Read, Write, and Navigation Awareness in Realistic Multi-View Collaborations

Sasa Junuzovic, Prasun Dewan
Department of Computer Science
University of North Carolina at Chapel Hill
Chapel Hill, NC, USA
{sasa, dewan}@cs.unc.edu

Yong Rui
Microsoft China R&D Group
Microsoft
Beijing, China
yongrui@microsoft.com

***Abstract*— Read, write, and navigation awareness allow users of a multi-view collaborative editor to get fine-grained information about whether others are reading what they are editing, where others are editing, and to which areas of the document others have navigated, respectively. We derive new high-level tools that directly support the three kinds of awareness. The results of a decision-making study involving the use of these tools revealed that write awareness helps with fine-grained conflict prevention, read awareness reduces unnecessary verbal communication, and navigation awareness is most effective when coupled with read and write awareness. Thus, the study identifies specific uses of the three forms of awareness and motivates tools for supporting them.**

Keywords- Awareness; view divergence; multi-user editor

I. Introduction

Synchronous multi-user editors [5][7] are a practical reality today. As workplace tasks increase in complexity, teams rather than individuals are needed to accomplish them. These teams are finding that traditional single-user editors are of limited use for collaborative tasks and are beginning to use synchronous multi-user editors. Driven by market demand, commercial enterprises are implementing synchronous multi-user editing capabilities in mainstream applications, such as Google's Docs & Spreadsheets, Microsoft's OneNote, The Coding Monkeys' SubEthaEdit, and Sun's JSE8. Such editors can be classified by whether they support view divergence [1][3][4][9][10][16] or not. If view divergence is supported, the participants can work on different areas of the document. We focus on those that support view-divergence.

Based on usage of the OneNote multi-view editor, we have found that view-divergence causes several specific problems. In particular, situations arise in which users' actions conflict, leading to lost work or confusing the users when their changes are applied in an unexpected way. Also, when a discussion of a particular document area is taking place, the discussion leader cannot determine that the discussion topic is stale when other users stop viewing the topic. Moreover, a late-comer may have to interrupt the current discussion to find out exactly what part of the document is being discussed, which can seem unprofessional and cause embarrassment to the late-comer. These problems occur because the multi-view editor they used creates the problem of discovering the co-workers' locations in the document and what they are doing [21].

The question for this paper was whether these problems can be overcome using mechanisms that provide read, write, and navigation awareness, which are not provided by OneNote. These are variations of the previous concepts of gaze and location awareness [12], which are defined as where co-workers are looking and working, respectively. Write awareness is a fine-grained version of location awareness that allows users to determine the exact location of each co-worker's last change. Similarly, read awareness is a special case of gaze awareness that allows users to determine who else can see their last changes. Navigation awareness is a version of both gaze and location awareness that allows users to determine to which portion of a shared document others have scrolled or navigated.

In general, there has been great interest in awareness as it is expected to increase productivity [11] by contributing to the feeling of "being there" [12]. Each of the three forms of awareness schemes has the potential for individually improving collaboration management. Write awareness can reduce the problem of concurrent conflicting changes. Previous work has shown that in WYSIWIS collaboration, where all users can see each other's activities, social protocol can reduce conflicts [6]. By showing the exact edit locations of all users, write awareness has the potential to reduce conflicts in non-WYSIWIS collaborations. Read awareness has the potential to identify stale discussion issues. In addition, [4] found that during a collaboration session, users frequently stop making changes and simply observe changes being made by others. Such information is helpful because when another user who is changing a paragraph has a question about it, then knowing who else is looking at the paragraph helps target the question at the right co-worker. By showing who else is looking at the local changes, read awareness seems like a natural lead-in for ad-hoc collaboration. Navigation awareness could help users synchronize their views. Without such awareness, late-comers to a meeting may have to ask which topic is being discussed, and meeting participants who multi-task may face a similar problem if the discussion migrates to a new topic while they are focused on issues not related to the meeting. In addition, navigation awareness may help a consumer of some information to monitor the progress of a producer of the

This research was funded in part by Microsoft and NSF grants ANI 0229998, EIA 03-03590, IIS 0312328, and IIS 0712794.

information. For example, if one user is editing figures while another is changing the figure data, the first user may use navigation awareness to quickly view the other's changes.

It is important to determine if the potential of each kind of awareness can actually be realized in realistic collaborations, that is, collaborations based on actual rather than hypothesized uses of collaborative tools. However, previous studies have not answered this question. The usefulness of a particular form of awareness is a function of the tools used to support it. The more effort required by the actor and observer to convey and interpret certain awareness, respectively, the less likely it will be that the awareness mechanisms are effective. To try and separate the low-level details for conveying such awareness from its usefulness, in our work, we derive new high-level or lightweight tools for supporting the three kinds of awareness. Using qualitative arguments, we show that our tools are at least as high-level as previous tools. As a result, we do not experimentally compare the various awareness tools with each other to determine how effective they are in conveying the various forms of awareness. Instead, we compare two versions of the OneNote multi-user editor – the original one without any awareness tools and a new one with our tools – to study if the lowest-cost mechanism we have identified for each kind of awareness can positively influence user tasks. This study, thus, determines the usefulness of including tools for supporting these three forms of awareness in a multi-view editor.

The rest of this paper is organized as follows. Based on the above scenarios, we first propose a set of awareness hypotheses that we evaluate in our work. Next, we derive the high-level tools needed to test these hypotheses. We then describe the user study designed to evaluate the effects of the three types of awareness in a decision-making task. Following this, we discuss previous studies of awareness mechanisms. We end with conclusions and directions for future work.

II. Hypotheses

Ideally, we should hypothesize all of the potential uses of the three forms of awareness, but it would be difficult to construct and carry out a lab study in which all of these uses are likely to occur. In this work, we hypothesize only one use of each of write, read, and navigation awareness. As write awareness has the potential to reduce the number of conflicting operations, our first hypothesis is:

H1: In realistic multi-view collaborative tasks, write awareness is useful for reducing concurrent nearby edits.

