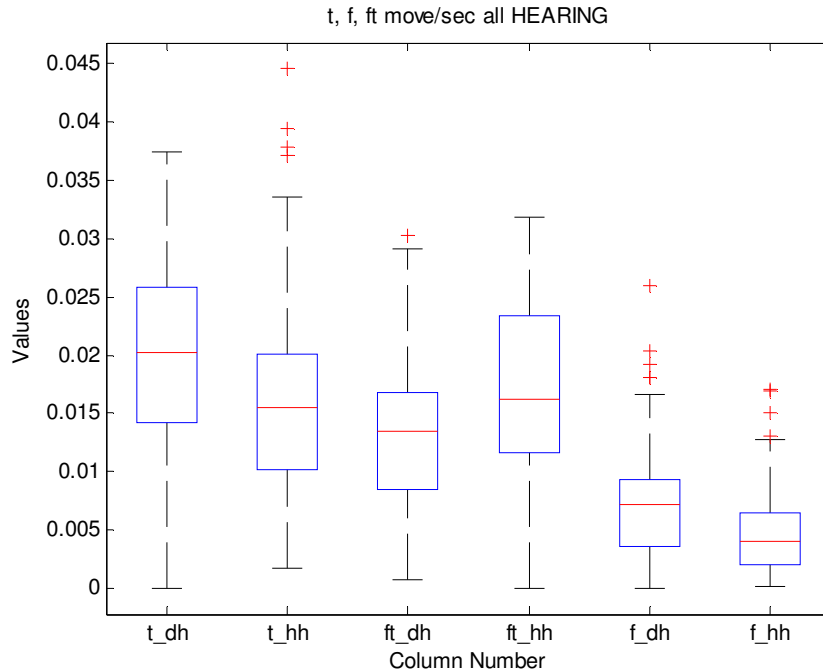
Figure 5-15. Mouse activity grouped by DH/HH pair and ordered by interface (t= telepointer, f= facetop, ft= facetop and telepointer)

Therefore we estimate that in the telepointer interface, about 45% of mouse activity for both participants is for controlling the checkers application. This is equivalent to the mouse activity in the Facetop interface to control the application. We attribute the remaining mouse activity to use as a telepointer by both participants.

Although in the Facetop interface the mouse is only used to control the application, use of the mouse impacts the participants' interaction with the Facetop interface. Observations from the recorded video illustrate how the mouse is used while gesturing at the video. Participants usually designated one person to control the checkers application while the other participant suggested ideas by pointing through the video. There is a disadvantage for the person controlling the mouse. The person controlling the mouse had to switch between pointing via finger and controlling the mouse. In some cases it seemed cumbersome to keep switching. As an alternative, some participants used their dominant hand to control the mouse and other hand to gesture and point. Of course, these observations are limited to the checkers task, which is an extreme case that is very dependent on pointing.

Diagram task

In the diagram task participants rarely made use of pointing. When pointing, it was used to reference words. Words were mainly referenced in conversation, verbally for HH pairs and in chat messaging for DH pairs. As described, occasionally words were referenced by editing or moving them in the shared workspace. Two pairs, however, made effective use of pointing.

One participant in a HH pair used pointing several times to reference words. The participant pointed with the telepointer in the telepointer interface. Without naming the word, he would point at the word and continue to suggest how the word could be categorized. The pointing was smoothly interweaved in the conversation and clearly understood by the other participant.

In another case, a participant in a DH pair used pointing to accomplish a specific task. He referenced words by pointing via finger through the video. In a circling gesture he pointed at a category and two nearby words. The category and words were placed as a group, however, they were not yet connected by lines. With the circling gesture, he was confirming with the other participant that the category and words should be grouped. With the other participant's approval, he proceeded to draw the connecting lines and complete that part of the diagram.

Gesturing

The Facetop interface is suited for using gestures at parts of the shared workspace, similar to being face-to-face. These gestures would be more involved than gesturing with a telepointer, such as flicking the pointer to indicate where to move a checkers piece. In the tasks for the user study, gestures were not used as much as we had anticipated. Participants used gestures in a few isolated instances.

In three instances, participants playing checkers in DH sessions used gestures to discuss a double jump. A double jump is when one side can take multiple opponent pieces if the necessary spaces are free to move into. A participant indicated a double jump by pointing with one hand at the first opponent piece to be taken and with the other hand gestured how the participant's piece would jump twice.

In the diagram task during a HH session a participant used a two handed gesture. The HH pair had completed their diagram and were commenting on it. One participant compared the width and height of the current diagram to the previous two diagrams. While commenting she traced the perimeter of the diagrams with two hands indicating the width and height of the diagrams.

On a lighter side, participants used the video to interact with each other informally. In the initial introduction to the Facetop interface a few participants teased each other by "poking" at the other participant's video image. In one case, at the end of a diagram task, two participants in a DH pair congratulated each other for completing the task by giving a virtual high-five gesture in the video.

Facetop video

A focus of this study is to evaluate the visual nature of the semi-transparent overlaid video used in the Facetop interface. We report on participants' observations as they compare the video of the Facetop interface to conventional video conferencing. Participants could successfully use the Facetop video for the duration of the session. The Facetop video is used in two of three parts. For the diagram task and checkers task the Facetop video is used for about 10 and 30 minutes, respectively.

The experience with two DH pairs illustrates that the video was clear to understand. The pairs independently chose to communicate via signing ASL through the video. They could communicate naturally and clearly. It seems communicating with ASL was more comfortable for them than refraining from ASL as requested for this user study. The experimenter asked them to communicate through chat messaging or general gestures in the video. This was to simulate the situation investigated in the study where the hearing person is assumed to not know ASL.

Despite the successful use of the Facetop video, however, participants overwhelmingly reported in the post-experiment questionnaire that they preferred the conventional video conferencing over the Facetop video. In discussion, participants indicated the video conferencing interface was preferred because the image of the workspace was clear and crisp. With the overlaid video the workspace, though recognizable, is diffuse and washed out.

The diffusion is caused by the melding of the workspace and video image. The transparency level balances the intensity of the workspace and video image. A suitable transparency level varies and depends on the visual content of the workspace, lighting in the room, and participants' background (a white screen in the user study).

Although the workspace and video are easily distinguished, the overlapping video of participants made it difficult to recognize the participants' images. For example, as the image of two faces overlap, it is difficult to identify the participants' expressions. Initially the participants' video is aligned on opposite sides of the workspace, which gives the impression of the participants sitting side-by-side. Over time, participants shift in their seats resulting in the video overlap. Overlapping video occurred particularly for checkers sessions, which lasted about 15 minutes.

Overlapping video, however, was not a distraction. The workspace was still recognizable and at least the participants in HH sessions reported their focus as being on the workspace rather than on the video of the other participant. More generally, in the post-experiment discussion participants in DH sessions commented that the other participant's movement in the video was not distracting.

The drawbacks to the Facetop video interface have inspired our research to improve the video interface. In (Gyllstrom, Miller et al. 2007) we propose image processing techniques to modify the video. For example, the workspace image is preserved by removing the background in the participants' video. In another example, the participants' video image can be processed to reduce details. Of course, the general acknowledgment gestures must still be recognizable for communication. We are also investigating how the Facetop video can be used only when participants desire to gesture. The remaining time, participants could use a conventional video conferencing interface.

5.3 Discussion

A deaf person working with a hearing person (a DH pair) benefit from the video conferencing, however differently than we hypothesized. DH pairs have a unique strategy for using the video conferencing and shared workspace compared to two hearing collaborators (a HH pair). A DH pair uses the video to communicate decisions through common head and hand gestures. The gestures we tallied during the sessions and

compared between DH pairs and HH pairs show that DH pairs used the video frequently while HH pairs make negligible use of the video. We observed that DH pairs exchange ideas by pointing at the shared workspace, editing the shared workspace, or using chat messaging.

Comparing DH pairs' completion of the checkers and brainstorm diagram tasks shows the pairs adopted the general strategy to the specifics of the task. In the checkers task, participants primarily use the telepointer to make suggestions about checkers moves. In the brainstorm diagram task, participants used the chat messaging to exchange ideas. In both cases participants used the video to gesture decisions, even occasionally when using chat messaging.

We compare a DH pair's use of the conventional video conferencing system and the Facetop video system. Overall, participants clearly prefer the conventional system over Facetop mainly because the image of the shared workspace is clear and crisp as opposed to the diffuse image in the Facetop system caused by the overlay video. Nonetheless, in this study it is inconclusive if one interface is better than the other for the collaborators to complete the tasks. In both cases, DH pairs can create a reasonable artifact in a session and the DH pairs use video to gesture acknowledgments.

Also participants prefer to point and gesture using a telepointer instead of gesturing by hand through the Facetop video. Participants self report that it is easier, faster, and clearer to point with a telepointer. In the checkers task, the gestures with the telepointer are sufficient to make suggestions about checkers moves.

Hand gestures made through the Facetop video are not used as we hypothesized. Participants used the hand gestures rarely in limited situations maybe because the hand gestures are ambiguous. The hand gestures that were used are unconventional and the observer has to interpret the gesture relative to the current situation. Hand gestures might be more useful if participants agree upon a convention amongst themselves or choose gestures common to the task.

It is informative that DH pairs complete the task in a similar fashion regardless of the video size. The small video in the conventional video conference system seems sufficient for participants to follow each other's gestures. We assumed participants might overlook the other participant, who is shown in a small video window besides the shared

workspace. Also other research indicates small video is not as useful (need Dave's references). The success of using small video might also depend on the task and video's resolution, latency, or other quality factors.

Future work can focus on understanding how collaborators manage to switch their attention between the shared workspace and video. There are three issues to consider. First, is to investigate the video configuration, such as the size and quality. Second, it is important to consider is the interaction of the collaborators. The collaborators probably switch attention to the workspace when one person proposes a move and on completion of the suggestion switch attention to the video to make the decision. Third, our user study shows that a participant is selective about the video he observes. Using the Facetop video system, hearing participants mention zoning out the video of the other participant except for watching the other participants pointing finger.

More generally, future work can focus on researching a video conferencing system which incorporates video of an interpreter. Having an interpreter is a more complete collaborative environment for DH pairs than our configuration intended to support ad hoc meetings when an interpreter is not available. Including the interpreter likely changes the collaborator's dynamic of taking turns and conversing. The collaborators would also need to find a strategy to point at the shared workspace while commenting on the item pointed at through the interpreter.

Chapter 6

Design of Deep View, an accessible diagram interface

An objective of this dissertation is to research how a blind person and sighted person can collaborate on discussing and editing node-link diagrams. Our approach is to provide each collaborator with the most sensory appropriate interface. The sighted person uses a conventional visual diagram application. The blind person uses the Deep View interface. We designed Deep View specifically to enable the blind person to collaborate with a sighted person. To facilitate the collaboration, the Deep View interface is also a complete solution to making node-link diagrams accessible to a blind person.

In this chapter we describe the design of the Deep View interface that enables a single blind user to access node-link diagrams. Deep View is generalized to support a wide range of node-link diagrams. Besides making diagrams accessible to a blind user, Deep View gives him full control over creating a diagram as would be necessary in collaboration.

Deep View provides several navigation techniques for a user to explore a diagram. The most significant technique is for Deep View to present high-level characteristics of a diagram. An example of a high-level characteristic in a diagram, such as a flow chart, is a loop, which represents a repeated process. We use graph algorithms to automatically detect the high level characteristics.

A significant feature of Deep View is to visualize the node-link diagrams for a sighted user. At the most basic level, a blind user can use a visual diagram to share it with sighted colleagues. We also use the visualization feature to support collaboration between a blind person and a sighted person. Our implementation demonstrates how Deep View can be integrated into a visual diagram application, such as Rational Rose, that is a diagram tool used by software engineering teams. This way a blind person can access and

edit existing diagrams in Rational Rose. Furthermore, a blind Deep View user and sighted Rational Rose user can collaborate using their respective interfaces to edit the same diagram.

6.1 Diagram representation

Deep View is designed to present a wide variety of node-link diagrams. This is possible because node-link diagrams have many common characteristics. Of course, domain specific terminology and details of the diagrams are accommodated in Deep View's textual description of the diagram.

We will use state-charts and Entity Relationship Diagrams (ERD) to demonstrate the range of node-link diagrams Deep View can represent. A state-chart is an example of a straight forward node-link diagram. A state chart is the diagrammatic representation of a state machine consisting of states and transitions between the states that depend on the inputs to the state machine. Figure 6-1 (a) shows a state-chart of the process used to reheat dinner in a microwave. On the other hand, an ERD is an example of one of the most complex node-link diagrams. An ERD captures the relationships between various data in a system. For example, a library system might have books. There are various kinds of ERD notations; we use the Barker's Notation (Halpin 2001), visually drawn with "crow's feet." Figure 6-2 a. shows an ERD of a library system with patrons that checkout books.

The common characteristics of node-link diagrams include nodes, links, and attributes. Nodes are discrete items, such as data, concepts, or processes, and links are relationships between the nodes. Attributes are additional details associated with a node or link. Although links and nodes might have attributes, Deep View supports attributes on nodes only.

Deep View presents different categories of nodes and links as demonstrated in the state chart and ERDs. A diagram can have different categories of nodes or links to represent different types of information.In a state chart, nodes represent a state, a stage in a larger process. As shown in Figure 6-1 (a), a state chart has three categories of nodes: a regular state, a start state, and an end state. The process represented by the state machine begins at a start state and completes at an end state. All the other states of the state

machine between the start and end states are regular states. A state node has a short label describing the stage in the state machine.

a) State-chart visual diagram

b) Initial list of nodes in logical sequence

Deep View (version 14.1)

File   Edit   Perspective   Visualize   Options   Colab

⊞ dinner, START STATE
⊞ Put food in microwave, STATE
⊞ Set time, STATE
⊞ Check temperature, STATE
⊞ Eat, END STATE

dinner

Put food in microwave

Set time
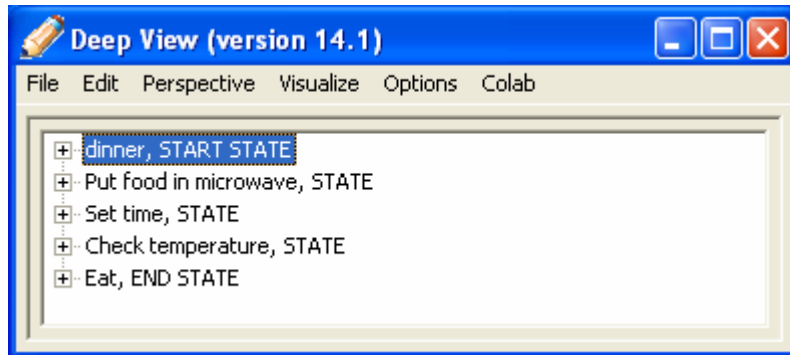
c) List of links after nodes are expanded

Deep View (version 14.1)

File   Edit   Perspective   Visualize   Options   Colab

⊟ dinner, START STATE
   └ TO Put food in microwave, TRANSITION
⊟ Put food in microwave, STATE
   ├ TO Set time, TRANSITION
   └ FROM dinner, TRANSITION
⊟ Set time, STATE
   ├ TO Check temperature, TRANSITION
   └ FROM Put food in microwave, TRANSITION
⊟ Check temperature, STATE
   ├ TO Eat, TRANSITION, named, right temperature
   └ FROM Set time, TRANSITION
⊟ Eat, END STATE
   └ FROM Check temperature, TRANSITION, named, right temperature

Check temperature

right temperature

Eat

Figure 6-1. State-chart demonstrates a diagram with nodes and links. This diagram has three categories of nodes: start state, end state, regular state.

95

b) Deep View ERD treeview show relationships and attributes of the Patron entity type
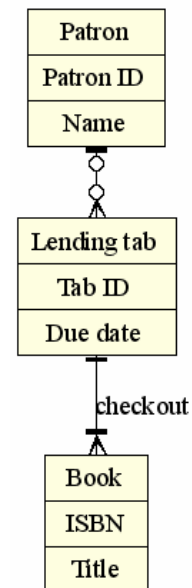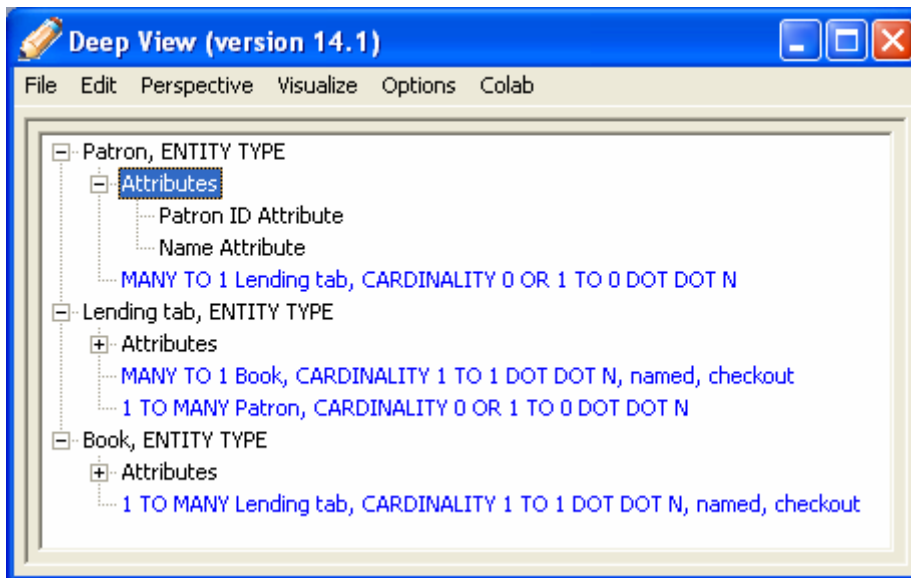
Figure 6-2. Example of ERD

a) visual diagram

a) Deep View treeview of sub-diagram
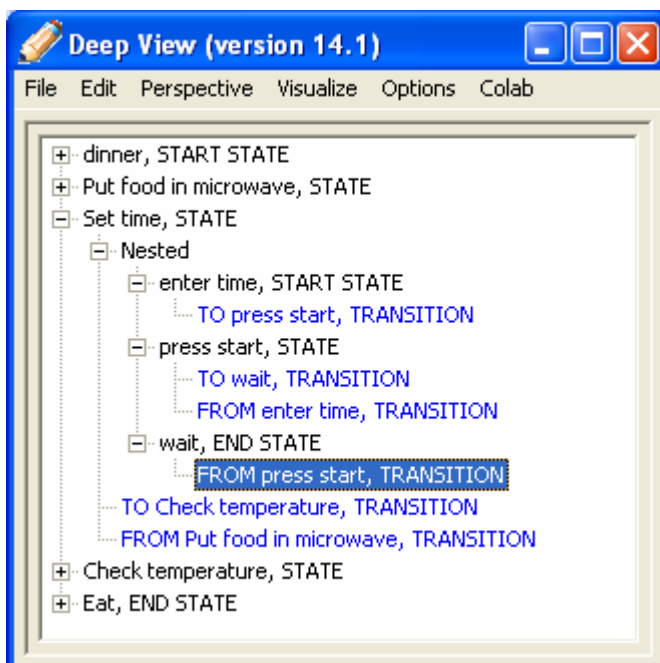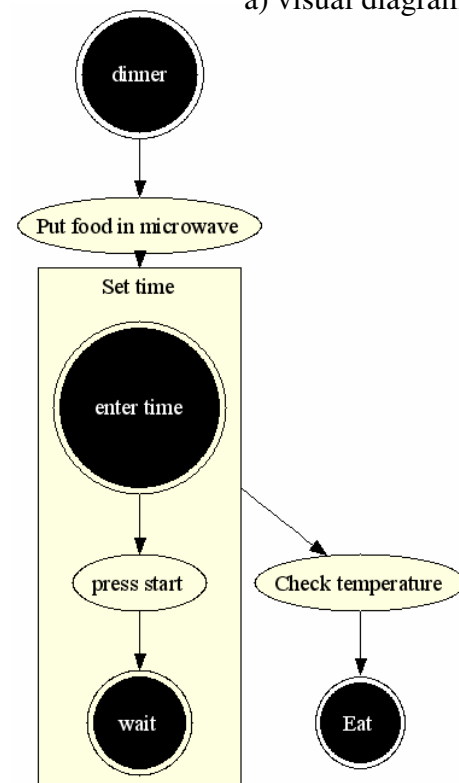
Figure 6-3. State-chart with a sub-diagram

96

Transitions between states are the only kind of link in a state chart. A transition represents how the state machine progresses from a given state to the next. The label on the transition describes inputs that initiate the transition. The transition is directional, such that a transition goes from node "set time" to node "check temperature." A start state has only outgoing transitions and an end state only incoming transitions.

An ERD encompasses a rich set of details about entities and their relationships. For simplicity we focus on a small set of details to demonstrate how ERDs are represented in Deep View. Our focus is on an ERD with two categories of nodes and four categories of links. We also limit our ERD example to the database relational model, which minimizes the number of categories of links. Our existing example of an ERD could be expanded to encompass more details of an ERD.

In an ERD, nodes of an ERD are named entity types that represent the data in a system. An instance of an entity type is referred to as an entity. Links are called relationship types and represent how entities are associated. The ERD we consider has two categories of nodes. The general entity type is a collection of entities (individual data). The example in Figure 6-2 (a) includes the "patron", "lending tab", and "book" entity types. An associative entity type is a special case of entity type in which there is a many-to-many relation between two other entity types. An associative entity type is used because a many-to-many relation between two entity types cannot be modeled in a database. Note, an associative entity type is not shown in Figure 6-2 (a).

A link in an ERD is a relationship type, which is an association between two entity types. In the Baker ERD notation, the relationship type is specified by the label on the link connecting two entity types. For example, in Figure 6-2 (a), an example of a relationship type is that the "lending tab" entity tallies the "book" entities checked out. In Deep View, relationship types can by unary (an entity type node links to itself) or binary (a link connecting two entity types).

Deep View has four categories of links for an ERD to represent the constraints on a relationship type. A constraint refers to the number (cardinality) of distinct entities that can be associated by a relationship type. In our simplified ERD, we model one-to-many relations and exclude one-to-one and many-to-many relations. For example in Figure 6-2 (a), one "lending tab" entity can have many "book" entities checked out. Deep View has

four categories of links to reflect the combination of minimum and maximum cardinalities of each entity type in a relationship type. In a relationship type, each entity type can have a cardinality of {zero or one, exactly one, zero or more, one or more}. A cardinality of zero indicates that the entities in a relationship type are optional. Considering only one-to-many relations means there are four possible links as follows (written as spelled out and in shorter notation):

1. (zero or one) to (zero or more), 0/1 to 0..N
2. (one) to (zero or more), 1 to 0..N
3. (zero or one) to (one or more), 0/1 to 1..N
4. (one) to (one or more), 1 to 1..N

Figure 6-2 (a) gives an example of two relationship types with different constraints. The relationship type between "patron" and "lending tab" has a cardinality of 0/1 to 0..N, respectively. And the relationship type between "lending tab" and "book" has a cardinality of 1 to 1..N, respectively.

The ERD demonstrates how Deep View presents extra details on nodes. In an ERD, an entity type has additional data named attributes. These attributes represent properties of an entity type. An example of an attribute is a primary key in a database. In the ERD convention we use, entity attributes are visually part of the entity; they are listed below the entity name as shown in Figure 6-2 (a). In other conventions, entity attributes are separate nodes linked to the corresponding entity. Deep View can be configured to support this convention too. When describing Deep View we will refer to extra details of nodes as a node's attributes (similar to an entity's attributes).

The state-chart demonstrates how Deep View represents sub-diagrams, a diagram within the main diagram. In a state chart, a state (diagram node) may contain a state-machine determining how the state operates. An example of a sub-diagram is illustrated in Figure 6-3 (a).

