Supporting Email Workflow

Gina Danielle Venolia, Laura Dabbish, JJ Cadiz, Anoop Gupta

Revised December 2001
(Original September 2001)

Technical Report

MSR-TR-2001-88

Microsoft Research

Microsoft Corporation

One Microsoft Way

Redmond, WA 98052

Supporting Email Workflow

Gina Danielle Venolia[1], **Laura Dabbish**[2], **JJ Cadiz**[1], **Anoop Gupta**[1]
Microsoft Research—Collaboration and Multimedia Group

ABSTRACT
As more people use email at home or on the job, more people have come to experience the pain of email that Denning first wrote about 20 years ago [5]. In this paper, we present data from a field study in our own company to add to the existing body of research about how people use email. We then use these data and prior literature to outline a framework of the five main activities that we believe people engage in when dealing with email. In particular, we focus on two activities that we believe have been under-studied: attending to the flow of messages as they arrive, and doing "triage" on a body of new messages. In addition, we outline potential design directions for improving the email experience, with a focus on email clients that group messages and their replies together into threads. We present a prototype of such an interface as well as results from a lab study of the prototype.

Keywords
Email, electronic mail, asynchronous communication, personal information management, task management, computer-mediated communication

INTRODUCTION

In 1982, Peter Denning (then the ACM President) first wrote about the pain of working with email, calling it "The Receiver's Plight" and asking, "Who will save the receivers [of email] from drowning in the rising tide of information so generated?" [5]. Twenty years later, we still don't have the answer. Numerous studies have continued to provide data outlining the plight of email users, and it seems the only thing that's changed is that the number of people experiencing this pain has risen dramatically. Feelings first expressed by the ACM President are now headlines in national newspapers: "Email overload taxes workers and companies" [13]. Furthermore, the trend isn't slowing. IDC reports that in the year 2000 there were 452 million email mailboxes and approximately 9.7 billion messages exchanged on an average day. In 2005, the numbers are predicted to jump to 983 million mailboxes and 35 billion messages [10].

Simply put, email has become a place where many of us now live—a "habitat" in the words of Ducheneaut and Bellotti [8]. As shown by Whittaker and Sidner [17], this place poorly supports the tasks we need to accomplish. As they note, email has become overloaded: The usage and uses of email go far beyond what we could have imagined twenty years ago, but the interfaces of mail clients have not kept pace. In many ways, email has become a victim of its own success.

PREVIOUS EMAIL LITERATURE

Researchers have been studying email for quite some time. Much of the early work on social and organizational aspects of email is summed up well by Sproull and Kiesler [14]. In this paper, we'd like to focus on identifying users' needs and suggesting design directions to support them.

The research on how people work with their email clients includes both studies of current use and studies of prototype interfaces. As noted in the previous section, Denning [5] was the first to point out that current email clients did little to help people who received lots of email. Denning proposed several solutions to the problem based on two principles: First, there should always be a special path for people to get urgent, certified, and personal messages; and second, all other paths should be filtered.

Six years after Denning's paper, Mackay [11] published results from an extensive study of email (based on the Information Lens system built by Malone et al. [12]). Her results included two primary findings: People use email in incredibly diverse ways, and people use email for much more than just basic communication (e.g. task management, task delegation, time management, archiving information for future use). She also found that people generally fell into one of two categories when it came to handling email: *archivers* or *prioritizers*. Archivers focused on strategies for making sure that they would see all messages and not miss anything important; prioritizers focused on strategies to limit the time they spent with email so that they could get other work done. In a nutshell, prioritizers controlled their email while archivers were controlled by their email. Mackay also classified people based on whether they were "overwhelmed", "on the edge", or "ok" when it came to handling all their email.

Eight years after Mackay's work, Whittaker and Sidner [17] published their study on email use within Lotus. Like Mackay, they found that email was being used for several tasks in addition to basic communication, calling the phenomenon "email overload." They also studied how people handled email overload when it came to filing messages and classified people as *no filers* (people who don't clean up their inbox but use searching tools to manage it), *frequent filers* (people who constantly clean up their inbox), and *spring cleaners* (people who cleaned up their inbox once every few months).

Five years after Whittaker and Sidner's work, Ducheneaut and

[1] One Microsoft Way, Redmond, WA 98052 USA
{ginav; jjcadiz; anoop}@microsoft.com

[2] Work done while a summer intern at MSR; currently at
746 East End Ave., 1st Fl., Pittsburgh, PA 15221 USA, dabbish@cmu.edu

Bellotti [8] published their study, which examined email usage in three organizations. Like the previous studies, they found that email is used for a variety of tasks, such as information management, coordination, and collaboration. In fact, they found that people used email so often for so many tasks that they called email not just a killer application, but a "serial killer," writing: "It is seriously overloaded and has been co-opted to manage a variety of tasks that it was not originally meant to support."

A MODEL OF EMAIL WORKFLOW

Based on our review of the literature and early attempts to design an improved email client, we developed a conceptual model of users' activities surrounding email. We identified five different activities:

Flow: As people are working on other tasks, they want to keep up with the flow of incoming messages as they arrive.

Triage: After people are away from their email for a period of time, they need to catch up and deal with all the email that accumulated while they were away.

Task management: People often use email to remind them what they need to do, and to help them get tasks done.

Archive: People store email so they can refer to it later.

Retrieve: After archiving messages, people need a method of retrieving messages.

While the latter three activities are often discussed in the literature, less attention has been paid to the first two.

To validate this model and understand its intricacies, we collected data using three methods, described in the next section. We then discuss a detailed understanding of the model and suggest a number of user interface improvements, both based on analysis of the data and information from previous literature. Conversational threads are a recurring theme in the UI improvements; they are discussed next. We then sketch out a design integrating the various user interface improvements. The design sketch centers around a thread-based email browser, which we prototyped and tested; the results are described last.

DATA COLLECTION

The model described above made intuitive sense to us, but that, of course, is a poor basis for a design. To validate the model and understand it in depth, we studied employees in our company using three methods: interviews, analysis of message archives, and a survey.