As described above, read awareness can inform a user if any other users can see the user's last edit position. Thus, compared to having no read awareness, we expect that given read awareness, users entering new information will less often verbally inform others of these changes. Hence our second hypothesis is:

H2: In realistic multi-view collaborative tasks, read awareness is useful for reducing unnecessary verbal exchanges.

Finally, as navigation awareness has the potential to improve the rate at which a user finds where other users are looking, our third hypothesis is:

H3: In realistic multi-view collaborative tasks, navigation awareness is useful for synchronizing views.

III. Deriving the Shadow-Based Awareness Tools

As mentioned above, whether these hypotheses are actually true depends both on the cost of the tools and the benefits of the awareness they provide. The main question of this work is can read, write, and navigation awareness tools overcome the problems of synchronous multi-user editing reported to us by OneNote users. Existing work has developed a number of awareness mechanisms that can be used to provide these three kinds of awareness to various degrees. Comparing the effectiveness of all (or most) of these tools to each other would in itself be an important contribution. However, to provide statistically significant results, the study sample size should be eight or more. In addition, ideally no study participant should participate in multiple experiments. Given the number of existing tools that should be cross compared, the number of participants and experiments that would need to be managed and performed is beyond the scope of a single project.

Fortunately, such a study is not required if qualitative arguments can be used to find mechanisms that are at least as low-cost as other competing mechanisms. To prove the three hypotheses, it is then sufficient to use these mechanisms in a single study that compares a multi-view editor with and without the mechanisms. This is the approach we have taken. It requires us to use the most lightweight or high-level awareness tools we could find or devise in our study. We define the notion of a ***high-level awareness mechanism*** as follows: mechanism A is more high-level than mechanism B if the overhead imposed on the actor and the observer by A is lower than that of B. Combining this definition with the awareness study by [4], we state our first two principles that guide the selection of the awareness tools for our study:

Principle 1: The awareness mechanism must automatically gather information about the actor; in particular, the actor must not be required to perform explicit actions to provide awareness to the observer.

Principle 2: The awareness mechanism must minimize interpretation difficulties compared to other competitive tools; in particular, the observer must be able to obtain awareness of the actor with the least effort.

These principles present the functionality criteria. Based on the comments by our users, screen space is at a high premium. In some cases, dual monitors or a single large display can alleviate the problem [22]. The reality, however, is that users usually do not have such resources available to them, especially when they are using their laptops away from the office. Therefore, awareness mechanisms must also satisfy the in-place design requirement, that is, they cannot occupy additional screen space. A similar requirement was proposed by [4] who state that awareness and the object it is related to should be presented in the same space to improve usefulness. Combining these requirements, we state our third principle:

Principle 3: The awareness mechanism should be in-place; that is, it should not occupy additional screen space.

Using these three principles and qualitative arguments, we derive our shadow-based tools from existing mechanisms and show that they are at least as high-level as previous tools. To follow the in-place principle, our tools are superimposed on the text-editing area and rendered transparently so that they hide no text. Like many previous mechanisms [1][10][18], our tools differentiate users by color. For illustration purposes, we use dark gray, medium gray, and light gray colors to demonstrate the concepts. In real systems, other colors can be used.

A. Write Awareness

As mentioned above, write awareness provides information about each user's exact last editing position. Previous work has identified a number of tools that can be used to provide write awareness to various degrees. One class of such tools present a user with the edit history of all users. One synchronous edit history mechanism color-codes the text by author [1] so that users are aware of who edited which parts of the document. If combined with a "fade-to-regular text color" effect, this solution can also convey how recent the changes are [6]. Another extension of the previous solution is to implement the cloudburst model [6], which hides the changes made by a co-worker from the local user until the co-worker stops making changes and the local user stops typing. When the changes are hidden, the local user sees a cloud over the area the co-worker is editing. This approach is designed to minimize disruptions to the local user. These mechanisms provide some write awareness, but they do not display the co-workers' exact edit positions, the key information needed for fine-grained conflict prevention. As a result, it is difficult for an observer to tell exactly where the actor is editing, which breaks our second principle of minimizing interpretation difficulties.

Unlike edit history mechanisms, telepointers can be used to provide awareness of the actors exact edit position, which means that they satisfy the second principle. On the other hand, to provide write awareness of text editing operations using a telepointer, a user must manually adjust the telepointer with each operation, which is extremely demanding on the actor. Even worse, the actor may forget to make the adjustments in which case telepointers provide no write awareness. Thus, telepointers as write awareness mechanisms break the first principle of automatically gathering awareness information about the actor.

One way to address this issue is to modify the telepointer to automatically follow the insertion point of the actor, that is, effectively create a telepointer-size magnification of a user's caret position and make it public. The work presented in [22] incorporates this idea in telecaret views, which show the immediate area around an actor's insertion point. The telecaret is displayed in the view by highlighting the actor's insertion point. Each time the actor performs an edit operation, the telecaret is positioned immediately after the character that was last changed, and whenever the actor uses the arrow and page up/down keys to move between lines, the telecaret also moves.

The former effect conveys precise write awareness of the actor. The latter effect, on the other hand, can potentially provide false write awareness. For example, when editing a paragraph, the user may use the page up key to temporarily scroll to an earlier part of the document, check that the current changes do not break document flow, and then scroll back down and continue editing the original paragraph. In addition, in our experience, users often click and highlight document areas they are reading, which updates the insertion position even though the users have no intention of making any changes. Hence, while the user is reading the previous parts of the document, another user may look at the changes the first user made, see that they are incomplete, and delete them, all because the pointer has followed the first user's insertion point. To solve this problem, the pointer could be changed to follow the insertion only when the user is editing; in particular, when the user is using the page up/down keys or scrolling, the pointer stays at the last edit position. Such an approach does not attempt to (incorrectly) guess what the user is about to do; instead, it relies only on the information it knows.

Another issue with telecaret windows is that they do not provide information in-place, which violates our third principle of reducing screen space. More subtly, it also violates our second principle of minimizing interpretation difficulties. In particular, to get write awareness, a user must map a location in the telecaret view to a location in the actual workspace, which can be tedious and time-consuming. This problem is solved by showing the pointer in the actual workspace instead of a separate window, thereby following the in-place principle proposed in [4].

