

Taking Email to Task: The Design and Evaluation of a Task Management Centered Email Tool

Victoria Bellotti, Nicolas Ducheneaut, Mark Howard, Ian Smith

Palo Alto Research Center

3333 Coyote Hill Road

Palo Alto, CA 94706 USA

+1 650 812 4000

{bellotti, nicolas, mahoward, iansmith}@parc.com

ABSTRACT

Email has come to play a central role in task management, yet email tool features have remained relatively static in recent years, lagging behind users' evolving practices. The Taskmaster system narrows this gap by recasting email as task management and embedding task-centric resources directly in the client. In this paper, we describe the field research that inspired Taskmaster and the principles behind its design. We then describe how user studies conducted with "live" email data over a two-week period revealed the value of a task-centric approach to email system design and its potential benefits for overloaded users.

Keywords

Email, task management, user studies, system evaluation

INTRODUCTION

An increasing body of literature points to the importance of email as a task management resource. Mackay [14] detailed how it supports a variety of time and task management activities. Whittaker and Sidner [21] extended her findings to show how the email inbox is a repository of "to-dos", "to-reads" items of "indeterminate status" and "ongoing correspondence" that can be difficult to deal with. More recently, we discussed how email is transforming into a "habitat", the central place from which work is received, managed, and delegated in organizations [9].

In spite of the fact that users' have co-opted this flexible application as a critical task management resource, the fundamental messaging metaphor of most clients is not optimized for task management. Email users clearly feel overwhelmed and daunted by the time it takes to deal with all the work coming in through this medium [13, 21].

Acknowledging this problem, a few recent projects have tried to overhaul email's interface. Several have diminished the user's cognitive burden by automating aspects of the filing and organizing of the myriad incoming messages [6, 16, 19]. But this only addresses the problem of managing the *volume* of email. Others have explored different

avenues. Among these, CAFE [20] is probably the most ambitious. Based on studies of email users' behavior, CAFE offers three modes (busy, cool and curious) to satisfy *varying needs in different situations*. Cadiz et al.'s prototype [7] focuses on keeping up with the flow of messages as they arrive and the activity of *email triage* on a body of new messages. In a similar vein Rohall et al. [18] describe visualization techniques to quickly identify *dependencies* between messages in a conversation. Finally, Gwidzka's work [12] focuses on the management of pending tasks and the process of *reminding* to act on them.

This body of research is, however, only in its infancy, with each prototype only addressing one facet of the problem of task management in email. The situation is similar with commercial software touting personal information management (PIM) features, such as Microsoft Outlook, which aggregate without integrating diverse components of task management. The task list in Outlook is, for instance, disconnected from the inbox, even though messages and to-dos are often indistinguishable [9, 21]. Outgoing messages, frequently conveying important task-related information (such as outstanding actions for others) are hidden, out of context, in the outbox. And time sensitive information relating to message content is restricted to the calendar.

Based on our studies of email use, we built Taskmaster, an email system entirely redesigned for task and project management. Taskmaster offers a new solution to the often-decried "pain of email" [7] by recognizing upfront that this technology is not simply concerned with messaging, but that *dealing with email and managing tasks and projects are indistinguishable* [14, 21]. We accomplish this goal purely through a redesign of email's user experience without changing its fundamental technical infrastructure.

In this paper we first summarize our investigations regarding task management in email. Based on these studies, we identify the major facets of email task management that require support. We then describe our implementation of such a system. Finally, we present the results of an extended evaluation, during which nine users managed their email with our prototype.

STUDIES OF TASK MANAGEMENT IN EMAIL

Based on our previous work [9, 10], and that of others cited above, we hypothesized that much of email's complexity (and overload) depends on the nature of the task management activities it is used to support. So we devised

DRAFT, PLEASE DO NOT CITE OR DISTRIBUTE.

