MessagePlus: Focus + Context Conversation Browsers

First Author Name (Blank if Blind Review)
Affiliation (Blank if Blind Review)
Address (Blank if Blind Review)
e-mail address (Blank if Blind Review)
Optional phone number (Blank if Blind Review)

Second Author Name (Blank if Blind Review)
Affiliation (Blank if Blind Review)
Address (Blank if Blind Review)
e-mail address (Blank if Blind Review)
Optional phone number (Blank if Blind Review)

ABSTRACT
Readers frequently encounter messages from online conversations out of their original context. We develop a family of algorithms for presenting parts of the conversation surrounding a focal message; we refer to them as MessagePlus views. In a lab experiment, subjects using a MessagePlus interface were able to complete tasks more quickly than those using full displays of the same threads.

Author Keywords
Email archives, email threads, focus + context, Usenet, threading, conversation

ACM Classification Keywords
H.4.3 Communications Applications: Information Browsers, **H.5.2** User Interfaces

INTRODUCTION

Online discussion boards and archives of email lists are valuable stores of information. Previous work has taken a variety of approaches to displaying the messages contained in these repositories. That messages should be displayed as a conversation is a recurring theme in existing research [1, 6]. Some approaches to displaying conversations focus on the content of messages within the lists, some on the relationship of messages to each other; other displays highlight information about the authors of messages and their relationships to and around the list.

Many discussion boards and email list archives are displayed in a threaded interface; messages are displayed in an outline-like list with other messages that reply to one another. The resulting display is a tree-structure. Threaded displays make visible the turn-taking nature of conversations [11] and helps provide some cues about activity in the thread (e.g., a message's depth in the conversation; busy branches of conversation; abandoned topics.)

Information surrounding a message, including its author, the time it was posted, its subject, the message to which it replies, messages that reply to it, etc. could provide useful cues about a message's content. However, standard interfaces leave room for improvement because they do a poor job of conveying this context [7, 12]. Existing thread interfaces provide some local context and may be more useful than strictly chronological views, but they leave room for improvement[13].

Visualizations and Controller Panes

Many research efforts and commercial products have developed visualizations for conversations. Often, a visualization pane is yoked to a pane that displays the conversation, so that the visualization acts as a controller pane. We briefly survey such approaches below.

Several projects create maps that represent the reply structure of a conversation. Thread Arcs are a visualization [13] for personal emails that combines chronology and reply structure to create a multi-pane visualization that displays message-to-message relationships through lines and chronology through space [4] (as shown in Figure 2). The Treetable approach offers an alternative map of a thread; one version treats the map as a controller pane [9, Figure 10]. As shown in Figure 1 the Google Groups interface to UseNet also offers a controller pane with a map of the whole thread on the left side; each message is represented by a single line, with indentation ed to show which messages reply to others.

Conversation Map offers both an alternative version of a thread map and several other visualization panes. It computes and displays social networks among authors, discussion themes, and semantic networks of the conversation[10]. This approach allows users to explore the conversation from a variety of angles: who's talking, what terms are being used, and what's being talked about.

Conversation Thumbnails provides another overview of discussions. It uses a three-pane display. One pane provides a thumbnail scale version of the conversation in which stacked rectangles represent messages, and indentation shows reply structure [14].

Figure 1: Google Groups controller pane interface.

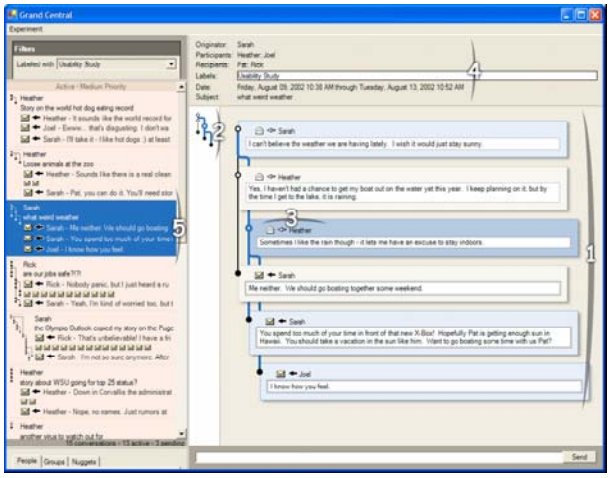

Figure 2. Mixed-model visualization (from Venolia & Neustaedter, 2003)

The multi- and controller-pane interfaces mentioned may be useful in some situations, but they have some drawbacks. They sometimes require specific user action, such as a button click, to access the actual content of messages. More critically, our informal observations revealed that users had some confusion about how controller panes were connected to the content pane and how exactly they affected one another.