Such pointer functionality is provided by a new tool we have created, which we call a write shadow. A "right-hand holding a pencil" shadow follows the local user's last edit position, while a "left-hand holding a pencil" shadow follows a co-worker's last edit position as shown in Figure 1 (left). The shadows are displayed in the workspace itself to minimize the mapping difficulties with telecaret views explained above. As Figure 1 (left) shows, the medium gray co-worker is editing the left note. The local dark gray-user can use the write shadows to avoid editing near or at the co-worker's editing position.

The shape of our write shadow was inspired by the work on shadow communications [15], which found that when people interact with another person with a shadow, they felt uncomfortable stepping on the shadow, and when the shadow touched their own, they felt as if the other person physically touched them. These results inspired us to transform physical shadows into virtual shadows in the workspace awareness domain. The effect of bodily communication, especially of movement, on coordination in a shared physical workspace has been recognized as a part of consequential communication [12], which is the information available as a consequence of a person's activity in the environment. For example, if two people seated at a table reach for the same tool on the table at the same time, they notice each other's motions, normally causing one person to back off. Write shadows are a way of conveying aspects of consequential communication that are based on physical shadows. For instance, in our personal experience, when a shadow crosses our workspace, we tend to look up to see whose shadow it was.

One issue that all write awareness mechanisms must consider is staleness of information. In particular, what happens when a remote user stops editing for a long time, perhaps when

Figure 1. (left) Local user's and a co-worker's write shadows, (center) local user knows that the local edit position can be seen on the two co-workers' screens, and (right) local user sees that the local scrollbar elevator overlaps with two other elevators.

viewing a different location in the document or temporarily switching to another application? One approach is to convey how long ago the change of which a user has write awareness was made. This can be done by fading out awareness information or showing it in different colors. For example, our write shadows could slowly fade out over time or turn to different colors. The basic problem with any such solution dates back to the problems of preemptively releasing locks – just because a user has stopped modifying an object does not mean the user has finished modifying the object. For fear of conveying incorrect awareness information, our write shadows simply always mark the position where a user made the last change (which is known) instead of guessing what the user is planning to do.

A recent study in single-display groupware [23] evaluated the effect of showing the area of a window that overlaps with another window transparently. This work was done independently of and contemporaneously with our research. The results of the study showed that this approach allowed the users to effectively partition screen-space and avoid conflicts. Our write shadows are a finer-grained version of this idea that address intra-window rather than inter-window conflicts.

B. Read Awareness

The counterpart of a user knowing exactly where others are editing is the user knowing when the co-workers can see the user's own changes, that is, having read awareness of the text being edited locally. Telepointers can be used to provide some read awareness. For example, whenever an actor is reading a part of the document, the actor could use the telepointer to trace the text being read much like one uses a pencil tip to follow printed text. An observer who is editing the document will see the actor's telepointer move on the local screen whenever the actor reads near the observer's edit position, thereby getting read awareness. But as in the write awareness case, the actor must manually invoke and control the telepointer to provide read awareness to an observer, which means that telepointers break our first principle of automatically gathering information about the actor. The view rectangles presented in [2] provide more direct support for read awareness as follows. Whenever a part of the local view-port of one user can be seen on the local view-port of another user, this part of the local view-port of the first user is overlayed with a transparent rectangle that identifies the second user (by color). Thus, when the first user is editing in the overlayed area, the user gets read awareness of the second user. This solution is more high-level than telepointers because it collects read awareness information automatically. One minor issue occurs in the case when users have different viewport sizes or are using different resolutions. Suppose there are three users whose viewports are all of different size. In this case, if the two of the largest viewports completely overlap with the smallest viewport, then the user with the smallest viewport will have to do color math to decide which other users can see local changes. In addition, overlapping view rectangles may make it difficult to read the workspace, especially in text editors, where color contrast is important for readability. Thus, view rectangles as read awareness mechanisms break our second principle of minimizing interpretation difficulties.

To solve these problems, we have replaced view rectangles with read shadows, which work as follows. When a co-worker can see the local user's last edit position, and the local user's last edit position is within the local user's view, a read shadow for the co-worker appears in the local user's view immediately above the user's last edit location. It is displayed above the last edit position of a user instead of the current caret position for the following reason. As discussed above, the current caret position does not necessarily indicate the user's current edit position. As read awareness is defined with respect to edit positions, it makes no sense to display it with respect to the caret position. The read shadow is displayed immediately above the local user's last edit position in order to grab the user's attention. As shown in Figure 1 (center), the read shadows provide awareness that the medium and light gray co-workers can see the local user's (dark gray) last edit position. A user who has not modified the workspace, and hence has no write shadow, can have a read shadow showing on one or more collaborators' screens. In keeping with the virtual shadow theme, the read shadow is in the shape of an "over the shoulder" silhouette to portray the fact that someone else can see the local user's last modification as if they were looking over the shoulder of the local user. The co-workers need not be editing near the local user. As Figure 1 (center) shows, there are no co-workers' write shadows.

Any read awareness mechanism, including telepointers, view rectangles, and read shadows, that automatically collects what users can see on their screens suffers from the same problem: the fact that an item appears on a user's screen does not mean that the user is actually looking at it. For example, in a co-editing scenario, if a user is editing a paragraph that appears on the bottom of a co-worker's screen, the co-worker may not actually be following the changes and is instead reading some other paragraph. Nevertheless, in some cases, the session context can make read awareness information more precise. For example, suppose two users are editing two far apart sections of a document so that their views do not overlap. At some point, one user makes some changes and asks the other user to look at them. In this case, when read awareness eventually informs the user that the co-worker's view overlaps with the local view, the user knows that the co-worker is looking at the changes. In our user study, we evaluate another scenario in which read awareness is "contextually precise."

So far, we have presented read and write awareness tools. We next describe our tool for providing navigation awareness.

C. Navigation Awareness

As stated above, navigation awareness allows a user to determine to which portion of a shared document another user has scrolled or navigated. One approach to providing navigation awareness is to display the participants' positions in a mini view of the document. Such a solution is known as a radar view [19]. Aligning position markers in the radar view achieves view synchronization. Radar views can also show rectangles around the document areas that the users are viewing [18], the user's mouse-cursor position [9], color-code content in the mini-view by author [1], and divide the workspace into regions and display region-specific information [13]. Fish-eye radar views allocate more space to document areas in which users are currently in. Some fish-eye views magnify these areas using mathematical functions [18], while others show only these areas and ignore the rest [8].