6.2 Deep View interface

The Deep View interface presents the diagram characteristics described in the previous section. Deep View's interface is a textual description that the user interactively navigates and reads. The language of the textual description incorporates the domain-specific terminology of a diagram. The specific structure of the textual description is described in more detail in the next section.

The Deep View interface is a graphical user interface (GUI) consisting of standard widgets shown in Figure 6-1 (b, c) and Figure 6-2 (b). A blind user accesses Deep View through a screen reader, which reads the textual description of the diagram. Blind users will be familiar with the standard widgets and can apply their previous knowledge, such as keyboard shortcuts, to navigating the widgets. Although the interface is a GUI a sighted person could use, the interface is optimized for access with a screen reader. Nonetheless, the sighted person can get an understanding for how the diagram is organized for the blind user.

Deep View mainly presents a node-link diagram in a treeview GUI widget, typically used by the operating system to show the tree structure of directories and files. The organization of the diagram characteristics in the treeview provides a consistent metaphor for a blind user to navigate. An advantage of the treeview is how it shows different levels of detail. The main diagram information is presented at the highest level separated from secondary information listed (collapsed) at lower levels, similar to how subdirectories are below a main directory in a directory structure. A user expands the levels in the treeview to access lower level details as needed. The treeview entries are short textual descriptions of a corresponding node, link, or node attribute. A user will typically interactively move quickly between the treeview entries listening to the entry and deciding which element to visit next.

Deep View's presentation of a node-link diagram is centered around a diagram's nodes. The nodes are at the top level of the tree and each node has a separate treeview entry. The text of the treeview entry includes the node name and is described more in the next section. The list of nodes gives the user an overview of the context of the diagram. Initially the nodes in the treeview are collapsed, so that other details of the diagram are hidden. The user expands the treeview entry of a given node to access a node's detailed

information, including links, attributes, and sub-diagrams. Expanding the treeview entry opens (expands) a list of treeview entries with the corresponding details.

The primary information listed below a diagram node (expanded treeview entry), is a list of links connecting the corresponding node to other nodes. The links incoming to a node are listed first followed by the outgoing links (in the case of directed links). Take, for example, the link (transition) connecting nodes (states) "set time" and "check temperature" in Figure 6-1. A typical link appears twice in the entire treeview – once under each diagram node ("set time" and "check temperature") that it connects. The text of the treeview entries describe the link's direction between the "set time" and "check temperature" nodes. An exception is a link that links a node to itself. Such a link appears only once under the node linked to itself (not shown in Figure 6-1).

In addition to the links listed below a diagram node, there is a treeview entry named "attributes" representing a node's set of attributes. Expanding the attributes treeview entry lists the individual attributes as separate treeview entries, as shown in Figure 6-2 (b). Attributes are nested at a deeper level in the treeview because the attributes are at a lower level of detail. Typically a user will inspect nodes and links before inspecting the specifics of the attributes.

The treeview representation provides a consistent metaphor for presenting sub-diagrams. A sub-diagram is visually drawn within a node of the main diagram. Similarly in Deep View, a sub-diagram is presented within the corresponding node of the main diagram as shown in Figure 6-3 (b). An expanded treeview entry corresponding to a node lists entries for links, attributes, and an additional treeview entry named "nested" corresponding to a sub-diagram. Expanding the nested treeview entry opens the sub-diagram; the nodes of the sub-diagram are listed. The sub-diagram functions like the main diagram. With this design the details of a sub-diagram are separated from the main diagram and a user can inspect each diagram separately. Sub-diagrams can further contain nested sub-diagrams.

6.2.1 Language of textual description

Elements of a node-link diagram have names and labels relevant to the concept expressed in the diagram. Deep View incorporates a diagram's domain specific

100

terminology into the textual description of diagram elements (i.e. treeview entries). The textual descriptions are short so that a user can quickly identify the node-link diagram element and decide where to navigate next.

Domain specific terminology is incorporated into Deep View by enabling advanced users to create new diagrams. A diagram is declared in a text file, which defines a diagram's name and terms to describe the categories of nodes and categories of links. It is possible to have one or more categories of nodes and links. Although terminology differs between diagrams, Deep View takes advantage of similar descriptions of diagrams. We consider a state-chart to illustrate the textual descriptions of nodes and links.

The textual description of nodes is straightforward. It is the node name followed by the name of the node category. See Figure 6-1 as an example; the textual description of the node of category STATE named "set time" is: "set time, STATE". The order of the name and classification are deliberately chosen to be suitable for the screen reader user.

The technique is used in other interfaces designed for blind users. In comparison, a visual representation would list the classification first to provide a visual cue for grouping similar elements. A sighted person can easily ignore the repeated classification and focus on an element's unique name. A screen reader, however does not ignore the repeated classification. As a blind user navigates a list of items with the preceding classification, the person has to wait for the screen reader to read the classification before the unique entry information, such as the name in the example, is available. So for the blind user, the unique information is listed first. The blind user can listen to the classification if necessary; or he can save time by cutting the screen reader short and move to the next treeview entry.

The language of a diagram link is more complicated than a diagram node. A link describes the relationship between two nodes. For example, a link in a state chart is a transition between two states (nodes). The description of a transition is: it goes from state "set time" to state "check temperature". Or that, the transition goes to state "check temperature" from state "set time".

In Deep View, the description of a transition (link) is relative to the state (node) it is listed under. The description should read like a natural sentence. A link's description

starts with a short term (we refer to it as a connection phrase) describing the link and the node it is connected to. The connection phrase is short, preferably one word. The textual description is followed by the category of the link. As an example, the text of the transition when listed under state "set time" is:

TO check temperature, Transition

And when the link is listed under state "check temperature" it is:

FROM set time, Transition

The connection phrases are "to" and "from", which are domain specific. For a link, the connection phrase differs because it describes the relative connection of a link between two nodes. In general, the connection phrases would be replaced by domain specific terms of a given diagram.

An optional label on a link summarizes the nature of the link. If a link has a label, Deep View places the label at the end of the textual description. The term "named" is added so that the description reads smoothly. This way the description reads naturally with or without a label. For example, in Figure 6-1 (c) the transition between nodes "check temperature" and "eat" has the label "right temperature". The textual description of the link relative to the "check temperature" node is:

TO eat, Transition, named, right temperature

An ERD demonstrates how the Deep View textual description can apply to a diagram with dramatically different terminology. An ERD has nodes of categories: entity and associative entity. These nodes are similar to states in a state chart. ERD relationships (links), however, are more complex than state transitions. ERD relationships represent the cardinality of the related entities.

Deep View has four categories of ERD links that represent the possible constraints on a relationship type. The connection phrases express the cardinality of the two entities. For example, in Figure 6-2 there is a one-to-many relationship between the "lending tab" and "book" entity types. In Deep View the textual description of the link is as follows:

Under the "lending tab" entity type the text describing the relationship is:

1 TO MANY book, cardinality 1 to 1..N, named, checkout

Under the "book" entity type:

MANY TO 1 lending tab, cardinality 1 to 1..N, named, checkout

The connection phrases of an ERD are "many-to-one" and "one-to-many", as is used when people verbally discuss an ERD relationship type. These phrases reflect the maximum cardinality. Although the minimum cardinality is also important, it is secondary. Therefore the complete cardinality (minimum/maximum) is listed at the end of the textual description. A user would listen to the entire textual description to get the most complete cardinality information. The complete cardinality information also happens to be the ERD's name of the corresponding link category. The name of the relationship type is also appended to the textual description.

The Deep View textual description formats are the result of incorporating blind users' feedback. Overall, blind users found the textual descriptions comprehensible. The feedback helped us choose punctuation (comma and periods) so that the screen reader pauses appropriately while reading the textual descriptions.

Feedback from Braille users and screen reader users, however, suggests that the phrasing of the textual description should be customizable. The comments apply specifically when a link has a label, which describes the relationship between two nodes. A Braille display user prefers the described technique when the connection phrase is first. This way the user can feel the first character on the Braille display to distinguish the outgoing links from incoming links. For example in a state-chart, the "t" in "to" phrase distinguishes it from the "f" in the "from" phrase. On the other hand, a screen reader user would prefer having the link label listed before the connection phrase to avoid listening to

the repetitive connection phrase. More details on user feedback are included in the evaluation Chapter 7.

6.3 Editing

Deep View provides a blind user full functionality to edit node-link diagrams. A blind user can create diagrams from scratch to present ideas to sighted colleagues. A blind user can also edit existing diagrams to contribute new ideas to the diagram. Editing involves the functionality of adding, removing, and changing elements in the diagram. The editable diagram elements include nodes, links, attributes and sub-diagrams.

The technique a sighted person uses to edit a diagram is not sufficient for a blind user. In a graphical diagram application, a sighted person relies on a mouse for editing a node-link diagram. The user uses a mouse to click on nodes and links to be edited. A user adds a link by dragging the mouse between two nodes to be connected. There is no keyboard equivalent. Keyboard shortcuts – when available – are limited to accessing a diagram element's specification, such as text or visual appearance.
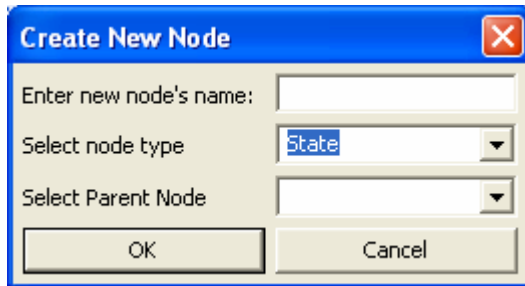
Deep View makes editing a diagram accessible through a series of dialog windows, which are shown in Figure 6-4. The dialogs consist of standard GUI widgets, which a blind user is familiar with navigating. To create or edit a diagram element, a user enters the relevant information in the dialog window. For example, to create a node, a user enters a node's name and selects the category of node (dialog in Figure 6-4 (a)). Optionally, a user can select a node in which the new node should be nested to create a sub-diagram.

When creating a new link, a user specifies the link name, and selects the nodes to be connected from two lists of nodes. The user also selects the category of diagram link. The category of link determines the labels describing the nodes to be connected. For example, in the dialog to add a transition in the state-chart the node labels are "from state" and "to state" (see Figure 6-4 (b)). On the other hand the labels to create an ERD relationship (see Figure 6-4 (d)) express the cardinality of the corresponding entity. When possible the dialog uses lists of options to mitigate invalid user input.

There is a similar dialog (shown in Figure 6-4 (c)) for creating an attribute of a node. In the dialog, a user selects the node the attribute should be appended to. The dialog applies
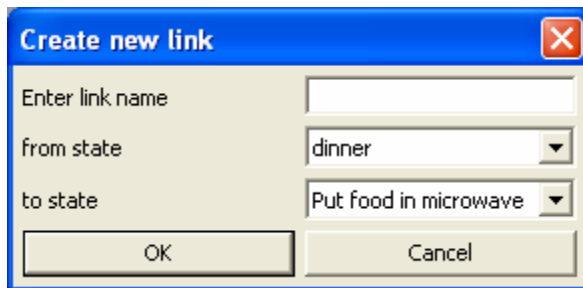
the attribute to the selected node in the treeview when the add attribute command is given.

a) Create a new node (state)
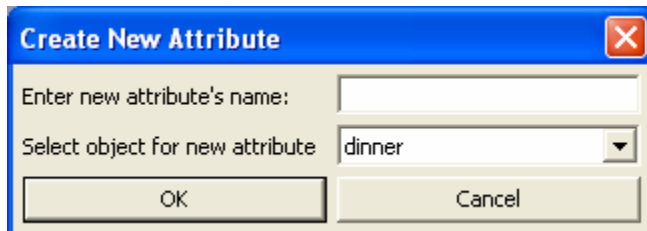
b) create a new link (transition)
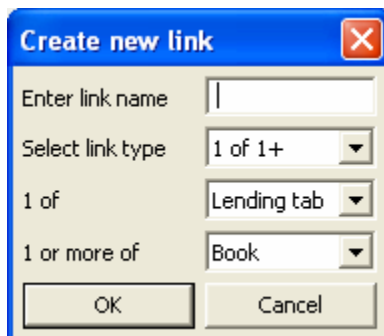
c) add an attribute to a state

d) Dialog to create a relationship in an ERD. There are several link types letting the user specify the cardinality of the entities.

Figure 6-4. Deep View dialogs to edit a state-chart diagram

Changing the name or label of a diagram element is straightforward. A user selects the element to edit and presses the edit shortcut key (Ctrl+E). The user makes the edit and accepts the change. The textual changes are reflected throughout all instances in the treeview. Labels and names of links and nodes appear multiple times in the treeview. A link appears under each diagram node entry it connects. A node's name also appears in the textual description of links.

Deleting diagram elements is also straightforward. Similar to editing a diagram element, a user selects the element to be deleted in the treeview and presses the delete key. Deleting a node with links to other nodes removes the links connected to the deleted node.

A limitation to Deep View's editing capability is that it does not enforce domain specific diagram restrictions. For instance, in a state-chart a user can incorrectly create a transition out of the end state or multiple transitions between two states. The onus is on the user to create valid diagrams. It is future work to enhance Deep View to enforce restrictions in a general manner.

6.4 Navigation

The main objective of Deep View is to enable blind users to explore and understand node-link diagrams. A Deep View user has a variety of techniques to navigate a node-link diagram. The navigation techniques are divided into three categories. The first is through the use of miscellaneous tools – a search tool and a diagram summary. The second is navigating individual nodes and links similar to other accessible diagram applications. The third is unique to Deep View and lets blind users access some high-level diagram characteristics that a sighted person readily recognizes from the visual layout of a diagram. A user can use a mix of techniques to explore the diagram from different perspectives.

6.4.1 Miscellaneous navigation

Two types of miscellaneous tools were suggested by Deep View users and added to Deep View. These tools are common in other applications. The first is a summary report of a diagram. It reports the number of nodes and links and high-level

characteristics described later. These facts might give the user an insight into the size and complexity of the diagram. Deep View automatically displays the summary when a new diagram is loaded and the user can retrieve it at any time.

The second tool is a search function. It can be helpful for finding diagram elements and navigating a diagram. In a separate dialog, a user enters a search term. The result of the search is a list of all treeview entries with the given term. Selecting a given answer and pressing enter, Deep View closes the search dialog and brings the widget focus to the entry the user selected. This way a user can search for and navigate directly to a given node, for example, rather than traverse the treeview entries searching for the node.

6.4.2 Navigate individual diagram elements

The most elementary navigation is for the user to interactively visit individual nodes and links in the treeview. A user's main interest is to explore a diagram's nodes and relationships between the nodes. Deep View supports the two forms of navigation non-sighted users found useful in the single user Deep View study (Chapter 7) when exploring a node-link diagram. The two forms of navigation are characterized as a breadth first traversal and a depth first traversal.

In the breath first traversal, the user is interested in all the nodes in the diagram. The nodes determine the scope and context of the diagram. Deep View's initial presentation of a diagram in the treeview is a list of diagram nodes. A user can easily browse the nodes by traversing the list of nodes. Information on links, attributes, and sub-diagrams are collapsed away under the nodes. The initial view of a sub-diagram also starts with a list of nodes in the sub-diagram to enable the breath first traversal.

Furthermore, Deep View places the list of nodes in a meaningful order when possible. In a state chart with a simple sequence of states the logical order of nodes is from the start state, through the regular states, to the end state. Deep View orders the diagram nodes according to a topological sort of nodes. It is an open issue how to order a diagram's nodes in a sequential order when a diagram has cycles or parallel paths. For now, the topological traversal is an initial attempt at ordering the nodes. Section 6.5.1 describes the procedure to perform a topological sort on a diagram with cycles, which

otherwise would not be possible. After a person uses the breath first traversal and understands the context of the diagram, he can continue to explore the relationships between nodes.

The relationship between the nodes can be explored in a depth first traversal of nodes and links. In this form of navigation the user traverses a path in a diagram from one node to another by following links connecting the nodes. Deep View provides a more efficient traversal of links than the tedious technique, as follows. In general, traversing a link starts with a user identifying a link to traverse to another node. It is tedious for the user to navigate the treeview to find the linked node as it requires the user to read several irrelevant treeview entries.

Deep View's innovation is to provide a hyperlink mechanism for the user to traverse a link. A user can quickly navigate relationships similar to a hyperlinked webpage. In the analogy, a diagram's nodes are web pages and diagram links are hyperlinks between pages. In Deep View, a user navigates a diagram link by selecting the link and pressing the enter key. This takes the focus to the linked node.

Deep View marks visited nodes in a way similar to how a web browser highlights links of visited web pages. A visited node is marked with an alteration to the visited node's textual description; it is appended with the phrase "visited". This way a user can quickly identify nodes that have not yet been explored.

A user can easily backtrack along a path of visited nodes (traversed links). The user presses the backspace button and Deep View sets the focus to the node of the last link traversed. This is useful, for example, when examining a tree structure. A user can traverse one branch and backtrack to explore other branches.

6.4.3 High-level queries

In a visual diagram a sighted person can recognize meaningful high level characteristics at a glance. A loop in a state-chart, for example, is a sequence of nodes connected by links that form a circular shape (see Figure 6-5 (a) for an example). The significance of a loop is that it indicates a process that is possibly repeated multiple times. A blind user does not have the same advantage of quickly recognizing high-level diagram characteristics. With the described elementary navigation it is a matter of hunt-and-peck

for a blind user to find information such as finding a loop. The user must use trial-and-error to traverse links in the hope of ending at the final node in the loop (the initial starting node). Even when the user completes the loop, it is up to the user to remember the path. Finding the same path again requires repeating the traversal.

Deep View identifies and presents three types of diagram characteristics. The first characteristic is a loop as described (see Figure 6-5 (a) for an example). The second characteristic is a path of links between two nodes the user specifies. A path signifies the relationship between the nodes. We refer to the third characteristic as a parallel path (referred to as internally disjoint paths in graph theory). These high-level characteristics are significant for many node-link diagrams. Although Deep View currently manages three diagram characteristics, future research could identify more characteristics that can be useful to present to the user.

Deep View has the unique feature of automatically identifying and presenting some high-level diagram characteristics to a blind user. For example, Deep View presents the blind user the sequence of nodes involved in a loop. In the next section, we explain how Deep View presents the high-level characteristics of a diagram in the Deep View treeview in a manner consistent with the rest of the diagram.

We will use state-charts with directed links (transitions) to explain the significance of the high-level queries. We will also briefly describe how the high level queries generalize to an ERD with undirected links (relationships).

a) visual diagram

b) Deep View presents the query for cycles in the diagram
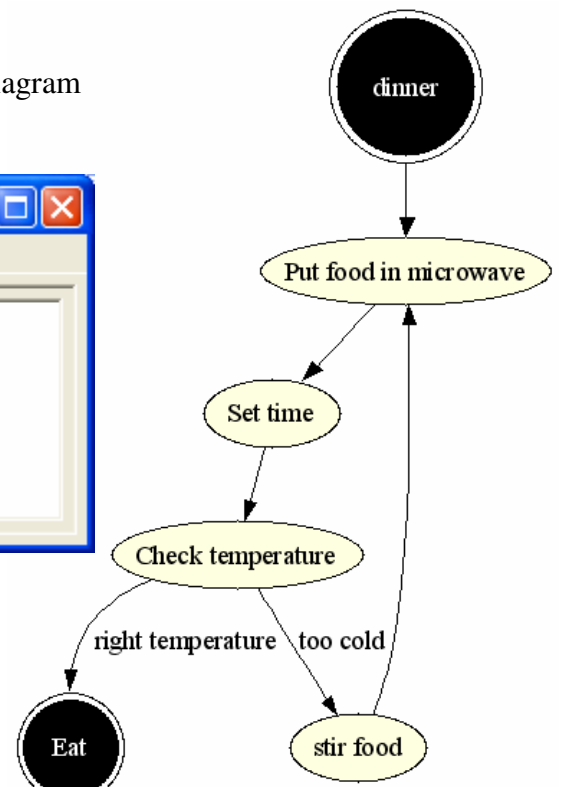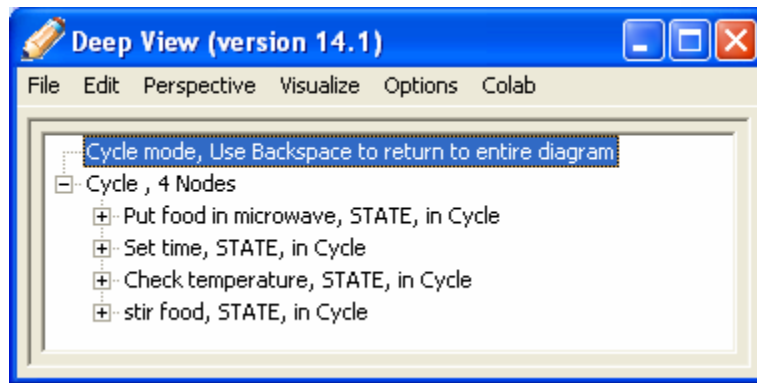
Figure 6-5. State-chart that demonstrates a cycle

b) Query for path from "start" state to "go to restaurant" state   a) Visual diagram

Deep View (version 14.1)

File Edit Perspective Visualize Options Colab

```
Path mode, Use Backspace to return to entire diagram
Forward path from, start,  to , go to restaurant, 3 Nodes
    start, START STATE, on Path
    dinner, STATE, on Path
    go to restaurant, STATE, on Path
```
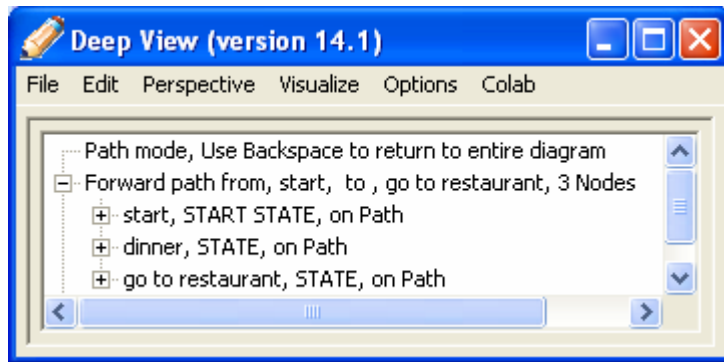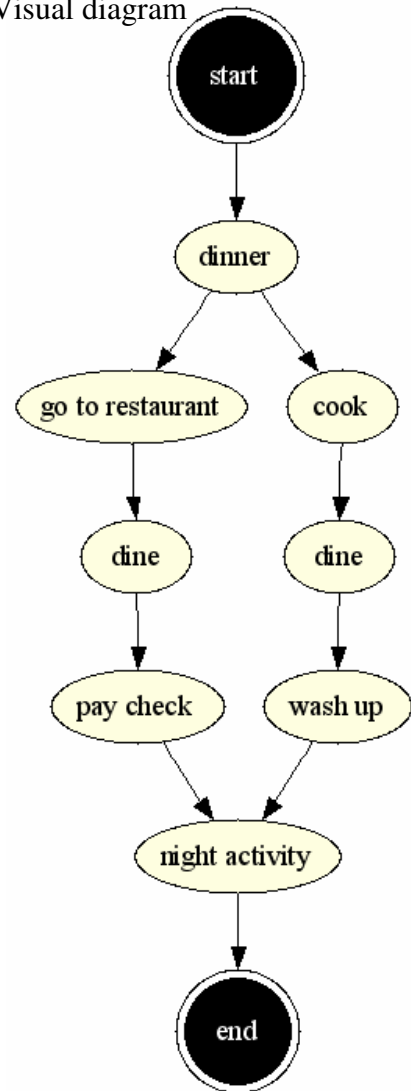
c) query for all parallel paths in the diagram

Deep View (version 14.1)

File Edit Perspective Visualize Options Colab

```
Parallel path mode, Use Backspace to return to entire diagram
Parallel Path from, dinner,  to , night activity, 2 Paths
    Forward path from, dinner,  to , night activity, 5 Nodes
        dinner, STATE, on Path
        go to restaurant, STATE, on Path
        dine, STATE, on Path
        pay check, STATE, on Path
        night activity, STATE, on Path
    Forward path from, dinner,  to , night activity, 5 Nodes
        dinner, STATE, on Path
        cook, STATE, on Path
        dine, STATE, on Path
        wash up, STATE, on Path
        night activity, STATE, on Path
```

d) Query for the path between the "go to restaurant" state and "cook" state. Has a common parent

Deep View (version 14.1)

File Edit Perspective Visualize Options Colab

```
Path mode, Use Backspace to return to entire diagram
Common parent, dinner, for , go to restaurant and , cook, 3 Nodes
    From parent to, go to restaurant1, 2 Nodes
    From parent to, cook1, 2 Nodes
Common child, night activity for , go to restaurant and , cook, 7 Nodes
    From, go to restaurant,  to child2, 4 Nodes
    From, cook,  to child2, 4 Nodes
```

Figure 6-6. State-chart diagram with parallel path to demonstrate queries for high-level diagram characteristics.