Our study focuses solely on the design of message view panes. In this approach, users control the scrolling and display of the pane directly rather than through another yoked pane with a different display. Even if visualizations will frequently be provided in controller panes, it will be helpful to research the design of main message panes. If they can be improved sufficiently, separate controller panes will be needed less often.

The Need for Context in Message Panes

It is natural to think of conversations unfolding over time, with participants relying on their memory of previous messages to make sense of new ones. Some people participate in Usenet and email discussion lists in just this way. It is quite common, however, for users to approach a new message not having read the other messages in the conversation, or not having read them recently.

One common scenario is that a user performs a keyword search of a group's message archive, or performs a more general search through a site such as Google. The search results generally provide a list of messages. The user clicks through on a particular message and is presented with that as a focal message; the user can scroll or click to see other messages from the same conversation thread.

SlashDot's meta-moderation interface provides another example. Moderators assign labels to messages. Other users meta-moderate: they say whether the moderators assigned labels fairly [5]. The interface presents the message and the label that was assigned to it, without any of the surrounding messages.

Even when participating in an unfolding conversation, readers may arrive at a message of interest without necessary context. They may skip some messages, especially if the system suppresses the display of low-rated messages, as SlashDot does. They may skim over other messages or jump to messages by authors they find interesting. And they may return to a conversation after a few days away and not remember the contents of all the messages they had read previously.

These are just a few of the scenarios in which users encounter a single focal message which is not necessarily the first message in a thread. In order to understand that message, users may need to read other messages such as the one to which their focus message is replying or the first message in the conversation. But if all the messages in a thread are displayed, the important contextual information may be hard to find. Because threaded views display all messages that branch off from that original head message, the view may become cluttered if the conversation tree is both wide and deep. Threaded views may display too much information at once; messages in email lists often contain extraneous material such as signatures and quoted text, and because this material uses screen real estate, reading messages remains a one-at-a-time activity [8]. Navigating through the clutter of a busy conversation thread can be both time consuming and frustrating for users.

The next section will describe the design space of single pane conversation viewers, with an eye towards alternative displays that provide some context from non-focal messages. We will then describe hypotheses about which choices in that design space will provide appropriate context from non-focal messages. We then describe a test interface based on those hypotheses and will also describe an experiment that compared that test interface to a normal threaded view. We describe the results of that experiment

and explore what those results mean for future research on the design of conversation viewers.

DESIGN SPACE

Within the realm of views that present information from one or more messages of a discussion, it is possible to characterize the design space of alternatives fairly cleanly. Any design of this kind can be characterized as a set of rules for filtering, sorting, and displaying individual messages. We first consider the message properties that can be used in defining the rules. Then we explore options for each of these in detail below.

The structure of the reply graph offers several useful ways of categorizing messages in relation to a focal message. There are direct replies to the focal message and indirect replies (replies to replies to the focal message, and so on). There are sibling messages that respond to the same parent message. There is the direct parent (the message that the focal message replies to), indirect ancestors (the parent of the parent, and so on), and the top-level ancestor that generated the whole reply tree that the focal message is part of. Potentially, different display options could be applied to any of those types.

There are several other kinds of relationships among messages that can be used to determine filtering, sorting, or display options. One is temporal ordering of when messages were posted. Another is co-authorship: other messages written by the same author as the focal message may provide useful context. Another potentially useful relationship is content similarity between the focal message and other message.

Finally, there are properties of individual messages that can be used in determining display options. For example, messages with more original text (as opposed to quotation of prior messages) or messages that include external links might be treated differently. Messages that have received high ratings according to some scoring system, or those posted by authors with high ratings might also be treated differently.