Structured Interviews

We interviewed ten individuals for the first part of our study. Participants included a systems engineer, a television studio engineer, an encyclopedia editor, a sales representative, an administrative assistant, a game tester, two project managers, and two training coordinators. All interviews were scheduled for one hour in the participant's office, to occur after a period of absence from the computer—first thing in the morning or after a meeting—so there would likely be some new messages waiting. In addition, participants were asked beforehand to refrain from reading new messages for the day prior to the interview.

Part of the interview was conducted as a contextual inquiry, where participants worked with their mail while thinking aloud. For the remainder of the interview the participants were asked a variety of questions, such as how often participants checked their mail, their folder hierarchies, how they handled each message, what they liked and disliked about the email experience, and so on.

Message Archives

We used a tool to collect ten message archives for the second part of our study. Seven of the archives were collected from our interview participants (technical difficulties prevented us from collecting archives from the other three interviewees) and three of the archives were from members of the authors' workgroup. The tool collected all the information in users' email archive including thread structure of messages, folder hierarchy, where messages were filed, whom messages were sent to, etc. The only information that wasn't collected was subject lines and bodies of messages.

Survey of Email Use

The last method we used in our study was a web survey. Based on our interview findings, we developed a survey asking a variety of questions about what makes a message important or unimportant, how people handle messages when they arrive, how people use email as a task planning tool, how people file messages, and how people retrieve older messages. This survey was sent to approximately 1,500 people via general-interest discussion lists, resulting in 406 completed surveys. The majority of the questions were answered using a 5-point Likert scale where 1 = "strongly disagree" and 5 = "strongly agree."

The Participants

It's important to note that we work for a software company, thus our study participants are arguably above average when it comes to technical expertise. At the same time, we were careful to include a wide variety of job roles. In may be that the email culture inside our company is a bellwether, predicting aspects of society in general as dependence on email increases.

All of our participants used Microsoft's Outlook 2000 or Outlook XP as their email client. While Outlook has its particular quirks, it is generally representative of the dominant commercial email clients.

THE EMAIL WORKFLOW MODEL IN DEPTH

This section discusses each of the five activities associated with email in depth. For each we discuss evidence for the existence of the activity, details about how people use Outlook to approach the activity, problems that they are having, and suggestions for user interface enhancements that might mitigate those problems.

Flow Activity

As stated by [8], email has now become a habitat that many of us live in. However, as much as we might like to, we can't live in email all the time. Eventually people have to do other work on their computers, and while they do, they like to keep track of incoming messages as they arrive, an activity we call keeping up with the "flow." This

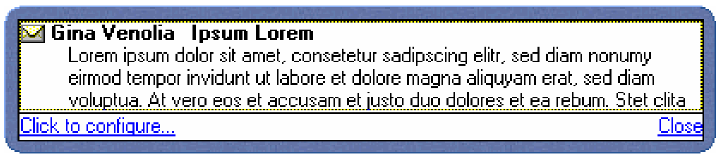

Gina Venolia Ipsum Lorem
Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita

Figure 1: A prototype email notification. This window appeared briefly when a message arrived. Clicking in the window opened the message.

desire to be aware of message arrival was clearly indicated in our survey responses. The median response to the statement, "When I'm at my computer and a message arrives, I immediately look at it" was 4 or "agree" (avg=3.7, sd=0.9).

Unlike the other four activities we discuss, the flow activity is typically a secondary, background activity that is unrelated to the primary task being performed (writing a document, reading a web page, etc.). Thus, when users receive a new message, a series of tasks is triggered revolving around evaluating the message and deciding what action to take.

Outlook provides three methods of being notified of new mail: playing a sound, displaying an icon in the Windows task bar, or briefly changing the mouse pointer. When users are notified of a new message, they have to stop what they're doing, switch to Outlook, and read the message in order to determine if they need to do anything. When finished, they have to remember what they were doing before and switch back to it. This context switching can be very painful. In fact, several of our interview participants said that when they were stressed or deeply involved in a task, they would ignore Outlook when a new message arrived, turn off new mail notifications, or shut down Outlook altogether.

In addition to the simple notification, another way that people use Outlook to support the flow activity is by keeping the inbox visible on the screen. The majority of survey respondents (61%) indicated that they keep Outlook visible two-thirds of the time or more.

By default Outlook generates the same notification for all incoming messages. Some survey respondents indicated that they used rules to give a different sound based on various aspects of the message (the survey didn't ask about this directly, but some used the narrative response fields to describe this customization).

UI to Support the Flow Activity

To support the flow activity, the client should provide enough information to decide an appropriate action. One way to do this is to pop up a window and play a sound when the message is received. Microsoft's Messenger service already does this with new Hotmail messages, but a similar feature doesn't exist for Outlook. Interestingly, when a prototype (unrelated to this project) that provided this feature for Outlook, shown in Figure 1, was distributed within our company, the data indicated that this was one of the most popular features, even though it was a relatively minor part of the prototype [4].

Of course the contents of that pop-up notification window must be designed to provide the appropriate information. Hotmail's notifications display sender and subject line, but as Table 1 shows, people may benefit from seeing additional information so they can decide whether to deal with the message.

Factor	Mean
Reply to my message	4.3
From Manager	4.2
I'm on TO line	4.1
"High Importance" flag	4.1
From project member	4.0
From direct report	3.9
From management chain	3.7
From peer	3.7
Interesting auto-preview	3.6
Interesting subject line	3.6
To fewer than five	3.5
From family member	3.4
I'm on CC line	3.2
From friend	3.2
Important DL on TO line	3.1
From administrator	3.1
To fewer than ten	3.0
From other person in org	2.9
Important DL on CC line	2.6
To more than ten	2.6
From unknown sender	2.1

Table 1: Factors in message importance. Mean responses to survey questions of the form, "A message is particularly important if…"

When the notification appears, the user may read it and choose to act on the message. There may be enough information in the notification window for the user to take decisive action: mark the message as "read," delete it, or initiate composing a reply to it. Or there may be only enough information to warrant opening the message to investigate it further. If no action is taken the message is left "unread" in the inbox, to be dealt with in the triage activity.

