ANCHORED CONVERSATIONS: CHATTING IN THE CONTEXT OF A DOCUMENT

Elizabeth F. Churchill, Jonathan Trevor, Sara Bly, Les Nelson, Davor Cubranic[†]

FX Palo Alto Laboratory Inc.
3400 Hillview Avenue
Palo Alto, CA 94304, USA
{churchill, nelson, trevor}@pal.xerox.com

Sara Bly Consulting
24511 NW Moreland Road
North Plains, OR 97133, USA
sara_bly@acm.org

ABSTRACT

This paper describes an application-independent tool called Anchored Conversations that brings together text-based conversations and documents. The design of Anchored Conversations is based on our observations of the use of documents and text chats in collaborative settings. We observed that chat spaces support work conversations, but they do not allow the close integration of conversations with work documents that can be seen when people are working together face-to-face. Anchored Conversations directly addresses this problem by allowing text chats to be anchored into documents. Anchored Conversations also facilitates document sharing; accepting an invitation to an anchored conversation results in the document being automatically uploaded. In addition, Anchored Conversations provides support for review, catch-up and asynchronous communications through a database. In this paper we describe motivating fieldwork, the design of Anchored Conversations, a scenario of use, and some preliminary results from a user study.

Keywords

Text-based chat, sticky chats, collaboration, conversations, CSCW, shared documents, synchronous communication, asynchronous communication

INTRODUCTION

The past few years have seen the development of a number of computationally lightweight text-chat systems that support synchronous and asynchronous communications between individuals and groups. Examples include Internet Relay Chat (IRC) and AOL Instant Messenger [1]. Once chat software of this kind is installed, initiating contact with others is easy, taking only a few keystrokes. By creating ‘buddy lists’ of regular contacts, starting a chat is even simpler – selecting a user-name initiates contact. Therefore, ongoing and frequent informal interactions are easily supported. It is also possible to have multiple simultaneous conversations with different individuals and groups by opening more than one chat window [3,5].

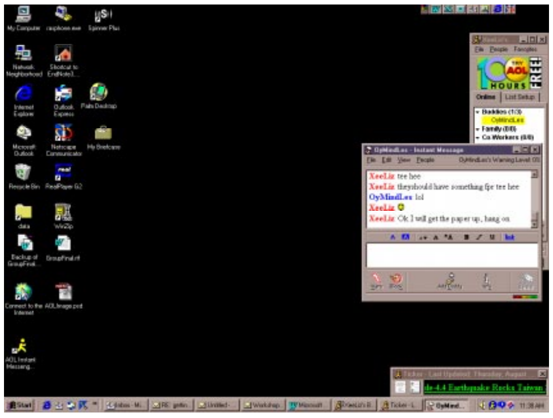
In many chat spaces, conversations are ephemeral, lasting only while the current conversation or session is alive. However, some applications and systems allow text-based conversations to be recorded so that they have persistence. Such text logs enable asynchronous messaging between collaborators who are not online at the same time, and can also be used for catch-up and review [2, 3, 5, 6, 16]. Much recent work has also focused on providing more social activity cues in the interfaces to such chat environments [6, 16].

The success of these chat tools is not entirely surprising – people like to talk and maintain contact with others and chat systems support a form of quick, lightweight, informal communication. These are not just important for recreational activities - in the workplace, informal quick conversations and exchanges are important for maintaining the social fabric as well as for specific information exchange. Undoubtedly, the success of messenger applications and chat spaces is also due to their being computationally and cognitively lightweight: they take little time and effort to set up, little processing power to run, and little maintenance to keep going. Yet they support rich, informal, ongoing contact whenever users require. Most people don’t even have to stray from their offices – everything happens on their desktops.

Problems With Chats At Work

Although chat tools are proving very useful in the workplace for maintaining contacts and exchanging information, they do not support the close integration of documents with ongoing conversations. Many work collaborations involve the sharing of documents: Web pages, presentations, spreadsheets and word processed documents. However, text conversations tend to occur in windows that are separate from, and have no relationship to, work-related resources. Figure 1 illustrates this point. In

[†]Current address: Dept. of Computer Science, University of British Columbia, 20-2366 Main Mall, Vancouver, BC, Canada. Email: cubranic@cs.ubc.ca


XeeLiz tee hee
XeeLiz theyshould have something fpr tee hee
OyMindLex lol
XeeLiz
XeeLiz Ok I will get the paper up, hang on

