Available online at www.sciencedirect.com
SCIENCE DIRECT
Int. J. Human-Computer Studies 64 (2006) 583–598
International Journal of Human-Computer Studies
www.elsevier.com/locate/ijhcs

A framework for asynchronous change awareness in collaborative documents and workspaces ☆

James Tam*, Saul Greenberg

Department of Computer Science, University of Calgary, Calgary, Alberta, Canada T2N 1N4

Available online 30 March 2006

Abstract

Change awareness is the ability of individuals to track the asynchronous changes made to a collaborative document or graphical workspace by other participants over time. We develop a framework that articulates what change awareness information is critical if people are to track and maintain change awareness. Information elements include: knowing *who* changed the artifact, *what* those changes involve, *where* changes occur, *when* changes were made, *how* things have changed and *why* people made the changes. The framework accounts for people's need to view these changes from different perspectives: an *artifact-based* view, a *person-based* view and a *workspace-based view.* Each information element is further broken down into distinguishing features and matched against these perspectives, e.g., *location history* within the *where category* prompts the questions 'where was this artifact when I left' in the artifact-based view, 'where in the workspace has a person visited' in the person-based view and 'where have people been in the workspace' in the workspace-based view. The framework can be used both to inform and critique change awareness tools.

Keywords: Change awareness; Asynchronous awareness

1. Introduction

People often collaborate (Nunamaker et al., 1997) for the purpose of creating and developing a work artifact over time such as when: people co-author papers, iterate designs from conception to final form, or negotiate a plan through an evolving blueprint. The participation of all partners is vital, perhaps due to the group's particular combination of skills and expertise, or because all participants are interested stakeholders, or because there is too much work for one person to do by themselves, or because involvement is required if participants are to buy into the final outcome.

While many episodes of collaboration often occur in face to face meetings over the work artifact, others frequently occur asynchronously. Asynchronous collaboration and evolution of the artifact can happen in several ways:

People may explicitly pass the artifact back and forth for comments and revisions (i.e., 'it is your turn, give it back to me after you have worked on it').
Individuals may work on the artifact as time and opportunities arise, without explicitly coordinating this with the other participants.
The group may drift in and out between collocated and asynchronous work (e.g., a group may begin work in an extended face to face meeting, but members may leave and return to the meeting over its course).

In same-time collaborations, we already know that people use *workspace awareness* not only to follow actions of others, but to understand and respond to any changes others make to the workspace artifact (Gutwin, 1997) (see Section 3). The problem is that when people interact asynchronously this awareness disappears; changes are only understood if one person tells the other what they have done (e.g., through prior coordinating talk, on-going

☆This paper expands considerably on an earlier conference paper (CRIWG'04 X International Workshop on Groupware, Lecture Notes in Computer Science (LNCS Number 3198), Springer Verlag). The significant major changes include all the new material in Section 5.

*Corresponding author. Tel.: +1 403 210 9455; fax: +1 403 284 4707.
E-mail addresses: tamj@cpsc.ucalgary.ca (J. Tam), saul@cpsc.ucalgary.ca (S. Greenberg).

1071-5819/$ - see front matter © 2006 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ijhcs.2006.02.004

emails, or notes attached to the artifact), or if the person can somehow understand the changes made by inspecting the artifact. If people cannot understand what has changed, collaboration can quickly spiral out of control. Missed changes made by one person can unintentionally wreak havoc on the work of others or even the entire project.

Within this context, our research interest is on *asynchronous change awareness of artifacts* (which we call change awareness for short), defined as the ability of individuals to track the asynchronous changes made to a collaborative document or surface by other participants. Our research concentrates primarily on how people maintain change awareness by inspecting the changed document, rather than on how change awareness is transmitted by other communications e.g., verbal or textual dialog that occurs directly between people outside the confines of the document.

Because change awareness is a broad subject, our immediate goal in this paper is to contribute a framework of the critical information people need if they are to maintain change awareness. Simply showing all the information may simply be too overwhelming. To achieve this goal, we took an existing framework for awareness in face to face collaboration, Gutwin's framework for workspace awareness (Gutwin, 1997), and modified and extended it to account for awareness of changes to graphical documents over time. At the same time, we developed a set of techniques for displaying awareness information in a drawing editor which further influenced the development of our framework.

This paper unfolds as follows. First, we set the scene with several examples of failures that arise when people have inadequate change awareness, and then describe how current systems try to provide this information. Second, we summarize Gutwin's earlier framework for workspace awareness for real-time interactions (Gutwin, 1997), because it acts as a theoretical precursor to our own work. Third, we introduce and describe in detail our framework for change awareness. Finally, we discuss several implications this framework has to practitioners and implementers of change awareness systems, and show how the framework can be used to critique change awareness features in an existing collaborative workspace.

2. Motivations and related work

2.1. Several true incidents give an example of the consequences of missed changes

The Town of Canmore in Canada has an administrative office that oversees all subdivision plans submitted by developers. In this process, the developers and the administrative office negotiate the plan through many back and forth exchanges. The administration would ask for changes so that the subdivision fit the needs of the town, and the developers would respond by incorporating these (and perhaps other) changes in a modified plan of the subdivision development. In one on-going subdivision plan, the developers added a gate to the main road entrance, a security measure that inhibits 'outsiders' from entering the grounds. The administrative office did not notice this addition, and approved that particular version of the plan. This oversight was only seen after the subdivision was built with the gate in place. The gate generated wide-spread controversy, where it made the front page of the local newspaper and became a heated issue for the town council, the town population, and the developers. Because Canmore was a small town fostering community values, the townspeople felt that this gated community created a private enclave that violated the sense of community held by its population, and that it would also set a bad precedent for future developments. The developers, on the other hand, believed that they had followed the planning process correctly, and because the plan had been approved they had the right to implement their vision. The developers were adamant that they had not tried to 'slip in' this gate amongst the other changes in the plan, and indeed had told a staff member about it. The administrative staff said that the key staff members did not see the addition of the gate; it was a visually small change in a complex plan that had many other changes within it. The town council stated that "they didn't know about the plan and wouldn't have approved it had they known" (reported in the Rocky Mountain Outlook, March 11, 2004). What happened was that the administrators lacked awareness of what had changed between documents, and thus missed a small but critical detail.

Another true example focuses on missed changes within TeamWave Workplace, a commercial room-based groupware environment (Greenberg and Roseman, 2003). Community members could enter a virtual room at any time and change its contents, i.e., a collection of persistent groupware applications and documents scattered on the wall of the room. The only difference between synchronous and asynchronous work was whether people happened to inhabit and make changes within the room at the same time or at different times. Independent evaluators of TeamWave found that its major usability problem was that people had difficulty maintaining awareness of whether anything had changed since their last visit, and what those changes concerned (Steves et al., 2001). "Participants reported that they were unable to effectively and efficiently detect modifications to artifacts made by others in the rooms they were using. TeamWave users devised [email workarounds] to provide the information that they needed" (p. 5). The problem was that TeamWave users had to resort to manual techniques to monitor changes. To see if anything had changed, they had to start and login to the application, navigate to a room, and visually inspect it. Because this is a heavyweight and error-prone process, people did not visit rooms often enough to notice if anything had changed, and when they did visit a room some changes were easily missed. In turn, this meant that people were increasingly reluctant to leave time-critical

information in a room, for they knew that others were unlikely to notice it until too late. The usefulness of the entire system suffered as a consequence.

Our third example illustrates the effort people go through if a system does not explicitly provide a mechanism revealing change awareness information. A typical strategy is to compare the two document versions—by memory or visual inspection—and to try and spot changes and decipher their meaning. This is both a daunting and error prone task. For example, Fig. 1(a) and (b) shows a before and after version of a UML class diagram, respectively. Although a developer working on the project may be able to eventually determine all the differences, it requires much time and effort to do so, and changes are easily missed. Try this for yourself: see how many differences you and your collaborators can spot in (say) 30 seconds. Have a first person try it when both images are side by side (for direct visual comparison). Have a second person try it when the images are on opposite sides of the same piece of paper (for short-term memory comparison). Have a third person look at the first image on the first day for 30 seconds, and then have them examine the second image on the second day for changes (for long-term memory comparisons).