111

Path

A path between two nodes is the relationship between the nodes. For example, in a state chart a path signifies the stages of a process between the start and destination states. In Deep View, a user instructs Deep View to find a path by entering the start and destination nodes of the path in a dialog window. Deep View returns all the paths between the nodes. In Deep View we identify three kinds of paths, representing different information. These paths apply to diagrams with directed edges.

The first kind of path is a simple forward directed path. The links from the start node to the destination node are always directed at the next node in the sequence. In a state chart, a forward path details the inputs from one state to another. Figure 6-6 a. shows the forward path from the "start" state to the "go to restaurant" state. The reverse of a forward path is the same path in the reverse direction between the start and destination nodes. Such a reverse path is called a back path. A forward or back path is encountered depending on the order in which the user specifies the start and destination nodes.

The second kind of path is a path where the user specifies an initial node and destination node that have a common parent or child node. In a state-chart, the common parent state of the initial node and destination node is the state where a decision is made between two processes (paths). For example, in Figure 6-6 a. a person decides between cooking dinner at home or going to a restaurant to eat. There is a similar relationship if two nodes in the diagram have a common child.

For another example, consider a tree data structure, such as a company organization chart. In this case a link is directed from the manager to the managed employee. A parent node in a path between two employees represents the manager the employees have in common.

The third kind of path between two nodes is where the direction of the links is arbitrary. In a state-chart it indicates that the states are connected through transitions but the nodes are different parts of the larger state machine.

Cycles

As in the given example, a cycle of nodes signifies a repeated process in a state chart as illustrated in Figure 6-5 (a). Deep View returns all cycles in a diagram when a blind user queries for cycles. Deep View currently only identifies directed cycles where links in the cycle are in the same direction, such as a loop. Undirected cycles might be significant too and could be indicated in future versions of Deep View.

Parallel path

We refer to parallel paths as two or more forward paths between a start node and a destination node as shown in Figure 6-6 (a). Considering the parallel paths independent from the remaining diagram, the start node is a source node (directed links are outgoing) and the destination node is a sink node (directed links are incoming). In Figure 6-6 (a). the source is the "dinner" state and the sink is the "night activity" state. In graph theory a parallel path is referred to as an internally disjoint or independent path. The significance of a parallel path is that there are multiple ways to traverse between the sink and source nodes. When a user queries Deep View for parallel paths, Deep View identifies all pairs of source/sink nodes and presents all the paths between each pair of nodes.

ERD and high level queries

An ERD demonstrates how the described high level characteristics apply to a diagram with undirected links. Although the ERD relationship types (links) are undirected, in Deep View we assign a direction. We define the direction of a relationship type from the entity type (node) with smaller cardinality to the entity type with higher cardinality. Remember in our example we only consider a one-to-many relation rather than a one-to-one relation or a many-to-many relation. This gives the entities with lower cardinality higher priority, although typically there is no hierarchy in an ERD. The implication, for example, in a simple ERD is that an instance of the main entity type is associated to one or more of the secondary entities. Important for the Deep View presentation, the prioritization contributes to determining the initial ordering of a diagram's nodes.

For a user the most relevant high-level characteristic of an ERD is finding paths between entity types. The typical use of an ERD is for the user to track the associations (paths) between entity types. The most likely path relationship between two entity types is that the directions of the links are arbitrary. Nonetheless, the user has the main desired information about how the two entity types are related.

A special feature of Deep View's path query is that it can identify the main entity type when the user queries for the two secondary entity types. In this example, Deep View would identify the main entity type as a common parent to the secondary entity types. This is a result of assigning the direction on the ERD's links (relationship types).

Entity types in ERD may have cyclical or parallel associations between them. In future research we would identify the significance of cyclical or parallel associations to expressing concepts in an ERD.

6.4.4 Displaying high-level queries

Deep View presents the results of the high-level queries in the Deep View GUI treeview used to present node-link diagrams. The intention of the design is for the user to access the query results in the same paradigm as the diagram. To start the high-level query a user gives a keyboard command to start the query. In the case of querying for a path, the user enters the start and destination nodes of the path in a dialog window.

Deep View responds to the query by presenting all instances of the queried characteristic; for example, a query for cycles in the diagram could result in three cycles. Each instance of a result is displayed as a treeview entry at the top level. Expanding the treeview entry lists the details related to the characteristic. In the example, there would be three entries for the three identified cycles. The emphasis of the presentation is on the nodes involved in the identified characteristic.

For one instance of an identified cycle (as shown in Figure 6-5 (b)), Deep View displays the sequence of nodes in the cycle starting at the start node of the cycle. Deep View uses a heuristic to identify the start of a cycle. The start of the cycle is the node with the most incoming edges; otherwise an arbitrary node is chosen. The technique works for the simplest case, such as a loop in a state-chart.

114

The results of the path query are presented similar to the cycle query. Each instance of a simple path query indicates its type of path: a forward/back path, path with a common parent or child node, or a path with arbitrarily directed links. A simple forward/back path lists the sequence of nodes in the path below the given treeview entry. Figure 6-6 (b) shows a forward path from the "start" state to the "go to restaurant" state.

The presentation of paths with a common parent/child is more intricate. Figure 6-6 (d) shows a path with a common parent between the "cook" state and "go to restaurant" state. Expanding the treeview entry for a path, there are two further treeview entries. Expanding each of these entries shows the path from the common parent/child to the respective nodes queried by the user.

Deep View returns all parallel paths in a diagram when the user queries for parallel paths. Figure 6-6 (c) shows the parallel path between the "dinner" source node and "night activity" sink node. At the top level of the treeview there is an entry for each pair of source/sink nodes. Expanding the entry of a source/sink pair lists entries corresponding to each path between the source and the sink. Expanding an entry for a path lists the sequence of nodes between the source and the sink.

Preliminary feedback from blind users indicates that the query feature is usable. The Deep View presentation, however, needs to be refined based on feedback from more usability studies. Although blind users could understand portions of the presentation, they were confused by the phrasing and organization. In some cases participants were confused because of a lack of experience dealing with node-link diagrams. Ideally the usability study to evaluate the feature would involve advanced Deep View users, familiar with the basic functionality and ready to use the advanced Deep View features.

6.5 Graph algorithms

Deep View automatically identifies the high level diagram characteristics by analyzing the relationships between nodes with a series of graph algorithms. A diagram is treated as a graph where a diagram's nodes are vertices in the graph and a diagram's links are edges of the graph. Although node-link diagrams can have directed or undirected links, Deep View treats all links as directed. Parallel paths, for example, are identified by combining the technique to identify cycles and paths between nodes. In the analysis a

node-link diagram is treated as a graph with directed or undirected links depending on the situation. Deep View's analysis is independent of the spatial layout of the diagram. However, future work could incorporate into Deep View the meaningful information inferred from the spatial layout.

We next describe the graph algorithms Deep View uses to identify the high level diagram characteristics. We implemented well known algorithms, described in detail for example in (Cormen, Leiserson et al. 2001). Performance of the algorithms is only a secondary concern. Although some of the algorithms are NP complete, the performance impact of computing the algorithms is manageable as most human readable diagrams are likely to contain less than approximately 20 nodes, as (Ishihara, Takagi et al. 2006) found in their evaluation.

A priority of future work is to improve the performance sufficiently to enable incremental update the diagram characteristics. Currently after each edit to the diagram, a query for high level characteristics recomputes everything from scratch. Performance can be improved by incrementally updating the characteristics based on the user's edit to the diagram.

6.5.1 Order of nodes

Deep View uses graph algorithms for one feature not related to the high-level diagram characteristics. The graph algorithms are used to order the initial sequence of nodes Deep View presents to a user when a diagram is loaded. For a logical ordering of the nodes, Deep View presents the nodes resulting from a topological sort, which sorts a direct acyclic graph (DAG) in the order that nodes are encountered from the start node.

A topological sort is possible only on a DAG. For the purpose of computation, Deep View converts the node-link diagram into a DAG by creating a spanning tree. The spanning tree is created with a depth-first-search (DFS) of the node-link diagram. As the original diagram is traversed, edges are added to the spanning tree if the edge does not connect to a visited node. This eliminates edges closing cycles, including back edges (a cycle). Finally the spanning tree is traversed with a DFS and the order of traversed nodes results in the topological sort. The DFS begins with nodes only with outgoing links.

116

Because the graph is not weighted, finding the spanning tree does not require a more complicated algorithm, such as Prins algorithm (Cormen, Leiserson et al. 2001)..

Ideally in a state diagram the start state is the first in the sequence and the end state is the last in the sequence. A topological sort will always make the start state the first node. The end state, however, might not be last. A topological sort can have multiple valid sequences of nodes. In future research we will investigate further heuristics and techniques to improve this case.

6.5.2 Paths

Paths are found by an algorithm to find all paths between two nodes specified by the user. The algorithm builds on a DFS. A DFS is typically used to find the shortest path between two nodes. To find all paths, a DFS is performed on every node starting with the start node and every node reached from the start node. This algorithm is NP-complete.

In a separate process, paths are categorized as a forward/back path, a common parent/child path, or a path with arbitrarily oriented links. Because we know we are working with a single path between two nodes, a node has at most two edges. The first and last nodes have only one edge. The process used to identify the category of path employs what we call a *swap vertex*, a node that has only incoming or outgoing edges. It is named swap vertex because the direction of the path changes at this node.

A path with a common parent has one swap vertex with incoming edges. The common parent node is the swap vertex. Similarly, the path with a common child has one swap vertex with incoming edges and the common child node is the swap vertex.

A forward/back path does not have swap vertices. The direction of the link of the node determines if it is a forward or back path; an outgoing edge means it is a forward path and an incoming link means it is a back path.

6.5.3 Cycles

Cycles are found with a separate procedure from paths. Ultimately Deep View will present directed cycles, where all edges are in the same direction. Detecting these, however, first requires finding all cycles regardless of direction. Hence the diagram is treated as an undirected graph. We use a typical algorithm to find cycles. First the basic set of cycles is calculated and then the basic set is combined to generate other cycles.

The basic cycle set is calculated with a spanning tree, computed as in the described topological sort. While creating the spanning tree, nodes revisited indicate a cycle. We refer to the closing edge as the edge reaching a visited node. It is not added to the spanning tree but it is recorded.

The closing edges and the spanning tree are used to calculate the basic cycle set. Each closing edge represents one basic cycle in the node-link diagram. Adding a closing edge to the spanning tree creates only one cycle. So a cycle is calculated by finding the path between the two nodes connected by the closing edge. The path is found with a DFS.

Further cycles are calculated by combining the basic cycle set. Cycles that share one or more edges combine to create a new cycle. The complete space of all possible cycles is the combination of all cycles in the basic cycle set. The entire space, however, does not need to be searched exhaustively. Rather a new cycle is calculated by combining a basic cycle with an existing cycle. At the start of the process, the algorithm attempts to combine cycles in the basic set with each other. Those pairs of cycles that share edges create a new set of cycles. On subsequent passes, the algorithm attempts to combine each of the new cycles with basic cycles previously not incorporated into the cycle (each cycle records the basic cycles it consists of).

Directed cycles (with edges in the same direction) are those presented in the Deep View interface. Although the cycles were calculated assuming an undirected graph, the direction of the edges is maintained. A cycle with edges in the same direction is determined by the number of swap vertices it contains; mainly a directed cycle has no swap vertices.

6.5.4 Parallel paths

Deep View identifies parallel paths by combining the techniques to identify cycles and general paths. The source and sink nodes of a parallel path are identified by the algorithm to find cycles. A parallel path is characterized by a cycle with two swap vertices. The swap vertex with outgoing edges is the source node of the parallel path. The swap vertex with incoming edges is the sink node of the parallel path. If there are more than two parallel paths, multiple cycles will indicate the same source/sink nodes.

The two or more paths between the source and sink nodes are found by searching for all paths between the source node and sink node. Finding all paths, however, might detect non-forward paths that are part of the parallel paths. Deep View presents only the forward paths.

6.6 Diagram visualization

An accessible node-link diagram is of limited use for a blind user unless the diagram can be visualized and shared with sighted colleagues. Deep View provides two mechanisms for blind and sighted persons to transparently exchange diagrams without needing to know about another person's visual abilities.

The first mechanism is the most basic and comes with the standalone Deep View application. The Deep View application uses a GraphViz web service (Gansner and North 2000) to automatically lay out and generate an image of a node-link diagram. The visual diagrams in this chapter were generated by GraphViz. GraphViz is a leading research tool in automatically laying-out and generating visual diagrams. Although automatically generated diagrams need to be aesthetically improved through further research, GraphViz manages to generate reasonable diagrams for most examples. Advanced Deep View users can customize the visual appearance of the GraphViz diagram. In the same file in which the user defines a new diagram and its terminology, the user specifies the visual characteristics of a diagram's elements, such as shapes, lines or colors, provided through GraphViz.

A blind person shares the diagram with a sighted person by copying the GraphViz image into a document, email, file, etc. Unfortunately a sighted person cannot readily edit the GraphViz visual diagram, which is an image file.

Deep View provides a second mechanism that enables a two-way exchange for viewing and editing diagrams between blind and sighted persons. Deep View is integrated into several visual diagram applications, which a sighted person uses to view and edit diagrams. The Deep View interface is accessed within the diagram application. The blind user can view and edit existing diagrams with the Deep View interface. This way the blind and sighted persons can transparently exchange diagrams because changes made in one person's interface are readily available in the interface of the other person.

We have prototyped Deep View plugins to various extents for the following three diagram applications.

- **IBM® Rational Rose ™**, a software engineering tool for UML diagrams. Deep View provides complete access to editing state-chart diagrams; edits in the Rational Rose or Deep View interface are updated in the other interface. We have also implemented a proof-of-concept for editing UML class diagrams with Deep View. Other diagrams could be accessible, too, if the Deep View extension were to be implemented.

- **Microsoft® Visio ™**, a general diagram application. Deep View can be customized to edit a large variety of Visio diagrams as long as they conform to the standard Visio diagram protocol. Our main emphasis for the Visio plugin is to make ERDs accessible. The Deep View plugin implementation enables a blind user to view the Visio ERD or to import a Deep View ERD into Visio. With further implementation, Deep View could support diagram edits interactively appearing in the other person's interface.

- **Eclipse and Omondo UML Live ™ tool**. Eclipse is a general integrated development environment (IDE) mainly used for Java applications. The Omondo UML tool is an Eclipse plugin specifically for viewing and editing UML diagrams of software projects in Eclipse. Our prototype is a proof-of-concept for how UML class diagrams can be accessed by a Deep View interface, which would be an independent Eclipse extension. Limitations in the Omondo programming interface, however, limit Deep View from having full control of UML diagrams. Therefore, the Deep View user does not have full editing control over the UML diagram.

The Deep View plugin into Rational Rose also enables a blind person and a sighted person to work on a diagram at the same time. As one person edits the diagram, the changes are updated in both the Deep View and Rational Rose interfaces. Microsoft Visio diagrams could be edited simultaneously, but the Deep View plugin implementation has to be extended. More information about the Deep View plugins is in the following implementation section.

6.7 Implementation

The Deep View implementation leverages a screen reader by providing a GUI with standard GUI widgets that the screen reader presents to blind users. Several other accessible diagram applications, such as TeDUB (Petrie, Schlieder et al. 2002) and DocExplorer (Ishihara, Takagi et al. 2006), take advantage of the same technique. As mentioned in the description of the Deep View interface, leveraging the screen reader and GUI interface is beneficial for blind and sighted users. Leveraging the screen reader is also beneficial for the development of accessible applications. A sighted developer can program the GUI with familiar standard widgets. The developer can then customize the control of the widgets to accommodate a blind user; for example, he would customize the hyper-linking interaction in the treeview in Deep View.

Using the screen reader also simplifies the implementation. The implementation does not need to include the complexities of setting and controlling a text-to-speech engine. This functionality is handled by the screen reader. However, a trade-off to this design is the requirement of a screen reader. Sighted users and developers are unlikely to own expensive screen reading software.

Deep View is a Java application. Deep View uses the Java SWT package to create the GUI. The unique feature of SWT is that it creates the interface from native GUI widgets. This means that the interface consists of widgets for which the screen reader has been optimized. The SWT package contrasts to the Java SWING interface, which creates custom GUI widgets. Accessibility of SWING GUI widgets is limited and complicated compared to that of native GUI widgets.

6.7.1 Deep View plugin

Deep View can be integrated into visual diagram applications that provide a mechanism for third party plugins. These applications provide an application programming interface (API) that exposes control over the application's diagrams. We assume the developers of the APIs intended them to enable third parties to customize an application to the processes used by the third party. In our case, we leverage the API to make the diagrams accessible. Of course, screen readers could provide accessible diagrams by using mechanisms similar to Deep View.

The available mechanisms to create third party plugins are specific to Windows and Java. Plugins for Visio and Rational Rose are integrated using Windows Active X components. The Deep View Java application is made into an Active X compatible component through the Java Active X Bridge. We use Visual Basic to integrate the Deep View Active X component into Visio. Similarly, with Rational Rose we use Visual Basic and the Rose Extensibility Interface (REI) to integrate the Deep View plugin. Eclipse provides its own Java extension mechanism, which was designed for programmers to enhance and create new programming tools.

Deep View has two objectives which drive the requirements for the visual diagram application. Deep View's primary objective is to provide a blind person an accessible interface to access and edit diagrams. The diagram application API should also support a secondary objective of Deep View, to support collaboration between a blind and sighted person working on a diagram at the same time. The sighted person views the diagram in the visual interface of the diagram application while the blind person accesses the same diagram through Deep View. As one person makes edits, the other person can see the changes in their corresponding interface.

These objectives for Deep View require the following functionality from the API. Edits refer to additions, deletions, or modifications of node, links, or attributes in a diagram.

1. The application model stores a diagram semantically and programmatically exposes the diagram model. Specifically the model would contain a diagram's nodes, links, attributes and their respective relationships.
2. The application model component must be editable.
3. As the Deep View user edits the diagram, Deep View's modifications to the application model component of the diagram must be automatically refreshed in the visual representation of the diagram.
4. As the visual interface user edits the diagram, the diagram application must trigger an event that Deep View can capture. Deep View uses the event to update its representation of the diagram and presents it to the blind user.

Requirements 1-3 are the core requirements for making the diagram accessible. Requirement 1 is the essence of Deep View's representation of the diagram. Requirements 2 and 3 allow the Deep View user to edit a diagram. Editing a visual diagram enables the blind person to transparently share diagrams with sighted persons. Requirement 4 enables the blind person to examine the sighted person's edits while the pair works together simultaneously.

It is convenient for a sighted person if the diagram application supports automatic diagram layout, saving the user the effort of laying out the diagram. However, the automatic layout may produce aesthetically poor diagrams and it may still be necessary for the sighted person to modify the visual layout of the diagram.

An alternative to Deep View making diagrams accessible is for the visual diagram applications to provide an accessible interface. Many visual diagram applications already have a browser view, which is intended as an alternate organization to the visual layout. The browser displays the diagram elements in a treeview widget similar to Deep View. However, the organization and interaction of the browser treeview is unlike Deep View. The existing treeview could be modified to display the diagram similar to Deep View and provide additional keyboard shortcuts. A drawback to realizing the accessible browser treeview is that it has minimal benefit (i.e. additional new features) for general users.

Chapter 7
Single User Deep View user study

7.1 *User study design*

We conducted a user study to evaluate basic usability of the Deep View interface. In the study, blind participants complete a series of diagram tasks, which we use to measure their understanding of node-link diagrams. We also observe a blind participant's strategy using Deep View to learn about a diagram.

We compare the performance of blind participants to sighted participants. Sighted persons complete the same tasks as the blind person but use a typical visual diagram application, in our case, Rational Rose. Of course, we expect a sighted person to complete the diagram tasks faster than a blind person because diagrams are suited to be processed visually. However, the sighted person's performance is the ideal performance we hope blind persons' could approach. We hypothesize:

Blind participants will be able to comprehend a node-link diagram using the Deep View interface similar to a sighted person.

Although this study investigates individual participants experience accessing node-link diagrams, the results are important for the following user study to investigate a blind person and sighted person collaborating to edit a node-link diagram. In order to collaborate, we need to verify that blind participants have a solid understanding the diagrams and can discuss them similar to a sighted person. Furthermore, the user study

implicitly practices the remote collaboration as participants must interact with the experimenter to discuss diagrams.

7.1.1 Independent variables

A participant's session consisted of the following. Participants read or edit six diagrams about familiar everyday topics. Diagrams are of two types to demonstrate the variety of diagrams Deep View can represent. Diagrams average ten nodes and ten links for a reasonable complexity for a participant to manage in a user study session. The diagrams are presented in a predetermined random order to counter ordering effects. Sighted participants completed the study in one hour whereby the blind participants used one hour for training and two hours for completing the diagram tasks. The training reviewed concepts about diagrams and instructions on using Deep View. Finally, participants completed a post study questionnaire. The questionnaire, instructions, and other forms used in the user study are available electronically for blind participants to access with a screen reader.

In the editing task, a participant must complete a diagram given a textual specification. The nodes of the diagram are provided and the participants must create the links between the nodes. In the reading task, participants are given a diagram and an overview of its context.

The two types of diagrams are a flowchart and a categorization diagram. An example of a flow chart is the procedure of heating food in a microwave; if the food is not hot enough it is heated again, which is represented as a loop in the flowchart. Participants are asked four questions about a flow chart. The first three questions are low level details about a specific node and its relationship to other nodes. The fourth question is about a higher level concept expressed in the diagram, such as what diagram nodes are involved in the loop to reheat the food in the microwave.

A categorization diagram is similar to a tree data structure. An example is categorizing common animals by species. A tree node represents a category (species) and leaf items are specific animals. A diagram link's direction is from a parent to a child; in other words, a child node is "in" a parent node category. There are three questions about

identifying the common category two leaf items share; for example, the first and simplest question is to identify the common parent of two immediate leaf nodes.

7.1.2 Measurements

Our measurements are used to compare the performance of blind participants to sighted participants as they complete the diagram tasks. A high level comparison is based on the time to complete the diagram tasks. The relative difference will indicate how much faster sighted participates are.

A participant's understanding of the diagrams is recorded in two metrics. The first metric is the accuracy with which participants complete editing diagrams. The second metric is the accuracy with which participants answer questions about diagrams. Furthermore, the experimenter asks if the participant answered the question from memory or by referring to the diagram. This question gives insights into how participants process diagrams.