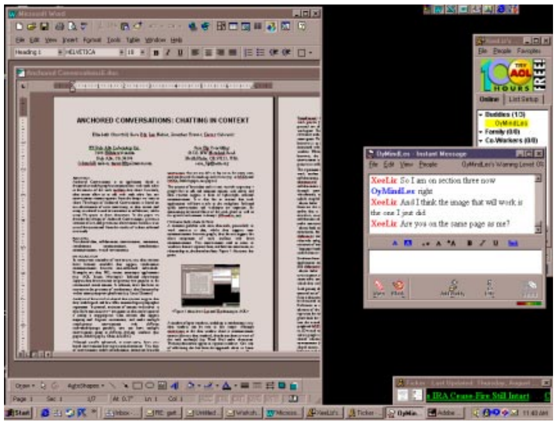

XeeLiz So I am on section three now
OyMindLex right
XeeLiz And I think the image that will work is the one I just did
XeeLiz Are you on the same page as me?

Figure 1. Chatting in messenger windows.
On the left a text chat is ongoing; on the right the word processor document that is being discussed has been opened on the desktop. Conversations and documents have no relationship

the first image a chat window is open on the desktop. In the second image, the chat window is shown alongside a word processor document that constitutes the subject of the conversation.

There are several problems with this arrangement. First sharing or copying the document that is to be discussed requires that users carry out extra steps using other applications (e.g. ftp the file, email an attachment, browse a shared repository, set up a shared window, etc). Secondly, when documents are sent or retrieved, issues arise around ensuring that all collaborators have the *same* document. Thirdly, once all collaborators have copies of the same document, establishing shared reference by navigating to the same portion of the document can be time consuming. Discussions about the content of the document require that navigation statements be typed into the chat window ("Look at section three, paragraph 2") or that the relevant materials from the document be copied and pasted into the text-chat window. In the case of shared windows, navigation cues in the form of shared pointers may be available, but shared windows tend to result in restrictions on what users can do with the content of the file itself and do not support asynchronous communications.

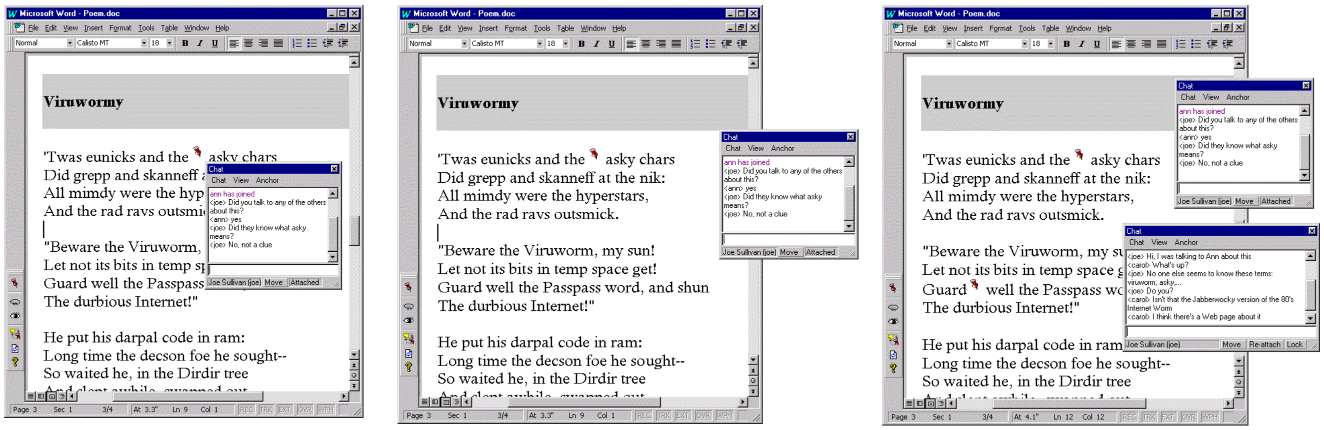
What we know about many collaboration situations is at odds with the technology affordances in this instance. Collaborators who are collocated and working in a "tightly coupled" way over those documents often converse about the details of documents: lines of text, figures or cells in a spreadsheet. Conversations in such collaborations have been characterized as "object laden" [7]; there is a high level of focus on the object or artifact that is under discussion and/or that is being co-constructed. Discussions are facilitated by (1) knowing which section is currently under discussion, (2) seeing the broader context in which the section is located, and (3) negotiating a shared visions of what is required and who will carry it out. Such conversations, therefore, crucially depend on shared context, i.e. that collaborators all have visual access to the artifact or object under construction or discussion. Here, "the object leads and language follows" [7].