There is a constant tension between attending to the primary task and the flow activity. It may be appropriate to suppress the notification under some circumstances. This may be as simple as identifying some email discussion lists as low-priority. Bälter and Sidner [1] suggest a message prioritization scheme for sorting the inbox which could be extended to control which messages generate notification. Horvitz et al [9] suggest a more involved approach where an intelligent agent infers over time what makes a message important to a user and dynamically estimates the interruptability of the user. These factors are used to adjust the salience and timing of notification.

Triage Activity

People often spend blocks of time going through their mail and deciding what to do with all their messages. We call this activity "triage."

Triage can be triggered by several events. First, nearly all our survey respondents indicated that they performed the triage activity on their inbox after being away form their mail for a while. The median response to, "When I get to work in the morning, the first thing I do is check my inbox" was 5 or "strongly agree" (avg=4.8, sd=0.4). The median response to "When I get back from a meeting, the first thing I do is check my inbox" was also 5 (avg=4.7, sd=0.6). Triage may also be triggered by a full inbox (median=4 ("agree"), avg=4.1, sd=1.1) or by the arrival of an important message (median=4, avg=3.9, sd=1.1). Note that performing triage on a single message as soon as it arrives is essentially the "flow" activity discussed in the previous section.

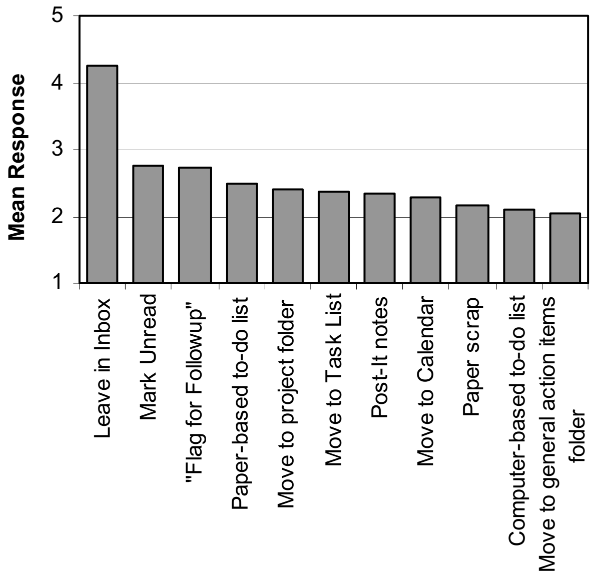


Figure 2: Users task management strategies. Average Likert scale ratings for various mechanisms for turning a message into a task reminder.

In our interviews we observed two dominant strategies for approaching the Triage activity: serial (3 of 10 interviewed participants) or prioritized (7 of 10). Participants who used the serial strategy read messages in the order of arrival, while those who used the prioritized strategy either skipped around picking out interesting senders or subject lines, or used sorting to group messages by sender. The dominance of the prioritized strategy was supported in the survey: The median response to "When I have a lot of mail to read through, I skip around to find important messages" was 5 (avg=4.2, sd=1.0).

We believe two reasons underlie the use of the prioritized strategy. First, people have a greater need to keep aware of things that are important to them and that have potential of greater impact on their life. Second, people may not be able to finish the triage task before they have to attend to some other task, thus people want to deal with the most important messages first.

UI to Support the Triage Activity

Thus, the key UI challenge for the triage activity is providing sufficient, relevant information for identifying important new messages. Bälter and Sidner [1] describe a prototype that divides the inbox into several distinct categories, arranged in rough order of importance by some simple, easily-customized rules. Another important aspect of the design is displaying the message characteristics (as in Table 1) that are associated with important messages.

Another strategy that may be employed in the design is to list conversational threads, rather than individual messages, in the inbox. This serves two purposes: The total number of items in the inbox is reduced, and messages are shown in their conversational context. We discuss a prototype thread-oriented message browsing in a later section.

Task Management Activity

It's clear that people rely heavily on their email clients to help them keep track of what they need to do. Mackay [11] found this, Whittaker and Sidner [17] found this, Ducheneaut and Bellotti [8] found this, and we found this in our study. Six of our 10 interview participants used email messages as their to-do lists, and on our survey, the median response to "I keep messages as reminders for later action when I owe a response" was 4 or "agree" (avg=4.3, sd=0.7). People also kept messages that they needed read later (median=4, avg=4.1, sd=0.8) and messages for which they were expecting a response from someone else (median=4, avg=3.9, sd=1.0).

However, the problem we observed is that there's no single successful method provided by Outlook for handling tasks. Although Outlook provides a separate Task list tool, only three of our interview participants used this feature. Furthermore, on our survey, we asked, "If a message needs action but I can't do it right away, I move it to the Outlook Task list". The median response was 2 or "disagree" (avg=2.4, sd=1.3).

In addition to the Task list, Outlook also provides several low-level methods for handling messages that need future action: leave in inbox, mark as unread, mark with a flag icon for follow-up, move to a specific to-do/project folder, move to calendar, and so on. As shown in Figure 2, by far the most popular strategy is keeping everything in the inbox. This strategy was so prevalent that in our interviews, we even observed the same thing Ducheneaut and Bellotti found: People place non-email related tasks in their inbox by sending themselves mail.

One benefit of keeping tasks in the inbox is that, since the inbox is often visible, pending tasks are visible. Of course, the problem with keeping everything in the inbox is that the inbox can quickly become swamped with messages, making it difficult to figure out what needs to be done. When we asked, "I can easily tell which messages I have kept as reminders," the median response was 3 or "neutral" (avg=3.2, sd=1.3). Whittaker and Sidner [17] also found this problem and made two suggestions for improving the interface to better support the activity of task management: Group messages by thread, and allow people to flag messages such that the system would remind them later about the message. Outlook supports the latter suggestion, but it doesn't appear to be widely used: When we asked, "If a message needs action but I can't do it right away, I use the 'Flag for Follow Up' feature" the median response was 2 or "disagree" (avg=2.7, sd=1.4). It's unclear whether the lukewarm use of this feature is due to inherent limitations or to other UI issues, e.g. the rather cryptic dialog box that appears in response to the "Flag for Follow Up" command. In a later section we discuss Whittaker and Sidner's other suggestion—grouping messages into threads.