Filtering

The first way that these message characteristics can be used is to determine filtering of content. Some messages may be omitted from the display entirely. Others may get only one line of the display. Others may get a few lines. Some messages may be displayed in their entirety.

In the original work on fisheye views of graph structures, an algorithm automatically determines a score for each node based on its distance from a focal node, and whether and how much of each node to display is determined by its score [3]. When applied to the source code, with program blocks as nodes with links to sub-blocks and individual lines of code, this technique produces displays that make it easy to grasp where any particular line of code fits into an overall program. This same approach, applied to a conversation, would cause messages near to a focal message to be displayed while those farther away would be hidden or displayed in abbreviated form. The definition of nearness to a focal message can rely on any of the characteristics described above, including distance in the reply graph, textual similarity, or common authorship.

	Display Options	**Examples**
Filtering	What text from each message is displayed (none, one line, the full text)	1. Fisheye view scores 2. Slash(dot) thresholds
Sorting	Order in which messages are displayed	1. Chronological 2. Reply structure 3. Message score
Formatting	display properties of messages	1. Highlighting focus message 2. Indented reply structure

Table 1. Summary of Design Space

Slash(dot) also employs a filtered view, where a comment may be suppressed from the display entirely, or displayed in abbreviated form. In the default view, for comments at the top level (i.e., direct responses to an initial news story) full text is shown if a comment has a moderation score of 1 or higher (scores range from -1 to +5). For comments deeper in the thread (i.e., replies to other comments), full text is shown if a comment has a score of 4 or higher, a single line is shown for comments rated 1-3, and the comment is omitted if it's score is below 1.

Sorting

The next design option is sorting. In principle, any of the message features could be used for sorting. In practice, however, only three kinds of message properties have been used for sorting: chronological ordering, reply structure, and scores for messages or authors. Flat displays based on post times are commonly used for archives of email lists, inbox-based interfaces to Usenet (e.g., MS Outlook's reader), and (in reverse chronological order) blog posts. Threaded views are based on a depth-first traversal of a tree of comments, where the tree is defined by the reply-structure. Among threaded views, there are still options about how to sort the messages that are all replies to a single message. The conventional method is chronological, although sort based on a relevance or quality score is another option. For example, in Slash(dot)'s threaded view (the default view at Slash(dot)), there is an option to sort messages that have the highest scoring top-level posts

Figure 3. Gmail interface for personal email archives

shown first, and sort the replies to each post as well, while still traversing the tree of replies in a depth-first manner. Amazon.com provides another example of the use of scores for sorting: reviews from their top-ranked reviewers are displayed first, above reviews from regular reviewers. The TreeTable approach explores a generalization of one-dimensional sorting to two-dimensional spatial layout [9, Figure 7].

Display Formatting

The third set of options concern the display formatting of individual messages. The full range of text formatting options are available, including indentation, text and background colors and shading, font, size, and bold and italics. It is common in threaded conversation views to indent replies a little more than the messages they reply to. This allows readers to find the parent message of any message by scrolling upward and scanning for the first place where there is less indentation. On the other hand, indenting replies can lead to inefficient use of screen real estate if there are many levels of nested replies. Other formatting options could be used to code for particular properties, such as immediate replies to a message or messages by particular authors.

HYPOTHESES

The theory of effective view traversal (EVT) draws out implications of two requirements for view navigability: 1) a limited number of outgoing links and 2) short paths from node to node [2]. By analogy, a major goal in picking a design from the design space is to identify the most likely messages that a user would navigate to from a focal message, and make those prominent, while suppressing other messages.

Two common tasks in which email list and Usent users engage are 1) searching for advice on an unfamiliar archive, and 2) determining the sincerity of a focus message. In the first task, it is likely that messages in the same branch of the conversation as the focus message will provide additional information about the scenario to which advice relates; other messages by the same author will provide cues about the author's reliability and level of engagement. In the second task, messages that follow the focus message will shed light on how others interpreted the post and reveal any clarifications made by the author.