In a post experiment questionnaire and discussion we gather feedback on the user interfaces. Specifically for Deep View we focus on suggestions to improve the usability of the Deep View interface.

7.1.3 Participants

Five sighted persons and five blind persons participated in the study. Three participants are congenitally blind, one is congenitally low vision, and one is low vision since childhood. All participants rely on a screen reader for their everyday tasks as a college student or working professional. The blind participants self report learning about node-link diagrams but rarely using them. Sighted participants use node-link diagrams occasionally or regularly. Our comparison does not factor in the different levels of experience blind and sighted participants have with diagrams. The group of sighted participants completed a total of 30 diagrams. The group of blind participants completed 26 diagrams in total because of time limitations.

7.1.4 Computer setup

The diagram editing applications are the main equipment used in the user study. Sighted participants edit node-link diagrams in Rational Rose, a software development application features various kinds of node-link diagrams. Blind participants access the node-link diagrams using the Deep View interface and a screen reader to navigate and read the textual content in the Deep View interface.

The user study is designed to accommodate participants remote from the experimenter in order to broaden the scope of people who can participate. Two of five sighted participants and two of five blind participants completed the study remote from the experimenter. We used a screen sharing application, such as VNC, for participant and experimenter to examine the diagrams. Working with a blind participant, the experimenter used the networking capabilities of the Deep View system to examine a blind participant's diagram as it is edited. For convenience, the experimenter examined the visual diagram generated by Deep View instead of the textual version the blind participant accesses.

7.2 Results

We compare the performance of sighted participants and blind participants completing the tasks of reading and editing diagrams. Our analysis treats the tasks with flowcharts and brainstorm diagrams in the same group because the diagrams are of similar complexity. The measures of performance include time to complete the task, accuracy in editing diagrams, and accuracy of answering questions about the diagrams. Overall, the blind participants have a good understanding of the diagrams, which in this study are relatively simple. The small sample size of the study means the results cannot be generalized to general populations. The results apply to the participants in the study and suggest trends that future studies could further substantiate.

7.2.1 Time to complete diagram tasks

As we expected, sighted participants complete the tasks of reading and editing diagrams considerably faster than blind participants. On average a sighted participant is 4.7 times faster than the blind participant reading an existing diagram. Figure 7-1 shows a

box-plot of the distribution of the participants' time to complete reading a diagram (we measured 15 and 13 samples for the sighted group and the blind group respectively).
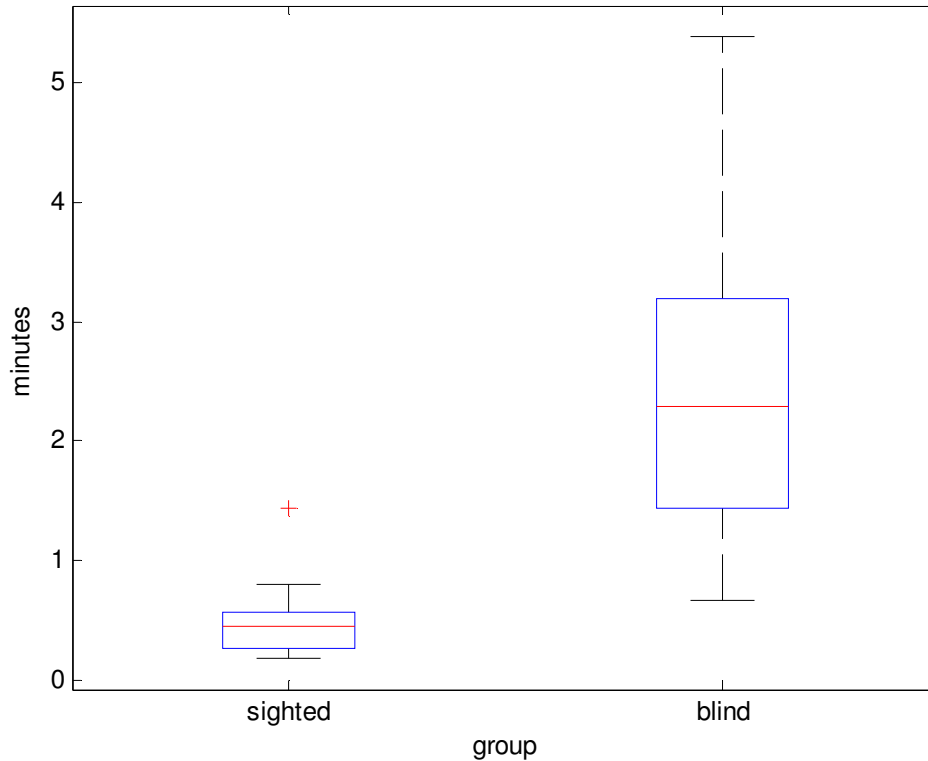
Figure 7-1. Average time to read existing diagrams by the sighted (N=15 samples) and blind groups (N=13 samples)

The sighted participants edited diagrams faster than blind participants; however the difference is smaller than for reading a diagram. On average a sighted participant is 2.99 times faster than a blind participant. Figure 7-2 shows a box-plot of the distribution of the participants' time to complete editing the diagrams.
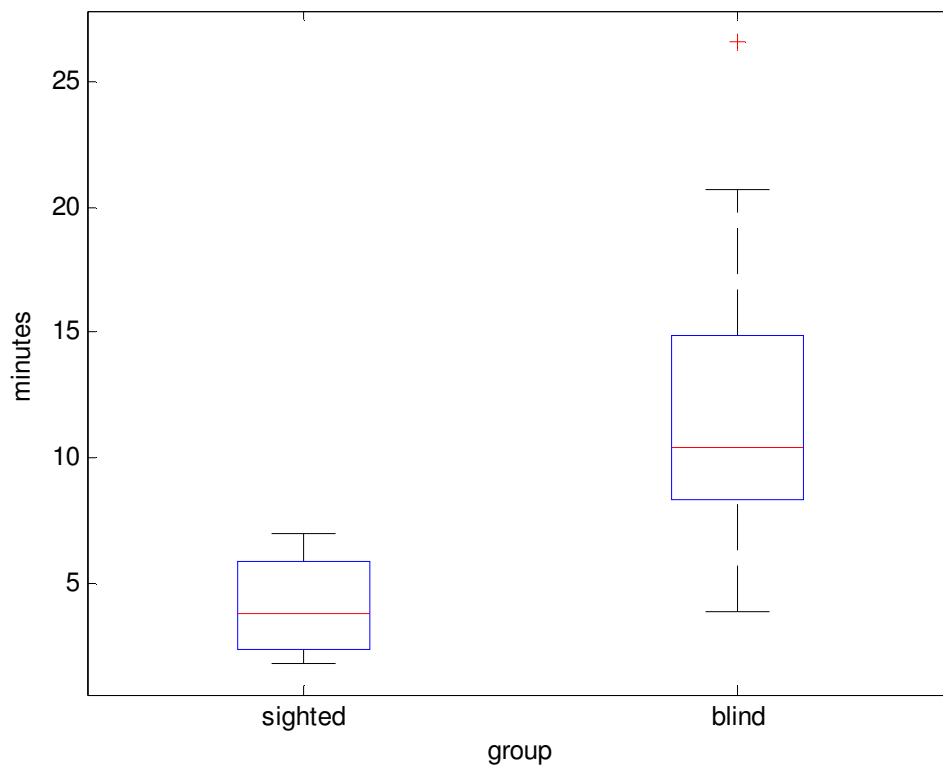
Figure 7-2. Average time to complete editing diagrams by by the sighted (N=15 samples) and blind groups (N=13 samples)

The discrepancy in time to complete the reading and editing tasks comes from the difference in nature of the tasks. We suspect the discrepancy is less when editing a diagram because a blind participant spends a smaller portion of time navigating the diagram. Both sighted participants and blind participants spend more time comprehending the textual description and making the edits to a diagram than simply reading a diagram. This is supported by the fact that the absolute time to complete the editing task is larger than that to complete the diagram reading task. All participants used a similar strategy to edit a diagram: reading the text description, editing the diagram (occasionally referring back to the description), and making a final review of the diagram by comparing it to the textual description.

A further study would be needed to compare the speed with which sighted participants and blind participants specifically complete edits to a diagram. The users' strategies are too different to otherwise compare. A sighted user adding a link to the

diagram must spatially layout the nodes to connect and then click-and-drag the diagram link between the nodes. The blind person completes the same edit using a Deep View dialog to create the link and selects the nodes to connect from a list of nodes.

7.2.2 Diagram understanding

We measured a participant's understanding of the read and edited diagrams in their answers to the questions about diagrams. It was trivial for sighted participants to answer the questions correctly. Sighted participants typically glanced at the diagram to find the answer. The remaining results discussed in this section focus on the blind participants' understanding of the diagrams.

The blind participants demonstrated a strategy different than sighted participants for studying and understanding a diagram. Blind participants answered most of the questions about the diagrams from memory. So although it takes longer for the blind person to read or edit a diagram, the blind person can discuss large portions of the diagram without referring back to it.

Blind participants have a solid understanding of the diagrams. In the post questionnaire, blind participants self reported their overall understanding of the diagrams being mostly confident but in some cases having some uncertainty. Relating specifically to the questions answered by participants, blind participants self report being very confident or confident about the answers to their questions. Of 77 total completed questions answered by blind participants about the diagrams, 95% of the questions were answered correctly. Five percent of the questions were answered incorrectly, however after the experimenter referred the participant to the diagram, the participant could identify the correct answer.

While blind participants answered most questions from memory, 22% (17 questions) were answered by referring to the diagram. In these cases, the participant knew the answer approximately but needed to refer to the diagram to confirm the answer or reference the exact name of the specific node. This demonstrates that participants can use Deep View to search and find specific information. The sighted person, however, is faster at looking up an answer; a blind participant takes an average of 23 seconds to look up an answer in the Deep View interface compared to the sighted person who can glance

at the diagram to identify the answer. Figure 7-3 shows the distribution of the response time when a blind participant uses Deep View to look up an answer to a question.
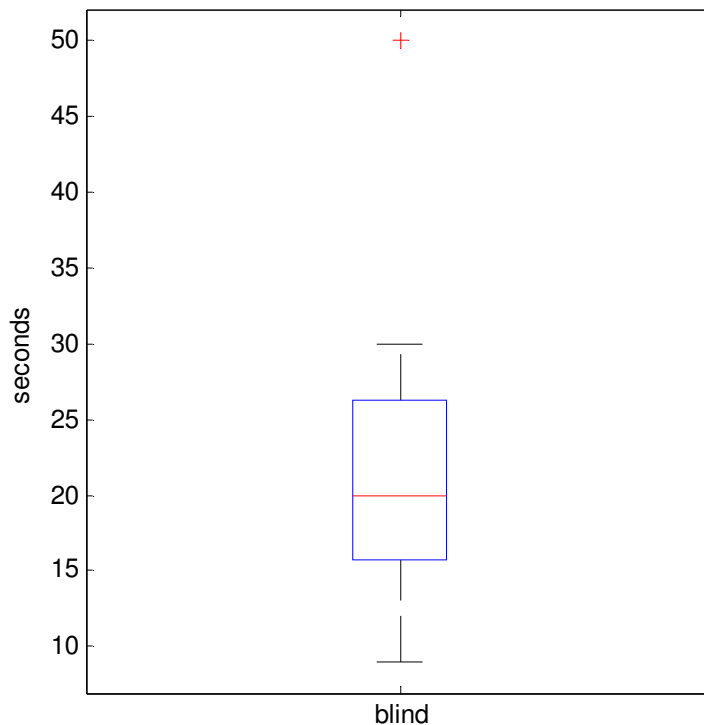
Figure 7-3. Time to look up answer in Deep View interface by blind participants (N= 77 questions)

Besides the mentioned 77 questions, there were an additional 16 questions about high level diagram characteristics in the flow charts. The blind participants answered the high level questions with varying degrees of proficiency. Three of five blind participants answered all the questions correctly. The other two blind participants could give information related to a question but not the exact answer; to the experimenter this seemed to come from a blind participant's misunderstanding of concepts about the high level characteristics. More training and experience with node-link diagrams would improve the blind participants' understanding of the diagram. For those participants that answered correctly, it shows an advanced understanding of a diagram. Of the questions about high level diagram characteristics, participants answered half of the questions from memory and the other half by referring to the diagram.

7.2.3 Editing diagrams

Editing diagrams is another way participants demonstrate their understanding of diagrams. It is straightforward for sighted participants to complete the task; editing the diagram is done with the familiar mechanism of clicking and dragging parts of the diagram in a visual interface. The blind participants completed the diagrams correctly with occasional errors described below. In the post-study questionnaire blind participants self report being very confident or confident about editing a diagram. However, the blind participants made two common mistakes, which are insightful for improving the Deep View interface to mitigate users' mistakes. As a whole the participants completed editing 13 diagrams.

The first common error occurred in editing three of 13 diagrams. It was omitting a node from the diagram, in other words, the omitted node did not have any links connecting it to the rest of the diagram. It was an oversight by the blind participant. Inspecting the final version of the diagram the omitted node is difficult to detect; the existing diagram would appear logically correct. The omitted node is simply an extra detail. The Deep View interface can be improved by adding a query for high level characteristics to group nodes connected to each other. For all connected nodes the query would return one group but omitted nodes would appear in their own groups.

The second common error occurred in three of 13 diagrams. It was placing multiple links between two nodes. In one case the blind participant could recognize the error and fix the problem by removing the duplicate links. Another blind participant did not recognize the mistake. Deep View could be improved by prompting a user when a duplicate link is created. This technique would have to be tailored to the diagram because some diagrams, such as UML sequence diagrams, can have multiple links between two nodes.

Finally, it should not be taken for granted that blind participants would create diagrams as the experimenter expected them; 12 of 13 diagrams were completed as expected. Completing the diagram as expected involves the blind participants properly creating cycles or parallel paths in a diagram. In contrast, one blind participant had a unique interpretation of the textual description and created a diagram very different from the expected result as shown in Figure 7-4. Figure 7-4 (a) shows that the expected result

has a cycle representing how paper repeats a process when it is recycled. Figure 7-4 (b) shows the blind participant's unique interpretation. It is logically correct and represents the inputs and outputs in the stages of the process.

a)   b)
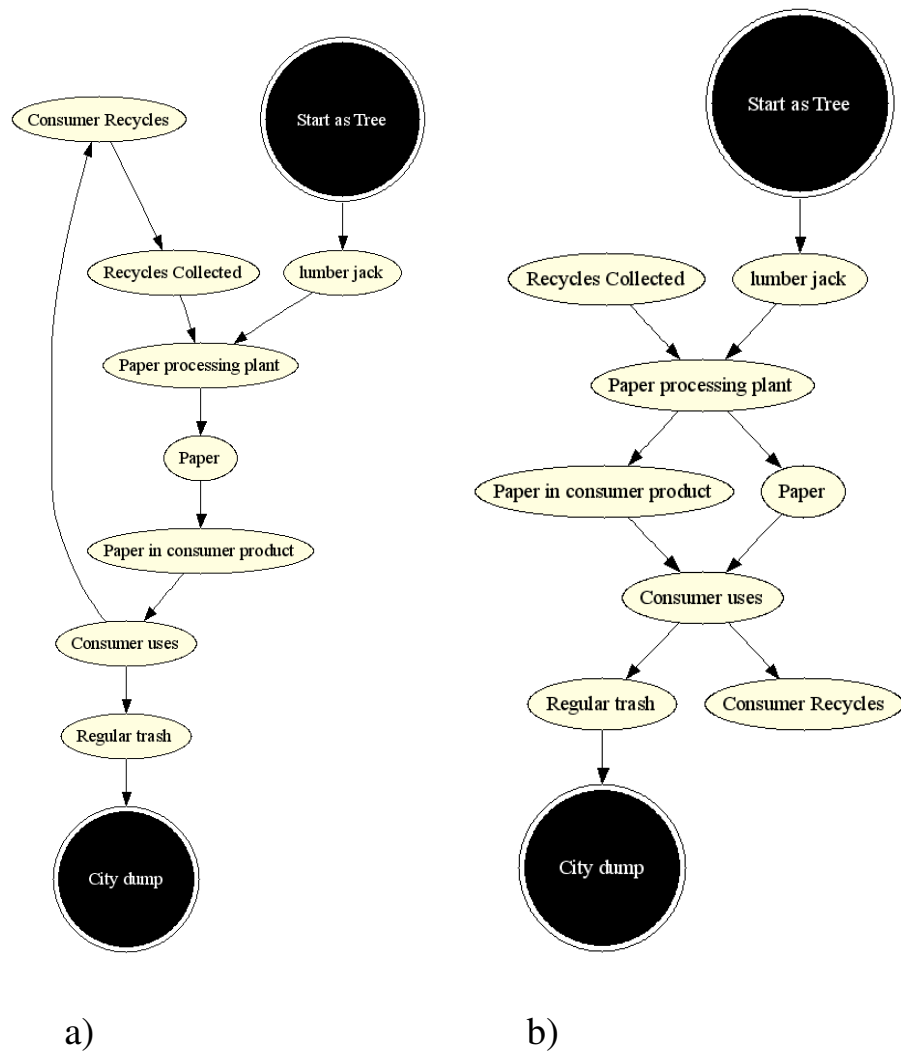
Figure 7-4. Diagram shows recycle process of paper. a) Expected result, b) One blind participant's unique interpretation

133

7.2.4 Spatial layout

Although Deep View does not present a diagram's spatial layout, the impact of spatial layout was evident, specifically in the brainstorm diagrams. The blind participants could successfully complete the tasks with the brainstorm diagram. In the initial training, however, blind participants were confused by the brainstorm diagram, which is similar to a tree data structure. The spatial layout of the brainstorm diagram conveys how categories and subcategories are grouped; a subcategory diagram is spatially placed below its parent category and items in the same category are placed at the same level.

Three of five blind participants were confused by how subcategories are represented in a brainstorm diagram. Consider the example of a brainstorm diagram in Figure 7-5 where the main category is North Carolina. As indicated by a link, the item "icy winter" is in the "weather" category, which is in the main category. The blind participants expected there to be an additional link from the main category, "North Carolina" to "icy winter" to represent that grouping too.
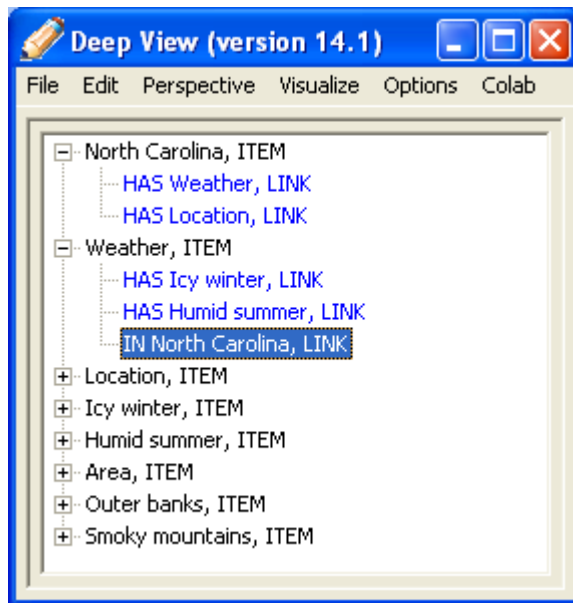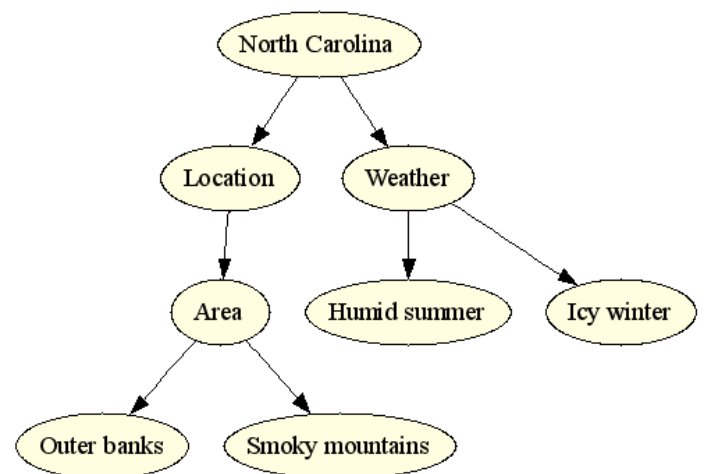
a)   b)

Figure 7-5. Brainstorm diagram with main category North Carolina; a) Shown in Deep View b) Visually represented as a tree structure

In Deep View a user determines how an item is categorized by navigating links from a given item through the parent categories to the main category. Alternatively the

user can query for the path from the main category to a given item. The queried path lists the subcategories between the main category and given item.

The Deep View interface can support a representation of the brainstorm diagram that the blind participants expressed a preference for. The preferred representation is to present the brainstorm diagram like directories in an operating system. The main category is the root directory and direct subcategories are directories within the root directory. Deep View can provide this representation with subdiagrams; one subdiagram represents a subcategory. Figure 7-6 shows the Deep View treeview when the brainstorm diagram is constructed with subdiagrams.

Figure 7-7 shows the visual representation of the diagram when the Deep View diagram is represented with sub-diagrams. Instead of a tree data structure, the visual diagram is similar to a Venn diagram. This experience suggests that the Deep View presentation in the treeview could be decoupled from the visual representation. Then the blind user could use the preferred subdiagrams in the Deep View interface and choose if the visualization is constructed as a tree structure or Venn diagram.
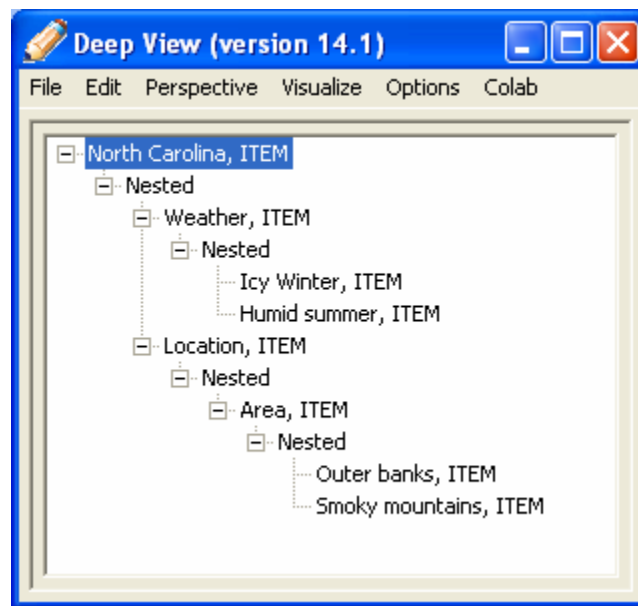
Figure 7-6. Alternative representation of a brainstorm diagram using Deep View's subdiagrams.
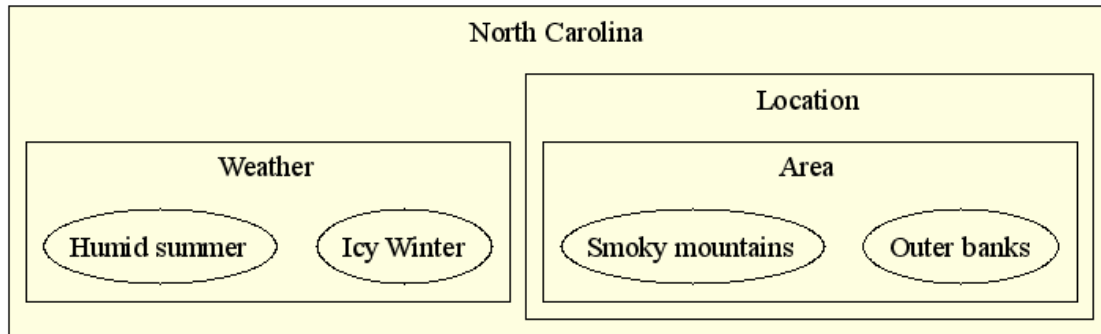
Figure 7-7. Alternative visual representation of a brainstorm diagram when the diagram is constructed with Deep View's subdiagrams.