FIELD WORK: CONVERSATIONS OVER WORK DOCUMENTS

Our intuitions about work collaborations were further developed in a series of field observations and interviews [3]. Our analyses focused on tightly-coupled collaborative work between non-collocated colleagues in a number of domains including software research, nuclear fusion experiments, geology field studies and after-sales software support. In each of these domains we noted issues arising around the sharing of artifacts.

In the first domain, software research, collaborators were using a text-based MUD to keep in touch and work together. Here, the tendency was to share files through email or by consulting shared file repositories. Specific details were shared by pasting text into the chat window and giving navigation cues like "Section 3, line 4" [5]. As well as being somewhat unwieldy and involving many steps, this copy-and-paste practice had the side-effect of taking the pasted-in material out of its local context. In the second domain, nuclear fusion experiments, problems arose around the sharing of experimental results in graphical and textual form. In the third domain, geology field experiments, we noted the need to support discussion around numerical data in a spreadsheet and graphs. In the final domain, after-sales software support, we noted the need to support collaborative problem solving over on-line software manuals.

In each of these cases problems arose around the establishment of shared context for conversations. The communication media (e.g. email, telephone and text-chat) were separate from documents under discussion and explicit navigation cues were required to literally "get everyone on the same page".


Microsoft Word - Poem.doc
File Edit View Insert Format Tools Table Window Help
Normal
Calisto MT
18
Chat
Chat View Anchor
Joe Sullivan (joe)
Move
Attached
Re-attach
Lock
Page 3 Sec 1 3/4 At 3.3" Ln 9 Col 1
REC TRK EXT OVR WPH

Viruwormy

'Twas eunicks and the asky chars
Did grepp and skanneff at the nik:
All mimdy were the hyperstars,
And the rad ravs outsmick.

"Beware the Viruworm, my sun!
Let not its bits in temp space get!
Guard well the Passpass word, and shun
The durbious Internet!"

He put his darpal code in ram:
Long time the decson foe he sought--
So waited he, in the Dirdir tree

ann has joined
<joe> Did you talk to any of the others about this?
<ann> yes
<joe> Did they know what asky means?
<joe> No, not a clue

<joe> Hi, I was talking to Ann about this
<carol> What's up?
<joe> No one else seems to know these terms: viruworm, asky,...
<joe> Do you?
<carol> Isn't that the Jabberwocky version of the 80's Internet Worm
<carol> I think there's a Web page about it

Figure 2: Chatting over a document using Anchored Conversations
Chat windows are anchored into word processor documents. Anchor points are represented as pushpins. There are no restrictions on the number of chat windows that can be anchored into a document.

ANCHORED CONVERSATIONS

The Anchored Conversations tool directly addresses the issue of coupling conversations and work artifacts when working remotely. Like chat spaces, the Anchored Conversations tool is computationally and cognitively lightweight, supporting synchronous and asynchronous communications. However, Anchored Conversations also allows text-based conversations to be coupled with the documents that provide the context for work discussions (e.g. word processor documents, presentation files, spreadsheet files, graphical simulations, etc). The user model is that of "sticky chats" - adhesive chat windows that can be stuck to documents for in-context conversations, much as a sticky note can be affixed to a printed document.

Although our examples in the following sections are all grounded in word processor documents, Anchored Conversations is application independent. Sticky chat windows can be placed into any application document, and can be moved between different application documents, just as a sticky note may be moved from one type of printed document to another.

Below we detail the features of Anchored Conversations. First we describe the "sticky chat" windows in which conversations take place. Then we describe how the Anchored Conversations tool supports document transport and sharing. Here we also describe how Anchored Conversations solves the problem of establishing and maintaining shared context.

Chat windows in documents

In Figure 2 we show the use of Anchored Conversations to support synchronous discussions over a text document. In the first image at the left of Figure 2, a text editor is shown with a document displayed. In the document is an anchored chat space, a "sticky chat" window. This is a standard chat space having a space for typing text, a window for viewing the ongoing conversation and a scroll bar at the right for viewing things that have already been said. The window can be resized easily. Again, as is standard, people's names are shown in the angled brackets at the left of the chat window.

The small pushpin icon to the top left of the chat window before the word "asky" indicates the location at which the chat is anchored - or, to put it another way, the context for the conversation. The chat space is literally *anchored* to the point where the pushpin is inserted. If the user scrolls the document, the chat window scrolls with it. It may even go off screen. On scrolling back, the sticky chat window will reappear, still attached to its anchor point. Similarly, if the user drags the application window around the desktop, the sticky chat window will stick to the application window at that location and move around with it.

Sticky chat windows may be stuck to the anchor as described above, stuck near the anchor or detached from the anchor. Thus, a sticky chat window's spatial location in relation to the anchor can be altered to prevent occlusion of the material in the body of the document.