THIS SPACE BLANK

FOR ACM COPYRIGHT NOTICE.

a multi-phase study of Outlook™ and Eudora™ users to understand this better. We recruited 11 people, though not all could agree to all the phases (and subsequent prototype use; see table 1.). The phases were as follows:

Phase 1. For 7 participants, we installed filters on all incoming and outgoing email. Participants were permitted to delete private content from the folders where the filters stored data (participants reported deleting about 10% of the messages). Two to three weeks worth of data in the middle of the time period in which the filters were installed was analyzed *by hand, message-by-message* to see who was being emailed about what and how messages related to each other, in order to determine how tasks and collaboration took place in email. So we focused on the following:

- *Addressing*: Whether each message was individual, multiple or list addressed, implying its likely import in terms of thread tracking (as argued in [2]).
- *Threads*: Series of topically related messages and replies (unlike [21] we analyzed content as well as subject lines and thus found a greater average density of threaded messages; 46.9%). Threads on distribution lists were only included if the participant contributed to them.

Phase 2. We video recorded 20 to 60 minutes of email ‘triage’ (this is the practice of handling incoming email and related to-dos and organization [7]). We arranged to be in the participant’s office at a normal time for this activity to occur. If it lasted less than 20 minutes, we returned for a second session. Participants were asked to describe what they were doing, but to try to work as normal.

Phase 3. These observations were shortly followed by interviews, covering the participant’s role and use of email. We asked open-ended questions (e.g., “what is your biggest organizational headache in email?”) and also used 5-point Likert scales to get participants to rate how much they did certain things or used certain features of their email client.

Phase 4. At the end of our filtered data collection period, we conducted follow-up interviews to complement and verify our analysis for accuracy of interpretation of the filtered message activity (we were close to 100% accurate).

FINDINGS: TASK MANAGEMENT IN EMAIL

Managing Tasks in Email: Quantity and To-Dos

Our results from phases 1-4 (see [5] for details) focusing on task activity in email, show that some tasks require only a simple *rapid response* taking just a few seconds (similar to the one-touch model described in [21]). In other cases, a response might be interrupted or delayed (for minutes, hours, even days) while one takes time to gather information (e.g. from the Web, or documents) to complete the task – we call these *extended-response* tasks. Here, it is the *volume* of email and *to-dos for oneself* that cause trouble to email users, not its structure.

Managing Tasks in Email: Complexity and Others’ To-Dos

A significant number of tasks in email are more complex rapid- or extended-response tasks. These are *interdependent* tasks; tasks with obligations that also depend upon the *to-dos of others*. These are characterized by complex threads of email (and other communication such as phone calls)

and are often associated with delays of anything from a few hours to weeks, waiting for responses that enable progress.

Role	Study Phase				Tool Use
Attorney (A)	-	2	3	-	-
Legal Secretary (LS)	-	2	3	-	-
Business Development (BD1)	-	2	3	-	2 days
Public Relations (PR)	1	2	3	4	1 week
Group Manager 1 (GM1)	1	2	3	4	-
Group Manager 2 (GM2)	1	2	3	4	-
Group Manager 3 (GM3)	-	2	3	-	Ongoing
Patent Process Manager (PPM)	1	2	3	4	-
Researcher 1 (R1)	1	2	3	4	1 day
Researcher 2 (R2)	1	2	3	4	2 weeks
Researcher 3 (R3)	1	2	3	4	Ongoing
Researcher 4 (R4)	-	-	-	-	Ongoing
Researcher 5 (R5)	-	-	-	-	2 weeks
Business Development (BD2)	-	-	-	-	1 week

Table 1. Roles and participation in study phases and prototype use. Due to the heavy commitments required, most participants took part in a subset of the phases.

The factors that seem to relate most to a sense of overload are the *number of threads* one is tracking per day, and the *length of the intervals between messages in those threads*. Multiplying these two factors for each participant gave us a simple metric that corresponded more closely than the number of messages per day with reports of overload. The explanation is simple: if one is keeping track of a thread with large intervals between messages, the last message reminding one about that thread drifts out of sight in the inbox as more email arrives. The more one gets involved in threads like this, the harder it is to keep track of them. This leads to significant amounts of time invested in ‘managing’ task-related content. From analysis of video recordings, we found that our participants spent about 10% of their email work time filing messages they thought they might need for future work (cf [1]) and about 8% scrolling around and inspecting folders to find messages associated with active threads. This problem is compounded by the fact that attachments and links often accompany these messages, and must also be acted upon (by skimming, or reading and then storing somewhere memorable) before any further progress can be made. Most importantly perhaps, these numbers do not factor in the time and organizational resources lost when a to-do has drifted out of sight and has not been acted upon, as well as the deep frustration experienced by email users when this situation presents itself [21].