UI to Support the Task Management Activity

Whittaker and Sidner identify reasons an old message may be kept in the inbox:

1. "*To dos*," reminders that the user needs to take action;
2. "*To reads*," long documents need to be read at some later time;
3. *Ongoing correspondence* where the user may *owe* or *be owed* a response; and
4. *Messages of indeterminate status*.

We suggest providing *pending flags* in the user interface that map onto these:

1. *Pending—Action required*
2. *Pending—To be read*
3. *Pending—Need to respond* and *Pending—Owed a response*
4. *Pending (unspecified)*

The user needs access to pending messages. Whittaker and Sidner point out that people keep task messages in their inbox simply because it's visible, and that hiding the pending messages in a different view leads to them being "out of sight, out of mind." Thus we propose that threads that contain messages flagged as pending appear clustered in the inbox, below the clusters of active threads.

While the cluster of threads with pending messages may keep reminders accessible and allow simple operations such as scanning for the next task and marking a task as completed, it may not be the correct information design for deeper task management operations. We envision that a separate application would allow in-depth task management, including prioritizing, treeing, clustering, ordering, etc.

Archive Activity

Some messages may have long-term value, so mail users retain them. They often develop rich folder hierarchies to associate related saved messages. We call the activity of marking messages to facilitate later retrieval the "archive" activity.

It's clear that archiving messages is very common. In our survey, when we asked, "I organize saved mail into folders," the median response was 5 or "strongly agree" (avg=4.5, sd=0.7). However, the frequency with which people archived messages varied: According to our survey data, 67% of respondents filed daily or weekly, 23% monthly, and 10% rarely or never, corresponding to frequent filers, spring cleaners, and no filers categories discussed in [17].

Using Folders

Outlook, like most other popular email clients, provides folders as the dominant means of archiving of messages. Previous research has examined users' folder structures, and we did the same. For the 10 people whose archives we studied, the average number of folders was 104 folders (min 11, max 309). These folders were organized in a hierarchy with a typical depth of 2 or 3, but one had 5 levels. Overall, these data indicate that the complexity of folder structures has increased since Whittaker and Sidner's [17] study in 1996 when on average *no filers* had 11 folders, *spring cleaners* had 61 folders, and *frequent filers* had 71 folders. Our numbers are also higher than the numbers reported by Mackay in 1988 [11] where the average number of folders was 33.

Not surprisingly, having so many folders can lead to problems, including folders having too many or too few messages to be useful [17], mail clients enforcing alphabetical ordering of folders, which isn't what users always want [8], having many folders that are no longer useful, and having so many folders that filing often requires scrolling through a long list of folders [11] [17]. Interestingly, users don't perceive a problem finding a place for messages to go. In our survey when we asked, "When filing a message, I know exactly where it should go," the median response was 4 or "agree" (avg=3.9, sd=1.0).

However, just because people know where to file every message doesn't mean that every message belongs in just one folder. Outlook (like most other popular email clients) allows a message to be stored in only one folder. The problem becomes more acute when dealing with entire threads of messages. In our analysis of message archives, we found that 23% of all message threads were spread across two or more folders, mostly because of Outlook's habit of automatically placing one's replies in the "sent items" folder, which guarantees that it's in stored in a different folder than its parent message.

The single-folder-per-message problem is complicated by the fact that the inbox itself is considered a folder. Thus archiving a message—i.e. moving it to a folder—removes it from the inbox. As a result the archive task interferes with triage and task management. This unnecessarily increases the number of times that the user must understand the meaning of a message.

As an aside, we should note that Outlook does allow users to associate multiple category labels with messages, which can be a way around the limitation of only being able to store mail in one folder. However, both the UI to attach and retrieve messages using categories is cumbersome, and they are not used: The median response to "I use the Categories feature in Outlook" was 2 or "disagree" (avg=2.1, sd=1.2).

Mail without Filing

Filing takes time and today's systems allow messages to be marked with only a single label. However, one system developed by researchers at the Compaq SRC labs called Pachyderm [1] introduces a system that solves both of these problems. Pachyderm is based on the notion that there should be no folders (in form of separate storage buckets) and all messages should reside in a single conceptual store. However, users can create folders using standing queries (search commands that are continuously updated). Thus, instead of creating a folder for all mail about project "Gresham", I can create a standing query for all messages sent to the "Gresham" distribution list and all messages containing the word "Gresham" in the subject or body. Users' collections of standing queries can be represented just like the folder hierarchy, with the advantages that no filing is required, and messages can exist in the results from several standing queries. Similar issues have been explored in the document management space by Dourish et al. [7].

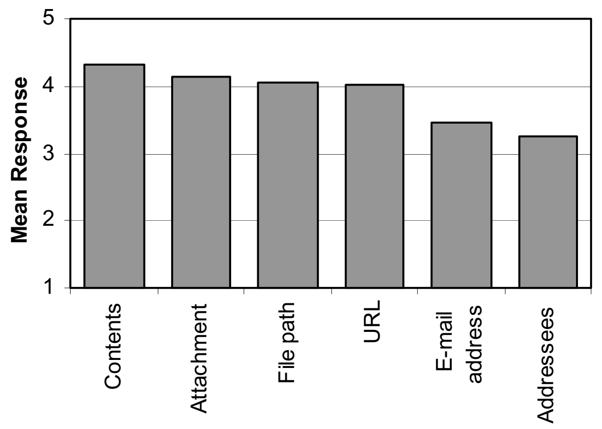


Figure 3: Reasons for filing. Mean responses to survey questions of the form, "I try to keep a message easy to find when I may later want…"

Reasons for Saving Messages

As shown in Figure 3, people archive messages for a variety of reasons. Clearly the information content of the message is important: The median response to "I try to keep a message easy to find when I may want the information it contains later" was 4 or "agree" (avg=4.3, sd=0.7), but we also found that people tend to file messages when they have objects in them that may be of future use (files, file pointers, web links, etc.). The second most popular reason for keeping a message is because of the attachments it contains. Ducheneaut and Bellotti [8] reported similar findings, noting that email is now the main method of exchanging documents.

Outlook provides a search feature that supports finding messages that contain particular words and finding messages that contain (or don't contain) attachments, but does not support searching for file paths or URLs categorically (though text matching can be used if part of the string is known).