In the second image in the center of Figure 2, the sticky chat window is shown having been moved to the right of its anchor point in the document in order to uncover the previously occluded text. The sticky chat window has been "locked" into this new location. The chat window is still attached to its anchor and will remain in this spatial relation to the anchor unless moved and again "locked" to a new location. So, if the user scrolls the document or moves the application window, the sticky chat will move as before.

Sticky chat windows can also be detached; in the third image at the right of Figure 2, two sticky chat windows are shown. The lower window in the image at the right of Figure 2 is detached. The toggle at the bottom of the chat window indicates this, as the user has the option to "reattach". When a window is detached, if the user scrolls the document or moves the application window, the chat window will not move with it. However, the sticky chat does not "forget" its "home" or context anchor point. By simply clicking on the 'Reattach' toggle at the bottom of the sticky chat window, the chat window reattaches to its "home" anchor location.

As shown at the right of Figure 2, users can have as many sticky chat windows anchored in the document as they wish. These sticky chats could be separate conversations between different groups of people or could be conversations that are separated due to theme or topic.

It is also possible to move a sticky chat window's anchor location by placing the cursor, and then selecting and clicking on the "Move" toggle at the bottom of the chat window. The new anchor location is set and the anchor is moved to a new place in the document. The sticky chat window is now anchored at this new location. When sticky chats are moved in one document, this change propagates: if I move a sticky chat to a new location, you will see the change and your chat will move too. Sticky chats can also be moved *between* documents. This is discussed below.

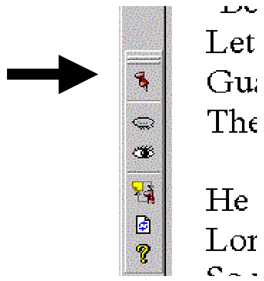


Figure 3 Pushpin Anchor Menu Item The Anchored Conversations tool adds a menu down the left side of applications. The pushpin menu item allows sticky chats to be attached to documents.

Starting conversations and sharing documents

The Anchored Conversations tool user interface consists of four parts: (1) the sticky chat window described above; (2) the pushpin that represents the point at which a sticky chat is anchored; (3) a menu bar that is added to applications and that appears down the left side of application windows (shown in Figure 3); and (4) a conversation coordinator window which interfaces to the Anchored Conversations database (shown in Figure 4).

Starting an anchored conversation requires few steps. By selecting the push-pin from the menu (shown in Figure 3) and clicking, an anchor is placed into the document at the current cursor point. Clicking on the push-pin in the document results in a sticky chat window appearing. Once the window is open, names can be selected from a "buddy" list in the menu bar. Selections result in invitations being sent automatically to invitees' desktops. This model is the same as for most messenger services. Also in accord with messenger services, once invitations are accepted, the invitees join the chat.

However, Anchored Conversations differs from other chat applications in that, on accepting an invitation to converse over a document, that document is *automatically* transported to the desktop of the invitee. The document itself opens automatically and is scrolled to the location where the sticky chat window is anchored. The sticky chat window is open and IRC available. Each person has his/her own copy of the document at this point; the sticky chat window is shared.

Anchored Conversations can also act as a simple chat application. If a chat window is not anchored within a document when created using the Conversation Coordinator, invitations result in a conversation client being opened on the desktop much as a standard chat client would. However, a chat can then be attached to a document and that document will be sent to others in the conversation.

Logging conversations and contexts

All conversations are logged in the sticky chat database (called the Conversation Database), and are available for review at any time – irrespective of whether the associated document is open. As all conversations are logged as text, searching and displaying by theme or person is trivial.

Conversation contexts are also logged. Sticky chats are not simply embedded chat objects. The "anchors" store information about the local context for the conversation. This context may be words that are nearest to the anchor, or cells in a spreadsheet – the specifics of context depend on the application in which the sticky chat is anchored. Context information of this kind is stored in the database along with information about the creation time and date of the anchor. All previous context locations in which the anchor has been inserted are also still available.

Users may query the database to see, for example, if an anchored chat has ever been located in a different place. All actions and conversations that take place within an anchored chat window are retained in the database, and are available for search, review and reuse. Even if the document in which the sticky chat was located has been deleted, it is still possible to review the conversations that were associated with the file – the filename and local context that the anchor was originally in are also preserved. Thus users can recreate conversation contexts if decisions need to be reconsidered. There is evidence that people review in this way occasionally [3].