Of course, the document itself can help reveal changes. For example, the version of the UML diagram depicted in (c) highlights specific blocks of changed items by bolding them and by muting unchanged items (coloring would be preferred, but this is not possible in this black and white reproduction). While limited, even this simple technique clearly allows people to spot changes faster and with fewer errors than by using manual comparisons.

These examples clearly show the need for effective awareness when working asynchronously in collaborative workspaces and documents. This is not a new idea, and other researchers (Morán et al., 2002) as well as developers recognize the importance of change awareness in these situations. Many commercial applications concerning sequential documents (e.g., collections of reports, papers, and source code) already detect and show change information between new and old versions. A sampling of standard techniques is listed below.

The first class of change awareness techniques contains several ways that differences between two versions of a document are visually displayed on the screen:

Sequential deltas, as exemplified by the UNIX *Diff* system (Hunt and McIlroy, 1975) insert instructions after lines that are deemed different. Instructions describe how the line(s) in the previous document can be transformed into the new document, i.e., instructions

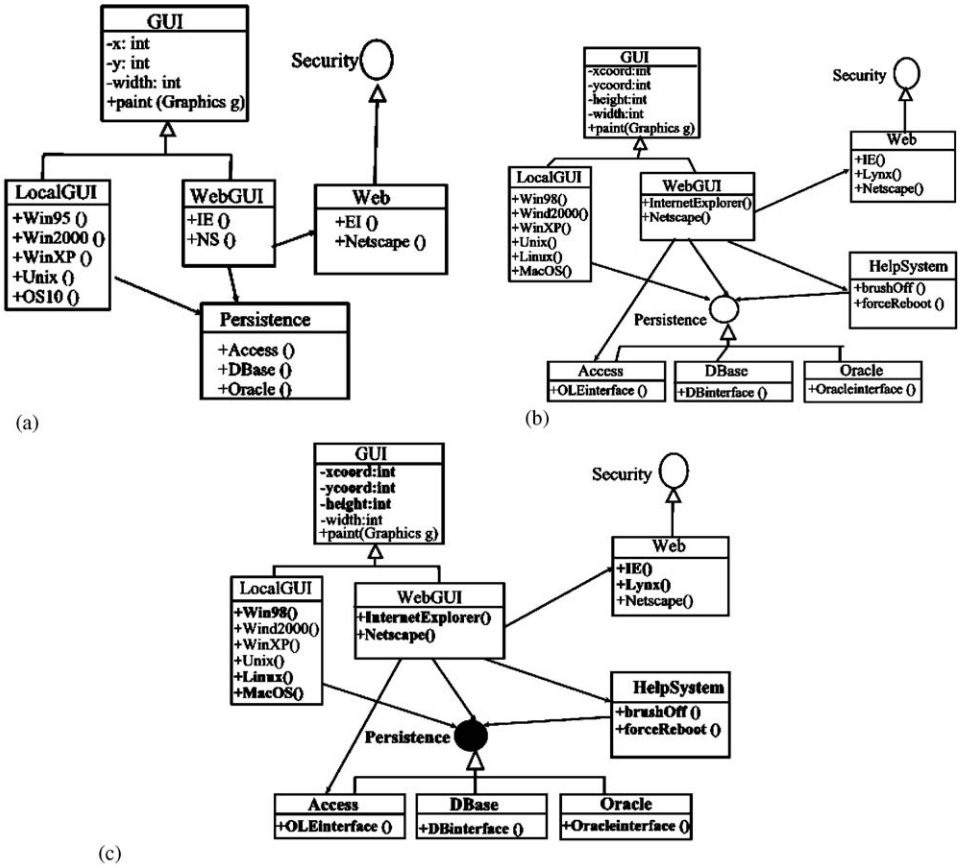

Fig. 1. A UML diagram in its original, changed, and visually enhanced changed form: (a) original version, (b) modified version, (c) modified, with changes now shown in bold and unchanged parts muted.

to add or delete new lines, or instructions that transform an old line into a new form.

Annotations and markups display differences as explanatory notes that points to the parts of the document that has changed. Microsoft Word's *track changes* capabilities, for example, optionally displays changes as margin notes (Microsoft, 2003). Flexible *Diff* (Neuwirth et al., 1992) accompanies the source text with multiple columns of annotations. The first and second columns contain the original and modified text. The third column shows only the differences (where thresholds can be specified to mute small differences), while the fourth shows explanatory annotations added by the author. Another example is how Rational Rose shows changes in object-oriented source code IBM. In one of its views, it displays objects within a hierarchical class browser, and marks items that have changed by special symbols positioned next to the item.

Highlighting displays the differences within the document itself by directly marking up the appearance of the document contents. One example is the UML diagram in Fig. 1c. Another is Microsoft Word's ability to show inserted text in a contrasting color, and deleted text as crossed out (changed text is treated as a combination of deleted and new text) (Microsoft, 2003).

Overviews communicate changes made to an entire document at a glance. They usually provide a graphical miniature of the document, and mark regions that have changed through highlighting. Attribute-mapped scroll bars (Hill and Hollan, 1992) provide indications of edit-wear (i.e., the areas where people have repeatedly edited the document) as marks within a scroll bar. Seesoft provides a zoomed out view of source code scattered over several files (Eick et al., 1992). Each column of the overview represents a file; each character is compressed into a pixel. Line coloring indicates change to the code, such as its age and the developer responsible for adding it. State TreeMaps uses a Treemap overview to emphasize changes made within multiple documents (Molli et al., 2001).

Graphical playback replays changes over time by showing all fine-grained editing actions, or by storyboarding major changes into easily comparable scenes (Kurlander, 1993; Tam, 2002).

The second class of change awareness techniques describes how document versions are maintained or specified, and how changes within them are detected or tracked over time. Each technique may use one or more of the change display mechanisms described above to reveal its information to the end user.

File differencing occurs when a user manually specifies two files and asks the system to compare them for differences (Hunt and McIlroy, 1975).

Real-time differencing lets the author turn on a facility to continually track changes. The document itself stores these changes internally. Microsoft Word serves as a good example, where text is marked up as the author types (Microsoft, 2003). All change information is kept until an author decides to accept or reject them.

Version control systems automate and enhance the process of manually tracking progressive versions of documents and their changes (Brown, 1970; Rochkind, 1975; Tichy, 1991; Magnusson and Asklund, 1996). The first version is typically created by freezing the current document. The frozen version can no longer be changed, and so editing it implicitly creates a new revision (Tichy, 1991).

History systems track all incremental changes made to a document, typically so they can be played back at a later time or so actions can be undone (Kurlander, 1993; Tam, 2002).

Most of the above are ad hoc solutions for sequential documents, and even then there is much they leave out. They typically neglect change awareness in two-dimensional graphical documents (Tam, 2002) such as: figures, photos, blueprints, concept maps, graphs, UML diagrams and collaborative workspaces containing spatially scattered artifacts. While 2D documents are widespread, techniques for displaying change awareness within them are undeveloped and are likely non-trivial (Tam, 2002); we do not even know what change information they should show to the end user. Our goal is to fill this void.

3. Theoretical foundations

The prerequisite to understanding change awareness is to determine what information is necessary if people are to comprehend change in the collaborative workspace. These informational elements of knowledge verbalize, categorize and explain what information should be tracked and captured by an application as a change occurs and how this information could be useful to an end user. Once the informational elements have been specified, we can then design an interface that captures and display this information in a meaningful and useful fashion. The details of the first step—the information elements—are the focus of the next two sections. Our theoretical foundations of these information elements have their roots in Gutwin's framework for workspace awareness (Gutwin, 1997), summarized in this section and in Tables 1 and 2.