Folder Types

There is a vast range of organization schemes that people use for grouping messages. The survey asked about three types of folders: for projects, by discussion list, and by sender. The median response to, "I have folders where I keep mail regarding a particular project," was 5 or "strongly agree" (avg=4.4, sd=0.8). The median response to, "I have folders where I keep mail from particular discussion lists," was 4 or "agree" (avg=4.1, sd=1.1). The median response to, "I have folders where I keep mail from particular people," was 4 (avg=3.6, sd=1.3). Narrative survey responses to, "What other folders to you have for old mail?" indicate that people have folders for a number of other reasons as well. Among the most common are folders for administrative, personal, and technical reference emails. A number of respondents file their old mail by date.

UI to Support the Archive Activity

A user interface to support the archive activity would improve the nature of labels, the process of applying them, and provide tools for managing labels. We propose two changes to the nature of labels. The first and most obvious is to move beyond the rather odd restriction that a message must live in a single folder, no more, no less. Instead, an arbitrary number of labels may be associated with an item. The second proposal addresses the problem of threads becoming spread across multiple folders: Labels should be associated with threads instead of messages. A beneficial side effect of this is that fewer objects need labeling, reducing the amount of work to be done in the archive activity.

There are several ways that the process of applying labels can be improved. The views that support the triage and task management should show active threads and threads with pending messages whether those threads are labeled or not. While some threads will be filed during triage, others will be done in bulk. To support the latter, it should be easy for the user to determine which threads are unlabeled, e.g. via a filter for unlabeled threads. Finally, as demonstrated by Segal and Kephart [14] it may be helpful if the interface were to suggest a few labels that are likely to apply to the thread being considered, e.g. by similarity of contents or participants.

It may be appropriate too to provide some high-level tools for managing labels, such as identifying underused labels that may be candidates for elimination, overused labels that may be split, pairs of labels that are similar and may be merged, etc.

As described in the preceding section, people have folders for by sender, by discussion list, and by date. There is no reason for the user to have the responsibility of filing based on these criteria since it can easily be extracted from the message header. Thus we can shift this burden from the archive activity to the retrieve activity, described next.

Retrieve Activity

People archive messages because they want to be able to recover them later. Thus clearly another main activity people need to do with their email clients is retrieve older messages. Just as with archiving messages, retrieval is a very common activity: When we asked the question, "I never access old messages," the median response was 1 or "strongly disagree" (avg=1.6, sd=0.8).

Clearly the archiving strategy affects the retrieving strategy, as explored by Bälter [1]. If the folder hierarchy is well-formed and well-used, retrieving messages should be easy. When we asked, "When I need to access an old message, I look in one of the folders I've created," the median response was 4 or "agree" (avg=4.4, sd=0.6). Furthermore, people didn't think they had many problems knowing which folder to look in. When we asked, "When I need to access an old message, I know the folder that I filed the message in," the median response was 4 (avg=3.9, sd=0.9). It's doubtful that people are as successful as they claim—during observation in the interviews, some participants had great difficulty when attempting to recall the folder where a message resided.

One surprising aspect of the data was the extent to which people look for messages in their "sent items" folder. When we asked, "When I need to access an old message, I first look in the Sent Items folder", the median response was 4 (avg=3.6, sd=1.1). This was consistent with the strategy of one interview participant who always deleted messages as soon as he responded to them. He figured that if a message was important enough to look for again, he likely had responded to it, thus there would be a copy in the "sent items" folder.

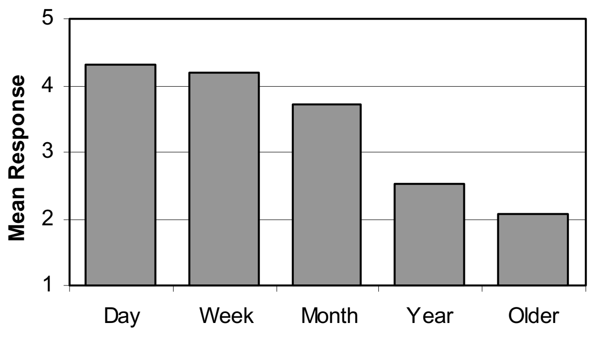


Figure 4: Age of retrieved items. Mean responses to survey questions of the form, "I frequently need to access messages that are a ___ old."

(However, this participant also admitted that he felt comfortable with this strategy because others in his group were extremely good at keeping copies of all important mail, thus he could always ask them for an old message if he couldn't find it. Note that the strategy of depending on others to be good archivers has been found to exist with archives of important paper documents [16].)

Another interesting finding from our survey with respect to retrieval was the age of messages that tended to be retrieved. As Figure 4 shows, survey respondents believed that as a message got older, the chances of them needing to retrieve the message declined. The implication is that day-old messages are still within the first screen of the inbox, thus a visual scan suffices and no search is needed.

People employed a variety of strategies to access their message archives using Outlook. According to survey responses, they use their archive folders (median=4, avg=4.4, sd=0.6) or the Sent Items folder (median=4, avg=3.6, sd=1.1); they sort by sender (median=4, avg=4.1, sd=0.8), date (median=4, avg=3.5, sd=1.1), or subject (median=4, avg=4.5, sd=1.1); or they use the Advanced Find dialog (median=4, avg=3.5, sd=1.3). A number of narrative responses to, "What other methods do you use to access old messages?" mentioned searching or sorting for messages with attachments. Sorting by sender or subject must be considered a crude form of filtering, since the user must scroll to find the sender or subject of interest. The fact that people use sorting instead of searching may be explained by the huge difference in system response time between the two operations.

UI to Support the Retrieve Activity

To support the retrieve activity the user interface should provide facile, quick tools for refining a query. Important filters include label, sender, discussion list, date, and attachments. In addition to these filters, full-text searches should be provided.

As noted above, people may save a message for the URL, file path, or email address it contains. These items should be programmatically recognized. Retrieving for one of these nuggets of information may be facilitated in two ways. First, filters can be provided to match only messages containing the nuggets. Second, the nuggets within the query results may be displayed in a separate pane.