We identified seven specific problems that participants in our study experience with task management in email:

1. Keeping track of lots of concurrent actions: One’s own to-dos and to-dos one expects from others.
2. Marking things as important or outstanding amongst the less important items.

3. Managing activity extending over time or keeping track of threads of activity and discussions.
4. Managing deadlines and reminders, which may be associated with particular messages or other content.
5. Collating related items (e.g., an extended thread or responses to a survey) and associated files and links.
6. Application switching and window management.
7. Most important, getting a task oriented overview, at a glance, rather than scrolling around inspecting folders.

Most of the solutions discussed in the introduction earlier focus on 5; collation of related incoming items, by topic or by thread. But this collation approach alone leaves the problems of interdependent task management unaddressed. So, in response to our findings, we began designing the Taskmaster system, using the eXtreme Programming (XP) approach [3, 4] to integrate fieldwork findings with design.

Taskmaster goes much further than previous efforts to address the seven problems above by repositioning email as task management, providing resources to reduce the time consuming work of overloaded multitaskers. In 2001 Gartner estimated that business users spend about 49 minutes each day processing their email. We felt that a tool that ameliorated problems 5, 6 and 7 could save users some of this time at the very least, and, by improving task tracking, would also address problems 1-4 and the indirect costs stemming from failing on one's email commitments.

THE TASKMASTER DESIGN PHILOSOPHY

A number of principles combine to distinguish Taskmaster from an ordinary mail tool. These are described below.

Thrasks: Threaded Task-Centric Collections

The first principle is that the main element of interest is *the task*, not the message. Our fieldwork shows that individual messages can represent tasks, but *interdependent* tasks (described above) comprise threads of messages files, links and drafts. So Taskmaster supports semi-automatic collections of these items, which we call 'thrasks.'

In the thrask model, any related incoming messages (replies in a thread, with any attendant files or links) are grouped, together based upon analyzing message data. This saves the effort we observed in problem 5 above, just as collation systems do. However, automatic filing via filtering or categorization can lead to problems: previous studies show users do not trust classifiers [17], and like to see messages before moving them anywhere. Categorization can also defeat the use of the inbox as a primary to-do list [19].

Taskmaster maintains the to-do function of thrasks by keeping them in its main list view (the top pane in figure 1) together with incoming new (non-reply) messages, which appear as single-item thrasks at the bottom of the list, rather like an email tool's inbox. For example, in figure 1, "G4 Tips" at the bottom of the top pane, is a new message that has been made into a single item thrask.

The middle pane displays thrask content (messages, attachments, and links) and the bottom pane displays a preview of the content of individual selected items (in figure 1, a thrask entitled 'CHI2003 Paper' is selected, and

a draft paper within that thrask is being viewed). So thrasks, unlike the folders in most collation systems, remain visible in the main top pane; they are optimized as reminders and repositories for ongoing tasks, in response to our observing that people often use email folders as secondary, activity-centric to-do list collections.

Taskmaster reflects our finding (see problems 1 and 3 above) that, when managing a task, one's own messages are often as important to keep track of as those of others (often representing to-dos for others). So, in a break from the standard email-as-messaging-system model (with in- and out-boxes), *incoming and outgoing* messages are viewed *together*. They appear as new thrasks in the top pane of Taskmaster or are added to existing thrasks (as if the sender were CC'ed on every message). As mentioned above, any message that is not a reply becomes a new thrask.

Meaningful Activities Not Just Message Threads

Since thrasks are intended to correspond to threads of activity, we allow users to rename them as we saw people do with folders (the default is the first message's subject line, but this may not be particularly meaningful).