3.1. Workspace awareness for real-time interactions

Gutwin focused on *workspace awareness* within real-time groupware environments. He was concerned with people's continuous maintenance of awareness of others in a visual workspace and how others were interacting with the artifacts held by that space. He articulated a broad set of awareness elements, where each element consists of the

Table 1
Elements of WA relating to the present, from Gutwin (1997)

Category	Element	Specific questions
Who	Presence	Is anyone in the workspace?
	Identity	Who is participating? Who is that?
	Authorship	Who is doing that?
What	Action	What are they doing?
	Intention	What goal is that action part of?
	Artifact	What object are they working on?
Where	Location	Where are they working?
	Gaze	Where are they looking?
	View	Where can they see?
	Reach	Where can they reach?

Table 2
Elements of WA relating to the past, from Gutwin (1997)

Category	Element	Specific questions
How	Action history	How did that operation happen?
	Artifact history	How did this Artifact come to be in this state?
When	Event history	When did that event happen?
Who (past)	Presence history	Who was here, and when?
Where (past)	Location history	Where has a person been?
What (past)	Action history	What has a person been doing?

information that people need to track events in the workspace as they occur. Table 1 shows Gutwin's elements of knowledge contained within a "who, what and where" category of questions asked about workspace events in the present.

For each of these categories of questions, there is a unique set of *informational elements* that provides answers to those questions. These informational elements are the specific pieces of information that a person would require in order to keep up with events as they occur in a collaborative real-time workspace.

For example, knowledge of the 'who' category simply means that you know the number of people who are present in the workspace (if any), their identity, as well as being able to attribute a specific person to each action that is taking place (Table 1, first row). In the 'what' category (second row), awareness of action and intention means that you know what a person is doing both at a rudimentary level (e.g., typing some text) and at a more abstract level (e.g., creating a title). Awareness of artifact is knowing what object another person is currently working on. Location, gaze, view and reach are all inter-related in the 'where' category (third row): location refers to the part of the workspace where a person is currently working; gaze is the part of the workspace where a person is currently looking at; view is where they can potentially be looking (i.e., their field of vision), and reach includes the parts of the workspace that this person can potentially change (Gutwin, 1997).

3.2. Workspace awareness for past interactions

Gutwin does mention elements for maintaining awareness of asynchronous changes in his framework, required for people to catch up with events that have already taken place in the workspace (Gutwin, 1997). Table 2 lists this second collection as elements of knowledge contained within the "who, what, where, when and how" categories of questions that may be asked of workspace events in the past.

Determining 'how' the workspace has changed involves two elements: *action history* and artifact *history* (first row, Table 2). Action history describes the unfolding of events that changed the workspace. Artifact history includes details about the process of how an object was changed over time. Information about 'when' something has occurred (second row) is described by the *event history* of the workspace. This element indicates the time at which things occurred in the workspace. 'Who' provides a *presence history* of people in workspace, that is, of knowing who has visited a particular location and when this visit occurred (third row).

Although there are potentially many aspects to the 'where' category of questions e.g., where did events take place, where have artifacts been etc., Gutwin mentions only the *location history* of other people, which indicates where a person has been in the workspace. Finally, the 'what' category of questions lists the *action history* of a person, which describes what actions have another person engaged in (last row).

Gutwin's framework is a good beginning. However, because he was mostly concerned with elements relating to the present, he did not elaborate on past elements beyond this initial list. The remainder of this paper tries to continue where he left off in developing a framework of change awareness.

4. Information elements for change awareness

In this section, we extend and elaborate the elements Gutwin identified for workspace awareness to create a more comprehensive change awareness framework for different-time asynchronous work. In this situation, a person has been away from the workspace for a period of time (hours, days, weeks) and must be brought up-to-date on what has changed in the interim.

When trying to catch up with changes the first piece of information that a person needs to know is "Is anything different since I last looked at the work?" Obviously, a change awareness system must provide the answer to this question in a very lightweight fashion. Afterwards, the person can then probe for further details by trying to find out the specifics of a change. The specific information that a person may require in order to track changes will vary from situation to situation. It will depend upon the task that is being performed, the person who is carrying out the task, as well as the surrounding environment.

In a manner similar to how Gutwin constructed his framework for workspace awareness, we can describe at a high level the questions that may be asked. This set of questions includes:

1. Where have changes been made?
2. Who has made the changes?
3. What changes were made?
4. How were things changed?
5. When did the changes take place?
6. Why were the changes made?

However, this does not suffice by itself. A change awareness framework must account for the fact that people may need to view aspects of the workspace in different ways at different times, i.e., from different perspectives. In particular, a person may query the workspace for changes in terms of:

- the artifacts that exist within it (artifact-based view),
- the people who work within it (person-based view), or
- the workspace may be viewed as one locale or as a collection of related locales (workspace-based view) where a person is interested in the changes and events that have taken place in one or more locales. This relates to Berlage and Sohlenkamp's artifact metaphor.

The specific perspective that a person has of the workspace will have an impact on the information that he or she is interested in and the way that the information is requested and represented. In terms of the artifact-based view, the person will be interested in changes made as they relate to particular workspace artifacts, and will make various queries about those changes. Examples include: how has this item changed, and what has been done to this item? From the person-based view, an individual wishes to know about the changes that were made by another collaborator. Queries about changes will therefore be focused on this person e.g., what did he do while I was away? On the other hand, someone with a workspace-based view would be interested in and inquiring about the events that have taken place in a specific location e.g., what changes and events took place in this space? This location can consist either of the workspace as a whole, or portions of the space that are somehow logically related e.g. a specific room in a room-based groupware system or a particular spatial region.

Of course there is a strong relation between the three workspace perspectives and the six categories of awareness questions. An individual that holds a person-based perspective will focus heavily on the 'who' category of questions. Someone that holds an artifact-based view of the workspace may focus instead on the 'what' category of questions and try to determine what changes were made to specific objects. Yet another person that holds a workspace-based view of the workspace may focus on the 'where' category of questions. Alternatively, the person with a workspace-based view may focus on 'what' was done in the part of project that he or she holds an interest in. The main point is that the person's particular view of the workspace will influence the value that he or she attaches to each category of question. As will be seen, however, the specific example questions that are unique to each of the six high level categories awareness questions can be asked from any of the three workspace perspectives.

The following subsections will describe in detail the informational elements associated with each category of question as well providing some specific example questions that a person might ask about changes.

4.1. Where?

Location in a 2D graphical workspace could be a simple Cartesian spatial region, or a direct digital analogue of physical demarcations, e.g. the rooms in a room-based system such as TeamRooms (Greenberg and Roseman, 2003) or locations may be more abstract in relating workspace entities to each other, e.g., the different logical or conceptual parts of a collaborative project. In all cases, the location of a change provides valuable clues regarding its context, which in turn guides people towards further exploration. For example, a person may ask if a given change was part of the project component that is undergoing extensive rework, that is, if the change occurred is in the same place where many other changes are also occurring.

Table 3 shows the specific questions that may be asked to learn 'where' changes have occurred with respect to each of the three workspace perspectives. As already mentioned, the difference is how queries about changes made to the workspace are formulated within each perspective. With the artifact-based view, the questions could be asked in terms of a specific object in the workspace. Where is it now? Where was it before? Where has it been since I have been away? From the person-based view, the example questions may be asked about a specific collaborator. Where has this person visited or looked in the workspace? Where did this person change things? From the workspace-based view, the questions asked would inquire about the different events that have taken place in the space since a person has been away. Where in the workspace have people visited? Where were the artifacts moved?