Finally, we noted above that a message may be meaningful only in the context of the thread that contains it. Viewing a particular message from the result set should present it as part of its thread.

CONVERSATIONAL THREADS

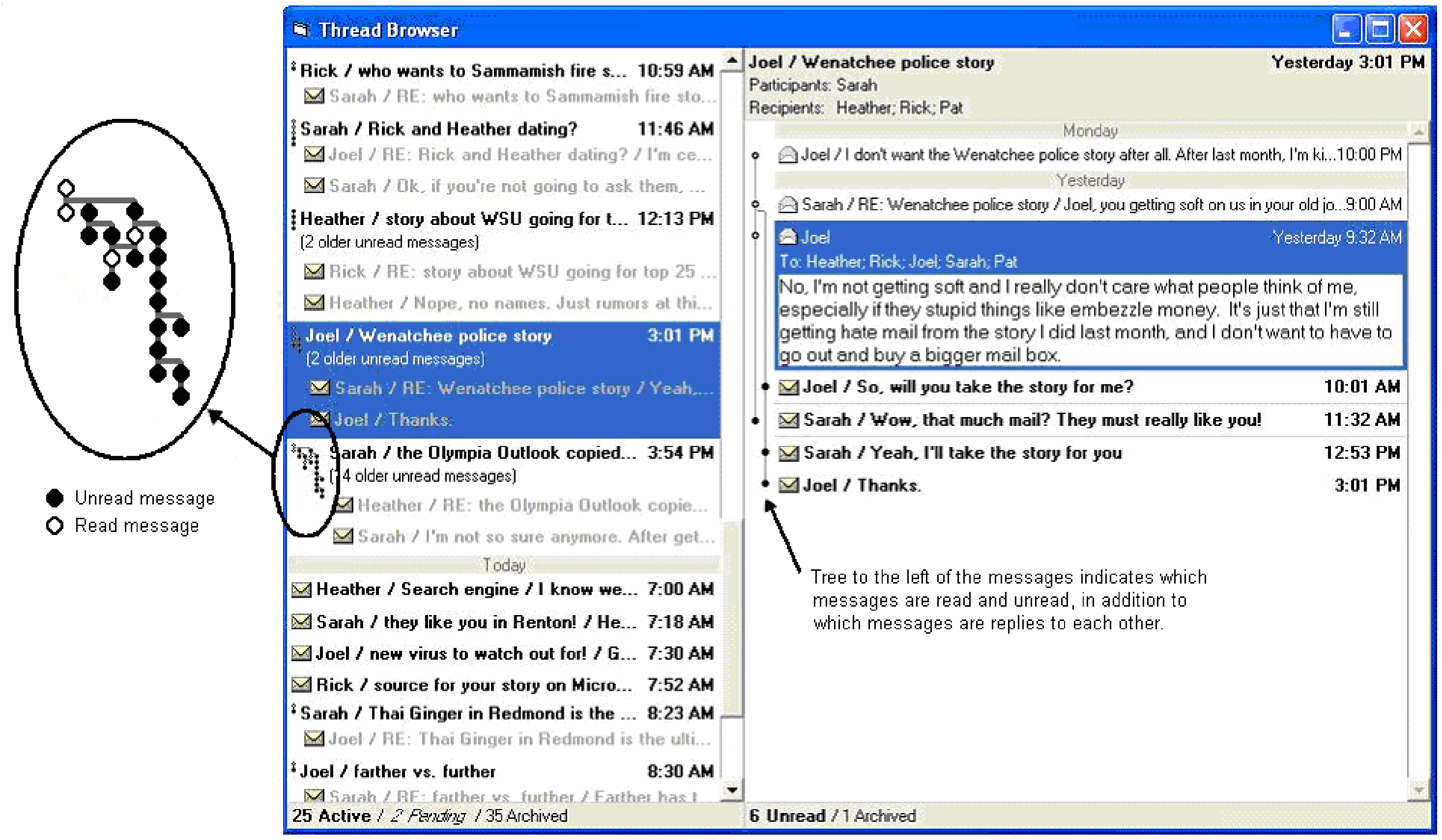
At several points in the previous section, we noted that grouping messages together that were part of the same reply tree would help alleviate some of the pain experienced by the users of email. This recurrent theme of threading deserves more discussion.

We believe providing a threaded email client has the potential to help users in three main ways. First, displaying a message along with all the replies above and below it in the chain provides *better local context*, which can help users better understand conversations that occur via email. Although this context is somewhat preserved by current email programs when they automatically include the text of all previous messages in replies, this method breaks down when multiple people reply to the same message, creating a complex, branching reply tree. Subsequent replies are another important part of a message's local context.

Second, by making the thread the main unit of display, more items can be shown at the same time, providing *greater global context*. As noted in previous sections, users' strategies often depend on how many messages they can view at once in the inbox. Thus, by collecting messages into threads, sets of messages that normally would have been displayed on several lines can be displayed on just one line, allowing people to view more items at once.

Third, when users work primarily with threads instead of individual messages, the interface can provide valuable *global operations*. Currently, if I receive five messages that are all part of the same thread, I have to perform five sets mouse and keyboard actions to work with all the messages (read, file, delete, etc.). However, if all the messages are grouped together, I only have to perform one set of mouse and keyboard actions. While this may seem like a small benefit, multiplied over the large number of email messages, the benefit translates to a significant saving. In our analysis of user's email archives, we found that 54% (sd=26%) of messages occur in threads of two or more messages (although this may be an underestimate given that people likely delete messages that are parts of threads). In addition, higher-level operations are also possible. For instance, if you start receiving messages on a topic that you're really not interested in, you could "unsubscribe" from the thread such that all current messages would be deleted, as well as all future messages on the same topic.

A thread-based interface is only as good as the underlying data that relates individual messages into threads. In the best of circumstances a field is provided in the message header that identifies the message being replied to. For example, the Standard for the Format of ARPA Internet Text Messages [5] defines such field, IN-REPLY-TO. Such a field may be used to construct the thread trees over a body of messages. To increase robustness, we can combine this kind of field with other information, such as matches in subject line, quoting of the original message, and similarity of addressing.