One problem with radar and fish-eye views is that they require devoted screen space, which means that they break our third principle of not occupying additional screen space. The head-up display presented in [9] addresses the problem by transparently showing the radar-view on top of the workspace. One issue with this approach is that it is possible that the artifacts in the head-up display overlap with workspace artifacts in such a manner that the user experiences double-vision, which could be both confusing and disorientating. It is perhaps for this reason that the authors in [9] note that the display works best in workspaces in which artifacts are sparse. In text based environments, however, there is very little space that is not covered by artifacts (i.e. characters), and therefore, the double-vision problem is particularly prominent.

Unlike the head-up display, which requires no dedicated screen space, multi-user scrollbars occupy a small amount of screen space but do not suffer from the double-vision problem. A multi-user scrollbar shows the scrollbar positions of all the participants in a co-editing session. Views are synchronized by lining up the elevators. One type of multi-user scrollbar displays each elevator in its own column [10]. A solution that requires less screen space (SubEthaEdit) displays all the elevators in a single vertical column. A third solution, which requires a little more space than the second one, but allows an easier comparison of the local user's position with the co-workers' positions is to display the local user's elevator in one column and all the co-workers' elevators in a single separate column [1]. However, this solution does not clearly display elevator overlap when multiple co-workers' views overlap.

To resolve this issue, the remote scrollbar elevators can be displayed in a single column but rendered semi-transparently. When multiple remote elevators overlap, the overlapped region is painted with a color that is a mix of all the overlapped scrollbar elevator colors. To help distinguish overlapped regions from non-overlapped ones, that is, help with color-math, the overlapped regions can be rendered darker and more opaque as the degree of overlap increases. Therefore, this is the functionality provided by our version of the multi-user scrollbar, which is called the shadowbar. A screenshot of the shadowbar that illustrates the overlap is shown in Figure 1 (right). In Figure 1 (right), the local user can tell that the medium and light gray co-workers can, in their remote views, see everything in the local view since their elevator bars encompass the local one. The local user also knows that the two co-workers' views display common information because the dark gray area is showing the overlap of their elevators. This effect is similar to what happens when shadows from multiple light sources cross – hence the name, shadowbar. Based on opaqueness and color darkness, the overlapping regions pinpoint where most users are looking and can be used to navigate to an area being viewed by a group of users.

The shadowbar, multi-user scrollbars, and the mini-view tools provide some read and write awareness, but because screen resolutions and text-editor window sizes vary between users, these tools are not precise, and ultimately, the user must decide if co-workers can actually see the local changes or if the user is about to conflict with another user, which is error prone and breaks our second principle of minimizing interpretation effort. In addition, traditional radar-views do not differentiate between where a user is viewing and where the user last edited [22]. In fact, this is true of all miniature views and multi-user scrollbars. As a solution, a radar-view that displays telecarets was presented in [22] to explicitly provide awareness of both viewing and working areas. However, the authors in [22] recognize the resolution problem with radar views, so to provide more precise awareness they couple the radar view with the telecaret views and over-the-shoulder views, which show more detailed viewing locations of remote users. Thus, the principle of reducing screen space is triple violated in this case. Finally, these tools and all other miniature views are shown in a separate window or to the side of the text editing window. When users are engrossed in typing, they may not notice the scrollbars moving or the mini-views changing [21].

IV. User Study

As mentioned earlier, the main goal of our work was to evaluate the effectiveness of write, read, and navigation awareness in realistic collaborations – to determine if the costs of providing and interpreting such awareness can be made low enough to use them to solve real problems of synchronous editing. Such an evaluation is a function of both awareness mechanisms and task type. By using our read and write shadows and the shadowbar, which we have shown using qualitative arguments in the previous section to be at least as high-level as existing awareness tools, we minimize the tool costs, thereby controlling the impact of the low-level details of these mechanisms on the effectiveness of awareness they provide. As mentioned above, our work was motivated by the issues encountered by users of the synchronous multi-user editor capabilities in OneNote in tasks that involved decision-making. For our study task, we chose one that mimics this real-world. We describe the study and the task next.

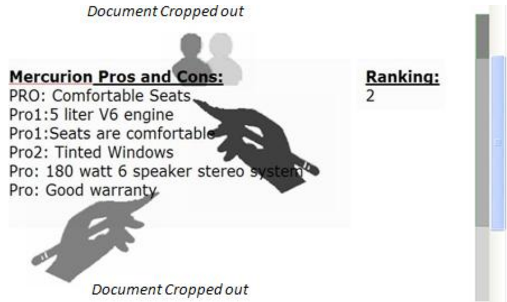

Figure 2. Partial screenshot from one study showing the shadow mechanisms in context.

A. Participant Selection Process

To obtain a proper evaluation of our tools, the study participants had to be representative of our target population: daily document editor users. As a result, our participant criteria were 1) between 18 and 55 years of age, and 2) mid-range expertise with Microsoft Office. The age criterion includes college students and office workers, two large text editor user groups. The mid-range expertise level with Microsoft Office criterion excludes participants whose potential difficulties with OneNote usage could affect their evaluation of the awareness tools. Also, to remove positive bias towards our mechanisms, we preferred users from outside of our organization.

Some previous studies used undergraduate and graduate students and academics as study participants. Such a selection process may yield technically-savvy participants which could bias the study results. To combat this, we offloaded the participant selection task to an external company whose domain-expertise is selecting usability participants. This firm has a rigorous selection process which can evaluate participants on a number of dimensions, including Microsoft Office expertise level. We asked for twenty-four participants. However, out of the twenty-four participants, eight cancelled shortly before their session times. In a best effort to run complete studies, which required participants in groups of three, four OneNote novices from within our organization, who did not know anything about this research, participated.

B. User Study Session

In our organization, we have access to a user study area in which one room, the study room, is separated from the observer-side room by a one-way mirror. Intercom speakers and microphones enable communication between the rooms. We used this area for our study. We provided the participants with non-tablet PC laptops. We seated them at a single table in a manner that did not allow them to see each others' screens.