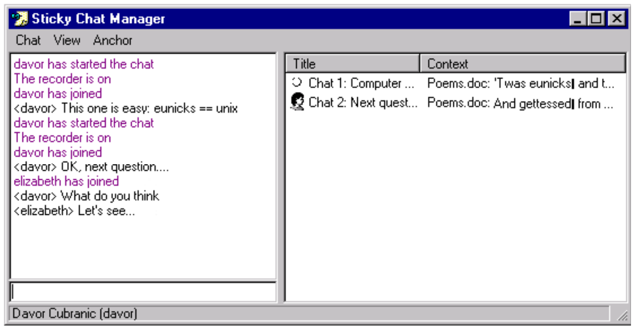
In Figure 4 we show our current interface to the database. Information about different active chats is shown (2 chats and their contexts are shown on the right of the window) as well as a log of all the current chats is shown on the left. By selecting the chats on the right, the user can navigate to a chat window in a document. We thus support our fourth goal, to provide review facilities for recreating conversations in context.

Scenario of use

In this section we present a short scenario to illustrate the use of Anchored Conversations in a distributed, collaborative work context.

Carol, Ann and Joe are colleagues working closely together on the collection and analysis of field data. They are currently preparing an executive summary of their recent work to distribute to the company management. Ann and

Joe work in the same geographic location but Carol works 4000 miles away with a 6-hour time difference.



davor has started the chat
The recorder is on
davor has joined
<davor> This one is easy: eunicks == unix
davor has started the chat
The recorder is on
davor has joined
<davor> OK, next question....
elizabeth has joined
<davor> What do you think
<elizabeth> Let's see...

Title	Context
Chat 1: Computer ...	Poems.doc: 'Twas eunicks and t...
Chat 2: Next quest...	Poems.doc: And gettessed from ...

Davor Cubranic (davor)

Figure 4. Interface to database. The area on the left displays the text chat and the area on the right lists the sticky chats that are currently open, their names and their current contexts. The icon beside the sticky chat name shows which chats are currently active.

Last week they had a telephone conference call so that they would all have a chance to agree on the general direction of their executive summary. In addition, they have been emailing messages back and forth daily. Today, Carol is working to incorporate her analysis into the draft document that Joe sent yesterday. She likes the graph showing the data points that they've chosen to use but thinks there is an additional interpretation that should be included based on the original data. However, it will require a substantial change to the summary graph and she's not sure how to best to represent her concern. She is anxious to interact with Ann and Joe about the data and the analysis. She has already made several modifications to the document itself. She has tried to summarize her concerns about the data in an email message. However, she wants to have a conversation about it. Although she knows she could call, Joe typically works at his home in the mornings while Ann will be in the company office. They would have to arrange for a conference phone call and still might have trouble knowing exactly what point in the original data was troubling her. She decides to use the Anchored Conversations tool. In the paragraph explaining the data, she inserts a sticky chat window and invites Ann and Joe. She knows they'll be getting into work soon and they'll see the invitation to chat with her. She then continues to edit other sections of the document, inserting and carefully positioning a second anchored conversation to explain why she reorganized the Related Work section.

As soon as Carol sees activity (she can see this both in the conversation coordinator window and in the sticky chat window itself), she returns to that section of the document. Joe is there and asks why she isn't happy with the data as presented. Carol immediately tells Joe she'll show him the issue with the original data and moves the chat window to the spreadsheet with all the original data. She anchors the conversation to the graph of data points that she thinks contradict the summary data representation. As she moves the sticky chat window, Joe's document automatically scrolls to the location Carol has selected and the sticky chat window in his document reanchored.

At this point, Ann joins in as well. She reviews the conversations that have taken place – seeing what was said and where it was said. She agrees with Carol's point and starts another sticky chat in a document where she had been trying various data representations. Neither Joe nor Carol has seen this document but copies open up for each of them on their respective desktops, and all the chats automatically follow. They continue to move around the data, the report document and their conversation, arguing and discussing the possibilities for their summary.

When Carol leaves for the day, she knows that the next morning she will have another go at the document -- and that Joe and Ann will have left conversations in critical places in the document to share their thinking with her as they continued to edit the report.

Anchored Conversations allows Joe, Ann, and Carol to work together on a shared document despite their differences in time and place. Carol was able to express her concerns about the data with a clear reference to the particular place in the data at issue. They were able to converse at the same time and to leave notes for each other at different times. Placing multiple conversations in the documents allowed different conversations to point directly to different topics. Also a conversation could move as the group discussion continued; it was not held to document boundaries or type. The anchored conversation serves as a means of annotation, of chat, of pointer in both real-time and at different times.