The informational elements that will answer the 'where' category of questions include Gutwin's location history (described in Section 3), and the new categories of gaze history and edit history. Location history refers to the parts of the workspace that have been visited by a person. Gaze history includes the parts of the workspace that a person has looked at. The difference between location and gaze history is that while a person may have been present in a general location, he or she may not have actively attended to everything that went on there. Although location and gaze history do not directly provide information about changes that have been made, they do indicate the parts of

Table 3
Where: information elements and workspace questions related to 'where'

Information elements	Specific questions for 'where'		
	Artifact-based view	Person-based view	Workspace view
Location history	Where was this artifact (when I left)?	Where in the workspace has a person visited?	Where have people been in the workspace?
	Where is the artifact now?	Where in the workspace has a person looked at?	Where were artifacts in the workspace?
Gaze history	Where has this artifact been during the time	Where in the workspace has a person made changes?	Which parts of the workspace have people looked at?
Edit history	that I have been away?		Which parts of the workspace have people made changes in?

Table 4
Who: information elements and workspace questions related to 'who'

Information elements	Specific questions for 'who'		
	Artifact-based view	Person-based view	Workspace view
Presence history	Who has looked at this artifact?	Who has this person interacted with?	Who has been in the workspace?
Identity	Who has changed this artifact?		
Readership history		Who made changes with this person?	Who has looked at the workspace?
Authorship history			Who has made changes to the workspace?

the workspace that have been visited or viewed and the frequency of these visits (Hill and Hollan, 1992). This provides strong clues as to where one should look for changes. Edit history, on the other hand, explicitly deals with the changes that were made. Awareness of 'where' edits occurred is vital to routine project management as it provides strong clues as to the progress that has made towards satisfying project-level goals. By this very fact, the location of changes provides strong cues to the answers to other "who, what, why, and how" category of questions.

4.2. Who?

Answers to questions concerning 'who' are important. In collaborative environments, knowing who made a change becomes an opportunity to query that person directly for further information. Also, people may attend to changes differently depending upon who made them. For example, Neuwirth et al. (1992) described how collaborators could be less interested in seeing changes made by co-authors that he or she has known for a long time and more interested in seeing changes made by less trustworthy co-authors.

In Table 4, we provide a more detailed breakdown of the 'who' questions from the different workspace views. To Gutwin's concept of *presence history*, we add *identity*, *readership history* and *authorship history*. While presence history tracks if anyone was present, identity indicates the specific individual associated with a change or event. Establishing identity is needed to provide the context for answering the person-based view of workspace changes. For example, a team member may be responsible for auditing and upgrading different parts of the project. If his or her identity is associated with a particular change, it may better help other team members (who may prefer their original version) accept that change. Readership history carries identity even further by listing all the people who have viewed a particular artifact, or conversely, listing all the items that a particular person has seen. This is important as knowing that someone has viewed the changes without raising any objections or making any further changes suggests an implicit acceptance of the current work. By a similar vein, authorship history can list the people who have made changes to an artifact, or list all the artifacts that a particular person has changed. Tracking readership and authorship history can, for example, be used to gauge progress of the project through a process-oriented lifecycle. In such a case knowing who has seen an object and 'signed off' on it is an important part of workflow management and document routing.

4.3. What?

The 'what' category of questions leads to answers that produce a picture of the *action history* of the workspace (Table 5). Gutwin described two ways that action history can answer these questions (Gutwin, 1997). First, it can be used to track all low level actions that a person has done, e.g., creating, labeling and positioning a circle in a diagram.

Table 5
What: information elements and workspace questions related to 'what'

Information elements	Specific questions for 'what'		
	Artifact-based view	Person-based view	Workspace view
Action history	What changes have been made to the artifact?	What artifacts has a person looked at?	What changes have occurred in the workspace?
		What artifacts has a person changed?	What artifacts were viewed?
		What activities has a person engaged in?	What artifacts were changed?

Table 6
How: information elements and workspace questions related to 'how'

Information elements	Specific questions for 'how'		
	Artifact-based view	Person-based view	Workspace view
Process history Outcome history	How has this artifact changed?	How has a person changed things?	How has the workspace changed?

Knowledge about actions that people have engaged in while one was away is important. When people are asked to describe what is new in the workplace it is frequently in terms of the actions and events that have taken place. Sometimes there is, of course, a need to put all of these lower level actions in the context of the higher goals. So Gutwin also described action history from a higher-level perspective of the low-level changes in a way that considers the goals that motivated these actions, e.g., the labeled circle was created in order to represent a new person in an organizational chart.

Yet the low-level questions and answers presented in Table 5 are often the only information that developers can hope to capture when they add change awareness support to an application. The problem is that it is difficult to ascertain and store the motives behind a series of low level changes. One could use spatial proximity (i.e., changes located near to each other) or temporal proximity (i.e. changes that occur at the same time) as predictors of inter-relatedness, but often these methods will fail because the sub-steps for achieving several different high level goals may often be interleaved. Thus, we will postpone discussing the higher-order view of changes until Section 4.6, and instead focus here on only the significance of low-level changes. It is important to point out, though, that people are able to relate and combine several low-level actions to derive a higher-level action if they are given enough contextual information to understand the relationships between these lower-lever actions.

The specific questions associated with the 'what' category varies depending upon the perspective that a person has of the workspace. These questions are shown in Table 5. For the artifact-based view, inquires are made about the changes that have been made to a particular artifact. From the person-based view, the questions ask about what actions has a person undertaken. With the workspace-based view, the questions ask about the actions that were undertaken within the workspace or actions that were carried out on the artifacts in the workspace.

4.4. How?

The 'how' category asks how the current workspace differs from the way that it was before (Table 6). The answers to these questions can be integrated to derive one of two historical views of changes. The first is in terms of the *process history* of the workspace, which indicates incrementally how the workspace evolved from its previous state (i.e., the state that it was in when one last looked at it) to its present state. This is useful when a person is interested in the mechanical means (the intermediary steps) that produced a change or group of changes as well as the end result. Thus process history is tightly coupled with action history. The difference is that the action history consists of all the actions that have occurred while one was away, while process history relates and abstracts a subset of all actions into a series of steps in a process. The process view is important for, as Gutwin described, people may have trouble interpreting instantaneous changes (Gutwin, 1997). Thus, describing all the sub-steps involved in a change may help to clarify what happened. Also, the process-oriented view describes the *evolutionary context* of changes (i.e., the specific details of the circumstances faced by the person who made the change at the time that the change occurred), and thus can provide valuable insight on 'how' and 'why' things came to be in their present state. Of course, a person may only be interested in the final result. This is the second historical view of 'how' a workspace has changed, i.e., the *outcome history*. The outcome history presents only a 'bottom line' understanding of a change where it highlights only those things that differ between the initial and the final state.

The choice of process vs. outcome history will depend largely upon the task at hand. For example, a graphic artist

Table 7
When: information elements and workspace questions related to 'when'

Information elements	Specific questions for 'when'		
	Artifact-based view	Person-based view	Workspace view
Event history	When was this artifact changed? When was a particular change to this artifact made? In what order were changes made to this artifact?	When did a person make changes? When did a person make a particular change? In what order did this person make changes?	When were changes made to the workspace? When did a particular change in the workspace occur? In what order did changes to the workspace occur?

Table 8
Why: information elements and workspace questions related to 'why'

Information elements	Specific questions for 'why'		
	Artifact-based view	Person-based view	Workspace view
Cognitive history Motivational history	Why was this artifact changed?	Why did a person make that change?	Why was that change made in the workspace?

may be interested in the technique used to produce some visual effect. In this case, this person will want to see (and thus learn) the process history of the workspace. On the other hand, a newspaper editor reviewing an article submitted by a reporter is far too busy to be concerned with the rough drafts produced by this person, and would thus be interested only in the outcome history of the article. Consequently, it is important that software support for change awareness provide the ability to discover both the process and outcome history of a workspace.