Thread Browser

Rick / who wants to Sammamish fire s... 10:59 AM
Sarah / RE: who wants to Sammamish fire sto...
Sarah / Rick and Heather dating? 11:46 AM
Joel / RE: Rick and Heather dating? / I'm ce...
Sarah / Ok, if you're not going to ask them, ...
Heather / story about WSU going for t... 12:13 PM
(2 older unread messages)
Rick / RE: story about WSU going for top 25 ...
Heather / Nope, no names. Just rumors at thi...
Joel / Wenatchee police story 3:01 PM
(2 older unread messages)
Sarah / RE: Wenatchee police story / Yeah,...
Joel / Thanks.
Sarah / the Olympia Outlook copied... 3:54 PM
(4 older unread messages)
Heather / RE: the Olympia Outlook copie...
Sarah / I'm not so sure anymore. After get...
Today
Heather / Search engine / I know we... 7:00 AM
Sarah / they like you in Renton! / He... 7:18 AM
Joel / new virus to watch out for! / G... 7:30 AM
Rick / source for your story on Micro... 7:52 AM
Sarah / Thai Ginger in Redmond is the ... 8:23 AM
Joel / RE: Thai Ginger in Redmond is the ulti...
Joel / farther vs. further 8:30 AM
Sarah / RE: farther vs. further / Farther has t...
25 Active / 2 Pending / 35 Archived

Joel / Wenatchee police story Yesterday 3:01 PM
Participants: Sarah
Recipients: Heather; Rick; Pat
Monday
Joel / I don't want the Wenatchee police story after all. After last month, I'm ki... 10:00 PM
Yesterday
Sarah / RE: Wenatchee police story / Joel, you getting soft on us in your old jo... 9:00 AM
Joel Yesterday 9:32 AM
To: Heather; Rick; Joel; Sarah; Pat
No, I'm not getting soft and I really don't care what people think of me, especially if they stupid things like embezzle money. It's just that I'm still getting hate mail from the story I did last month, and I don't want to have to go out and buy a bigger mail box.
Joel / So, will you take the story for me? 10:01 AM
Sarah / Wow, that much mail? They must really like you! 11:32 AM
Sarah / Yeah, I'll take the story for you 12:53 PM
Joel / Thanks. 3:01 PM
6 Unread / 1 Archived

Tree to the left of the messages indicates which messages are read and unread, in addition to which messages are replies to each other.

Figure 5: Thread-based mail browser. Messages that are replies to each other are grouped together into one item and displayed on the left. Clicking on one of these items displays all the messages inside the thread on the right. Messages are displayed in one-line preview format on the right, and clicking the message displays the entire message. In both the left and right panes, a thread tree is displayed to help the user determine what the structure of the thread is and how the messages relate to each other.

A SKETCH OF THE CLIENT USER INTERFACE

In the preceding sections we suggested UI improvements to benefit each of the five activities in our model. Many of our suggestions are not novel. We believe that our approach, however, will lead to a significantly improved email client for two reasons. First, we believe that no single technique will be the "silver bullet" that results in the degree of improvement we seek; rather that it is a synergy among the solutions that is required. Second, the activity model provides a structure that guides the user interface design process, so that its result supports tasks in an appropriate organization and at a reasonable level of abstraction. This section sketches out the mail client user interface design that follows from the activity model.

This mail client consists of two components: an application and a notification window. The application window is divided into two panes, like the prototype shown in Figure 5: The left pane shows the list of threads, and selecting a thread displays it in the right pane. Each entry in the thread list shows a number of attributes of the thread: a thumbnail of the thread tree, the sender, the subject, the time of the most recent message, and a line with sender and subject per unread message (up to three). A thread tree thumbnail appears to the left of each thread that contains two or more messages. This compact visual representation gives users a high-level sense of the structure of the thread, as well as which parts haven't yet been read by the user. Among the commands that may be applied to the selected thread are the top three labels suggested for the thread.

The order of the threads in the thread list is controlled so that active threads (those with unread messages) appear at the top, followed by threads with pending (but not unread) messages, followed by recently-active threads. The active threads are clustered and arranged as suggested by Bälter and Sidner [1], so that threads likely to be important appear toward the top. Threads with pending messages are clustered by the type of pending flag (e.g. Pending—Action required, etc.). Finally, the recently-active threads are organized with the most-recent at the top, clustered by day.

A variety of filters may be applied to the thread list: label, sender, discussion list, date, attachments, "important" flag, etc. Multiple filters may be applied simultaneously. A few selections for likely labels, people, and discussion lists are chosen automatically based on the evolving result set and presented as shortcuts to the user.

The right pane, which we call the *thread browser*, consists of a header and a list of messages. The header shows the sender of the first message in the thread, others who have sent messages, others who have received messages in the thread, the title of the original thread message, and the date of the most recent message in the thread. The message list is sorted by message date (oldest to newest), and grouped by day. Each entry in the message list shows an icon indicating its read/unread status, the sender, the subject (if different from the previous), the first few words of the body, and the time. The selected message is expanded inline, showing sender, subject, "to" addressees, "cc" addresses and the complete body. Just to the left of list is a parallel depiction of the messages as a thread tree so that users can see how the individual messages relate to each other.

One design decision to note is the departure from indenting messages to signify replies. Most Usenet browsers display messages in a thread as an indented tree. First, the tree display has a couple of flaws: Deep trees, the typical shape for email conversations, result in substantial indenting, wasting valuable display space. Second, the newest messages are distributed almost randomly through the list of messages, making it difficult to perform the triage activity. Finally, when writing mail, it is not uncommon to refer to *any* prior message, not limited to the ancestors in the tree. The tree display destroys the temporal order, making the complete message context difficult for the reader to understand. We chose instead to sort the message list in the thread browser by date, avoiding all three problems, while retaining information about the reply structure in the tree drawn in the margin. In addition, it allows grouping by day, helping to give a sense of the temporal characteristics of the conversation.

The notification happens in a separate window, much like the one shown in Figure 1, which appears briefly when a message arrives. The contents of the window are much like an entry in the thread list. The user may invoke commands to mark as "read," delete, open or reply to the message. These commands may be invoked by a mouse click in the window or by speech recognition.