Since OneNote is relatively new, the participants first received individual OneNote training. Then they participated in a training co-editing session during which they learned about our awareness tools. After the training period, the participants performed the main tasks, one with and one without the awareness tools. Each main task had three stages, which we describe in the next section. In the final part of each session, the participants completed a questionnaire and a short debrief interview during which we delved into specific issues observed during each session. A session ran for about 75 minutes.

C. User Study Scenario

We based our decision-making task on the war-room and job-candidate ranking meetings, both of which typically have three stages. In the first stage, users concurrently enter data: progress reports in war-room and candidate pros and cons in job-candidate meetings. In the second stage, participants discuss this data to make sure everyone is aware of all of it. In the third stage, one user summarizes the meeting. Meanwhile, the co-workers may update previously entered data. For example, interviewers may change their opinions of a candidate. All users must ensure that the summary reflects the newest information.

At the start of our task, the participants were informed that they were acting as employees from a car-buying consulting firm. They were asked to recommend a minivan and a sports coupe for two different customers based on their own research and the customers' preferences. The participants were free to behave as they would in a regular face-to-face meeting. We chose a car-ranking task for two reasons. First, many people car shop or give car advice to friends at some point in time, and thus the task should be of interest to more participants than a candidate-ranking task. Second, car-recommending is, unlike the war-room meeting, not technical and hence should be fun for the participants.

In the first stage of each task, the participants entered pros and cons for the cars based on research data provided in paper form. The participants had similar, but not identical, information as they might during the war-room or candidate ranking meetings. The participants were given eight minutes to enter the data. In the second stage, the participants discussed the top three pros and cons for each car, and were given five minutes to complete this stage. This is similar to discussion stage of the war-room meeting. In the third stage, one user volunteered to rank the cars (for both user study tasks) based on the pros and cons, just like one of the participants in war-room and candidate-ranking meeting volunteers to summarize the meeting. The remaining users updated the pros and cons based on new (provided) research. All users were told that the final ranking must reflect the newest information. This stage lasted another eight minutes.

Figure 2 illustrates the third stage of the task, showing a partial screenshot of one of the users, who was not the ranker. This user's right-handed write shadow is shown in dark grey. In addition, the picture shows shadows for the ranker and one other collaborator, in light and medium grey, respectively. The local user is editing data for the Mercurion car at the location indicated by the dark-grey write shadow. Concurrently, the non-ranking collaborator is editing another pro of the same car, at the location indicated by the medium grey left-handed write shadow. Both collaborators can view the changes made by the local user, as indicated by the two read shadows above the right-handed write shadow. We also see two scrollbars in the picture. The right one is the regular OneNote scrollbar while the left one is the shadowbar. The shadowbar shows the scroll positions of the two collaborators. It is divided into three regions, showing the document portions visible to the medium-grey, both, and light-grey collaborators, respectively.

Table 1. Questionnaire questions and answers (14 questions omitted due to space limits).

#	Questionnaire Question	N	Avg	Std Err	t	DF	p
1	I felt uncomfortable when my write shadow crossed a co-workers "hand with pencil" shadow.	20	2.65	0.3101	-1.13	19	5.462
2	I tried to keep my write shadow away from a co-worker's "hand with pencil" shadow.	19	2.53	0.3852	-1.23	18	4.692
3	I felt more comfortable making changes to the document when I knew where my co-workers were.	19	4.37	0.1746	7.84	18	<.0001
4	It was easy to find my co-workers' positions in the document.	20	4.45	0.1141	12.7	19	<.0001
5	Awareness tools helped me coordinate my work with that of my co-workers.	20	4.15	0.15	7.67	19	<.0001
6	With awareness tools, I was interrupted more by my co-workers. *	20	3.85	0.2436	3.49	19	0.05

* the null hypothesis assumes positive questions so original value of 2.15 is interpreted as 3.85 since the question was negative

We also see here how shadows are useful for this collaboration task. Users are able to avoid conflicts by ensuring that their write shadows do not cross. The shadowbar and write shadows allow the ranker to poll for the changes made by other users. In turn, a user can use the ranker's read shadow, together with the task context which implies that the ranker is polling, as an indication that the ranker is looking at the local changes.

D. Data Collection

In total, twenty users participated in seven sessions: six sessions with three people and one with two. To control for ordering effects, in four (including the single two-user session), the participants performed the first task (minivan) with and the second task (sports car) without the awareness mechanisms. In the remaining three sessions, the users performed the tasks in the same order but did the first task without and the second task with the awareness mechanisms. We collected study data through a questionnaire, debrief sessions, and OneNote logs.

The questionnaire had three sets of questions that 1) evaluated the users' awareness mechanism experience, 2) compared working with and without the awareness mechanisms, and 3) measured the most and least favorite mechanism. The first two sets of questions were answered on a Likert scale of 1 ("strongly disagree") to 5 ("strongly agree") and balanced positive and negative questions. Each question also had an "N/A" option. The third section was multiple choice. For each questions in the first two sections, we pose (the standard notion of) a null hypothesis that an average value of 3 or less (reversed for negatively stated questions) implies that the users do not agree with the question. We chose 3 as the value because if the average of the users' answers is higher than 3, then on average users had to agree or strongly agree with the question. Using a one sample t-test, for each question we test the average of the users' answers against the null hypothesis to see if it is significantly different. We used a Bonferroni correction to reduce the potential of calculating a significant effect by chance.

We used the debrief session to delve into any issues we observed during the user tasks. For example, we asked users who seemed confused when performing the tasks without the tools to recall what caused their confusion. The debrief sessions were videotaped for later review.

Finally, we recorded logs of the edit and view locations for each user, one log for each main task. The log entries were recorded whenever awareness information (a shadow or shadowbar positions) changed. To reduce the size of the logs, we recorded entries at one second intervals. Since a single user cannot do many operations in one second, and since edit and awareness updates were sent to other users every one and a half second, recording at one-second intervals captured the information required to retrace users' exact edit and view positions later. We excluded two users' logs, because the users did not follow the instructions. This left 10 logs for users who did the first task with the tools and 8 logs for the other users.