Thus, we would expect that an interface that highlights replies, parents, and messages by the same author as a focus message would be most useful for these tasks. Messages in other branches of the conversation and other messages not directly related to the focus message are likely to produce noise rather than to provide context. Special text formatting to encode information is not likely to be obvious to novice users, so should probably only be used in situations where heavy use rather than one-time use is expected.

	Gmail	**Slash(dot) default view**	**MessagePlus**
Filtering	• Full text of focus message • First line of other messages • Hides quoted text	• Full text of high scoring messages • Zero or one line of other messages	• Full text of focus message • Zero, one or three lines of other messages • Hides quoted text
Sorting	Chronological	Reply structure	Reply structure
Formatting	• Color coding for author • Bold for unread • Gray outlines for each message	• Indented replies • Italics for original text • Bold for author and time	• Indented replies • Bold for author name • Black outlines for each message

Table 2. Gmail, Slash(dot), and MessagePlus Design Space Choices

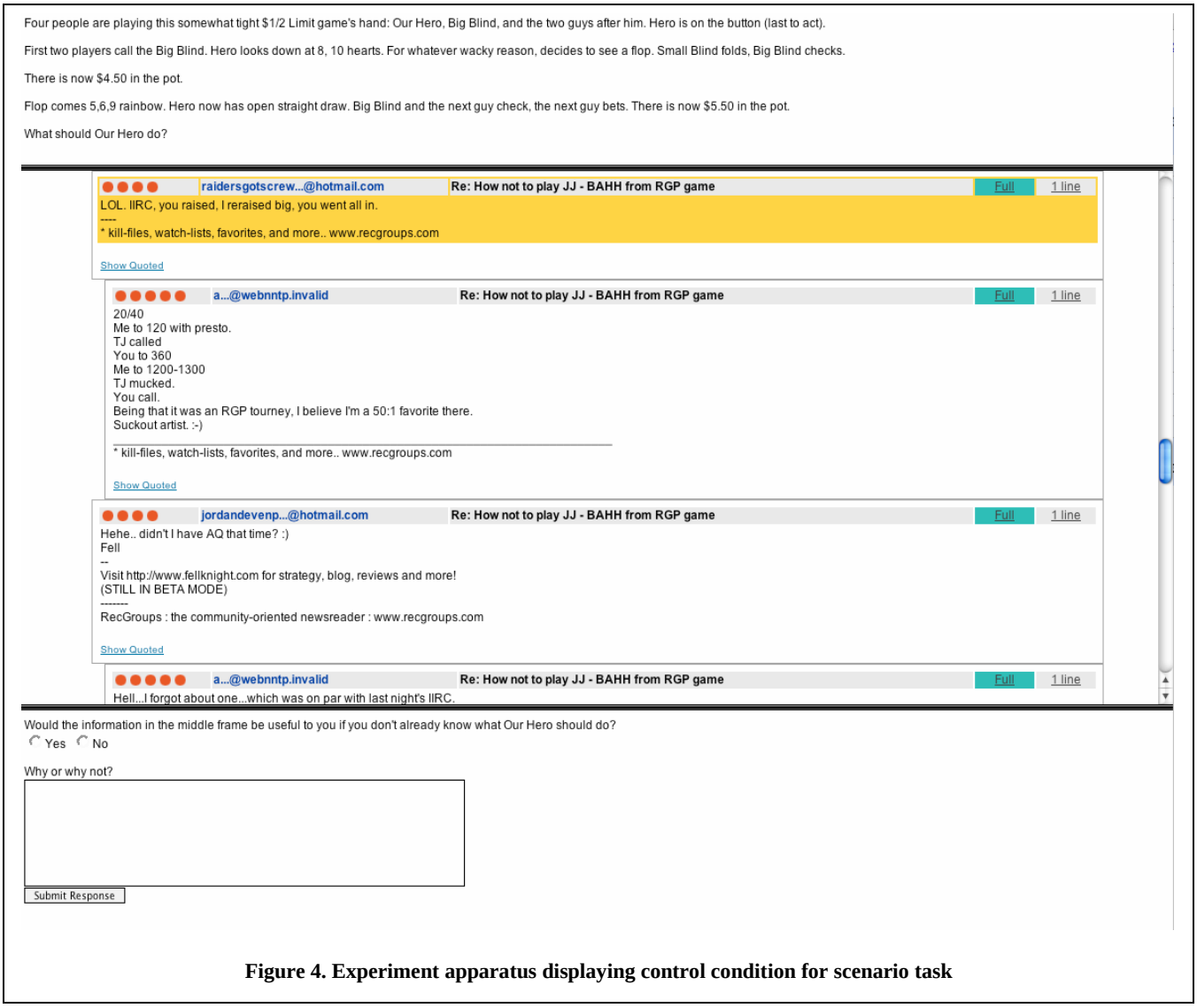

Figure 4. Experiment apparatus displaying control condition for scenario task

METHODS

The primary goal of this study was to test whether displaying some filtered, sorted context for an individual message allows users to perform common tasks more easily. We constructed an interface that maintains the reply structure of a conversation and allows users to make decisions about what additional information is displayed or hidden. We assigned subjects to one of two versions of the interface and compared task completion times across the two conditions

Apparatus

The MessagePlus system encompasses a family of display algorithms. All the algorithms employ sorting based on thread structure, and all employ the same formatting styles. They differ only in what the initial filtering state of each message is: full original text, 3 lines, 1 line, or hidden. With any of the display algorithms, the user can override the initial state manually.

Our implementation represents each message within a thread in one of four ways: as a hidden element, as the first line of original content, as the first three lines of original content, or as a full view of the original content. Here, original content refers to the text within a message that has been added by the most recent offer; MessagePlus hides quoted text that writers had included from earlier messages but allows readers to display that text if they choose.

To render the HTML page that displays the conversation, we build 3 HTML elements for each message - one for each view of the message except as a hidden element. We then use cascading style sheets (CSS) to adjust display properties on those elements to show or hide them based on the MessagePlus algorithm which is described in more detail later in this section. When the