4.5. When?

The timing and ordinality (sequential order) of changes is revealed by the answers to the questions of 'when' changes occurred as listed in Table 7. The time when a change occurred, particularly if it overrides an earlier change by another person, is often of great significance and affects the perceived importance of a change. For example, a person may only be interested in recent workspace events, or a person may only be interested in changes that occurred within a specific period of time.

The timing and ordinality of changes constitute the *event history* of the workspace, and it provides the *chronological context* for understanding and interpreting changes giving clues to the 'where', 'who', 'what', 'how' and 'why' categories of questions.

4.6. Why?

Knowing the thought and motives behind a change can be important for accepting the changes that others have made. The questions that a person will ask to discover 'why' changes were made are summarized in Table 8. A historical view of 'why' changes were made includes both the *cognitive history* and the *motivational history*. Cognitive history describes the logic or reasoning that may be behind a change, which is a rational reconstruction of the person's goals and plans. Motivational history deals more with the impulses or desires that are the impetus for making a change, which is the actual reason why a person did something at a moment in time. The reason that they are separate elements is because a change may be based upon a well thought out and carefully conceived plan or it may be more of a spur of the moment thing as one reacts to the current situation. Also, some changes are completely unintended accidents.

Although it is not always needed, knowing 'why' a change was made is obviously an important step for coming to understand and accept it. For lower-level changes that are the parts of a grander higher-level change, the motivating factors may be painfully obvious. In this way providing the motivational history for simple changes may be too effortful (and distracting from the main task) to explain. Also, describing all the motives behind a change and detailing all the reasoning behind each event is extremely difficult for computers to do automatically. This is because understanding the 'why' often draws upon a person's accumulated technical expertise and implicit or 'hidden' cultural information relating to group priorities, work practices, and short and long term goals. Today's computing systems lack the ability to sense these technical and cultural factors that motivated a change. They also lack the intelligence needed to produce a truly comprehensive picture of the cognitive history of the workspace. Consequently, most 'why' information will likely be generated explicitly by authors, e.g., as annotations added to changes or as design rationales.

4.7. Discussion

We just described in detail the information that can be used by a person to track changes made by others over time in a collaborative project. The informational elements were classified according to several categories of change awareness questions. These categories are inter-related and inter-dependent. When a person is tracking changes he or she may start with the highest-level question, 'Has anything changed?' From that point the person will make inquiries about changes from one or more of particular perspectives—artifact, person, or workspace-based—that make the most sense to him or her and the work context. The inquiries can be directed towards a specific collaborator, 'What has this person done?' Alternatively, the process of inquiry can take the form of an examination of a particular artifact, 'How does this object differ from before?' or it can take the form of an inspection of a select portion of the workspace, e.g., 'What has happened in this corner of the project?'

Within the bounds of the chosen perspective, a person can ask specific questions from these categories to probe for further details of changes. The specific category that a person begins with (where, who, what, how, when or why) is not fixed. As mentioned previously, it will be influenced by the workspace perspective that is held e.g., if a person is making inquiries from a person-based view then the 'who' category may be of the most pressing urgency. Also, as we have shown in the previous sections, queries made in one category of question are not isolated from the other categories. The process of inquiry can occur in parallel as someone delves for answers in more than one category at once. For example, take the case of a project manager who is reluctant to have certain parts of the project undergo anymore changes, or who has severe misgivings about the work of specific team members. When the manager returns to the project after a period of absence and discovers that many changes have been made, he may immediately try to determine exactly where changes were made and specifically who made those changes.

The answers to the questions from one category may also inspire additional inquiries in another category. For example, a person who is tracking the historical context of the changes to a software system may start by asking about 'when' most of the changes occurred. Upon discovering this information she notes that the code was most volatile during a port between operating systems. Since she knows that there is a methodological way to do this, her queries then focus on the process history of the software as she tries to determine exactly what the programmers did during the port.

Furthermore, the answers to the questions that a person asks about one category of question may directly provide him with further information. For example, when a person knows exactly where changes occurred, she then knows who made those changes (because she knows who is responsible for which portions of the project). Or the person may be able to make predictions about the answers to the other categories of questions based upon the information that she gets from one category. When a person learns that a specific team member made a change, she can guess as to how the change was made. These guesses are based upon her personal knowledge of the person who made the changes and the techniques that he has employed in the past.

Although a single answer may provide information about multiple categories of questions, the main point of the framework is to ensure that designers of change awareness systems actually consider what change information should be captured if the system is to provide its end-users who are tracking changes with the information they need to answer these questions. At the very least, the designer can use this framework to prioritize what change information is needed, and to focus on those with the highest priority. The framework ensures that the designer will not neglect certain categories because he or she is unaware of the need to track it.

The framework also allows a designer to evaluate and critique existing systems for their offerings of change awareness information. We do this in the next section, where we apply the framework to understand the limitations of a change awareness tool that we built prior to the framework's development.

5. Applying the framework in the critique of a change awareness tool

Our interest in a change awareness framework emerged out of our design of PastDraw, a 2D structured drawing application augmented with various techniques for capturing and displaying change awareness information (Fig. 2). After completing an alpha version of this application, we realized that there was something not quite right about how we were portraying change awareness within it. Yet, we could not precisely articulate what was wrong. Consequently, we abandoned the implementation in order to develop the framework described previously.

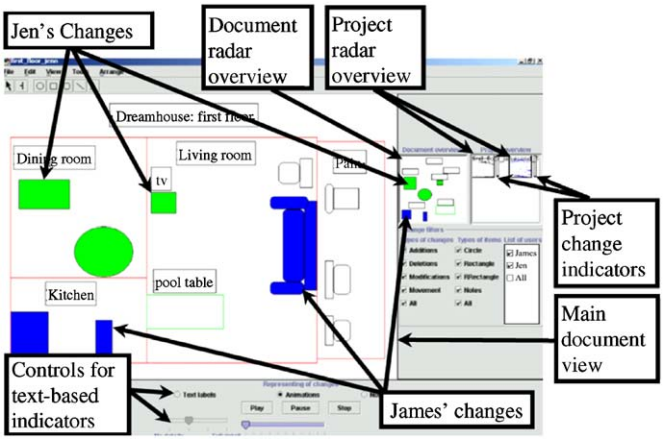

Fig. 2. Overview of the main features of PastDraw.

In this section we return to PastDraw. Our goal is to demonstrate how we can use the change awareness framework to appraise change awareness tools in a fashion that is similar to how Nielsen's heuristics (Nielsen, 1993) can be applied in usability evaluations.

5.1. Introduction to the base system and a usage scenario for PastDraw

PastDraw is built atop of the *GEF* Graph Editing Framework (Tigris, 1995), an open source Java library (Sun Microsystems Inc., 1996) that enables the construction of many different types of 2D graphical applications, i.e., basic structured drawing tools to more specialized tools such as the UML editor Argo/UML (Tigris, 2000).

Within PastDraw, users have a conventional drawing surface (called the 'main view', center of Fig. 2). The system also provides two overviews. First, a high level project overview shows all diagrams (top right corner) as thumbnail images. Second, a real-time radar overview (Gutwin, 1997) shows a miniature of the entire drawing document, a portion of which is currently displayed in the main view. Changes made from the present back to a predetermined point in time are displayed in all these views, as detailed below. In the main view changes are shown either in the form of labels and graphical cues or as animated replays both of which can be filtered. Details are described below.

Changes in the overviews: Within the project and document radar overviews, changed objects are colored to indicate which PastDraw user made the changes. In Fig. 2, for example, objects in the document radar overview that were changed by James are colored blue (or black if the image is in black and white) while those by Jen changed are green (or medium gray). Change indicators, visualized as change bars to the right of these overviews, indicate the amount of changes that have been made. Together, the project overview and the document overview provide the current user with a rough idea of where changes occurred, what objects were changed, as well as who made the changes. Text labels (described below) also appear in the overview, but because of their size reduction they are not readable; thus they are best considered as visual markers.