7.2.5 Deep View usability comments

Overall the blind participants reported having a positive experience using the Deep View interface. Their suggestions for improving the interface are related to streamlining the interaction with the interface. This includes reducing redundancies so that a user can complete a task faster or less tediously. Following are three suggested improvements.

The first suggestion is to reduce redundant information in the dialogs to create a new link or node. In the dialog, the user can select the category of node or link. It is tedious for the user to review this option when there is only one category to choose from. The improvement is to remove the selection of the category when there is only one possible category. This suggestion has been implemented in Deep View.

The second suggestion is related to searching for items in a diagram. In the search dialog the user enters the term to search for. The suggestion is to remember the last search term. This way a user can switch between the search dialog and full diagram without having to reenter the search term.

The third suggestion is related to the tasks in the user study and might have less general applicability. At least three of five blind participants suggested creating multiple links within one Deep View dialog. For example in the brainstorm diagram, multiple items would be linked to the same category.

136

7.3 Discussion

Our user study shows that blind participants develop a solid understanding of diagrams when using the Deep View interface. The blind participants demonstrated their comprehension through high accuracy answering questions about the diagrams and correctly editing diagrams. This suggests blind participants can use Deep View to independently study and create diagrams. The blind participants demonstrated a unique strategy of memorizing large portions of the diagram compared to sighted participants' strategy. Furthermore, blind participants are considerably slower completing tasks related to reading and editing diagrams. These findings have several implications for the blind person working with diagrams and sharing diagrams with sighted colleagues. As mentioned, the results apply only to the participants in the user study because the sample size is too small to represent the general population.

The blind participants' strategy for reviewing a diagram has implications for blind users comprehending larger diagrams. Given the long time to review a diagram, a blind user might need to spread the review over several sessions so that the user can comfortably concentrate on reviewing the diagram. Also a large diagram might be more than a person can comfortably memorize. Of course, blind participants will be able to take advantage of grouping information and using their familiarity with the diagram's subject matter to comprehend large and complex diagrams.

We suggest three strategies a blind user could use to manage reading and editing large diagrams. The first strategy would be to use Deep View in conjunction with an accessible diagram interface that presents a diagram's spatial layout, which provides the user a different perspective on the diagram. The second strategy is to use advanced navigation tools, such as Deep View's query for high level characteristics (to be evaluated in other user studies). In the third strategy, blind participants could adopt techniques sighted persons use to study a large diagram that is too large and complicated to simply process by glancing at the diagram.

Ultimately a blind person would collaborate with a sighted colleague to discuss and edit diagrams. Results from our user study contribute to insights into one aspect of how the collaborators could work together. Specifically it would be helpful to the collaboration if the collaborators can work at a similar pace so that one person does not

have to wait much for the other. Our results suggest that the collaborators could fluently discuss a diagram that both collaborators are familiar with. The collaborators can easily and quickly reference information in a diagram; a sighted person glances at the diagram while the blind person has large portions memorized. Also, the blind person can reference information in the diagram when needed.

Furthermore, it might be time-efficient for the blind person to study an existing diagram independently because it takes the blind person considerably longer to study a diagram than a sighted person. In future research we will investigate how it is for a blind person and a sighted person to create and edit diagrams at the same time. Although the blind person edits a diagram slower than a sighted person, the interaction is more involved as the collaborators are exchanging ideas in a conversation.

Chapter 8
Deep View collaboration interface

8.1 Shared workspace

Collaborative applications are necessary for a blind person and sighted person to work together. Two persons without disabilities can use a single user application to view the same information and communicate face-to-face. However, A blind person and sighted person can not easily collaborate on a single user application. The conventional screen reader a blind person uses to control the computer is difficult to use with two people; when one person controls the computer, it is difficult for the other person to follow. Instead collaborative applications such as the Deep View system enable both persons to access the same diagram through the most appropriate interfaces.

The Deep View system provides a loosely coupled workspace to support collaboration between a sighted and blind person. Loosely coupled refers to the feature of a shared workspace where the presentation and control of the workspace interfaces differ for the collaborators. In our research, the presentation of the sighted user's and blind user's workspace is different to accommodate their needs. The sighted person uses a typical visual diagram interface. The blind person uses an audio interface, i.e. the Deep View interface. The workspaces have in common that they represent the same diagram's nodes and links between nodes. The Deep View system maintains the common model for the user's workspaces.

The blind person and sighted person use the shared workspace to interact with a diagram at the same abstract level. The language the collaborators use to describe the diagram is in the same frame of reference, such that referring to a specific diagram node

or link is understood the same by both persons. To avoid confusion, the collaborators should not refer to details of the specific interface, for example, the sighted person referring to colors of nodes or links in the visual interface.

Besides different presentations of the workspace, the users' control of the workspace is independent. The collaborators can equally add, remove, and change node or links of a diagram. One person's changes are immediately reflected in the other collaborator's interface. The collaborators can also navigate the diagram independently. A sighted person can, for example, scroll the diagram window, without impacting the blind person's interface. Independent navigation is vital for the blind user because to understand the diagram he must navigate between a diagram's nodes and links in the interface. For example, as the sighted person edits the diagram, the blind person can independently explore information related to the edit besides the immediate edit. This loosely coupled workspace contrasts to a tightly coupled workspace where each person's manipulation of the workspace, such as scrolling the window in the visual interface, is mimicked in the other collaborator's interface.

As mentioned, participants refer to a diagram's nodes by their names or descriptions. Alternatively, Deep View provides a mechanism for the participants to explicitly point at diagram elements in the interface. In contrast, two sighted collaborators would use, for example, a telepointer to point at diagram elements in a visual interface. Deep View provides a semantic pointing mechanism where one user selects one or more items to point at and the Deep View system highlights the corresponding items in the other person's interface. The pointing mechanism is described in Section 8.6.

The collaborators exchange ideas by communicating verbally. The Deep View system does not provide an audio connection between the collaborators. Instead the collaborators can use other technology, such as a telephone call or an audio connection provided by instant messaging programs.

8.2 Basis of collaboration

The basis for a blind-sighted pair to collaborate differs from two sighted persons collaborating. The blind-sighted collaborators inevitably infer different information from the diverging representation of a diagram. A diagram, however, is only a representation

140

of the larger concept to discuss. The collaborators can still achieve the larger objective of discussing the concepts expressed in a diagram; for example, for a flow chart the collaborators can discuss the process expressed in the flow chart. The collaborators will be able to build on their knowledge of the domain specific task in which the diagram is used.

Although Deep View makes many aspects of a diagram available to a blind user, it cannot represent all characteristics. The most obvious information Deep View does not represent to a blind user is the layout of the diagram. The layout reveals information such as the hierarchy of a diagram's nodes represented by arranging elements from top to bottom or grouping related elements in close proximity.

Deep View does present some characteristics of a diagram represented in the visual layout of a diagram, such as a cycle in a flow chart. A sighted person and blind person, however, must think about the diagram differently to access the same information, such as a cycle. For a sighted person the diagram layout draws his attention to the cycle characteristic. On the other hand, the blind person must consciously take the initiative to identify a cycle. Specifically, the blind user must have the idea to search for a cycle and then query the Deep View interface for cycles. Then Deep View returns all cycles and the diagram nodes included in the cycle.

Although the blind and sighted collaborators might have different understandings of a diagram, the pair can still collaborate on exploring and editing a diagram. The different understanding can shape the relationship between the collaborators. The pair might be able to work as equals in editing the diagram. On the other hand, one person might take the position as an instructor. For example, the sighted person can provide the blind person with insights to the diagram the blind person does not get from Deep View. In either case, the collaboration can be productive.

8.3 User interfaces

We designed the Deep View system to support a practical situation where a blind person and sighted person would collaborate on diagrams. In particular we enable the collaborators to discuss state-chart diagrams in the Rational Rose application. Rational Rose is a popular software engineering tool for teams to design computer systems. With

Deep View, blind team members can participate in the design process by accessing diagrams and editing them collaboratively with sighted colleagues. The current implementation supports only state-chart diagrams but could be expanded to support other Rational Rose diagrams. A state-chart is one example of the many node-link diagrams that can be presented in the Deep View interface.

As the blind person and sighted person collaborate, each person uses the most appropriate interface; the sighted person uses the visual diagram editor Rational Rose and the blind person uses the Deep View interface. The collaborators can use different computers and their computers are connected through the Deep View system.

8.3.1 Sighted person's interface

The sighted person uses Rational Rose to access and edit a state-chart diagram. The Rational Rose interface shown in Figure 8-1 is an example of a typical visual drawing application. The user controls the diagram edits with a mouse and the diagram tool bar. A user creates a new diagram state by clicking the corresponding button in the tool bar and clicks on the diagram canvas to place it. A user creates a new diagram transition by clicking the corresponding button on the tool bar and dragging the mouse between the states to be connected by the transition. The diagram is visually rearranged by dragging the diagram elements within the diagram canvas.

Although Rational Rose is a proprietary application, the Deep View implementation uses techniques provided by Rational Rose to manipulate the diagram. The Deep View system can add or remove nodes and links; change text and font; and change the color of nodes. These techniques are used to reflect actions by the blind user.
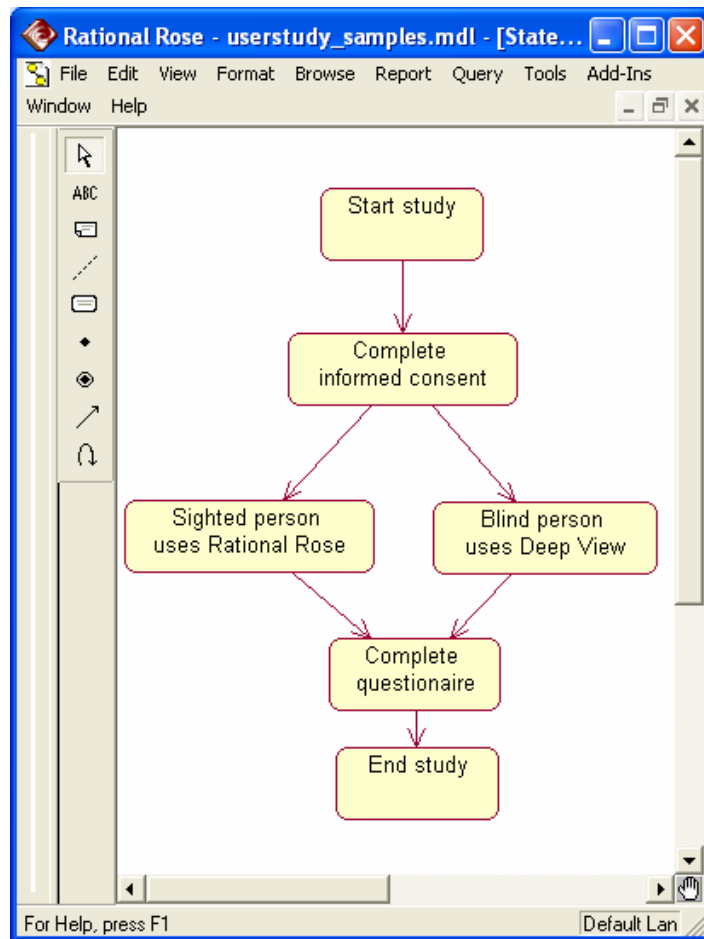
Figure 8-1. Rational Rose state-chart diagram for a sighted user. Two users are pointing at different items as indicated by green and purple colored states.

8.3.2 Blind person's interface

The Deep View interface is designed to enable a blind person and a sighted person to collaborate. A brief review of the interface: For a blind user, Deep View is an audio interface that the user interactively navigates to learn about elements of a diagram. Concretely, the Deep View interface is a GUI consisting of standard GUI widgets as shown in Figure 8-2. The blind person's screen reader reads aloud the textual description of a diagram. The main widget in the interface is a treeview, typically used to represent directories on a computer. The highest level of the treeview lists the nodes of the diagram and collapsed below each node are the links connected to it. Deep View provides a blind user several innovative mechanisms to navigate the elements in the diagram. A sighted person could use the interface but it would not be as practical as the visual interface.

143
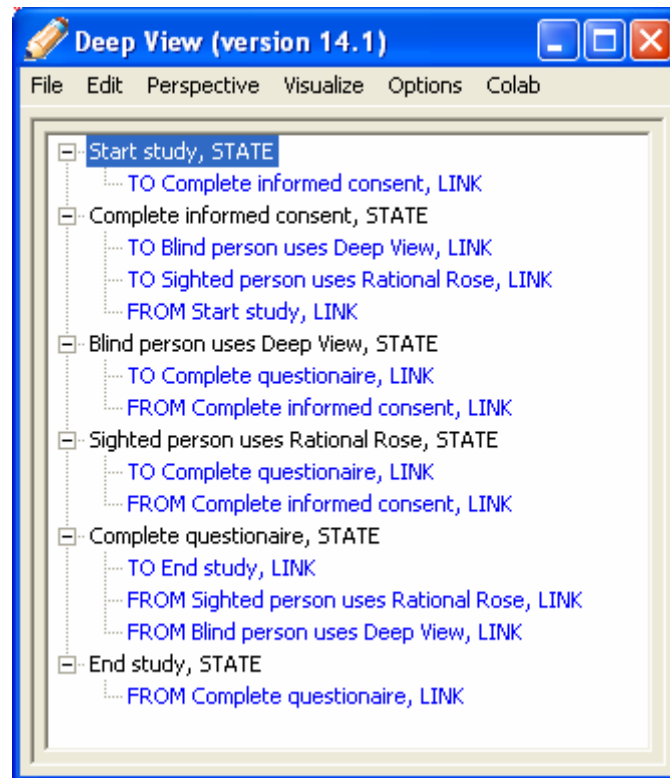
Figure 8-2. Blind user interface to access a state-chart diagram.

Deep View has four design features that specifically support collaboration. The first feature is that Deep View makes a node-link diagram accessible to a blind user. The second feature is that the blind user has complete control to edit the diagram including adding, removing and changing nodes, links, and attributes in a diagram. The third feature is that a blind user can interactively explore a diagram; at each step through the diagram a blind user listens to a short textual description, which identifies the corresponding diagram element, such as a node or a link. While collaborating, the blind user can quickly move between the diagram elements to focus on the elements being discussed with the sighted collaborator. Also, the textual descriptions are brief so that it is less disruptive when the sighted person wants to interrupt, for instance, to start a new thread of conversation. The fourth feature is that the blind person and sighted person can

144

use their interfaces to explicitly point at diagram elements. The pointing mechanism is described in more detail in Section 8.6.

8.4 Editing the shared workspace

The blind person and sighted person have similar controls to edit a diagram and contribute ideas to the collaboration. Each user's interface reflects the edits the other person makes. A significant issue is for the collaborators awareness of each other's edits. They must be able to identify when an edit occurs and recognize the details of the edit. We describe a sighted person's and a blind person's experience when editing a diagram.

When collaborating with a blind user, the sighted person is in the unique situation of solely managing the visual layout of a diagram. The sighted person manually lays out the nodes and links he and the blind person create. The Deep View system automatically places nodes created by the blind user in rows across the top of the visual diagram's canvas. As a blind user would add at most a few nodes at a time, it is manageable for the sighted person to keep up with arranging the visual layout.

A sighted person tracks edits by naturally detecting changes to the visual appearance of a diagram. For example, a new diagram link created by the blind user will appear as a line between the nodes connected by the link. Furthermore, the collaborators' discussion will likely already focus the sighted person's attention on the region of the diagram that will change.

In the other case, the blind person must keep track of the sighted person's edits. The sighted person's edits to the diagram appear asynchronously in the blind person's Deep View interface. For example, a new diagram node created by the sighted person is added to the list of nodes in the treeview. Deep View provides two mechanisms for the blind person to keep track of changes.

The first mechanism is auditory feedback. As the diagram is edited, Deep View plays a short audio icon. Each kind of edit (addition, removal, change) is categorized by a different audio icon. This is similar to an instant message exchange where a short audio icon signifies new messages. The audio icon informs the blind user about the change. The blind user can use this as a simple confirmation that the other person completed the agreed upon edit.

The second mechanism is for the blind user to retrieve a summary of changes to the diagram. The blind person uses a keyboard shortcut to bring up a dialog with a summary of the edits as shown in Figure 8-3. The dialog lists each edit and who made the change. The blind user can quickly confirm changes by reviewing the list. This is faster than traversing all diagram elements in the treeview to search for changes.
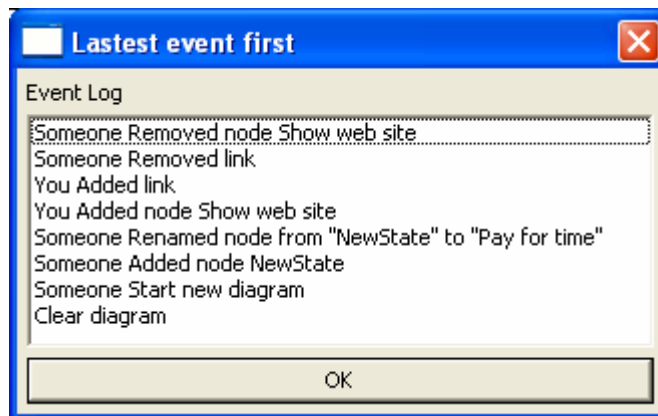
Figure 8-3. Deep View event log of edits for blind user to monitor

8.5 Connecting applications for collaboration

The collaborators use separate diagram applications and can work on different computers. The Deep View system includes a server through which the users' applications communicate with each other about changes to the diagram. As one person makes an edit, their diagram application informs the server, which sends the edit to the other person's diagram application. The server maintains a consistent model of the diagram.

To initiate the collaboration, the collaborators must connect their diagram applications to the Deep View server. The Deep View user connects to the server through options in the Deep View interface main menu. The user selects the "Colab" menu and the "Connect" sub-option. The user receives a dialog with a message indicating if the connection to the server was successful or not. In the current Deep View implementation, the server is fixed so that the user does not need to specify it. Of course, in a more general Deep View implementation a user would specify the server.

The sighted person operating Rational Rose uses the same mechanism as in Deep View to connect to the server. In fact, an instance of the Deep View interface is started

from within Rational Rose. The Deep View plugin is started from the Rational Rose "Tools" menu. Then the sighted person uses the Deep View interface to connect to the server as described. After that the sighted person does not need to use the Deep View interface which runs as a separate window to Rational Rose.

Once connected, each user can start a new diagram or load an existing diagram to share in the collaboration. In the Deep View interface, a user can start a new diagram or open an existing diagram from the main menu. In Rational Rose, a user selects one diagram in the Rational Rose project and uses the mouse context menu (right-click) to select the option "load into Deep View". As for saving a diagram, each user's copy of the diagram is stored independently: Rational Rose stores the diagrams in its format and Deep View stores the diagram in its own format, which is a simple text file.

8.6 Semantic pointing in interfaces

The Deep View system provides a unique mechanism for the collaborators to explicitly point at elements in the shared workspace. Through a semantic pointing mechanism collaborators can focus their attention on the same part of a diagram. One person selects an item to point at and the corresponding item is highlighted in the other collaborator's interface. This assumes that an item pointed at is realized in each collaborator's interface. Currently the interfaces in the Deep View system support pointing at states in a state-chart diagram, although there are other elements, such as transitions.

Three issues must be resolved to enable the collaborators to point at elements of the diagram. First, the users must be able to select the diagram elements to point at. Second, the diagram elements must be highlighted and brought to the other person's attention. Third, the interface must make the user aware of the event that items were pointed at. Following we describe how these requirements are realized in the Rational Rose and Deep View interfaces.

A feature of semantic pointing in the Deep View system is that collaborators can point at multiple items at the same time. A group of items pointed at by one collaborator are distinguished from items pointed at by the other collaborator.

8.6.1 Semantic pointing in visual diagram interface

The mechanism we use to highlight diagram nodes pointed at is to change the visual appearance of the corresponding node. Specifically, we change the background color of the node. Other visual characteristics could be changed too, such as a diagram node's text color or font, or outline color.

The diagram nodes pointed at by a specific collaborator are distinguished by a unique background color for nodes pointed at. The sighted person identifies his designated color when he points at nodes and they are highlighted. The sighted person learns what the blind collaborator's identifying color is when the blind person points at nodes and the background color of the associated nodes changes. Figure 8-4 shows a diagram where two collaborators are each pointing at multiple diagram nodes.
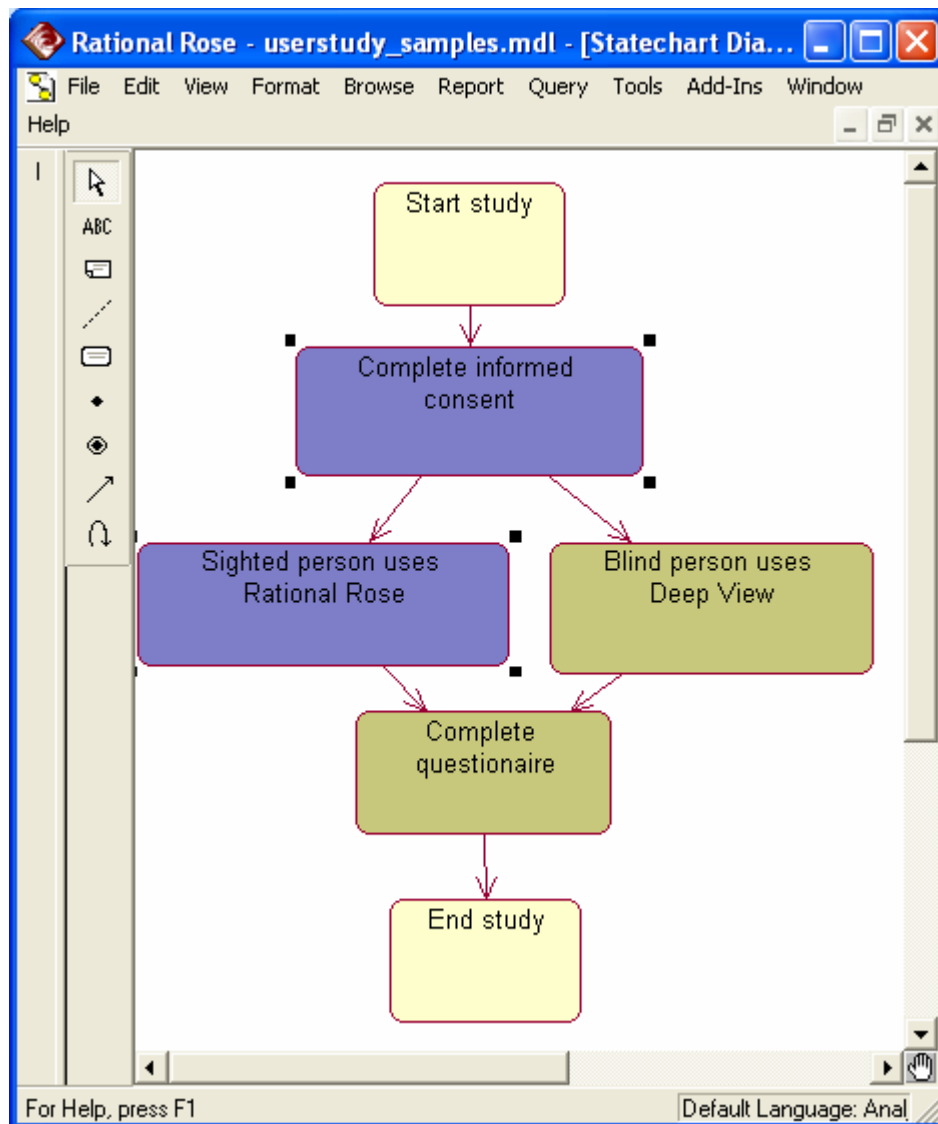
Figure 8-4. Multiple diagram nodes pointed at in the Rational Rose interface. The sighted user's nodes are highlighted in purple (dark color) and the blind user's nodes are highlighted in yellow/green (lighter color).