E. User Study Results

1) Write Shadows: To test if write shadows, and hence, write awareness, will be used for conflict avoidance, we included questions Q1 "I felt uncomfortable when my write shadow crossed a co-workers 'hand with pencil' shadow" and Q2 "I tried to keep my write shadow away from a co-worker's 'hand with pencil' shadow" (Table 1) in the questionnaire. The users neither agreed nor disagreed when asked if they avoided crossing write shadows, 2.7/5, or if they felt uncomfortable when their write shadows crossed with those of the co-workers, 2.5/5, but these averages were not significantly different from the null hypothesis. It *appears* that the write shadows were not used for conflict avoidance. However, the debrief sessions revealed that write shadows were used precisely for conflict avoidance. One user expressed that "if there was a hand that was near [my hand] typing something, I knew to press enter and do [my own] entry, and then it wouldn't interfere with [the co-workers]." Another user commented that with the awareness tools, he avoided conflicts because "if I saw that someone was typing in the same area, I would go just up a little bit or down a little bit … and start a new line." The same user also said that to avoid conflicts without awareness tools "I choose some very far away section [from others]." Furthermore, the write shadows allowed fine-grained conflict avoidance as users could "go just up a little bit or down a little bit" from a co-worker's write shadow to make their own changes. When asked if they felt more confident making changes to the document knowing the co-workers' positions in Q3 (Table 1), the mean response, 4.4, was significantly greater than the null hypothesis of 3 ($t(19)=7.84, p<0.0001$); thus, user responses indicate that this was the case, which is consistent with the comments above.

Table 2. Percentage of time user's edit positions were 0, 1, and 2 lines apart when editing nearby

Lines Apart	0	1	2
1st With	23.3	20.8	21.4
2nd Without	15.9	16.4	30.1
1st Without	11.9	24.3	30.4
2nd With	6.9	11.7	21.0

A careful log analysis revealed interesting findings. We analyzed the recorded logs for the vertical distances between edit positions when users were editing near each other, that is, when users could see write shadows of other users, which includes any conflicting edit operations. To measure the effect of the write shadows, we measured the time users spent editing at various vertical distances (in multiples of line heights) apart as a percentage of the total time when they edited near each other, that is, when one of the users could see one or more other user's write shadows. We defined the height of a line to be the height of the 16 point Verdana font as this was the font size used in the study. We calculated this data for the first task with and without the tools, and similarly for the second task.

Table 2 shows the percentage of time when users were 0-2 lines apart from each other, where edit conflicts were most likely to occur. Our first hypothesis is that the tools will decrease the number of nearby line edits (conflicts). While the tools may change the conflicts, familiarity of the OneNote system may also change the conflicts. This is the well-known ordering effect. We next analyze how these two factors explain the data in Table 2. Let α be the conflict decrease caused by the familiarity of the OneNote editor, that is, the ordering effect. Let γ be the conflict decrease caused by the awareness tools. Our goal here is to data mine γ out of the data in Table 2, taking into account of the ordering effect α. After taking out the ordering effect, α, we have $\gamma = 12.0$. A logarithmic regression analysis revealed that the result is not statistically significant. Nevertheless, the 12% reduction in concurrent edits 0-2 lines apart offers some support for the effectiveness of write shadows as conflict avoidance tools. Thus, the log analysis offers further support for our hypothesis and matches the debrief interview comments. Interestingly, the users who did the second task with our tools reduced edits 0 lines apart by 5.0% ($Z = -2.29$, $p = 0.0219$) and edits 1 line apart by 12.6%, ($Z = -2.99$, $p = 0.0028$) in the second task. These users made a transition much like the one real-world users of a multi-user editor without awareness mechanisms would if our tools were added to the editor. As a result, their performance is perhaps more relevant than that of the other users.

During the interviews, some users expressed that conflicts did not occur as much with the write shadows and that the write shadows were "necessary." In addition, one group of users expressed that in absence of explicit conflict resolution tools such as locks, they could avoid conflicts by "knowing where someone was [with which] the hands helped the most."

2) Read Shadows: Optimally, to test if the read shadows, and hence read awareness, reduce verbal communication, the conversation amount with and without the awareness mechanisms should be compared. However, quantifying "conversation" is difficult, so instead, we delved into communication issues during debrief interviews. User comments seem to indicate that the read shadows did decrease unnecessary verbal communication. As one user expressed it, with "[read] shadows, I saw that he was looking at what I was typing, [so] I didn't need to tell him." Another user stated that "I actually liked knowing where people were viewing things, […] and the [read] shadow was nice because I would know 'oh they saw that' and that was helpful." Both users were talking about the ranker viewing their changes. The debrief session comments also explain why the positive effect was not stronger. Some users who made no use of read shadows expressed that they would have used them in different scenarios. One user was more specific and stated that "with time constraints, you tended to rush more. If [in] a different scenario, you could actually take time when you saw that someone was looking, and allow them [to see]," the read shadows would be useful. Users in one group who did not use the read shadows described a scenario in which they would have used them. According to them, the read shadows "would help if you wanted to make sure someone was reading your stuff." They added that this could happen if one user said "everyone go to paragraph two and [the user] would know that everyone is there." Thus the results offer further evidence for our second hypothesis: read awareness reduces unnecessary verbal communication.

3) Multi-user Shadowbar: To test if the shadowbar, and hence navigation awareness, helped synchronization user find where other users were looking, the questionnaire contained a question about to the relative ease of locating co-workers in the document. The mean value of the user responses to question Q4 "It was easy to find my co-workers' positions in the document" in Table 1, was significantly greater than the null hypothesis ($t(20)=12.7$, $p<0.0001$); thus, the users responses indicate that the users were able to find the co-workers' positions easily with the awareness mechanisms.