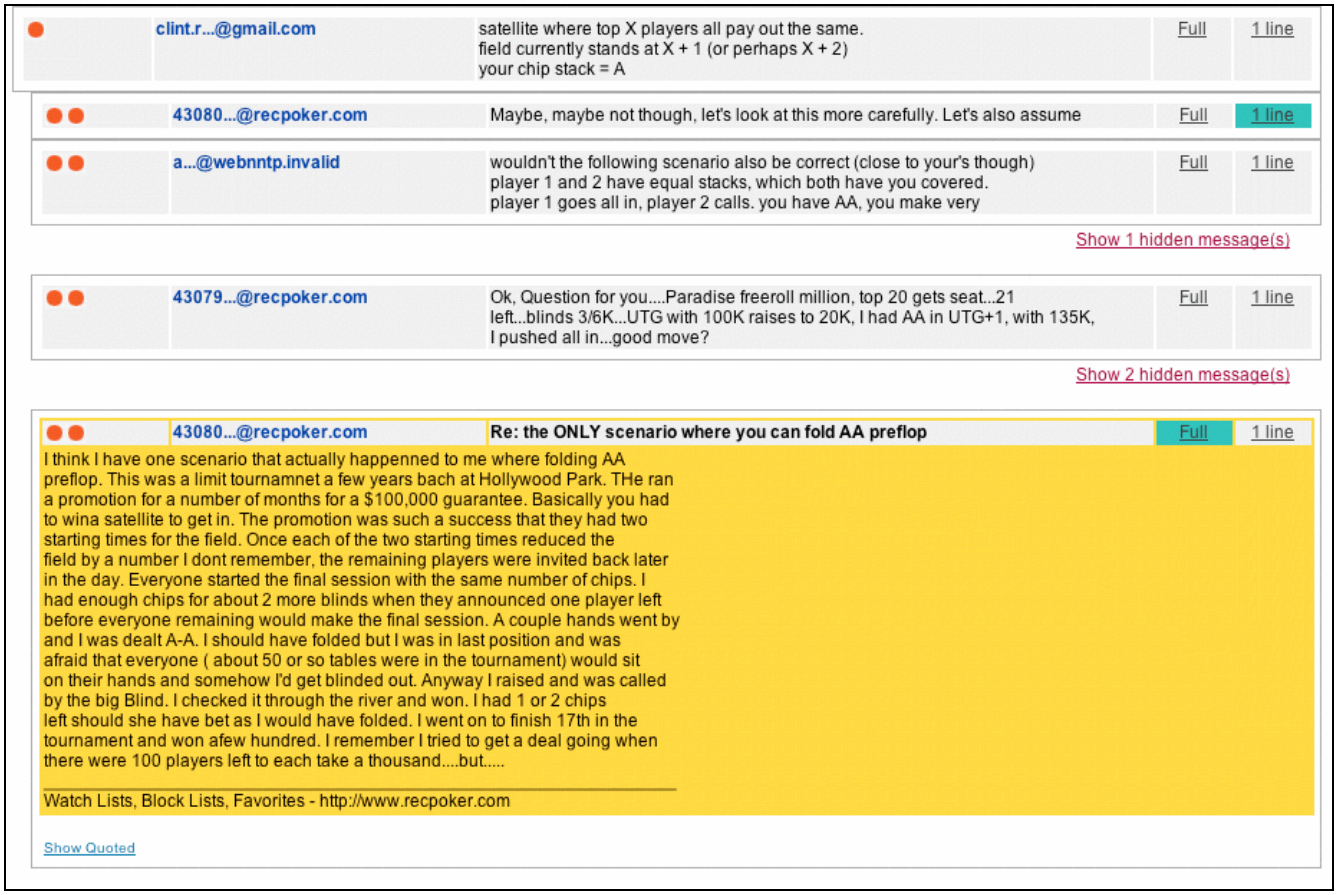

Figure 5. MessagePlus Interface

page is first rendered, only one element for each message is displayed. By default, only the focus message is shown in full; it is also highlighted so that users can keep track of it even when other messages are shown in full. Links are available to allow users to toggle among the four view options for each message. Each link calls a JavaScript function that changes the CSS properties of the affected elements.

This approach allows MessagePlus to minimize interaction with the server and to utilize a single database call when rendering the conversation. All rendering changes are executed client-side via JavaScript.

The tree structure of the conversation is maintained by indenting each message's elements according to its depth in the conversation. In addition, the depth of a message is encoded in the number of orange dots at the left of the message's top line. The order of messages is determined by the fixed reply structure defined by the References: headers of the Usenet messages..

Two versions of the display algorithm were used for comparison in the experiment. In the Everything Visible condition, all messages began in the "full display" state. In the MessagePlus condition, only the focal message began in that state; other messages were hidden or displayed in 1-line or 3-line format, as detailed in Table X. For each group of hidden messages, the interface contains a link that, when clicked, shows one line of each of the hidden messages. All other aspects of the interfaces were identical. Figures 4 and 5 show the two versions of the interface.

We chose to test MessagePlus against a full view of every message because MessagePlus employs our best guess at the right algorithm for displaying appropriate context; the results of this experiment will determine whether this algorithm is good enough that it beats displaying everything.

Interested readers can examine the experimental apparatus by visiting www.libbyh.com/research/messageplus/messageplus.php. You will be randomly assigned to use of the two versions of the interface and will be walked through the experimental tasks.

Procedures

We selected a set of conversations taken from the Usenet mailing list rec.gambling.poker. We selected rec.gambling.poker because it is an active list with a wide variety of conversations and authors. We selected conversations to include based on the material they covered, their length, and their reference to common concepts. The resulting group of conversations varies in length from 3 messages to 45 messages, with a median of 17. They cover a variety of topics from rating professional poker players to generic discussions of pot odds calculations and general poker betting strategy. We limited the conversations to those about the Texas Hold 'Em variant of poker.