It is straightforward for a sighted person to point at one or more diagram nodes using the mouse. The user clicks on one or more nodes (using the shift key) to select the items to point at. Right-clicking on the diagram canvas, the user selects the "point at" option from the context menu. The selected nodes change their background color and give the sighted user feedback that the pointing action was successful. The nodes' background color remains until the sighted user explicitly clears the selection with another context menu selection, which is "clear point". When cleared, the background

149

color of the selected nodes changes back to the default color. To note, it is unusual to require a user to right click on the diagram canvas instead of one of the nodes to point at, however, it is a constraint of the Rational Rose mechanism for third party plugins, such as Deep View.

As the blind user points at diagram nodes, the background color of the selected nodes in the visual diagram canvas changes immediately. For a sighted person we assume it should not be too distracting from their task when the color of items pointed at changes. With a glance the sighted person should be able to recognize the items pointed at.

Like with nodes the sighted person points at, the nodes pointed at by the blind person remain highlighted until the blind user unselects them. It can be advantageous to keep diagram nodes highlighted until the collaborators unselect them. During the conversation, if the sighted person forgets the nodes pointed at, he can refer back to the highlighted nodes to remember. Furthermore, the continuous highlighting is useful if the diagram canvas is larger than the Rational Rose window and must be scrolled. The sighted person can identify the nodes pointed at by scrolling the diagram searching for the highlighted nodes.

In our Rational Rose implementation, we have not implemented a mechanism for two users to point at the same diagram node. In that case, the diagram nodes pointed at should have a different visual appearance compared to if only one person points at the node.

8.6.2 Semantic pointing in Deep View

We specifically designed the Deep View interface to support semantic pointing. The concept to enable pointing is that the shared workspace consists of discrete elements. This is realized in the Deep View interface by making the diagram nodes, links and attributes discrete entries in the treeview widget. It is straightforward for the blind user to point at nodes in a diagram. The blind person selects the nodes to point at and issues the "Point at" keyboard command.

It is more involved for the blind user to examine the nodes the sighted person is pointing at. Two issues must be overcome, which are caused by the fact that a blind person must use keyboard commands to explicitly navigate to the information of interest.

First, the blind user has to identify the nodes pointed at in a practical way. It would be impractical for the blind user to search all entries in the Deep View treeview for items marked as pointed at. Second, the sighted person's pointing action should not intrusively interrupt the blind user from his current task. For example, the Deep View interface would be intrusive if it automatically changed the focus of the screen reader away from the blind person's current focus.

Our novel solution is for the Deep View interface to present the blind user with a list of only those nodes that the sighted person is pointing at. When the sighted collaborator points at some diagram nodes, the blind person's Deep View interface provides a brief audio icon notification. The brief audio icon will not interrupt the blind user from listening to text currently read by the screen reader. When the blind person is ready to examine the nodes pointed at, he issues the keyboard command to list only the items pointed at. This changes the mode of the treeview from displaying the entire diagram to the pointing mode. Figure 8-5 shows the nodes that the blind person and another collaborator are pointing at.
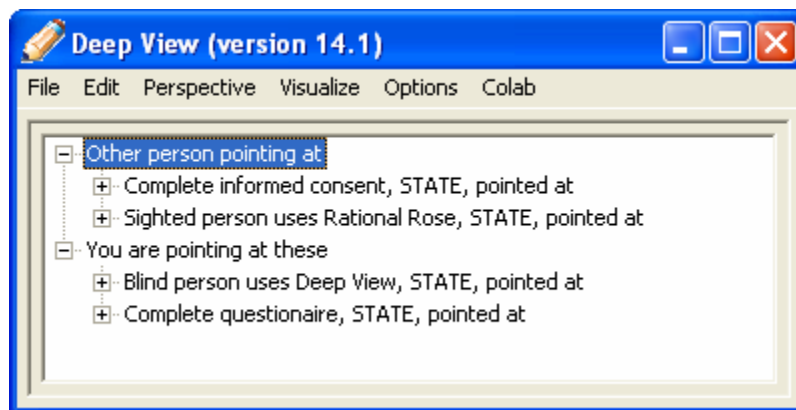
Figure 8-5. Multiple diagram nodes pointed at in the Deep View interface.

In this pointing mode, the top level of the treeview list has an entry for each collaborator. The description in the treeview entry distinguishes between nodes pointed at by the sighted person and the blind person. The treeview entry phrase for the blind person is "You are pointing at these". The phrase for the sighted person is "the other person is pointing at these". Expanding a treeview entry lists the nodes the corresponding user is

pointing at. Under the entry for the blind person, the blind person can verify the nodes he is pointing at. Under the entry for the sighted person are the elements the sighted person is pointing at. The lists of nodes will automatically update as the selection of items pointed at changes. The blind user returns the treeview from the pointing mode to displaying the diagram mode by a keyboard command, which is backspace.

Optionally, the collaborators can be further distinguished by their names. In this case the collaborators would enter names in the Deep View interface by selecting the "Colab" menu item. Given a name, the sighted person will be referred to by name instead of the phrase "other person". With two collaborators it is clear who "you" and the "other person" are. However, the names would clearly distinguish the collaborators when there are more than two collaborators.

8.6.3 Follow-me pointing

A unique feature of the Deep View system is that the sighted person can observe the node the blind person is currently inspecting. The blind person's current location in the diagram is selected and tracked in the Deep View treeview widget. Correspondingly, the blind person's current diagram node is visually highlighted in the sighted person's interface. Specifically, the node is highlighted by bolding and enlarging the font of the selected node as shown in Figure 8-6. This form of highlight distinguishes it from the diagram nodes explicitly pointed at. As the blind user moves to another node, the font of the original node returns to the default setting.
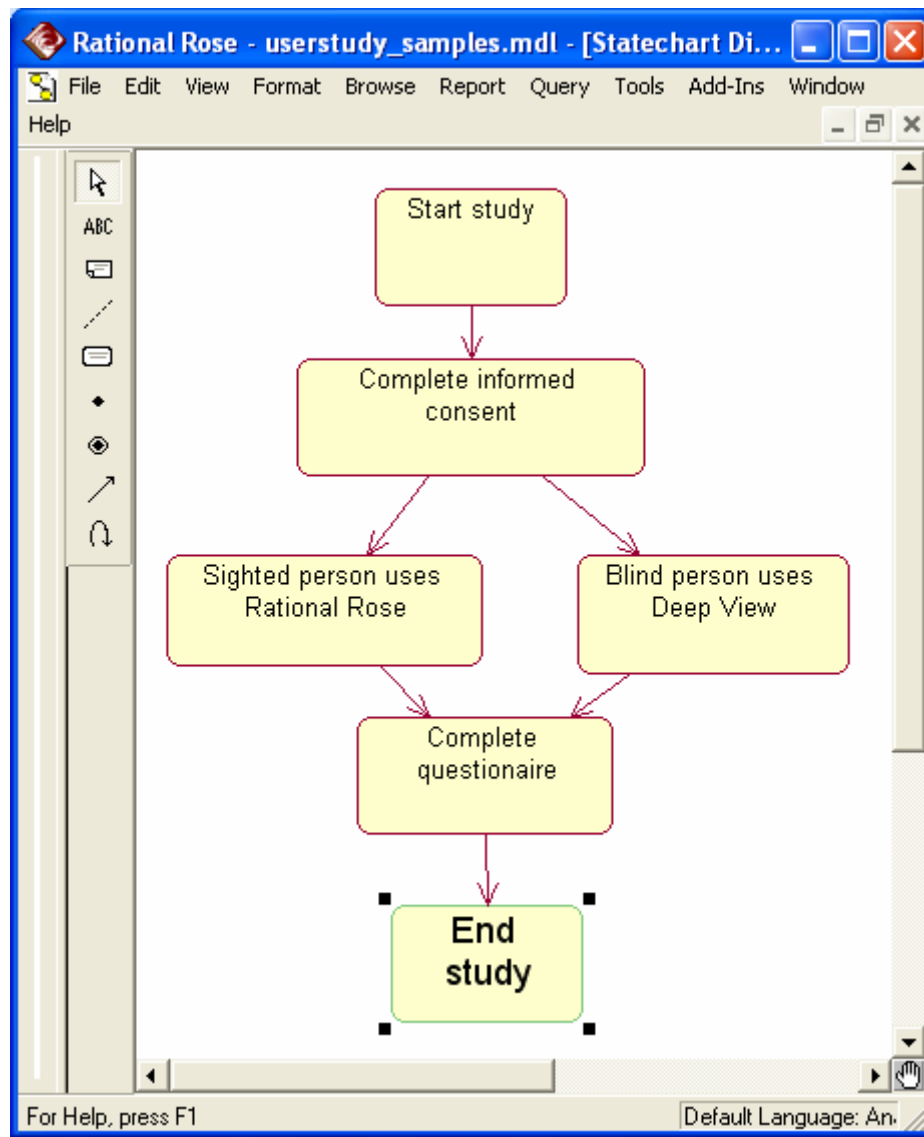
Figure 8-6 Diagram in Rational Rose indicating that the "end study" node is where the blind user is currently focused.

This feature is a form of a pointing mechanism and we refer to it as "follow-me" pointing. The sighted person can understand the blind person's reference to the current node without the blind person having to give the node's name or a description of the node. It is convenient for the blind person that it does not require an additional action, such as a keyboard command.

Besides a pointing mechanism, the follow-me feature gives the sighted person an awareness of the blind person's focus. The sighted person's awareness can help him plan his actions, such as choosing conversation topics or deciding on edits to the diagram. In

general, the follow-me pointing mechanism contributes to the CSCW research area of increasing collaborators' awareness of each other.

Unfortunately, the Deep View interface does not inform the blind user what the sighted person is viewing. The sighted person's current attention depends on where the user is looking which the visual diagram application does not track. Furthermore, the sighted person's mouse location does not accurately reflect where the sighted person's attention is focused because the mouse could be at any location while the sighted person examines the diagram.

8.7 Pointing in other accessible diagram interfaces

In the Deep View design we specifically choose to emphasize the discrete elements of a diagram to support semantic pointing between a blind person and sighted person. An alternative approach to making diagrams accessible emphasizes the spatial layout of the diagram. Such interfaces would require a different mechanism for a blind and sighted person to point at elements of a diagram. We have considered the concept of how such a pointing mechanism might function. A complication with such a pointing mechanism is to convey the spatial location of the item being pointed at.

Consider a node-link diagram that consists of discrete elements of nodes and links. The accessible diagram interface reads or sounds information about the diagram elements near the user's cursor location. For the blind person to realize what the sighted person is pointing at, the blind user's cursor must be moved to the position of the item pointed at. The pointing mechanism for such an interface differs depending on the two techniques by which the user controls the cursor. We consider the case in which the sighted person points at one item; pointing at multiple items would be more complicated.

In the first technique a blind user moves their hand over the space of the diagram. The user might be holding a pen, such as on a Tablet PC, or simply their finger on a touch screen. Working side by side, a sighted person could physically redirect the blind person's hand to the item the sighted person intends to point at. Alternatively the blind person's cursor stays put and the diagram is panned so that the referenced item is moved to the cursor's current location. Panning the diagram, however, has a drawback because it offsets the absolute locations of all elements in the diagram. A blind person remembers

154

the absolute locations of diagram elements so that he can return to them later. Therefore panning the diagram invalidates the absolute locations he had memorized.

In the second technique, the cursor is controlled by a mouse or a track ball. A user moves the cursor relative to the current location of the cursor. In this case a pointing mechanism could be implemented by automatically moving the cursor to the location of the element pointed at. The diagram interface can read and sound information about the element pointed at. The user might know the location of the element pointed at if the user is already familiar with the element. On the other hand, a user might be disoriented about the location of the element. Moving the user's cursor automatically makes it difficult for the user to identify how the current location relates to the last location the user was aware of.

8.8 *Deep View implementation for collaboration*

In the collaborative situation we research, collaborators use different interfaces on different computers to discuss and edit diagrams. This requires the Deep View system to manage the communication between the collaborators' applications. Deep View uses the Sync project (Munson and Dewan 1996) as a subsystem to support the communication. Sync is ideally suited for our collaborative situation. We first review the main design choices for Sync and then describe Sync and the Deep View implementation using Sync in more detail.

Although the collaborators' interfaces are different, the collaborators' applications maintain a shared model of the diagram the collaborators are editing and discussing at an abstract level. The Sync system is specifically designed to provide a shared data model among several collaborators' applications. In the Deep View application the shared data model are the elements of a node-link diagram, including the overall diagram properties, nodes, links, and attributes. Furthermore, the shared data model includes data necessary to enable the Deep View semantic pointing mechanism.

Sync completes a series of tasks to maintain the shared data model as the collaborators edit the diagram. Sync manages the network connections between the collaborators' applications, which must connect to a central Sync server to begin a

session. Furthermore, Sync manages concurrency control, which is the case when two or more collaborators want to edit the same diagram element.

Although Sync maintains the consistent shared data model within the Sync subsystem, the Deep View plugin must independently maintain the shared data model within the Rational Rose application. The techniques the Deep View plugin uses for this are an extension of the mechanisms Deep View uses to monitor and manipulate the Rational Rose model of a diagram when a single blind user is accessing a diagram.

A unique artifact of the Deep View implementation with the Sync subsystem is that the single user Rational Rose application becomes a remote synchronous collaborative application. With our design, the Rational Rose interface is unchanged (it cannot be changed in any case because it is proprietary). Instead, the Deep View plugin manipulates the Rational Rose diagram model to reflect the edits of other collaborators and Sync maintains the shared data model between the collaborators' applications.

Although our research focuses on supporting two collaborators, the Deep View system with the Sync subsystem can support two or more collaborators. A collaborative session can consist of any number of sighted persons or blind persons. In future research, we could study how the interfaces can support unique situations beyond the focus of our research. For example, if multiple sighted users collaborate, they have the unique situation of independently managing the visual layout of their copy of a diagram shown in Rational Rose. In another situation, multiple blind users could work together. Deep View's pointing mechanism already supports the concept of multiple users pointing at multiple diagram nodes.

Before discussing the design of the shared data model, we review the overall Deep View system. The Deep View system consists of three main components shown in Figure 8-7. The components are the Deep View interface for the blind person, the Rational Rose interface for the sighted person, and the shared data model, which includes the Sync subsystem. The Deep View system uses a model-adapter-view paradigm to manage the components. This way the shared data model of the diagram and pointing mechanism is separated from the views, i.e. the user interfaces.

The adapter manages messages between the components. An interface will send event messages as a user edits the diagram model or changes the diagram elements

pointed at. The adapter relays the message to the model. If multiple collaborators are working together, changes to the shared data model also trigger the Sync subsystem to propagate the changes through the Sync server to the shared data model components of the other collaborators. In turn, the shared data model component generates an update message passed through the adapter to all collaborators' interfaces, which can refresh the current state of the diagram.

The adapter has an additional task of managing unique IDs of elements in the shared model. The Deep View system assigns a unique ID to data elements in the shared data model. However, the corresponding model elements, such as nodes and links have a different ID within an instance of the Rational Rose application. Therefore, the adapter has the task of translating between the local ID in the Rational Rose application and the Deep View system shared data model. Also this design enables two sighted persons to collaborate because Deep View manages the unique IDs in each instance of Rational Rose.
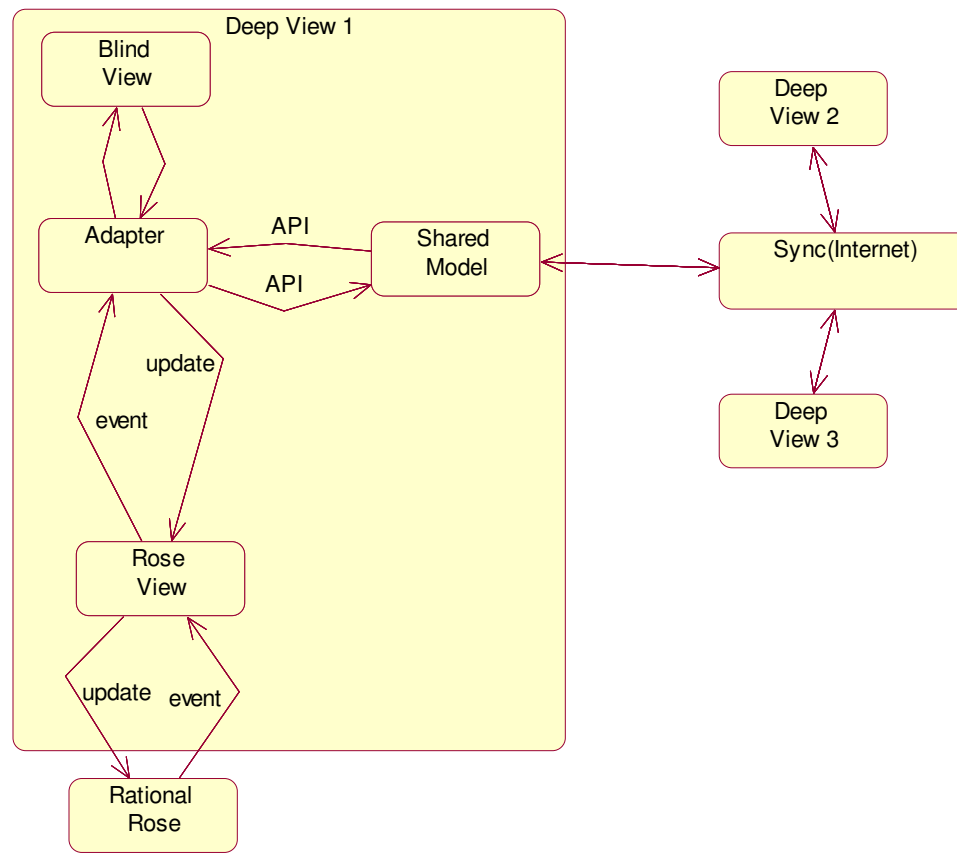
Figure 8-7: Deep View model-adapter-view paradigm

8.8.1 Background to Sync infrastructure

Sync is a Java based infrastructure for developing collaborative applications. Sync provides an object-oriented replicated model between instances of a collaborative application connected over a network. A model's replicated objects are comprised of arbitrary Java objects designed by a developer for a given application domain. In the client-server Sync model, clients are the instances of the collaborative application; each client stores a local copy of the replicated model. The server collects changes to the model from a client and propagates the changes to the remaining clients. The user interface representing the model is implemented separately and a developer can customize the interfaces for the needs of a collaborative application.

Sync was originally designed to support multi-user mobile applications, although Sync's features are more broadly suited to collaborative applications. To support mobile applications, emphasis of the Sync design is on devices with intermittent network connectivity. The general benefit is that it is straightforward to switch the application from running stand-alone to running as a collaborative application.

Sync also minimizes network bandwidth to accommodate the limited network resources of mobile devices. This is accomplished by synchronizing changes at the lowest level of granularity to the level of basic Java types (e.g. int). A general benefit is that changes can be tracked in detail and when updating the interface only the part corresponding to the changed data has to be updated instead of updating the entire user interface.

A single user application can easily be modified into a Sync collaborative application by following a few patterns. Sync maintains the replicated model through Java introspection, Java Bean property change events, and Java Bean property change listeners. Sync automatically identifies the properties of an arbitrary replicated object through Java's introspection; that is Sync identifies the getter/setter methods corresponding to an object's properties. Sync uses the Java Beans listener to capture a change to an object's property, which are triggered in the property's setter method. Sync asynchronously propagates the change to the remote clients and invokes the corresponding setter method for the changed property.

Sync also supports replicated Java style hash tables and vectors, significant in creating a replicated model with a complex data structure. Sync provides a custom listener pattern because Java Beans does not support these data structures by default.

The Sync server and clients exchange messages to manage changes to the shared model. The server and clients control each other through Java Remote Method Invocation (RMI). The server features various synchronization mechanisms, such as merging changes to the same object and forwarding changes only when the property is set to a different value (not just set to the same value again).

8.8.2 Overview of Deep View shared data model

The shared data model consists of two parts. The first part is that representing the node-link diagram, which consists of nodes, links, and attributes. Each diagram element is implemented in a Java class. And each instance of a diagram element has a unique ID. The overall model of the diagram stores nodes, links, and attributes by their unique ID in hash tables corresponding to the type of diagram element. Hash tables are used to facilitate easy access to individual diagram elements. Furthermore, the diagram elements store the unique IDs of related diagram elements. This includes the following relationships:

- A node stores references to its links. The references are stored in a vector to maintain an order to the links. Although the order is currently not used, in future work we will take advantage of ordering the links.
- A link stores references to the nodes it connects and how the link is oriented between the nodes. The references to the nodes are stored in a vector and the vector index identifies the start node and destination node of the link.
- Nodes and links reference corresponding attributes. Attributes are used in node-link diagrams in general, but not in state-charts specifically.

The UML class diagram in Figure 8-8 represents the shared diagram model and relationships between the diagram elements. Figure 8-8 is a simplification of the actual implementation to emphasize aspects of the shared diagram model; for example, each getter method in a class has a corresponding setter method.
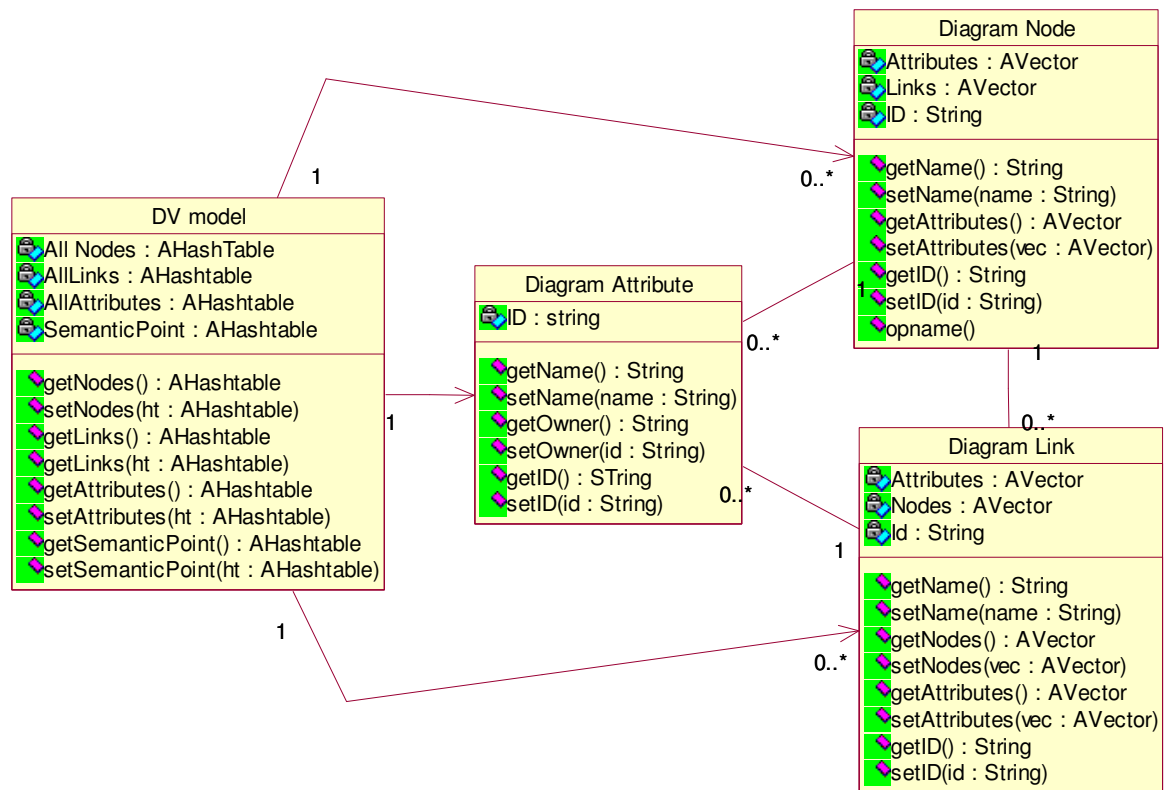
Figure 8-8: UML Class diagram of Deep View shared model.