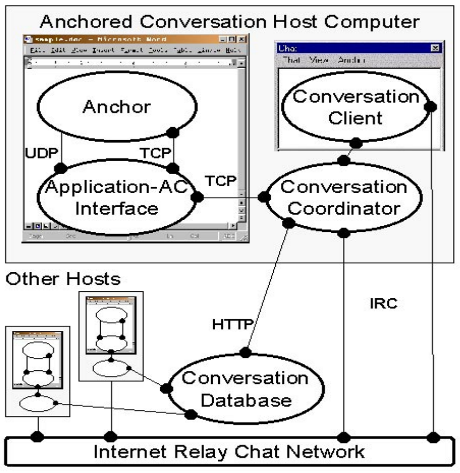


Figure 5 Architecture for Anchored Conversations

Implementation

The Anchored Conversations software design (Figure 5) consists of several components: (1) the anchor that locates a conversation within a document context; (2) the conversation client that provides the interface for one conversation; (3) the conversation coordinator that handles all conversations occurring on a host computer; (4) the

Application-Anchored Conversation interface that embeds the necessary functionality in a application required to enable conversations; (5) the Conversation Database that maintains access control and conversation history; and (6) the communication channel.

To facilitate document artifacts being accessed within their native applications (e.g. Word, PowerPoint, Photoshop) on a variety of host computers, the Anchored Conversations components are designed to be independently implemented and to communicate with each other through standard protocols (e.g. UDP, TCP/IP, HTTP). These protocols are selected for throughput and functionality needed for the different messaging requirements of the component interfaces.

The main conversational elements (Coordinator, Client, and Database) for the current prototype Anchored Conversations implementation are written in Java. The communication channel is IRC, which has been ported to a number of different systems. The Conversation Client is based on a slimmed down IRC client for text chat. Anchors are constructed using Microsoft's ActiveX technology, which enables embedding of anchors in a wide range of Microsoft Windows applications. In our study (described below), we used Microsoft Word 97, SR-2, running on Windows NT 4.0 and using different Pentium-type processors. The Word Application-AC Interface is implemented as a set of Word macros (Visual Basic for Applications, Version 5.0).

The detailed description of the software and flow of conversation and related document and control information starts with the anchors. Each anchor consists of an instance of an Anchored Conversations ActiveX control wrapped in a Word object (of type *InlineShape*). The control represents a specific point of conversation (by the pushpin graphic shown in Figure 2) and can determine and export its screen condition (location, focus, exposure etc.). Inserting an anchor thus causes a new Anchored Conversations control to be inserted within a context of the document. The context used in the current Word-based prototype to determine the conversation location consists of a range of text surrounding the anchor. Whenever the anchor object changes in visibility (by scrolling in the document) or moves (by window movement or by the user moving the anchor itself), UDP messages communicate these events. The Coordinator distributes anchor change events locally to the Client chat windows. The windows are positioned near to, and automatically follow their anchoring points within a document.

Significant events like anchor creation or deletion are distributed to other hosts using the Conversation Database. Each Coordinator maintains a local copy of the relevant parts of the database that gets updated as the system state changes. The chat client provides a user interface by which a conversation about some aspect of the artifact may be conducted. The text conversation messages are distributed from the chat clients to corresponding clients using the IRC network. IRC provides the basic machinery for invitation to a conversation. The Coordinator then copies the documents to the local host when an invitation is accepted and opens the document to the context of the inviting anchor.

There are a number of ways the software architecture permits the anchored conversation approach to be generalized beyond the current prototype including:

- The reference frame for anchor location may vary depending on the type of application and artifact (e.g., character position within a text file or [x,y] coordinate in a window, a range of text within a Word file).
- Anchor representation may include a bitmap image, text, or some combination of text and graphics. The representation may be static or may change in value based on information obtained directly from user input, from the Client, or other external sources.
- Clients may be associated with several anchors (multiple topic conversation) located in one or more artifacts. In the last case, one anchor can be considered its primary visual attachment to which the conversation homes.

Currently, we have implemented an open model of sharing: people who are invited to chat in any window associated with a document are able to access all other chats. However, more conservative, selective models of sharing can also be implemented. We have also sidestepped the issue of version control by providing users with a means of explicit resynching document versions. Prompts are given when anchor contexts across the open clients are out of synch and if users wish to synch up, they can click on a menu button to automatically update their documents to be in synch. Currently all documents synch to that of the original inviter.

RELATED WORK

The Anchored Conversations tool fits within the genre of lightweight communications technologies like chat spaces and messenger applications [1,8,14]. Unlike many of these applications Anchored Conversations retains logs of conversations and the contexts in which they took place. Further, chat windows in most applications cannot be associated with places (i.e. cannot be anchored into documents) and do not retain a history of their locations and usage in an integrated management tool. They cannot therefore be used for artifact/document navigation.