During the debrief sessions, we investigated how the users found their co-workers. For some users, the general document location provided by the shadowbar was enough. In fact, some of these users did not use the read shadows because they obtained read awareness from the shadowbar. For example, one user said that the read shadows were not useful because "I've already got the scrollbars to tell me where they are on the page." However, for some users, the shadowbar was useful only in conjuction with one or both of the other mechanisms. One user stated that to find the co-workers, "the bar gives the general area […] and then if there is a hand or the shoulder [you would know more]." Other users expressed similar opinions. To find out where a co-worker was editing, the multi-user shadowbar was used to jump to a general area and then the co-worker's write shadow would be used to find the co-worker's exact position. If a user was interested in read awareness of the local changes, the shadowbar was used in conjunction with the read shadows. One user said that "if you knew where [the co-workers] were scrollbar-wise and they were looking over your shoulder, you knew where they were." Finally, while we did not specifically test for the shadowbar usage when the elevators overlap, these situations did occur during the study. During the ranking stage, the users entering new data sometimes concurrently changed the same car. Thus, on the ranker's screen, their shadowbar elevators overlapped, informing the ranker where to look for new changes.

4) Other Results: Apart from evaluating each mechanism individually, it is also useful to compare the mechanisms with one another. For example, investigating which mechanism was the most or the least liked can offer insights into how to change the mechanisms or to decide which mechanism to remove. In the questionnaire, the users voted for their favorite and least favorite awareness mechanisms were. The write shadows, the shadowbar, and the read shadows received 11, 7, and 2 most favorite, and 4, 2, and 2 least favorite votes, respectively. Thus, the write shadows were liked the most. There could be multiple reasons for this. First, as shown above, they were used by users for fine-grained conflict resolution, and as such instilled confidence in users when making changes. A second, more interesting, reason could be that when searching for other users, looking for write shadows can be as effective as using the shadowbar, especially in short documents.

We also measured the effectiveness of our tools for coordinating changes and reducing interruptions. Because the mean values of the users response to questions Q5 "Awareness tools helped me coordinate my work with that of my co-workers" and Q6 "With awareness tools, I was interrupted more by my co-workers" in Table 1, were significantly higher than the null hypothesis ($t(20)=7.67, p<0.0001$; $t(20)=3.49, p=0.05$), the users' responses indicate that they did not feel that they were interrupted more by their co-workers with enabled awareness mechanisms. This was confirmed in the debrief sessions. One user stated that "[without awareness tools] I was like scroll, scroll, scroll. Where are they editing? Come on guys tell me where you are editing!" Another user even explicitly stated that "if you want minimal verbal communication [the tools] are good!" Also, the users felt that our awareness tools helped coordinate changes, which is arguably the most important task of the awareness mechanisms.

Finally, user comments like "Get this on the market!" and "How many years before we see these tools?" speak for themselves.

V. Related Work

Awareness mechanisms were studied in the context of ShrEdit [4], a synchronous multi-user editor, which the users used to design an automated post office. But instead of evaluating awareness or defending or comparing specific tools, they propose awareness tool design guidelines. As mentioned before, we use these guidelines to define our principles for selecting which awareness tools to use in our study. Another study [1] compared the mechanisms of multi-user scrollbars and text-specific radar-views, and reported that the users preferred having them to not having them without explaining the ways in which these tools were used. A later study [10] extended these results by evaluating the usability of telepointers, a number of radar view variations, and a multi-user scrollbar. They found for collaborative newspaper layout tasks, the users preferred the miniature and radar views over the other widgets in terms of the type of and ease of interpretation of the information they displayed. Again, the focus was on the usability of the tools rather than kinds of uses to which the tools were put. As the study found that the users liked the tools, it can be concluded that some form of awareness supported by them was useful. It is not clear from this study which specific forms of awareness defined here were used. In fact, based on qualitative arguments we presented in Section 3, it is likely that neither read nor write awareness was used. Moreover, it is not clear if the task required any view synchronization, and if it did, if users actually used the studied mechanisms to achieve it. In fact, the system used in the study provided a special button to teleport to another user's view. The paper indicates that several users did not notice/remember this feature and does not indicate if anyone actually used it. A more recent study [22] compared a basic radar view with a combination of radar, over-the-shoulder, and telecaret views. The results revealed that users preferred the combination to the simple radar view and that the users thought it was important to distinguish where a user is viewing from where the user's insertion point is. Like the previous study, the focus was on the tool usability rather than what exactly the tools were used for. Another study [11] involved "electronic welding of a pipeline." In this task, a director asked a performer to execute certain actions (determined by a script provided by the experimenters) and checked that the actions were executed correctly. The results showed that radar views reduced verbal communication and task completion time, but did not indicate which kind of awareness supported by these views contributed to these improvements. An issue with this study is that if a director knows what exactly is to be done, she could execute the tasks alone – there is no need for the performer. If the idea is to train the performer, then the computer could use the script to direct and check the pupil, as is done in simulation programs – there is no need for the director. To the best of our knowledge, this task is not representative of any actual usage of a collaborative editor to solve a real activity reported in the literature. A highly scripted exercise such as this one is ideal when comparing alternative awareness tools as the task steps are more or less the same in different instances of the exercise using different tools. However, it leaves open the question of what specific forms of awareness are useful in actual multi-view collaborations.

In summary, none of the previous studies listed above and others we are aware of [3][14] has either formulated any of our hypotheses or reported any of our specific results.

As mentioned above, during a co-editing session, users' actions may conflict. It is important therefore to provide both "syntactic consistency," that is, ensure consistency of users' displays, and "semantic consistency," that is, guarantee that the results of concurrent actions are semantically meaningful. As OneNote provides syntactic consistency, we studied only the prevention of conflicts that lead to semantic inconsistency. One way to handle semantically conflicting edits is to not let them happen in the first place, which is the reasoning behind locking approaches. However, locking schemes present several difficulties one of which is selecting lock granularity [6]. Write awareness addresses these issues by relying on social protocol for prevention of semantically conflicting edits.

VI. Conclusions and Future Work

Perhaps the earliest question in CSCW [20] is whether users should see the same or different views. Sharing the same view has the practical advantage that a generic application sharing system can be used to support this interaction for an arbitrary application. From the users' point of view, it allows users to know what their collaborators are seeing. However, it requires constant synchronization of user views, which is costly. Moreover, it does not support concurrent manipulation of independent parts of the shared object. Therefore, there has been much work in systems supporting view divergence. A variety of tools have been created for such systems to provide users with awareness of the views of others. Moreover, studies have been done to evaluate these tools. *However, no previous work has answered the following question: What specific problems are created by view divergence in realistic collaboration tasks and which awareness solutions can overcome some of these problems?* The answer to this question can increase our understanding of this domain, lead to more targeted research, and influence commercial products. This work has addressed this question.