8.8.3 Shared model and semantic pointing

Besides representing a diagram, the shared data model stores the data for semantic pointing, including the follow-me feature. The follow-me feature is treated the same as general semantic pointing, except that the two features are controlled differently in the interface. The current implementation supports pointing at diagram nodes. The nodes pointed at are in the collaborators' common frame of reference; which includes the elements stored in the shared model. Otherwise, if one person were to refer to an item outside the common frame of reference, the item would be unknown to the other person and cause confusion. To point at items in the shared model, however, an item should be realized in each collaborator's interface.

Following we describe the infrastructure of the shared data model related to semantic pointing and the follow-me feature. We also describe Deep View's programming commands (API) available to an interface to control the semantic pointing.

161

Overall, the infrastructure stores the multiple diagram nodes that collaborators are pointing at. The elements pointed at are determined on-the-fly as the collaborators use their application interface.

Pointing infrastructure data

Two forms of information are stored for what each collaborator is pointing at. The first information includes the unique IDs of one or more nodes a collaborator is pointing at. The second information is data uniquely identifying a collaborator in the user interfaces. For a visual interface, the shared data model stores the color associated with the collaborator. A collaborator's color is the same in all visual interfaces so that when two sighted collaborators refer to a color, the reference is clearly understood. For the Deep View interface, the shared data model stores a collaborator's name; the name is part of the textual description that the screen reader reads to the blind user.

The data related to pointing is stored in vector and hash table data structures. In the Deep View prototype, a hash table stores an entry for each collaborator hashed by the collaborator's unique ID. An entry for a collaborator stores a vector of the unique IDs corresponding to the diagram nodes pointed at by that collaborator. A separate hash table contains the collaborator's identifying characteristics and entries are hashed by the collaborators unique ID.

The vector storing the diagram nodes pointed at is tagged as either general semantic pointing or the follow-me pointing. The visual Rational Rose interface uses the tag to visually highlight the appropriate diagram nodes. Currently, the Deep View interface only presents nodes explicitly pointed at and does not highlight the follow-me nodes.

Pointing infrastructure control

The Deep View system provides several commands to enable pointing at diagram nodes or clearing the selection. However, the use of the commands differs slightly between general semantic pointing and follow-me pointing.

In general semantic pointing, a collaborator gives a command to point at a set of nodes. The Deep View system adds the unique IDs of diagram nodes pointed at to a vector corresponding to the collaborator. Finally the Deep View system sends a notification for the other interfaces to refresh as opposed to multiple individual notifications sent for each node pointed at. Otherwise, an interface would not know when the group of items pointed at is complete. Also in the Deep View application it would be inconvenient for a blind person to be notified for each node pointed at if there are many nodes. Clearing the selection happens in a similar order, except that IDs are removed from the vector and the notification indicates a cleared selection.

The follow-me pointing mechanism uses the same commands to indicate to the sighted person where the blind user is currently focused. Deep View automatically tracks the blind user's focus instead of requiring the user to issue an explicit command. As the user switches between diagram nodes in the Deep View treeview, the current node is changed. The previously selected node is unselected from the nodes pointed at. The command to point at a node is issued for the newly selected node. The vector storing these nodes is associated with pointing follow-me mechanism and the visual interface highlights the diagram node accordingly.

Chapter 9
Deep View collaboration user study

9.1 *User study design*

We conducted a user study to evaluate the described Deep View shared workspace. A sighted person uses visual diagram editor in Rational Rose to access and edit the same node-link diagram as the blind participant, who accesses the diagram through the Deep View interface. The Deep View system connects the participants diagram interfaces and maintains a consistent model of the diagram. We hypothesize:

> *Participants prefer when the blind participant can contribute to editing the diagram through the Deep View system*

We find the hypothesis supported if participants complete the tasks successfully and self report making steady progress doing so. Furthermore we seek to observe the strategies collaborators use to complete two aspects of the diagram task. In the first aspect, we compare three situations where the participants share control of editing the diagram. In two situations either the blind participant or sighted participant solely controls editing a diagram while the other person is allowed to review the diagram in his interface. In the third situation, both participants have equal access to editing the diagram. Collaborators completed the three situations in a predetermined random order to counter order effects. The current state of the art technology for a blind person and sighted person to collaborate is for the sighted person to be the sole editor; Deep View is the first shared workspace to do so.

The second aspect we observed is how collaborators complete the diagram task using the semantic pointing mechanism to point at diagram nodes in the diagram.

Participants are instructed to use semantic pointing instead of referring to diagram nodes by name. Although this requirement is unnatural for a realistic collaboration, the situation emphasizes the use of the pointing mechanism.

9.1.1 Independent variables

The collaborators' task is to construct a brainstorm diagram given a common knowledge topic. The visual representation of a brainstorm diagram is a tree structure where the main topic is the root and subcategories are linked below the root. Brainstorm diagrams are used as a writing strategy to organize ideas before writing about them. In the study, participants independently list five words they associate with the main topic. Then the participants group the words in categories or subcategories. A brainstorm diagram promotes that participants work together because they must agree on the categories to create.

9.1.2 Measurements

The main measurement is the participants' subjective reflection on the collaborative experience. The feedback is collected in a post experiment questionnaire and discussion. In the discussion participants are asked to explain unique situations. Secondary measurements are the time to complete creating a brainstorm diagram. We also record the diagrams created by the participants.

9.1.3 Participants

Four pairs, each consisting of a blind participant and a sighted participant, completed the user study. Three blind participants are congenitally blind and one participant became visually impaired in childhood. All participants are college students or working professionals. Prior to the user study, sighted and blind participants completed the first Deep View user study to learn how to use the diagram interfaces and review the concepts of node-link diagrams.

9.1.4 Computer setup

We designed the user study so that participants could participate remotely similar to how the shared workspace would be used in a realistic scenario. In one session the participants were collocated. In three of four sessions the participants were remote.

The Deep View system manages the networking between clients working on separate machines. Blind participants could install Deep View on their computer and use their screen reader to access it. Alternatively blind participants could remote desktop to a computer with Deep View and the JAWS screen reader. Remote sighted participants used Window's remote desktop to access the computer with Rational Rose. The collaborators conversed through an audio chat application.

Formalities of the study are handled electronically, which facilitates remote participants. The informed consent is emailed and the post questionnaire is an online web form. Even if participants and experimenter could meet in person, the electronic documents are necessary for blind participants to access them.

9.2 Results

We group the results of our user study into three categories of results. The first category is the participants overall impression using Deep View. The second category consists of observations of participants strategies to successfully complete the task. The last category is on observations of how participants used the pointing mechanism.

Overall the participants successfully completed the tasks and had a fluid exchange of ideas. We consider a diagram task successful when the participants complete the task, have followed the instructions, and produce a reasonable diagram. Participants self report that they enjoyed the collaboration and that pairs of participants contributed equally to the diagram task. However, a sighted participant provides additional information about the diagram to a blind participant. The most obvious situation, which occurred in almost all diagram tasks is that the blind participant would ask which items still have to be integrated into the diagram. It was straightforward for the sighted person to answer. Working with other diagrams might provide different situations in which a sighted person can provide additional information for a blind participant.

Besides reflecting on the overall experience, participants reflected on their experiences in the three editing situations. We summarize the results from the post experiment questionnaire in Table 9-1. Overall, pairs of collaborators favored the situations in which the blind participant could contribute to editing the diagram. Three pairs favored the situation when both participants could edit the diagram. One pair favored the situation when the blind participant controlled editing the diagram. In the favored situation, blind participants and sighted participants felt the situation is preferred and they felt their productivity was highest.

The three pairs that favored the situation where both participants edit the diagram shared the preference for similar reasons. The participants enjoyed the interactivity of working together and the interactivity of taking turns to make the edits. One participant could verify the other participant's edit in the interface and therefore have confirmation of the other participant's involvement.

The pair of collaborators that favored the condition when the blind participant controlled the diagram editing had a different explanation for the preference. The blind participant commented that the interactive nature of editing the diagram in Deep View was the best way for him to comprehend the diagram. The sighted participant also preferred the situation so that the blind participant could set the pace of the editing. The sighted participant was hesitant about making his edits too quickly because he was uncertain if the blind participant needed additional information or extra time to process the edit.

The situation in which participants had the best understanding of a diagram varied. Two blind participant and two sighted participants felt their understanding was similar in all three situations. One blind participant and one sighted participant preferred when they controlled the editing. As mentioned, controlling the editing can help the editor's understanding of a diagram because the editor can set the pace of the interaction. The interactive involvement of the editing deepens the understanding: One blind person had the best understanding when participants edited the diagram together.

One pair's observation reflects on the impact of using different representations of a brainstorm diagram, specifically related to creating links. The blind participant's thought process of creating the diagram is a bottom up approach, which is to first connect

categories to specific items and then connect the categories to the main category. On the other hand, the sighted participant connected nodes of a diagram in a top down approach. The sighted participant's approach is determined by the visual layout of the diagram and he completes missing links passing over the diagram canvas from top to bottom. The blind participant and sighted participant could comprehend either approach but the participants determined the specific approach in their discussion.

Table 9-1. User preferences comparing the three editing situations (n=4 blind-sighted pairs)

		Blind editor	sighted editor	both editors	all situations equal
I preferred it when	blind	1		2	1
	sighted	1		3	
We were most productive when	blind	1		3	
	sighted	1		3	
My understanding of the diagram was best when	blind	2		1	1
	sighted		1		3

The data samples are too few to recognize trends about the collaborators' time to complete the diagram tasks, which are shown in Figure 9-1. Also there is a large variation in the times ranging from 15% to 60%. Variations of 15% (about two minutes) might be explained by differences in the main topic of the diagram. It might also be caused by tangential discussions collaborators have, such as telling a story or a joke.
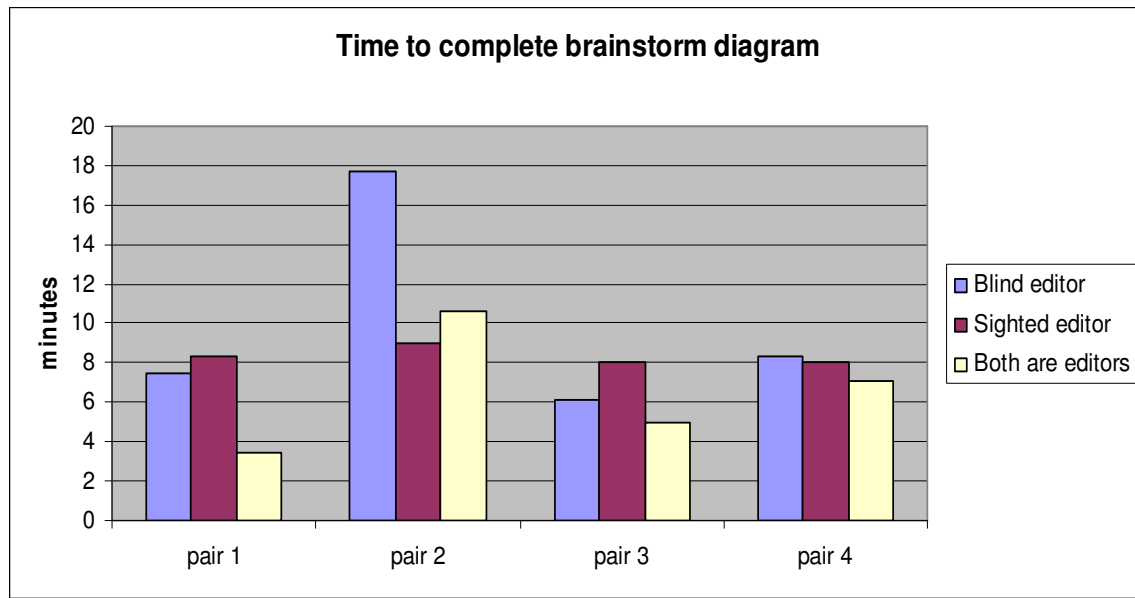
Figure 9-1. Time to complete brainstorm diagram (N = 12 diagrams)

9.2.1 Collaborators' interaction

During the diagram tasks the experimenter observed how collaborators interact to complete the task. The collaborators must manage interleaving conversation with editing of the diagram. The same interaction occurs in all three editing situations. Overall, the collaborators take turns exchanging ideas and deciding together which categories to create based on their associations with individual words. The participants discuss the concepts of the diagram at an abstract level; for example, related to a diagram, the collaborators spoke about creating and connecting words and categories (diagram nodes and links).

There is no apparent obstruction to the interaction by using different representations of the interfaces. The collaborators avoided confusion by not referring to the specifics of the interface; such as the color of a node in the Rational Rose interface or a treeview entry in the Deep View interface.

In the remainder of this subsection we discuss how participants used their user interfaces to coordinate their actions relative to editing a diagram. The interaction involves participants agreeing upon edits, one participant making the edit, and the other participant receiving an acknowledgement of the completed edit. From the context of the collaborators' conversation, one participant could anticipate an edit made by the other

participant. The edit is confirmed to a participant when the diagram interface gives a cue. When a blind participant makes an edit, a sighted person watches for the automatic update to the visual presentation of the diagram. The visual update gives complete details on the edit. Anticipating the edit, the sighted participant's attention is already focused on the area of the edit.

When a sighted participant makes an edit to a diagram, a blind participant awaits Deep View's audio icon indicating the edit is complete. Blind participants found the audio notification useful. However, at least one participant suggests one audio icon is enough instead of a different audio icon for each type of edit. Sometimes the audio icon is enough for a blind participant to confirm the completion of the edit.

Two blind participants had different techniques to review an edit in more detail. For a new node added to a diagram, a blind participant can access the node quickly. One blind participant noticed new nodes are added to the end of the treeview and he could find the new node there. Another participant found new nodes through hotkeys, where typing the first letter of the new node focuses the screen reader on nodes beginning with the typed letter. Identifying other types of edits to a diagram, however, is not as straightforward; for example, a new link is listed within a node in the treeview and not explicitly visible as a treeview entry. An alternative technique for tracking edits to a diagram is through Deep View's summary of edits, but for simplicity it was left out from the user study instructions. Although it is possible for the blind participant to follow the sighted participant's edits, the Deep View interface should be further enhanced to simplify the process, for example, by making it faster and easier for the blind participant to access the information about edits.

The interaction of the collaborators is inevitably influenced by a screen reader, which the blind participant listens to. Although the vocalization of the screen reader could continue when the collaborators talk, collaborators dealt successfully with the screen reader. Sighted participants say they were not distracted by the screen reader. The two sighted participants remote from the blind participant, heard the screen reader faintly through the microphone used to capture the blind participants voice. The sighted participant collocated with the blind participant was not distracted by the screen reader either. The participant treated the screen reader's vocalization as background noise

because the computer synthesized voice and pace of the voice is not easily understood to an unaccustomed ear.

A blind participants' technique to deal with the screen reader is to preempt the screen reader with one keystroke when the sighted participant starts talking. The technique works well in general for all blind participants. One blind participant, however, commented that the sighted person's interruption would inhibit understanding a more complicated diagram. To properly understand the diagram, the blind participant needs to explore several elements of a diagram.

9.2.2 Pointing

In the user study we evaluated the pointing mechanism. In the case of the simple brainstorm diagram, the pointing mechanism is less relevant as participants could easily reference nodes in the diagram by their unique names. The pointing mechanism is more applicable in a larger diagram, possibly with nodes with ambiguous names. Nevertheless, the user study provided insights on how the pointing mechanism could be used and improved. Overall, it is easier for a blind participant to point at diagram nodes and have the sighted participant identify the nodes. It is more complicated for the blind participant to identify the nodes a sighted participant is pointing at because the blind participant must complete multiple steps to identify the nodes.

The most useful pointing mechanism is the follow-me feature, where the visual interface indicates the current node the blind participant is focused on. Besides the blind participant explicitly pointing at a diagram node, the follow-me feature gave the sighted participant a sense of awareness that was beneficial in two ways. First, when the participants are switching between topics, the current node of the blind participants suggests to the sighted person the next node the blind participant wants to discuss. In anticipation, at least one sighted participant started to arrange the spatial layout of the given node as the blind participant started talking about it. Second, when editing together, the blind participant's current node suggested what the blind person was editing next. The sighted person could accordingly plan his edit to avoid conflicting with the blind participant's edit.

One blind participant's comment is an inspiration for an enhancement to the interfaces for a blind participant that makes him aware of the sighted person's focus. The Rational Rose interface detects a node or link currently selected in the visual interface and the selected item might be the next diagram element the sighted person edits. The Deep View interface could annotate the selected diagram item in the Deep View treeview. This way the blind participant would know the sighted participant's focus when the blind participant selects the diagram element in the treeview. This awareness information is helpful to the blind person to prevent both participants from editing the same diagram element.

It is straightforward for a blind participant to point at one or more diagram nodes. The participant used the follow-me feature when pointing at one diagram node. The blind participant could also easily point at multiple diagram nodes with the general pointing mechanism. One blind participant suggested an improvement to the general pointing mechanism; there should be feedback, such as an audio icon, to confirm that the command to point at a node is successful. It is also straightforward for the blind participant to recognize the highlighted nodes that the blind participant points at.

Although a blind participant can understand what a sighted participant is pointing at, the pointing mechanism must be streamlined to be faster to use. The issue is that it takes a blind participant too long to look up the diagram nodes pointed at. A blind participant must use at least four key strokes to access the diagram nodes pointed at: a keystroke to switch to the pointing mode, a keystroke to access the sighted participant's entry in the treeview, a keystroke to expand the treeview entry, and then keystrokes to access the list of diagram nodes a sighted participant is pointing at.

The general pointing mechanism can be improved by adding a technique specialized for two collaborators. The current general pointing mechanism is designed to support two or more collaborators pointing at multiple diagram nodes. In the proposed design, the blind participant uses one keyboard shortcut to cycle through the diagram nodes pointed at; concretely the Deep View treeview sets the focus of the screen reader to the corresponding treeview entries.

There is one situation in which the sighted participant pointing with the general pointing mechanism at multiple items worked well. In the example, the sighted person

points at multiple nodes that still need to be connected to the rest of the diagram and the blind participant brings up the list of nodes in Deep View. The blind participant can now easily point sequentially at a subset of nodes through the follow-me feature, which continues to highlight the blind participant's selected nodes in the Rational Rose interface. So in this way the blind and sighted could smoothly exchange ideas through each person pointing. This benefit was observed in the user study sessions and was not an original design goal.

9.3 Discussion

The results of the study suggest that the Deep View shared workspace enables a blind person and a sighted person to collaborate. The collaborators successfully completed the tasks and self-report a positive experience. It is practical that the collaborators could complete the brainstorm task in all three editing situations. In a work environment it might not be possible for both collaborators to have access to the shared workspace, for example, when one person is away from a computer. The collaborators could still discuss the diagram while one person edits the diagram.

The participants favored the situations where the blind participant could contribute to editing a diagram – with either both collaborators controlling the diagram or only the blind participant controlling the diagram. These situations expand on the current state of the art where only the sighted person makes the edits to the diagram. With Deep View the blind person's ability to edit the diagram deepens the person's involvement in the task, for example, to contribute ideas in more detail. With more collaborative sessions between a blind person and a sighted person, collaborators can shape their roles to accommodate how each person perceives a diagram and edits it.

The collaborators' use of Deep View's design features while collaborating indicates that the features where useful to support the collaborators' communication. One helpful feature is Deep View's mechanism of notifying participants of diagram edits visually or auditorily. The notification is an asynchronous event the user perceives but that does not distract the participant from the current task. The auditory notification is especially useful to blind participants who process the interface sequentially through the screen reader.

Another helpful Deep View feature is the follow-me pointing mechanism, which is well received by sighted participants. Besides being an explicit pointing mechanism for a blind user, it provides the sighted participant awareness about the blind participant's current activity. The awareness provided by the follow-me feature seems to benefit sighted users similar to other CSCW aware specific tools, such as multiple scroll bars for a text widget described in (McDaniel and Brinck 1997).

The semantic pointing mechanism would be more useful in a more complex diagram. As an example, the experimenter experienced one incident in the first user study where a blind participant had omitted a node in editing the paper recycle diagram. In the discussion, the blind participant was confused about which node was omitted. In that case, the experimenter could have used the semantic pointing mechanism to identify the specific node and clarify the ambiguity.

Chapter 10
Conclusion

Our research results are a testament to people's versatility in adapting to various situations in order to communicate with each other. In the user studies we conducted, participants with and without disabilities collaborated to successfully exchange ideas and create an artifact while completing a task.

From our results we draw conclusions in three areas. The first area is knowledge gained about how our interface design features support communication between collaborators regardless of disability. The second area is lessons learned from designing interfaces with cooperation and feedback from the communities with hearing impairments and the communities with visual impairments. The third area is identifying future work that will improve the user interfaces to further enhance the communication between collaborators. For simplicity, we continue to use abbreviations to refer to the pairs of collaborators. We abbreviate a deaf person collaborating with a hearing person as a DH pair; a blind person collaborating with a sighted person is a BS pair.

10.1 Interface design features to support communication

People who are either deaf or blind benefit directly from our research. It contributes to assistive technology to enable collaboration between people with and without disabilities. A person without disabilities uses his senses of hearing and vision simultaneously to process channels of information used to perform a task and communicate with a collaborator. The situations we research provide the unique perspective of isolating the hearing or vision senses used while communicating. We find the design features we research help deaf persons or blind persons by conveying similar

information in the visual and auditory interfaces. More generally, however, our research expands knowledge about user interfaces that support collaborators' communication: Designers of future interfaces can choose the most appropriate visual or auditory design feature and sensory channel to communicate desired information.

Overall, collaborators with disabilities use their senses to manage multiple channels of information, which include conversing with a collaborator, watching the shared workspace, and manipulating the shared workspace. Our design features facilitate the communication, specifically related to the three fundamental aspects of communication between collaborators described in Chapter 3. These include the collaborators' use of a shared workspace, coordinating their actions, and referencing the shared workspace.

10.1.1   Using a shared workspace

In our research the collaborative tasks are based on a shared workspace to motivate the collaborators' interaction. In their communication, DH and BS pairs use a shared workspace as part of their common ground, and use the shared workspace in ways similar to those of collaborators without disabilities. Specifically, the shared workspace maintains the current state of the task, and, by referring to it, collaborators can confirm that actions are completed as expected.

In the user studies we conducted, DH pairs and BS pairs mostly completed the tasks as expected by having a fluent exchange of ideas. An exception is the two DH pairs who did not complete the brainstorm diagram task as instructed, and used ASL to communicate (a relevant result but not within the scope of this work). All participants self-report an enjoyable experience while completing the tasks.

Given the visual nature of shared workspaces, DH pairs can use existing shared workspaces without modification. On the other hand, BS pairs require customized shared workspaces to accommodate the blind person's need to access the visual content. We found that the Deep View interface and system is an example of a full-fledged loosely coupled shared workspace to access and edit node-link diagrams. With the Deep View system, each collaborator uses the most appropriate interface.