Changes in the main view: The main view (center of Fig. 2, with further details in Fig. 3) uses color to indicate changed objects, as well as text labels and other graphical cues to accentuate and detail the differences between the initial and final view of an object. Through ghosting and outlining, the main view includes additional graphical cues about 'how' certain types of labeled objects changed. For example, in Fig. 3, we see that the TV object (top center) was added (ADD) since we last looked. By tracing back from the arrow, we see it was originally created it in the bottom right corner and resized (the two ghosted outlines and the MOD indicator) and finally moved (MOV) to its final destination.

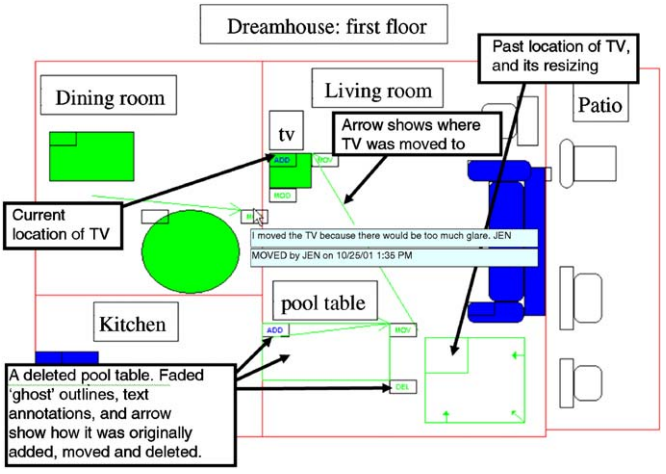

Fig. 3. Detail of the main view, showing color, text labels, and graphical cues.

Animating the main view and the document overview: In contrast to the static images seen in Figs. 2 and 3, the animation controls (bottom middle of Fig. 2) lets the viewer literally replay all changes that happened over a given time period. These changes appear both in the main view, and in a visually reduced form in the document overview.

Viewing personal and system-created annotations about changes to objects: People can explicitly add comments as annotations to objects. These are then displayed as the viewer mouses over the object, e.g., as in the upper note by JEN in the center of Fig. 3. By default, each object also contains a note that indicates what action was performed on it, who did that action, and when that action occurred e.g., as in the lower note.

Filtering the main view: To give the viewer some control over what changes they want to see as well as the level of detail they desire, PastDraw provides several filters to hide or show these details in the various views. First, by manipulating the checkbox controls (mid-right of Fig. 2), the viewer can filter the types of changes that should be displayed (i.e., additions, deletions, modifications…), and/or the types of drawing items that should show changes (i.e., circles, rectangles, notes, …), and/or by whose changes they want to see (i.e., one, some, or all past editors of the document). Second, the viewer can filter how much detail is shown by the text labels and their accompanying graphical cues through the slider control (Fig. 2, bottom left). This is illustrated in Fig. 4, where we see the effect that the different slider levels have on the display of changes for the TV object. "No details" turns off the display of all change information (Fig. 4a). The 2nd level provides only color cues for changes (4b). The 3rd level adds descriptive text labels and outlining (4c), while the "Full detail" of the 4th level augments the text labels with arrows used for spatial moves.

Accepting changes: Finally, the viewer has the ability to 'accept' agreed-upon changes so that they are no longer tracked or displayed.

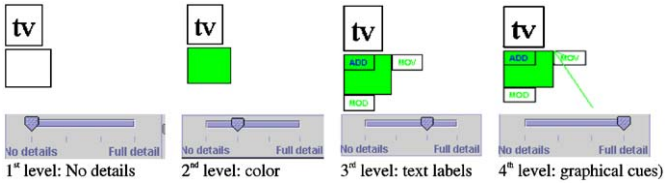

Fig. 4. Change details added at four levels of detail for the TV (the bottom box).

5.2. Analysing PastDraw's change tracking according to the framework

Using the informational elements from Section 4, we reviewed and categorized whether or not PastDraw tracks changes for a particular element, and if it does whether this tracking is complete or partial. Table 9 summarizes our efforts. We see at a glance that PastDraw is quite incomplete in how it tracks change history, where at best it tracks only partial information for most elements. While we may later decide that there is no need to track all these informational elements completely, the point is that the framework has helped us articulate the actual coverage of PastDraw.

In the remainder of this section, we briefly discuss our analysis of several examples of PastDraw's change tracking to illustrate that the process of applying the framework to critique a system is fairly straightforward.

First, we examined which questions in the various tables are typically answered by the dominant PastDraw change awareness mechanisms. From this set, we then scanned the tables to see if these particular questions were biased toward the artifact vs. person vs. workspace-based view, and what type of information element was favored the most. By doing so, it was immediately apparent that PastDraw's design is heavily biased towards an artifact-based view, where it predominantly showed action and edit history of changes of artifacts. That is, the system is designed mostly to display what changes have been made to individual artifacts.

The next step is to go through these informational elements in turn. We began with the action and edit history, as it has already been identified in the previous step as the dominant information provided by PastDraw's change awareness mechanism. As described in Section 4.3, the action history of an artifact includes all of the low-level edits in the workspace (what has changed in that artifact), while the edit history says where these changes occurred in the workspace. This information is likely of paramount importance as it is perhaps the most rudimentary information that a person will use when catching up on changes. We see that PastDraw captures the action and edit history by tracking the addition and deletion of primitive objects, as well as movement and resizing modifications. Even so, these histories are incomplete, for changes to text are not tracked (which is why it is marked as only partially supported in Table 9). This could be a serious omission if knowing about this type of change is important to the viewer.

Table 9
Summary of PastDraw's change tracking

Category of questions	Informational elements	Does PastDraw track the element?
Where	Gaze history	~
	Location history	~
	Edit history	✓
Who	Presence history	~
	Identity	~
	Authorship history	~
	Readership history	~
What	Action history	~
How	Process history	~
	Outcome history	~
When	Event history	✓
Why	Cognitive history	~
	Motivational history	×

✓—PastDraw does track changes for this informational element.
~—PastDraw indirectly or only partially tracks changes for this informational element.
×—PastDraw does not track changes for this informational element.

Because informational elements are inter-related, people can likely infer partial information about other elements from the action and edit history—presence, location, readership and gaze history (rows 5, 3, 8 and 2, respectively, in Table 9). If we know that changes have been made we can infer that someone has loaded up PastDraw (i.e., someone was present in the workspace) as well as being able to infer some information about where the person was located (the changed documents) and what was 'read' or 'gazed' upon (the changed objects and the approximate area around them).

However, PastDraw provides this information only if changes have been made. If no changes have been made, as happens when people read and review documents without editing them, then the viewer has no information about presence, location, readership and gaze history of others. Those others could have simply loaded up PastDraw and looked around without editing anything. Yet this information could be quite useful for the viewer. For example, knowing that others have been present and have looked at particular regions of the document (gaze and location history) may be valuable in cases where (say) one is waiting for approval by the other of particular changes that have been made, or where one is waiting for the other to get 'up to date' as a prelude to talking about the drawing. Or perhaps knowing that the second person has already looked and not expressed any objections or made any further changes may imply implicit acceptance of the current version of the document. From this, we can argue that PastDraw should at least provide details about presence and location history even when no changes have

been made. This level of knowledge should be sufficient. Except in special circumstances, it is not the individual elements in a diagram that count but it is their combination (in the form of location history) that matters. Thus, the lack of direct information about gaze and readership history is not of grievous concern.