Specifically, our contribution consists of the design of new awareness tools and the evaluation of the awareness they provide in a realistic decision-making task. To perform the study of the effectiveness of awareness, we derived three new awareness tools from existing mechanisms that directly provide write, read, and navigation awareness, which are at least as high-level as existing mechanisms. Our experiment with these tools shows that 1) write awareness reduces nearby concurrent edits, and hence conflicting user operations, 2) read awareness reduces unnecessary verbal communication, and 3) navigation awareness is most effective when coupled with read and write awareness. Thus, the design and evaluation components of our work had a symbiotic relationship in that the tools enabled the study, which in turn, justified including them in a synchronous multi-view editor.

In addition, we have identified three general phases of two decision-making tasks in which views divergence is actually used: war-room and candidate-ranking meetings. Moreover, we have identified a lab task that includes these three phases and, thus, can be used as a basis of other studies of awareness tools. This task is perhaps our most important contribution it is both realistic and allows specific benefits of awareness tools to be studied. Furthermore, we have identified the new metric of vertical line separation to evaluate collaborator adjacency. Finally, we qualitatively evaluate how well existing mechanisms can be used to convey write, read, and navigation awareness. In fact, in many cases we point a number of subtle but important issues with many of these tools regardless of what kind of awareness information they are conveying. Future versions of these mechanisms can become even more useful by addressing these issues found in their predecessors.

There are a number of possible ways to extend our work. Naturally, a whole range of additional user studies of awareness is desirable. Our work does not imply that alternate mechanisms that provide same or lower-level support for the three kinds of awareness will not yield the benefits that resulted from our mechanisms. This work should encourage studies that determine if the cost of these mechanisms is also low-enough compared the benefits. The task we have devised can be used to conduct such studies. We have taken the in-place requirement imposed on us by OneNote designers as an axiom. It would be useful to experimentally determine how crucial this requirement is to users. It is important to experiment with potential uses of the three forms of awareness identified here that were not covered by our study. In addition, exploring the effectiveness of read and write awareness in graphical and other text-based collaborations would be interesting. The main future direction triggered by this paper is further research in high-level tools for supporting the three forms of awareness. The shadow-based tools we describe here are only a first-cut at devising such tools and could be augmented/replaced with mechanisms supporting, for example, other shadow representations and sizes, or metaphors other than shadows.

REFERENCES

[1] R.M. Baecker, D. Nastos, I.R. Posner, and K.L. Mawby, “The user-centred iterative design of collaborative writing software,” INTERCHI 1993.

[2] M. Beaudouin-Lafon and A. Karsenty, “Transparency and awareness in a real-time groupware system,” UIST 1992.

[3] J. Begole, J., M.B. Rosson, and C.A. Shaffer, “Flexible collaboration transparency: supporting worker independence in replicated application-sharing systems,” TOCHI, June 99.

[4] P. Dourish and V. Bellotti, “Awareness and Coordination in Shared Workspace,” CSCW 1992.

[5] C.A. Ellis, S.J. Gibbs, and G.L. Rein, “Design and use of a group editor,” EHCI 1990.

[6] C.A. Ellis, S.J. Gibbs, and G.L. Rein, “Groupware: some issues and experiences,” CACM 1991.

[7] G. Foster and M. Stefik, “Cognoter: theory and practice of a collaborative tool,” CSCW 1986.

[8] S. Greenberg, C. Gutwin, and A. Cockburn, “Awareness through fisheye views in relaxed-WYSIWIS groupware,” *GI 1996.*

[9] Gutwin, C., Greenberg, S., and Roseman, M., “Workspace awareness support with radar views.” *CHI 1996*.

[10] C. Gutwin, M. Roseman, and S. Greenberg, “A usability study of awareness widgets in a shared workspace groupware system,” CSCW 1996.

[11] C. Gutwin and S. Greenberg, “The effects of workspace awareness support on the usability of real-time distributed groupware,” TOCHI 1999.

[12] C. Gutwin and S. Greenberg, “A descriptive framework of workspace awareness for real-time groupware,” JCSCW 2002.

[13] D. Li and J. Lu, “A lightweight approach to transparent sharing of familiar single-user editors,” *CSCW 2006.*

[14] I. Marsic and B. Dorohonceanu, “Flexible user interfaces for group collaboration,” IJHCI 2003.

[15] Y. Miwa and C. Ishibiki, “Shadow communication: system for embodied interaction with remote users,” CSCW 2004.

[16] J.S. Olson, G.M. Olson, M. Storrøsten, and M. Carter, “GroupWork close up: a comparison of the group design process with and without a simple group editor,” TOIS, 11(4), Oct 1993.

[17] G.K. Raikundalia and H.L. Zhang, “Newly-discovered group awareness mechanism for supporting real-time collaborative authoring,” AUIC 2005.

[18] W.A. Schafer and D.A. Bowman, “A comparison of traditional and fish-eye radar view techniques for spatial collaboration,” GI 2003.

[19] R.B. Smith, T. O’Shea, C. O’Malley, E. Scanlon, and J. Taylor, “Preliminary experiments with a distributed, multi-media, problem solving experiment,” ECSCW 1989.

[20] M. Stefik et al., “Beyond the chalkboard: computer support for collaboration and problem solving in meetings,” *CACM 1987*.

[21] M. Stefik, D.G. Bobrow, G. Foster, S. Lanning, and D. Tatar, “WYSIWIS revised: early experiences with multi-user interfaces,” TOIS, 5(2), Apr 1987.

[22] M.H. Tran, Y. Yang, and G.K. Raikundalia, “Extended radar view and modification director: awareness mechanisms for synchronous collaborative authoring,” AUIC 2006.

[23] T. Tsandilas and R. Balakrishnan, “An evaluation of techniques for reducing spatial interference in single display groupware,” ECSCW 2005.