Our single-editor user study (Chapter 7) shows that blind participants develop a solid understanding of node-link diagrams from using the Deep View interface. We found sighted participants, however, completed reading and creating diagrams 4.7 and 2.99 times faster, respectively, than blind participants. From the blind participants' feedback, we identified enhancements that should be made to the Deep View interface to alleviate the problems some blind participants had in understanding node-link diagrams through the interface. Blind participants will also improve their understanding of diagrams with more practice and experience, which is now limited because node-link diagrams are in general not accessible with commercially available software.

10.1.2   Coordinating actions

The collaborators must coordinate their actions to coherently exchange ideas. They meticulously time taking turns so each person can express an idea and the other person can follow. The collaborators switch attention between the shared workspace and conversing with each other. Our observations of collaborators' interactions in the collaborative user studies (Chapters 5 and 9) indicate how the collaborators use the design features of the interfaces to coordinate their interactions. As expected, the collaborators benefit from their shared common ground, such as, the context of the conversation lets the collaborators coordinate switching their attention between the different channels of information. For example, after one person suggests a checkers move, the collaborators both watch the video to decide on completing the move.

The interface for DH pairs presents the visual information on the screen in the form of a shared workspace, video of the collaborators, and a chat messaging window. To successfully collaborate, the collaborators have to manage the information by switching their attention between all of these sources of information. We observed this to be the case, for example, when participants used gestures in the video. Participants had to have been watching each other's video to perceive and respond accordingly to gestures, which included double acknowledgements, negative acknowledgements, and gestures of indifference.

Furthermore, our observations of the DH pairs revealed a unique strategy of using the sources of visual information. DH pairs rely on video to communicate, unlike

collaborators without disabilities, who rarely use the video. DH pairs use the video to make decisions about a suggested edit to the shared workspace. The suggested edit, however, is made by pointing at the shared workspace (specifically in the checkers task), editing the shared workspace, or writing chat messages (specifically in the brainstorm diagram task). Of course, collaborators make decisions in chat messages, but some decisions are still communicated in the video.

Future research could investigate the DH participants' strategy to manage the visual information sources, such as identifying cues from the interface that alerted participants. Collaborators without disabilities would use cues from the verbal conversation to direct attention. Furthermore, it would be illunimating to consider the cues and strategies two deaf collaborators use to converse with ASL while using a shared workspace.

In the Deep View collaborative study we also observed BS pairs manage multiple sources of information. A sighted collaborator interacts with the visual shared workspace while conversing with a blind collaborator. The blind participant manages several auditory information sources, including conversing with the sighted collaborator, listening to the screen reader, and listening to sounds from the Deep View interface. Although sighted persons and blind persons are familiar with managing such information from other experiences, in the collaborative situation we research it is unique for the BS pairs to track edits to the shared workspace.

One collaborator must confirm the other's edits in order to finally complete the current transaction. It is straightforward for a sighted person to detect changes to the visual interface as it is automatically updated. The Deep View interface provides blind persons an audio icon to notify them about the sighted person's edit to the node-link diagram. Although Deep View does provide mechanisms for the blind participant to look up the details of an edit, feedback from blind participants indicates the usability of the mechanisms can be improved.

10.1.3   Referencing and pointing at workspace elements

In our research we observed two ways collaborators express ideas by referring to the details in a shared workspace. The first way is to explicitly point at the shared

178

workspace with a telepointer or the semantic pointing mechanism. The second way is to refer to the names of shared workspace elements in conversation either in chat messaging or verbally.

DH pairs use a telepointer to explicitly point at items in a shared workspace. In the checkers task, a telepointer is sufficient for gesturing suggested checkers moves without extra explanation. The gestures work well for communicating about a spatial aspect in the shared workspace. However, in the tasks we researched, a telepointer is not useful for communicating decisions as is evidenced by our observations of the collaborators' use of video to make and confirm decisions.

In our collaborative Deep View user study (Chapter 9) we observed that BS pairs can explicitly reference the shared workspace by using the semantic pointing mechanism (including follow-me pointing) provided by the Deep View system. Through the semantic pointing mechanism each participant has equal ability to point at diagram nodes and access the diagram nodes the other person is pointing at. Although the pointing mechanism is functional, its usability can be improved for the blind person to faster access information about nodes pointed at by the sighted person.

Collaborators can refer to shared workspace elements indirectly in conversation. In the brainstorm diagram task, we observed that DH pairs and BS pairs can easily refer to diagram nodes by using their names. In this task, diagram node names are unique and the reference is unambiguous. Furthermore, the design of the Deep View system allows the collaborators to discuss the brainstorm diagram at an abstract level, which both collaborators understand; the collaborators do not need to refer to the specifics of the interfaces. As the interfaces are different for each collaborator, direct reference to them would cause confusion.

10.2 Collaborating with communities with disabilities

A priority of our research was to collaborate with members of communities with disabilities in order to make our research relevant for the people it is intended for. Throughout the research we gathered potential users' feedback. We proceeded in three main stages: first having general discussions to understand the communities' needs, second creating a prototype, and third gathering user's feedback from using the prototype.

Our user studies included people with disabilities, who might benefit from the interfaces in their own work. We tailored the user study material to communicate clearly with participants, such as using closed caption or ASL videos for deaf participants and electronic documents for blind participants.

We collaborated with members of the Deaf community through several organizations in North Carolina. Originally we spoke with two UNC Chapel Hill students with hearing impairments. After preparing the video conferencing user study, we had the great fortune of working with the Raleigh and Wilson N.C. Department of Health and Human Services offices for the deaf and hard-of-hearing, and Department of Disabilities at UNC Greensboro. Staff members at these organizations introduced us to their members with hearing impairments and gave us permission to recruit participants.

Our research on video conferencing will benefit the Deaf community in future designs of the technology. A more immediate benefit from our research for the deaf community, however, is a tangential project suggested by one of the UNC Chapel Hill students. The suggestion resulted in the Facetop Tablet project. This project addresses a deaf person's difficulty of taking notes during a meeting with hearing persons or deaf persons. While the deaf person takes notes, he misses the conversation being signed by an interpreter or other deaf person. With the Facetop Tablet project, a deaf person can watch a video of the signer on the screen next to the notes they take. Many people with hearing impairments we explained the project to could relate to the difficulty and provided examples of other situations in which the problem occurs. Several people were interested in trying Facetop Tablet for their own work; however, logistics prevented us so far from following through.

We worked with members of the blind community in the development and evaluation of the Deep View interface and system. The original idea to research accessible diagram interfaces came from the frequent discussion topics on the blind programming mailing list. During the development of Deep View, we gathered feedback from experts in accessibility to iteratively enhance the features of early prototypes.

We evaluated Deep View in the narrow scope of user studies and more generally with blind professionals. Participants for the user study were recruited from contacts

made through the research. We worked with two blind persons to prepare Deep View for their work with node-link diagrams as described below.

In the first case we worked with a blind instructor teaching a database course. Deep View's template for Entity-Relationship Diagram (ERD) diagrams was developed for this purpose. We also created the Deep View extension for Visio so that sighted students could use a familiar application to generate diagrams for the blind instructor. We made one trial run, where a sighted student created ERD Visio diagrams, which the blind instructor could access. An open issue is managing the logistics of using Deep View for all assignments and students in the course.

In the second case we are working with a software engineer, whose team will use UML diagrams to document their software design. Based on feedback, we have been streamlining the Deep View usability to make it more practical for the software engineer to use on a daily basis adding features, such as saving files, creating new templates, or adding dialogs to expose more information.

10.3 Future work

We have identified two main directions for future work. One relates specifically to video conferencing and the other to improving interfaces for DH pair and BS pair collaboration.

10.3.1   Designing future video conference systems

Although we did not develop a novel video conferencing application, knowledge from the user study can be used to design future video conferencing systems. There are two possibilities.

First, knowledge about the DH pairs' use of gestures is useful to users regardless of disability. The user study supports the concept that video is important for making decisions. Previous studies on video conferencing involving collaborators without disabilities find limited use of video. In limited instances, however, researchers observed that collaborators use video to interpret the other person's intentions when making a difficult decision (Hudson, Helser et al. 2003).

Second, the experimenter's observations of the DH pairs indicate that the gestures are limited to a small set of acknowledgements. In the future, one could abstract the gestures and eliminate the video conferencing, which has high bandwidth requirements. The gestures could be represented as animated icons. A user could trigger the animated icon through a keystroke. With this abstracted interface, collaborators might be able to communicate as well as with video conferencing. If video conferencing proves more useful than the animated icons, however, it would suggest that other valuable information is conveyed through the video.

10.3.2   Improving collaboration between a DH pair and a BS pair

Results from our user studies indicate areas of future research to enhance user interfaces to support collaboration between DH pairs and BS pairs. In our user studies the BS and DH pairs take longer to complete the brainstorm diagram task than collaborators without disabilities. The distribution of times for the groups to complete the brainstorm diagram task are shown in Figure 10-1. Times for the groups without disabilities and deaf-hearing group were recorded in the user study investigating deaf-hearing pairs (Chapter 5). Times for the blind-sighted group were recorded in the collaborative Deep View user study (Chapter 9). The time to completion for the task, however, cannot be statistically compared for various reasons including:

- The data samples are too few to get statistically valid results.
- The tasks were completed under different conditions, for example collaborators without disabilities used a tightly coupled interface while BS pairs used a loosely coupled interface.
- In the case of DH pairs, only one pair completed the task successfully as expected. Two pairs did not follow instructions and one pair did not complete the task successfully.
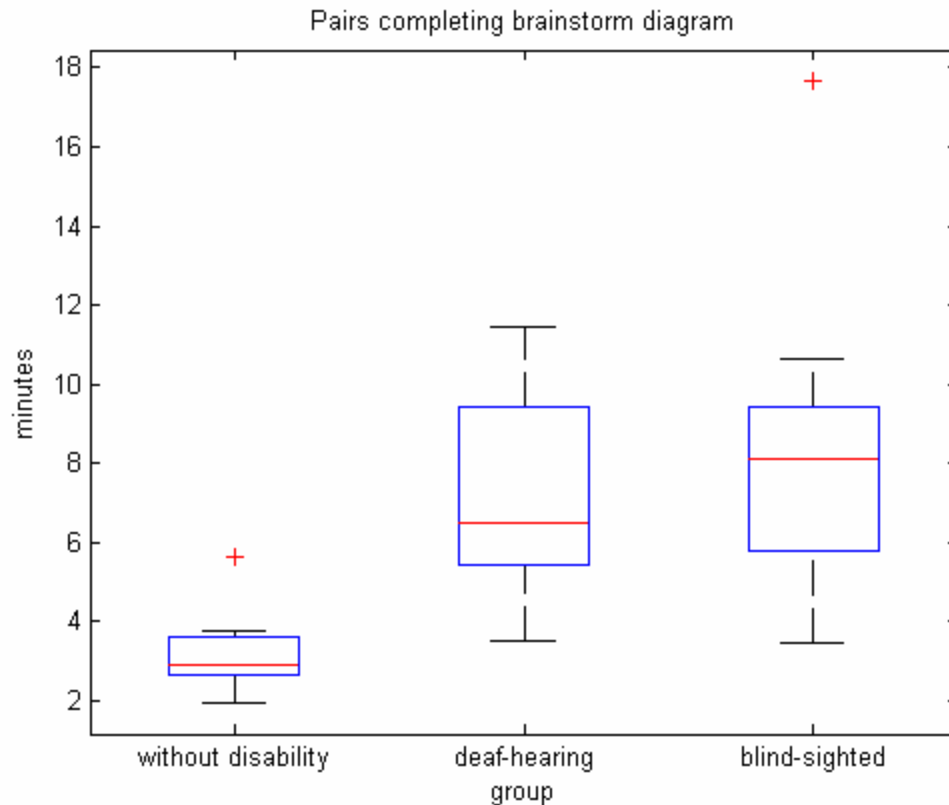
Figure 10-1. Time to complete the brainstorm diagram task (N= 9 diagrams for group without disabilities, N=12 diagrams for deaf-hearing group, N= 9 diagrams for blind-sighted group)

Although the sighted collaborators' faster time to completion is not statistically significant, it reflects a typical artifact of assistive technology: In some cases a person with a disability using assistive technology completes the task more slowly than a person without a disability. For example, blind users' access to diagrams is slower than sighted persons' because diagrams are optimized to be perceived visually. Note, however, advanced screen reader users in some cases can complete a task as fast as or faster than sighted persons who have less experience with the task. Also, taking longer to complete a task does not necessarily reflect the quality of the final product or the complexity of material covered. However, completing a task faster can increase the amount of material covered in a collaborative session.

We have two suggestions for future research for enhancing interfaces to support DH pairs and BS pairs. The first suggestion is to identify why the DH pairs and BS pairs are slower than collaborators without disabilities. Then the interfaces can be improved to

make the DH pairs and BS pairs complete the task faster. One could analyze collaborators' sessions and categorize the situations in which collaborators spend their time. Our hypothesis is that pairs of collaborators without disabilities complete the task faster because they divide the task between themselves while DH pairs and BS pairs do less so. DH pairs and BS pairs might be able to divide the task; however, they did not do so in the user study maybe because they were becoming accustomed to the new experience of collaborating. It seemed collaborators wanted to work together to confirm the task is completed as expected.

The second suggestion is to tailor tasks and user interfaces to direct the collaborators' interactions. For example, DH pairs playing checkers did not have the in-depth deliberation about checkers moves that collaborators without disabilities had. The DH pairs might have benefited if they could have used chat messaging to communicate ideas. However, the difficulty having a discussion about checkers using chat messaging is that it is difficult to refer to checkers pieces, which are best referred to through pointing. In the case of BS pairs, the blind person would benefit from declaring a pause, where the blind person can examine the diagram without interruption from the sighted person.

We hope our research contributions will help researchers in the future be more successful in improving assistive technology for people with disabilities.

References

AFB (2005). "Facts and Figures on Americans with Vision Loss (American Foundation for the Blind)." http://www.afb.org/Section.asp?SectionID=15&DocumentID=4398.

Andrea, R. K. (1996). Audiograf: a diagram-reader for the blind. Proceedings of the second annual ACM conference on Assistive technologies. Vancouver, British Columbia, Canada, ACM**:** 51-56.

Benford, S., C. Greenhalgh, et al. (1998). Understanding and constructing shared spaces with mixed-reality boundaries ACM Transactions on Computer-Human Interaction 185 - 223.

Bly, S. A., S. R. Harrison, et al. (1993). "Media spaces: bringing people together in a video, audio, and computing environment." Communications of the ACM **36**: 28 - 46.

Brown, A., S. Pettifer, et al. (2004). Evaluation of a non-visual molecule browser. Conference on Assistive Technologies Atlanta, GA, USA 40 - 47.

Chen, W.-C., H. Towles, et al. (2000). Toward a compelling sensation of telepresence: demonstrating a portal to a distant (static) office. IEEE Visualization Salt Lake City, Utah, USA, IEEE Computer Society Press 327 - 333.

Clark, H. (1996). Using Language. Cambridge, UK, University Cambridge Press.

Cohen, R. F., R. Yu, et al. (2005). PLUMB: displaying graphs to the blind using an active auditory interface. ACM SIGACCESS Conference on Assistive Technologies Baltimore, MD, USA 182 - 183.

Cormen, T. H., C. E. Leiserson, et al. (2001). Introduction to Algorithms, 2nd Ed., McGraw-Hill Higher Education.

Dewan, P. and R. Choudhard (1991). Flexible user interface coupling in a collaborative system. Conference on Human Factors in Computing Systems New Orleans, Louisiana, USA.

Eckman, P. and W. Friesen (1969). "The repertoire of non-verbal behavior: Categories, origins, usage, and coding." Semiotica **1**: 49-98.

Egido, C. (1988). Video conferencing as a technology to support group work: a review of its failures. Computer Supported Cooperative Work Portland, Oregon, USA**:** 13 - 24.

Eichelberger, H. (2002). SugiBib. Lecture Notes in Computer Science: Graph Drawing. Berlin / Heidelberg, Springer **2265/2002:** 581-584.

Engelbart, C. and W. K. English (1968). A research center for augmenting human intellect. AFIPS Conference Proceedings for Joint Computer Conference. San Francisco CA, USA.

Fish, R., R. Kraut, et al. (1990). The VideoWindow system in informal, communication, . Computer Supported Cooperative Work Los Angeles, California, USA 1-11.

Fussel, S. R., L. D. Setlock, et al. (2003). Effects of head-mounted and scene-oriented video systems on remote collaboration on physical tasks. Conference on Human Factors in Computing Systems Ft. Lauderdale, Florida, USA 513 - 520.

Fussell, S. R., R. E. Kraut, et al. (2000). Coordination of communication: effects of shared visual context on collaborative work. Computer Supported Cooperative Work Philadelphia, Pennsylvania, USA**:** 21 - 30.

Gallaudet (2005). "How many deaf people there are in the United States?" http://gri.gallaudet.edu/Demographics/deaf-US.php.

Gansner, E. R. and S. C. North (2000). "An open graph visualization system and its applications to software engineering." Software—Practice & Experience (John Wiley & Sons, Inc) **30**(11): 1203 - 1233.

Gaver, W., A. Sellen, et al. (1993). One is not enough: multiple views in a media space. Human factors in computing systems (INTERCHI). Amsterdam, The Netherlands, ACM Press**:** 335 - 341.

Grayson, D. M. and A. F. Monk (2003). "Are you looking at me? Eye contact and desktop video conferencing." ACM Transactions on Computer-Human Interaction **10**(3).

Gyllstrom, K., D. Miller, et al. (2007). Techniques for improving the visibility and "sharability" of semi-transparent video in shared workspaces. ACM Southeast Regional Conference Winston-Salem, North Carolina, USA.

Halpin, T. (2001). Information Modeling and Relational Databases: From Conceptual Analysis to Logical Design San Francisco, Libri.

Hellström, G. (1999). Proposed appendix to H.263: Sign language and lip-reading real time conversation usage of low bit rate video communication, ITU - Telecommunications Standardization Sector, Document Q15-G-47.

Holt, J., S. Hotto, et al. (1994). "Demographic Aspects of Hearing Impairment: Questions and Answers." http://gri.gallaudet.edu/Demographics/factsheet.html.

Hudson, T., A. Helser, et al. (2003). Managing collaboration in the distributed nanoManipulator. IEEE Virtual Reality Conference Los Angeles, CA, IEEE Press**:** 180-187.

Ishihara, T., H. Takagi, et al. (2006). Analyzing visual layout for a non-visual presentation-document interface ACM SIGACCESS Conference on Assistive Technologies Portland, Oregon, USA ACM**:** 165 - 172.

Ishii, H. (1990). TeamWorkStation: towards a seamless shared workspace Computer Supported Cooperative Work Los Angeles, California, USA**:** 13 - 26.

Ishii, H., M. Kobayashi, et al. (1993). "Integration of interpersonal space and shared workspace: ClearBoard design and experiments." ACM Transactions on Information Systems (TOIS): 349 - 375.

ITU (2006). "H.323 : Packet-based multimedia communications systems " http://www.itu.int/rec/T-REC-H.323/e.

Jancke, G., G. D. Venolia, et al. (2001). Linking public spaces: technical and social issues. Human Factors in Computing Systems Seattle, Washington, USA**:** 530 - 537.

Kamel, H. M. and J. A. Landay (2002). Sketching images eyes-free: a grid-based dynamic drawing tool for the blind. SIGACCESS Conference on Assistive Technologies Edinburgh, Scotland ACM Press**:** 33 - 40.

Keating, E. and G. Mirus (2003). American Sign Language in virtual space: Interactions between deaf users of computer-mediated video communication and the impact of technology on language practices. Language in Society Cambridge University Press. **32:** 693-714

Kendon, A. (1981). "Geography of gesture." Semiotica **37**: 129-163.

Kuzuoka, H. (1992). Spatial workspace collaboration: a SharedView video support system for remote collaboration capability. Human Factors in Computing Systems Monterey, California, USA**:** 533 - 540.

Mace, R. "Center for Universal Design, College of Design, North Carolina State University." http://www.design.ncsu.edu/cud/about_ud/about_ud.htm.

McDaniel, S. E. and T. Brinck (1997). "Awareness in Collaborative Systems." SIGCHI Bulletin **29**(4).

Miller, D., K. Gyllstrom, et al. (2007). Semi-transparent video interfaces to assist deaf persons in meetings. ACM Southeast Regional Conference Winston-Salem, North Carolina, USA**:** 501 - 506.

Morikawa, O. and T. Maesako (1998). HyperMirror: toward pleasant-to-use video mediated communication system. Computer Supported Cooperative Work Seattle, Washington, USA, ACM Press**:** 149 - 158.

Muir, L. J. and I. E. G. Richardson (2002). Video telephony for the deaf: analysis and development of an optimised video compression product. International Multimedia Conference Juan-les-Pins, France, ACM Press**:** 650 - 652.

Munson, J. and P. Dewan (1996). A concurrency control framework for collaborative systems. Computer Supported Cooperative Work Boston, Massachusetts, USA, ACM Press**:** 278 - 287.

NIST (2002). NIST 'Pins' Down Imaging System for the Blind. http://www.nist.gov/public_affairs/factsheet/visualdisplay.htm. web site.

Olson, J. and G. Olson (1997). Face-to-face group work compared to remote group work with and without video. Video-mediated communication (Technical Report Number: CREW-97-07). Hillsdale, NJ, Lawrence Erlbaum Associates**:** 157-172.

Parente, P. and G. Bishop (2003). BATS: the blind audio tactile mapping system ACM South Eastern Conference ACM Press.

Patterson, J. F., R. D. Hill, et al. (1990). Rendezvous: an architecture for synchronous multi-user applications. Computer Supported Cooperative Work Los Angeles, California, USA, ACM Press**:** 317 - 328.

Petrie, H., C. Schlieder, et al. (2002). TeDUB: A System for Presenting and Exploring Technical Drawings for Blind People. Lecture Notes In Computer Science, Springer-Verlag.

RealVNC "Real VNC." http://www.realvnc.com/.

Section508 "Section 508 of the Rehabilitation Act." http://www.section508.gov/.

Sellen, A. J. (1992). Speech patterns in video-mediated conversations. Human Factors in Computing Systems Monterey, California, USA 49 - 59.

Smith, A. C., J. S. Cook, et al. (2004). <u>Nonvisual tool for navigating hierarchical structures</u>. ACM SIGACCESS Conference on Assistive Technologies Atlanta, GA, USA ACM.

Stallmann, M., S. Balik, et al. (2007). ProofChecker: An Accessible Environment for Automata Theory Correctness Proofs. <u>Tech. Symp. on Computer Science Education</u>. Dundee, Scotland**:** 48 - 52.

Stotts, D., J. M. Smith, et al. (2004). FaceSpace: endo- and exo-spatial hypermedia in the transparent video facetop. <u>Conference on Hypertext and Hypermedia </u>Santa Cruz, CA, USA 48 - 57.

Tang, J. C. and S. L. Minneman (1991). "Videodraw: a video interface for collaborative drawing." <u>ACM Transactions on Information Systems (TOIS)</u> **9**(2): 170 - 184.

Vertegaal, R., R. Slagter, et al. (2001). Eye gaze patterns in conversations: there is more to conversational agents than meets the eyes <u>Human Factors in Computing Systems </u>Seattle, Washington, USA, ACM Press**:** 301-308.

Winberg, F. and J. Bowers (2004). Assembling the senses: towards the design of cooperative interfaces for visually impaired users. <u>Computer Supported Cooperative Work </u>Chicago, Illinois, USA 332 - 341.