Type of Message	Default View in MessagePlus
Focus message	Full original text
First message in thread	3 lines
Parent of focus message	3 lines
Sibling of focus message	3 lines
Ancestor of focus message	3 lines
Offspring of the focus message	1 line
Messages by the same author	1 line
All other messages	Hidden

Table 3. MessagePlus Default Display

Subjects were randomly assigned to one of two experimental groups: Everything Visible and MessagePlus. Before beginning the experiment, subjects were given a few minutes to explore the interface's functions so that they understood what each link meant and were familiar with the actions before the experiment began. Subjects also completed a questionnaire that asked how often they played poker, used Usenet, used email lists, used online discussion boards, and read poker books.

Tasks

Our informal observations of rec.gambling.poker Usenet revealed that searching for hand advice was one of the most common user tasks. In the first task, which we call the scenario task, subjects were asked to determine whether or not the information displayed would help a poker player, given information about his hand and how others around him were betting. They were shown eight separate conversation threads and asked to assess the utility of each. In each case, the display automatically began on a focal message (not the first message of the thread); users could scroll up or down from there.

Identifying "trolls" or messages posted in jest is another important task in any email discussion archive. After the scenario task, subjects performed the troll task. Subjects were shown 10 messages, each embedded in a conversation, and were asked to determine the sincerity of the focal message.

Subjects

We recruited 30 subjects for our study; each subject had at least some experience playing Texas Hold' Em and using online discussion boards or email lists. We report findings from 28 subjects; two subjects experienced network connectivity problems during the study that prevented them from completed the tasks, and we removed their data from our analysis. 15 subjects were in the Everything Visible condition, and 13 were in the MessagePlus condition. The system assigned subjects to conditions based on their subject id numbers, and one subject had to restart the experiment at the initial screen, due to a system error; this restart changed his group assignment. This explains why we have groups of N =15 and N = 13 instead of two groups of N = 14.

Most subjects were graduate and undergraduate students. The experiment took about one hour, and subjects were paid $15 for their participation. Payment did not depend on the speed or accuracy with which they answered scenario and troll questions.

We recorded task completion time for each subject reading and assessing each thread. We also recorded each JavaScript call made by each participant; this data tells us what messages subjects chose to show and hide, whether subjects changed the default display by collapsing or expanding any messages, and when they chose to view quoted text.

RESULTS

All participants performed the scenario task first and troll task second; all participants saw the threads in the same order. For each thread, we computed mean scores for task completion and ran *t*-tests to determine the significance of any mean differences. We also calculated the overall task completion time for the scenario task and troll task and performed t-tests to compare between conditions.

Scenario Task

For each thread of the scenario task, subjects in the MessagePlus condition completed the task faster; for half of the threads this time difference was statistically significant for $p < .05$. The overall task completion time for MessagePlus subjects was also significantly. See Table 4 for a summary of these results.

In summary, the scenario task showed a significant effect for interface for half of the threads, and a strongly significant effect for overall task completion time.

Troll Task

Thread	Condition	Mean Time in Seconds (s.d.)	Mean Difference
1	Everything Visible (EV)	347 (195)	-175**
	MessagePlus (MP)	172 (105)	
2	EV	195 (131)	-98*
	MP	98 (68)	
3	EV	232 (173)	-64
	MP	180 (114)	
4	EV	197 (132)	-93*
	MP	104 (62)	
5	EV	221 (157)	-91
	MP	130 (69)	
6	EV	153 (61)	-15
	MP	138 (83)	
7	EV	111 (49)	-20
	MP	91 (61)	
8	EV	231 (106)	-81*
	MP	150 (84)	
Total	EV	1688 (701)	-624*
	MP	1064 (463)	

*$p < .05$, **$p < .01$

Table 4. Scenario Task Completion Time Comparison

For 9 of 10 threads of the troll task, subjects in the MessagePlus condition completed the task faster; however, none of these differences were statistically significantly different. The overall task completion time for MessagePlus subjects also showed no significant difference ($t(26) = -1.16$, $p = 0.13$). See Table 5 for a summary of these results.

DISCUSSION

Our results revealed that MessagePlus provided an advantage for task completion time, but the significance of this advantage depends on the task being performed and the thread being read. In the scenario task, where subjects were asked to determine the usefulness of a given thread, MessagePlus proved significantly faster for half of the threads. This tells us that MessagePlus is better than displaying everything but does not necessarily support all aspects of our default algorithm. Subsequent analyses will explore user behavior data to determine whether MessagePlus users chose to hide or display other messages often; this analysis will provide more support or suggest revisions to our default display algorithm.