Our analysis [10] showed that threads of activity in email do not always correspond to straightforward message threads, so we let users fine-tune the contents of a thrask by adding items and thrasks to other thrasks or by moving items or sub-thrasks out. In this way topic-drift in a thread is accommodated (a thrask can be split into multiple thrasks) and technically unrelated threads can be combined. So Taskmaster differs from collation systems that track threads [7] since thrasks go beyond system-defined threads to encompass user-defined task-centric collections.

Drafts in Context

Since we have observed people creating messages slowly in *extended-responses*, Taskmaster permits users to save drafts (the label [Saved Message] in the middle pane of figure 1 indicates a draft). This parallels Outlook's drafts, or use of the task-bar, for extended-response messages. However the advantage in Taskmaster is that drafts can be saved *within the thrask* to which they relate, in the context of the related messages and documents that may need to be accessed in their creation. Classic mail tools put drafts in a separate folder out of context, where they may easily be forgotten.

Equality for All Content

Our second principle (relating to problem 5 above) is that we do not regard messages as always taking precedence over attachments and links as classic mail tools do. Messages are often deleted while an attachment must be filed or a link must be bookmarked. So our fieldwork shows that users need to cut across application boundaries in their work. Thus, further departing from the messaging-system model, thrasks contain not only messages, but also *attachments and links as first-class citizens*. Users can also include items from their desktop or useful links that have never been sent in email. Just as email message content can quickly be previewed, so can the contents of other types of items such as web pages, spreadsheets, presentations, and documents, reducing tiresome application start-up time and window management (problem 6 above).