The animation feature in PastDraw supports the process history—the sequence of changes in which the workspace evolved from its past state to its current state. Yet this is largely limited to an artifact point of view. It is easy to see how particular artifacts have changed over time by replaying their action history, but it is harder to see it from other points of view. As well, the animation playback is not supported in the project overview, which only shows a static change diagram (Fig. 2, top right). This means that it is hard to tell what other parts of the project co-evolved (or changed) as this particular document was changed.

Similarly, the views provided by both the overviews and the main views give a partial outcome history. By showing changes made in the 'after' state of the workspace, one can infer the 'before' state and how the new stated differs from it. However, this can be hard to do. A quick glance at Fig. 3 suggests that it would be difficult to tease out the before image from an after image cluttered with change marks. Yet this outcome history could be valuable if one is to put all the changes made into perspective. We should also mention that the outcome history as presented is predominantly artifact-oriented, although a person-oriented view is possible by filtering the view by person.

PastDraw does track authorship history, and this knowledge may be useful for reconstructing what a person has done, or simply to find out who the viewer should talk to for further information. However, the only authorship information tracked by PastDraw is the editor's name. Ideally authorship information should include additional details, such as the email address and phone number, to facilitate getting in touch with him or her.

Event history (12th row) is tracked in PastDraw both in terms of the exact timing of events (date and time) as well as the order in which they have occurred. This timing is used internally to drive the animation. Although it is more likely that a person would be interested in having only a rough indication of when a change occurred rather than the exact point in time in and of itself, storing this level of precision is not problematic. What is relevant is that PastDraw only displays event timing as text (i.e., a pop-up raised by mousing over any object, as in Fig. 2) which is hard to use usefully in practice. Alternate strategies could, for example, show event changes by aging objects through visual cues rather than by providing individual text timings.

PastDraw does not automatically track cognitive history, although it does allow users to manually enter their reasons for the changes that they made. Because these entries are attached to artifacts vs. the workspace or person level, and because they are only seen by mousing over object (as in Fig. 2), it will likely be difficult for a viewer to get a good sense of the high-level motivations behind a large set of changes created by a person.

We could go on, but these examples should be sufficient to show how we can use the framework from Section 4 to determine what change information is and is not tracked and displayed by PastDraw. In some cases, the absence of an informational element may have little consequence within our context of a simple graphical editor. In other cases, this absence can seriously impair a person's ability to track changes at the right level.

5.3. Analysing PastDraw's display of changes and suggested improvements

The framework can also be used to analyse what informational elements are supported by particular display mechanisms, and from this we can infer whether the display mechanism would be effective. As Figs. 2 and 3 showed, PastDraw has three main mechanisms for displaying changes: animations, short text labels combined with graphical cues, and the overviews. All of these mechanisms are augmented by applying color to changed objects (in an early pilot, test participants liked the quick overview of changes that color provided (Tam et al., 2000; Tam, 2002)). Yet we will see that none of PastDraw's display mechanisms fully display the informational elements, as summarized in Table 10. In the remainder of this section we will discuss a subset of PastDraw's display mechanisms to illustrate how this analysis can be performed.

As seen in the 'animations' row of Table 10, the animations employed in PastDraw focus primarily on the edits made to diagrams: what these edits were (action history), where they occurred (edit history) and the process of change that the workspace underwent (process history). The order of changes can be determined by the order of the animations (partial event history). The main draw back of the animated replays is that the viewer has to watch the full sequence of changes in the form of process history, with no ability to playback only the changes that took place during a particular period of time. Similarly, inconsequential edits are captured. That it, replaying the entire event history at full fidelity becomes too much of a good thing. Also, no timing information for 'when' is provided during the animations, which would have also been useful because then the viewer of the changes could have at least noted the point at which the animations started showing the changes made. While it was possible for a viewer of PastDraw to filter out his own changes or the changes of others, PastDraw does not indicate the person who was responsible for each change as the replay is occurring. Finally, no information is provided about 'why' changes were made. This deficiency suggests some enhancements. Perhaps animations could include a high-speed 'skimming' or compression mechanism that allows the user to quickly review overall details of changes and then view at normal speed the details of interesting changes. Or animations

Table 10
Summarizing PastDraw's display of informational elements

Display mechanism	Categories of questions							
	Where	Who		What	How		When	Why
	Edit history	Identity	Author-ship history	Action history	Process history	Outcome history	Event history	Cognitive history
Animations	~	×	×	~	~	×	~	×
Project overview	✓	~	~	×	×	×	×	×
Document overview (animations)	✓	×	×	~	~	×	~	×
Document overview (text labels)	✓	~	~	✓	×	×	×	×
Personal annotations	×	~	~	✓	×	×	~	~
Color	✓	~	~	×	×	×	×	×
Text labels	✓	×	×	✓	×	×	×	×
Graphical cues	~	×	×	~	×	~	×	×

✓—the display mechanism does display changes for this informational element.
~—the display mechanism only partially displays changes for this informational element.
×—the display mechanism does not display changes for this informational element.

could be supplemented to display 'why' changes were made by showing text documentation during the animations.

Text labels within the document works well for displaying abstract information (such as 'why') or precise information (such as 'when'), although within PastDraw they do not show 'how', 'when' and 'why'. Even if they did, it is difficult to represent some types of changes in text (e.g., spatial movements or the physical resizing of objects). Thus, text can, in practice, only do a partial job of revealing many of the changes made. This suggests a design tradeoff: use text to represent abstract information that cannot otherwise be easily understood, but use other mechanisms to represent changes that are inappropriate for a text display. A further problem is that the current text labels are too artifact centric: they only display what primitive actions occurred and where they occurred. Yet individual changes are usually one of a larger set comprising a complex change, the brief text descriptions may be inadequate in explaining what is really going on (e.g., cognitive history, process history, and outcome history at the person or workspace perspective).

The document overview displays the same animated change information as the main view (when the animation control is used) as well as the text labels, only in a significantly miniaturized form. The advantage to having the document overview is that while parts of the document may be obscured in the main view, the entire document is shown in the document overview, thus reducing the possibility that important changes will be missed. Of course, these changes are visually quite small. While the animation and text information is there, it would be hard for the viewer to decipher its meaning. In essence, the viewer will be able to tell that something has changed, but would have a hard time determining exactly what has changed. The document overview also excludes many information categories: 'how' changes occurred, 'when' they were made and 'why' they occurred. Yet we have to be careful not to overload this simple overview, as this view is only meant to provide a rough idea of the changes that occurred to a diagram. Viewers can look to the main view to see the additional details. Still, we can now ask ourselves if the appropriate coarse-level information is provided.

We can see in the Row 7 that color not only indicates where changes occurred but also partial information on who made changes. Objects in PastDraw are filled in with the color of the last person who changed it. For the text labels, the text is also filled with the color of the last person who made that particular change. However, this use of color is limited. It does not suggest 'how', 'when' and 'why' changes were made. As an alternative, we could perhaps augment PastDraw to assign colors to more than just identity, i.e., by allowing a person to map color to some of the other categories of change awareness questions. As well, color fading could have been exploited to show change aging, which in turns gives an approximate event history.

As seen in the table, we found that a great deal of change information was not represented effectively with the PastDraw change awareness mechanism. At the end of this exercise, we realized that while PastDraw used a 'grab-bag' of change awareness techniques, it did not present the viewer with a comprehensive understanding of changes.

5.4. Discussion of the critique

In the beginning of Section 5 we described the difficulties that we encountered when trying to develop PastDraw. We focused heavily on implementation details because at the time we did not have a clear understanding of what were the important concepts for change awareness. When we retrospectively applied the principles described in the framework to the design of PastDraw, we quickly realized that the system was woefully inadequate. While these problems could appear obvious after the fact, as seen in

much of our discussion in this chapter, we fell into quite a few pitfalls without it.