The results from the troll task were less clear. For 9 of the 10 threads, MessagePlus subjects performed the task faster, though in no case was the difference statistically significant. It may be that when determining the affect intended in a post, the replies to the focus post are most helpful in clarifying or extending that affect. Replies to a particular message are fairly easy to find in a full threaded view, simply by scrolling. In our MessagePlus display, only the first line of children of the focus message are displayed,

Thread	Condition	Mean in seconds (s.d.)	
1	Everything Visible (EV)	223 (248)	-227
	MessagePlus (MP)	94 (58)	
2	EV	164 (103)	-2
	MP	162 (217)	
3	EV	90 (55)	-21
	MP	69 (57)	
4	EV	107 (70)	-37
	MP	70 (64)	
5	EV	159 (162)	-15
	MP	144 (240)	
6	EV	91 (44)	-22
	MP	69 (68)	
7	EV	79 (46)	-31
	MP	48 (35)	
8	EV	109 (47)	9
	MP	113 (127)	
9	EV	87 (61)	-23
	MP	64 (49)	
10	EV	75 (41)	-24
	MP	51 (37)	
Total	EV	1183	-399
	MP	884	

Table 5. Troll Task Completion Time Comparison

and the troll task results may indicate that either more first lines of children (reply messages) need to be read, or more child nodes need to be expanded in order to understand the affect of a given message. Both of these behaviors would lead to faster task completion with the full display. Despite that, the data weakly suggest that the MessageDisplay led to faster task completion times than the full threaded display.

CONCLUSION

The results of our test interface were promising. On both tasks, the limited display seemed to lead to faster completion times, though the difference was statistically significant only for one of the tasks. We were able to determine that some hiding is helpful, especially for determining the relevance of a given message, and that users were able to understand the show/hide functions of an interface that filters messages.

ACKNOWLEDGMENTS

REFERENCES

1. Comer, D. and L. Peterson (1986). "Conversation-based mail." TOCS **4**(4): 45-49.
2. Furnas, G. W. (1986). Generalized fisheye views. CHI.
3. Furnas, G. W. (1997). Effective View Navigation. CHI, ACM Press.

4. Kerr, B. (2003). THREAD ARCS: An Email Thread Visualization. IEEE Symposium on Information Visualization, Seattle, WA.
5. Lampe, C. and P. Resnick (2004). Slash(dot) and Burn: Distributed Moderation in a Large Online Conversation Space. CHI, Vienna, Austria, ACM Press.
6. Lewis, D. and K. Knowles (1997). "Threading Electronic Mail: A Preliminary Study." Information Processing and Management **33**(1): 209-217.
7. Newman, P. Treetables and Other Visualizations for Email Threads, Human-Document Interaction Group at Xerox PARC.
8. Newman, P. (2002). Email Archive Views Using Subjects Indexes. CHI, ACM Press.
9. Newman, P. (2002). Exploring Discussion Lists: Steps and Directions. Joint Conference of Digital Libraries.
10. Sack, W. (2000). Conversation Map: A Content-based Usenet Newsgroup Browser. IUI.
11. Smith, M., J. J. Cadiz, et al. (2000). Conversation trees and threaded chats. ACM Conference on Computer Supported Cooperative Work, ACM Press.
12. Smith, M. A. and A. T. Fiore (2001). Visualization Components for Persistent Conversations. CHI, ACM Press.
13. Venolia, G. D. and C. Neustaedter (2003). Understanding Sequence and Reply Relationships within Email Conversations: A Mixed-Model Visualization. CHI, Ft. Lauderdale, FL, ACM Press.
14. Wattenberg, M. M. and D. R. Millen (2003). Conversation Thumbnails for Large-Scale Discussions. CHI, Ft. Lauderdale, FL, ACM Press.

Contributions and Benefits:

We develop a family of focus+context views for message threads; lab test of one shows it's superior to displaying complete threads.