Anchored Conversations provide some of the functionality of collaborative document production tools like Quilt [9] that support collaborative document work. In addition, the storage of anchor context information is similar to that described for annotations in Quilt (e.g. creator, creation time, activity logs, notification and triggering mechanisms). In Quilt, discussion ideas in the form of annotations can be "attached directly to the relevant points in documents rather than indicated by indirect reference to those points". The authors argue that this "direct manipulation allows discussion threads to be followed more easily". However, Quilt does not support synchronous communications in the context of the text being discussed. Anchored Conversations is primarily a conversation tool rather than a shared annotation tool.

Also similar is Microsoft's Office 2000 facility in Word which supports in document threaded conversations. Whilst this is close to our notion of in-context conversation, within document threaded discussions cannot be viewed independently of the document. By contrast, sticky chats can be accessed via the database as well as via the document in which they are anchored. Further, Anchored Conversations is application independent – sticky chats can be moved from text documents to graphical documents to spreadsheets.

There are other means of sharing contexts for conversations (e.g. 2D graphical MUDs [14] and 3D virtual worlds). Shared context is established by re-representing artifacts in virtual worlds. This process is time consuming and represents a barrier to quick and easy contact, communications and collaborations; task related spaces must be created before virtual meetings can take place. At the level of the design metaphor, Anchored Conversations inverts this model. In the Anchored Conversation model, instead of taking artifacts into the virtual place, the conversations are taken into the pre-existing work artifacts that are the focus of conversations. The pre-existing artifact becomes the environment, backdrop or periphery for the conversations.

Document sharing and conversations are also supported in work environment systems (e.g.[10,11,12,15]). Here, applications which are relevant to a work project are all "corralled" into a Web page or window-based workspace. Thus, artifacts of relevance and the chat window are all in the same desktop window space. The association that occurs, however, is at the application level – the chat window is associated with the work artifact applications *because* it is in the same workspace. These tools do not integrate the conversation or chat spaces with the text of the documents; sharing is at the level of files and folders.

USER STUDY

An informal user study was carried out using our Anchored Conversations prototype. The goal of the user study was to evaluate the basic features of Anchored Conversations, to begin to observe the ways in which people might appropriate the functionality, and to obtain feedback on possible applications of the software in people's working lives.

Six teams of three participants were asked to share a document and to answer questions regarding various parts of the document. The questions directed the conversation to specific pages, paragraphs, individual words, and picture content. One person was elected "team leader" and took responsibility for inviting the others to help with the task. This person took responsibility for recording answers to questions, and her/his document was considered to be the 'master' document. The task was intended to be one that would engage the group in the content of the document rather than focusing on prototype features and that would require use of the anchored conversation space in joint activity. All groups were given a demonstration and training instructions, then 5-10 minutes practicing with the system and then 30 minutes to answer the questions. After training, group members were separated into three different rooms so that their only communication was by using sticky chats. Their instructions were to answer each question in the conversation space, reaching agreement among the three of them. For most questions, there was no single right or best answer. A follow-up discussion with group members included questions about their overall experience using the prototype and whether or not they could imagine using such an application in their own work.

All the participants said they enjoyed the exercise and all groups answered at least 3-5 questions. The "sticky chat" metaphor proved easy to grasp. A preliminary look at the data suggests that most groups chose to use multiple conversation spaces, that the participants were highly interactive and that the basic functionality was used without problems. In follow-up discussions, participants were enthusiastic about their experiences and provided numerous ideas for future design and use. One of our participants stated "This would be really good for online critiquing and discussion, like in the magazine Websites I use". More detailed analysis of the study will be forthcoming.

SUMMARY AND FUTURE WORK

In this paper we have described a tool, Anchored Conversations, which supports synchronous and asynchronous communications over documents. This application allows conversation places in the form of chat spaces to be associated with specific locations within work documents. The Anchored Conversations application and the document applications within which conversations can be anchored are associated by means of the preserved context information. Anchored Conversations windows, or sticky chat windows, are not embedded *within* documents – they are associated with locations in documents. Thus conversations may be accessed from within documents *and* from the Anchored Conversation coordinator window.

Anchored Conversations is both cognitively and computationally lightweight in the following ways:

- By taking conversations to documents, Anchored Conversations does not require re-representation in a virtual world before shared context can be established. Rather, Anchored Conversations provides a means of placing conversations within the already existent work contexts - documents.
- The Anchored Conversation tool set up shared contexts for interaction over documents automatically; users do not need to invoke other applications. Navigation to shared contexts is also automatic. Further, application control is not taken away and each collaborator has access to the application's full capabilities. There is also an easy mechanism for resynching documents if necessary.