The thread list and browser together directly support triage, basic task management, and basic archiving. Using filters, the thread list supports the retrieve activity. The notification window supports the flow activity. Secondary applications, not described here, are provided to support in-depth task management and archiving.

TESTING A THREADED EMAIL CLIENT

In the previous section, we described how a thread list and thread browser combined to form the bulk of the email client application user interface. In this section we discuss an early prototype of an interface we developed to facilitate the use of threads, along with results from a lab study that tested this interface.

To explore benefits of a threaded email client, we built an early prototype using Visual Basic. To begin, we wanted the prototype to support just one of the five email activities. We chose the triage activity: We wanted to see how a threaded email client could help people process a very large amount of unread messages. This prototype is shown in Figure 5.

This prototype was a subset of the user interface described in in the previous section. The most notable difference is that the prototype's thread list is categorized only by day. Also, there are no commands in the prototype for labeling threads or filtering the thread list.

To test the prototype, we recruited sixteen participants who had used email for their job for at least 6 months and received at least 15 messages on a typical work day. Participants were asked to pretend that they were a journalist who had just returned from vacation. Their goal was to go through 200 email messages that had accumulated and enter all the tasks they had to do in a spreadsheet (the email messages were generated by the experimenters). Participants were given 25 minutes to complete the task. Half the participants were randomly assigned to use the thread interface while the other half used the same interface with threading turned off (the left pane of the interface shown in Figure 5 just showed the list of all 200 messages, and clicking on a message displayed it in the right pane).

In a post-test questionnaire, participants responded to a number of questions on a 5-point Likert scale, where "strongly disagree" was 1 and "strongly agree" was 5. For the question, "I didn't like using this email program to read the messages," the median response of subjects who used the message prototype was 4 or "agree" (avg=3.6, sd=0.9) while the median response of those who used the thread prototype was 2 or "disagree" (avg=2.3, sd= 0.5). Analysis by a Mann-Whitney U test found this difference to be significant (z=−2.8; p=0.007), thus the thread prototype was preferred.

Users who used the threaded interface also commented that the threads helped them perform their task better. One participant wrote, "All messages referring to one idea were grouped together. Made it easy to read & refer back." Another participant wrote, "I could easily see if something was resolved before I spent time on it myself."

CONCLUDING REMARKS

In this paper we have identified five major activities surrounding how people use email. In particular, we've highlighted two activities—keeping up with the flow of incoming messages, and triaging existing messages—that we believe are important, but haven't been widely covered by previous studies. For each activity we have discussed the mismatch between user needs and what one commercial mail client interface supports, how the problems have changed (or not) during the past decade, and possible solution directions. It is quite amazing how the majority of problems have remained unchanged and unaddressed. Finally, we've presented an early prototype of a thread-based email client, as well as results from a lab study evaluation. The results demonstrate clear benefits for a thread-oriented display for the triage activity.

Our investigation and discussion has centered on a particular email client: Microsoft Outlook. We believe that our results are broadly applicable for two reasons. First, Outlook is typical among mail clients in its design and the services that it offers. Second and more importantly, we believe the email activity model transcends the specific client. Other clients may have features that address the user needs implicit in the email activity model to a greater or lesser extent, but none to date directly and fully addresses those needs.

REFERENCES

1. Bälter, O. (2000). Keystroke Level Analysis of Email Message Organization. *Proceedings of the ACM Conference on Human Factors in Computing Systems* (CHI 2000).
2. Bälter, O. and Sidner, C. (2001). Bifrost Inbox Organizer: Giving Users Control Over the Inbox. *NADA Technical Report* TRITA-NA-P0101. Royal Institute of Technology, Stockholm, Sweden.
3. Birrell, A., Wobber, E., Schroeder, M., and Perl, S. Pachyderm, https://pachyderm.pa-x.dec.com/. Personal communication from Andrew Birrell.
4. Cadiz, J., Venolia, G., Jancke, G., and Gupta, A. (2001). Sideshow: Providing Peripheral Awareness of Important Information. *Microsoft Research Tech Report* MSR-TR-2001-83.
5. Crocker, D. (1982). RFC822: Standard for ARPA Internet Text Messages. http://www.w3.org/Protocols/rfc822/.
6. Denning, P. (1982). Electronic Junk. *Communications of the ACM*, 25(3), March 1982.
7. Dourish, p., Edwards, K., LaMarca, A., and Salisbury, M. (1999). Presto: An Experimental Architecture for Fluid Interactive Documents Spaces. *ACM Transactions on Computer-Human Interaction*, 6(2).
8. Ducheneaut, N. and Bellotti, V. (2001). Email as Habitat. *Interactions*, September/October 2001.
9. Horvitz, E., Jacobs, A., and Hovel, D. Attention-Sensitive Alerting. (1999). *Proceedings of the Conference on Uncertainty and Artificial Intelligence* (UAI '99).
10. Levitt, M. (2000). Email Usage Forecast and Analysis, 2000-2005. *International Data Corporation*, IDC Report #23011, September 2000.
11. Mackay, W. (1988). Diversity in the Use of Electronic Mail: A Preliminary Inquiry. *ACM Transactions on Office Information Systems*, 6(4), October 1988.
12. Malone, T., Grant, K., Turbak, F., Brobst, S., and Cohen, M. (1987) Intelligent Information-Sharing Systems. *Communications of the ACM*, 30(5), May 1987.
13. Schwartz, J. (2001). Email overload taxes workers and companies. *USA Today*, June 26, 2001, pg A-1.
14. Segal, R. and Kephart, J. (1999). Mailcat: an intelligent assistant for organizing email. In *Proc. 3rd Int. Conf. on Autonomous Agents* (Agents 99).
15. Sproull, L. and Kiesler, S. (1991). Connections: New Ways of Working in the Networked Organization. MIT Press: Cambridge, Massachusetts.
16. Whittaker, S. and Hirschberg, J. (2001). The Character, Value, and Management of Personal Paper Archives. *ACM Transactions on Computer-Human Interaction*, 8(2), June 2002.
17. Whittaker, S. and Sidner, C. (1996). Email Overload: Exploring Personal Information Management of Email. *Proceedings of the ACM Conference on Human Factors in Computing Systems* (CHI 1996