As a reminder, this paper is not about PastDraw. Rather, PastDraw was used as an example to show the utility of the framework. We saw how we can articulate what informational elements are supported by a system, and how the framework can help one understand the strengths and weaknesses of a change awareness implementation. What we did not show (but which should be obvious) is that the framework can also be used to inform design. In the case of PastDraw, determining what information that was present or absent for different mechanisms could help us see how augmenting several mechanisms and/or combining the strengths of particular mechanisms could offset their individual weaknesses.

6. Conclusions

In this paper, we have contributed a theoretical framework for asynchronous change awareness in collaborative documents and workspaces that can be used in several ways. First, designers can use it as a high-level guide for determining what change information should be tracked and displayed to participants, and what perspectives of viewing this information are relevant to the end user. This cannot be done blindly, for it requires deep knowledge of the end users, their tasks, and the context of their work. We suspect that this analysis would become part of the questions asked and analysis made during a requirements elicitation exercise, e.g., Contextual Inquiry (Holtzblatt and Jones, 1993) and/or Task Centered System Design (Greenberg, in press; Lewis and Reiman). Alternatively, it could be used to seed the questions asked in focus groups. Second, evaluators can apply the framework after the system has been built in a fashion similar to Nielsen's Heuristics (Nielsen, 1993), where the framework is used to critique the design. As with Nielsen, we suspect better analyses will include at least three to five evaluators: system designers, groupware experts and experts in the work domain (i.e., double experts Nielsen, 1993), and so on. Third, and perhaps most importantly, the framework will sensitize designers about the need and the importance of change awareness. Currently, if change awareness is included at all, it is usually done as an afterthought.

However, there is still much left to be done, and these are directions for future research.

First, we must recognize that this framework is an initial version, and is likely incomplete. For example, there is no mention in the framework of how conflicting changes by different people would be handled. As well, there is no discussion of what happens when people hand off responsibility (or ownership) of changes to another person. Neither is there any discussion of the differences between the perceived relevance of a change between (say) the author and one or more of its reviewers.

Second, the theoretical concepts in the framework do not dictate how they should be assigned into a system as particular interface features. As PastDraw has shown, just making the change information available in the interface may not suffice to make it clearly legible to the end user. While there are many ways to implement these theoretical concepts as interface components, it is likely that the interface will have to undergo a significant number of design iterations before they prove truly usable in practice. For example, simply making all change information visible will likely overwhelm the end user; yet it is unclear if filtering is the best option to reduce complexity. That is, while the theoretical framework can instigate research into change awareness methods, the hard work of interface design is still required.

Third, while the framework meets common sense analysis, it is not formally validated. Future work would validate the importance of the framework attributes against particular user and task needs. This could perhaps be done by building a prototype or system that follows the framework (perhaps using the interface components from the above step), and performing a usage analysis of its change awareness features through extensive observation and testing. Of course, validation should be done by several groups so that the differences can be collected and analysed.

Fourth, we have to analyse the need for change awareness in particular application domains. By understanding these differences, custom versions of the framework tailored to that domain may be produced.

Acknowledgements

Special thanks to the participation of Frank Maurer in this project, as well as the useful comments by members of the University of Calgary Grouplab team. Partial funding was provided by Canada's National Science and Engineering Research Council.

References

Berlage, T., Sohlenkamp, M., 1997. Visualizing common artifacts to support awareness in computer mediated cooperation. CSCW 8 (3), 207–238.

Brown, H.B., 1970. The clear/caster system. In: Proceedings of the NATO Conference of Software Engineering.

Eick, S., Steffen, J., Sumner, E., 1992. Seesoft—a tool for visualizing line oriented software statistics. In: Card, S., MacKinlay, J., Shneiderman, B. (Eds.), Readings in Information Visualization. Morgan Kaufmann Publishers Inc., Los Altos, pp. 419–430.

Greenberg, S., 2004. Working through task-centered system design. In: Diaper, D., Stanton, N. (Eds.), The Handbook of Task Analysis for Human–Computer Interaction. Lawrence Erlbaum Associates, London, pp. 49–66.

Greenberg, S., Roseman, M., 2003. Using a room metaphor to ease transitions in groupware. In: Ackerman, M., Pipek, V., Wulf, V. (Eds.), Sharing Expertise: Beyond Knowledge Management. MIT Press, Cambridge, MA, pp. 203–256 (January).

Gutwin, C., 1997. Workspace awareness in real-time groupware environments. Ph.D. Thesis, Department of Computer Science, University of Calgary, Calgary, Canada.

Hill, W.C., Hollan, J.D., 1992. Edit wear and read wear. In: Proceedings of the ACM CHI'92, pp. 3–9.

Holtzblatt, K., Jones, S., 1993. Contextual inquiry: a participatory technique for system design. In: Schuler, D., Namioka, A. (Eds.), Participatory Design: Principles and Practice. Lawrence Earlbaum, Hillsdale, NJ, pp. 180–193 (April).

Hunt, J.W., McIlroy, M.D., 1975. An algorithm for differential file comparison. Computing Science Technical Report No. 41, Bell Laboratories.

IBM Corporation: Rational Rose. http://www.306.ibm.com/software/rational/

Kurlander, D., 1993. Graphical editing by example. In: Proceedings of the ACM CHI'93.

Lewis, C., Reiman, J., Task Centered User Interface Design: A Practical Introduction. University of Colorado, Boulder. This is shareware book that is available online at the following url: ftp://ftp.cs.colorado.edu/pub/cs/distribs/clewis/HCI-Design-Book/

Magnusson, B., Asklund, U., 1996. Fine grained version control of configurations in COOP/Orm. In: Proceedings of the Symposium on Configuration Management, SCM6, Berlin, Germany.

Microsoft Inc., 2003. Microsoft Word, as included in Microsoft Office Professional.

Molli, P., Skaf-Molli, H., Bouthier, C., 2001. State Treemap: An Awareness Widget for Multi-Synchronous Groupware. Seventh International Workshop on Groupware—CRIWG'2001.

Morán, A.L., Favela, J., Martínez-Enríquez, A.M., Decouchant, D., 2002. Before Getting There: Potential and Actual Collaboration Proceedings. Springer, CRIWG.

Neuwirth, C.M., Chandhok, R., Kaufer, D.S., Erion, P., Morris, J., Miller, D., 1992. Flexible differing in a collaborative writing system. In: Proceedings of the ACM CSCW '92, pp. 147–154.

Nielsen, J., 1993. Usability Engineering. Academic Press, San Diego, CA.

Nunamaker, J., Briggs, R.O., Mittleman, D.D., Vogel, D.R., Balthazard, P.A., 1997. Lessons from a dozen years of group support systems research: a discussion of lab and field findings. Journal of Management Information Systems 13 (3), 163.

Rochkind, M.J., 1975. The source code control system. IEEE Transactions of Software Engineering 1 (4), 364–370.

Steves, M.P., Morse, E., Gutwin, C., Greenberg, S., 2001. A comparison of usage evaluation and inspection methods for assessing groupware usability. In: Proceedings of the ACM Group '01.

Sun Microsystems Inc., 1996. *Java*. http://java.sun.com/j2se/1.3/

Tam, J., 2002. Supporting change awareness in visual workspaces. M.Sc. Thesis, Department of Computer Science, University of Calgary, Alberta, February, unpublished.

Tam, J., McCaffrey, L., Maurer, F., Greenberg, S., 2000. Change awareness in software engineering using two dimensional graphical design and development tools. Technical Report 2000-670-22, Department of Computer Science, University of Calgary (Calgary, Canada), http://www.cpsc.ucalgary.ca/grouplab/papers

Tichy, W.F., 1991. RCS—a system for version control. Software—Practice and Experience 15 (7), 637–654.

Tigris, 1995. GEF (the Graph Editing Framework). http://gef.tigris.org/index.html

Tigris, 2000. Argo/UML. http://argouml.tigris.org/