- Anchored Conversations is easy to learn to use. As said, user's existing expertise and knowledge of how to use the document application remains relevant. Users are not placed into unfamiliar contexts to do work. By contrast, tools which re-render work artifacts require that users learn how to interact with those artifacts in new ways, and often familiar application types must be re-implemented.
- The Anchored Conversation tool is also low overhead as its use is uniform across different applications. Thus, inserting a sticky chat window into a spreadsheet requires the same actions as inserting one into a word processor document. Similarly, holding a discussion in a sticky chat window where an image provides the background (e.g. in Adobe PhotoShop) requires the same user actions as holding a discussion in a spreadsheet.
- Users can move between different documents and applications following conversation links. Following conversation links in this way results in the background documents being opened automatically, and circumvents the need for the user to search for the relevant documents to open. The Conversation Coordinator, not the user, handles the navigation to locations in documents in different applications and not the user.

There are a number of issues we are currently addressing in our work on Anchored Conversations. These include issues about sticky chat privacy and accessibility, implementation across different document types, database expansion, and shared editing of underlying documents. Our current model of document sharing is that, once invited to a single chat in a document, all other chats are available. A more conservative model may be appropriate with certain documents where security and privacy of conversations is of import.

We have successfully implemented a prototype that works with MS Word™ and are now facing the challenge of supporting sticky chat windows across different application types. Preserving context will then require appropriate representations in the database. We are also concerned to make the database more dynamic; we would like to support access to documents as well as chats via the database. Finally, Anchored Conversations could benefit from being hosted in underlying document applications that themselves provide shared editing and collaborative use.

ACKNOWLEDGEMENTS

The poem shown in Figure 2 is Virmuwormy by Ed Combs. The authors would like to thank all our study participants, and also Joe Sullivan, Cathy Marshall and Bill Schilit for their input to this project.

REFERENCES

1. AOL Instant messenger -- http://www.aol.com/aim/
2. Bradner, E., Kellog, W.A. and Erickson, T. The adoption and use of BABBLE: A field study of chat in the workplace. In Proceedings of ECSCW '99, September 1999.
3. Bly, S. and Churchill, E.F. Design Through Matchmaking: Technology is Search of Users. *interactions*, March–April 1999, 23-31
4. Churchill, E.F. and Bly, S. Virtual Environments at Work: Ongoing Use of MUDs in the Workplace. *Proceedings of WACC'99*, San Francisco, CA, USA. ACM Press, 1999.
5. Churchill, E.F. and Bly, S. It's all in the words: Supporting work activities with lightweight tools. To appear in Proceedings of Group'99, November 1999.
6. Erickson, T., Smith, D.N., Kellogg, W.A., Laff, M.R., Richards, J.T. and Bradner, E. Socially translucent systems: Social proxies, persistent conversation, and the design of 'BABBLE'. *Proceedings of CHI '99,* Pittsburgh, PA, USA, ACM, New York, 1999.
7. Fleming, D. Design talk: Constructing the Object in Studio Conversations. In Design Issues, Volume 14, Number 2, Summer, 1998.
8. ICQ -- http://www.mirabilis.com/
9. Leland, M.D.P., Fish, R.S. and Kraut, R.E. (1988) Collaborative Document Production Using Quilt. Proceedings of CSCW88, Portland, Oregon, USA, September 26-28, 1988.
10. Lotus Domino -- http://www.software.ibm.com/
11. Roseman, M. and Greenberg, S. TeamRooms: Network Places for Collaboration. In Proceedings of CSCW'96, Cambridge, MA, USA. ACM Publications.
12. Sohlenkamp, M. and Chwelos, G. Integrating Communication, Cooperaton and Awareness: The DIVA Virtual Office Environment. Proceedings of CSCW'94, Chapel Hill, NC, USA, 1994.
13. Stefik, M., Bobrow, D.G., Foster, G., Lanning, S. and Tatar, D. WYSIWIS revised: early experiences with multiuser interfaces. ACM Transactions on Information Systems, Vol. 5, No. 2 (April 1987), 147-167, 1987.
14. The Palace -- www.thepalace.com
15. TeamWave (http://www.teamwave.com/)
16. Viegas, F.B. and Donath, J.S. Chat Circles. Proceedings of CHI'99, Pittsburgh, PA, USA. ACM Press, 9-16, 1999.