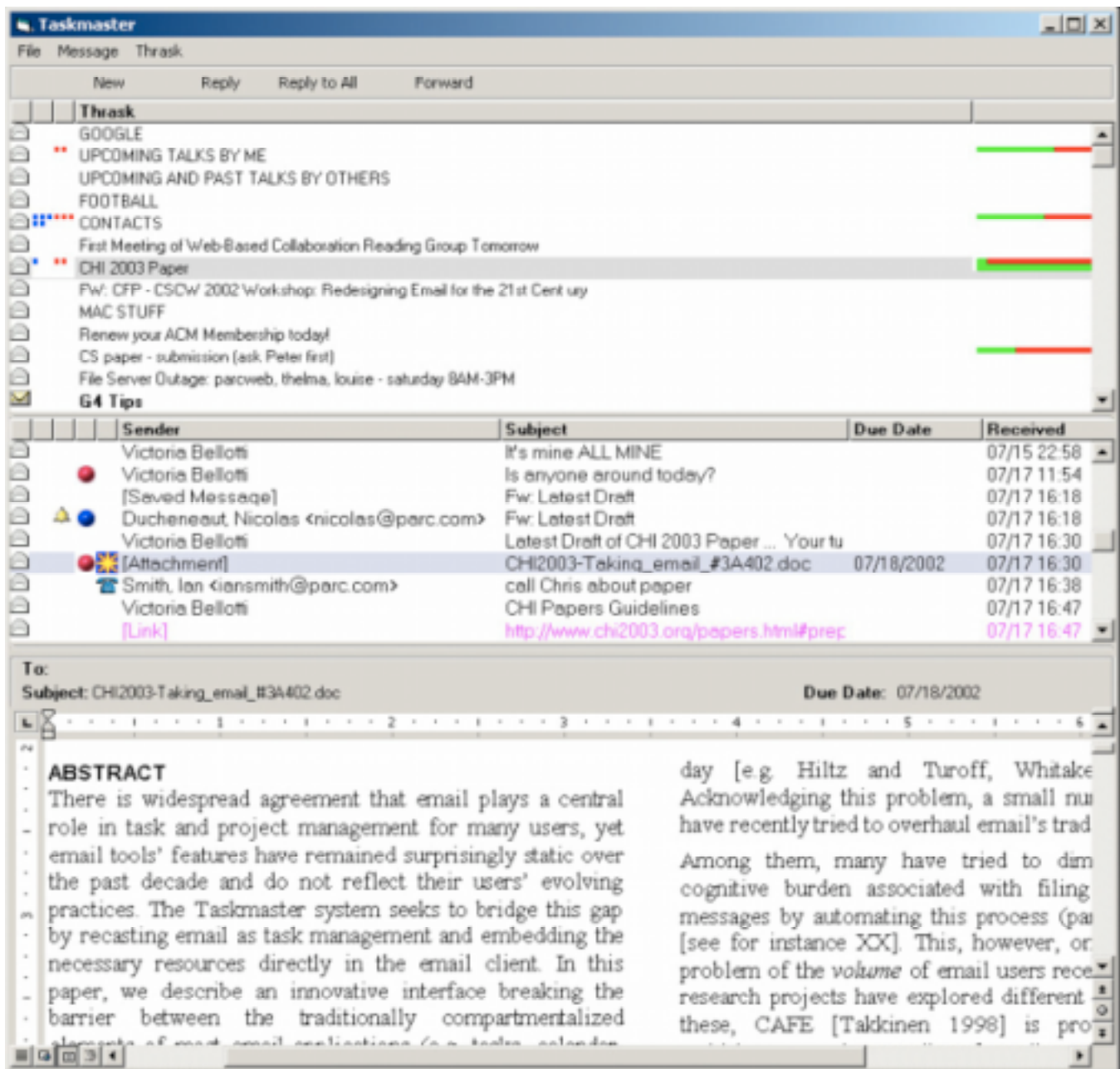


Figure 1. A screenshot of Taskmaster. The top pane is the thrask list viewer; the middle pane is the message and other thrask member items list viewer, and the bottom pane is the content preview.

This feature opens up the intriguing possibility of being able to use Taskmaster as a bookmarking tool for favorite URLs such as our organization's phone list, or Google™.

In this way Taskmaster feels less like a classic application and more like a general task-management environment, handling a variety of types of media.

Task-centric Meta-information for All Items

Our third principle is that any item in Taskmaster can have meta-information such as a deadline, reminder, action or a color code. Even if an item is a document or a link it might still represent a to-do, just as a mail message might.

Taskmaster users can assign meta-information directly to items either from the thrask collection view, or from within an open item. They no longer have to copy information into a separate tasks resource or into their calendar as we have often observed people doing. Deadlines and reminders cause notifications to appear when they are due. Actions are represented as red or blue balls (to represent actions for oneself or another). Color coding or iconic flags (e.g., the star and the telephone in figure 1) can be applied to items to distinguish them. However, these resources are semantically neutral to the application itself; they merely make messages more distinctive to users.

Aggregations of Information for an Overview

The fourth principle in Taskmaster is that thrasks afford an abstraction mechanism for aggregating over a collection to display useful information at the top level (addressing problem 7 above). We implemented three aggregations:

Warning bars. These represent the nearest upcoming deadline and reminder for a given thrask (shown on the right hand side of the top pane in figure 1). Each bar represents two weeks and the proportion of the bar from the left end displayed in green shows what proportion of that time period is left before the deadline or reminder arrives. The rest of the bar is red, so the red part grows larger and the green shrinks as the date approaches.

Action clusters. These represent clusters of actions associated with a thrask and are shown as miniature balls on the left hand side of the top pane in figure 1, next to the thrask they relate to. Adding a new action (and action ball) to an item in the thrask causes a new tiny ball of the corresponding color to appear in the action cluster area.

Task-Specific Contact lists. These are task-centric pop-up lists of the names and email addresses of all senders and recipients associated with items in a thrask. The user can mail all or a subset from a pop-up list attached to a thrask.

We designed these mechanisms to permit users to get, at-a-glance, a sense of their obligations and upcoming deadlines and to be able to contact relevant collaborators without spending time searching through thrasks and inspecting individual items.

IMPLEMENTATION OF TASKMASTER

Taskmaster is implemented in Visual Basic™ as an add-on to an Outlook client configured for the Exchange mail server. It duplicates incoming messages and passes outgoing messages back to Outlook, so all traffic appears in the user's Outlook client in the in- and out-boxes. This strategy provides a prototype with a 'security blanket' backup of all email in the user's Outlook client. In this way we avoided users fearing that transferring to our tool would cause them to lose all their archived email when our study ended. It also meant that should Taskmaster fail to provide any essential advanced features, users would be able to retreat back to Outlook for those features.

EVALUATING EMAIL AS TASK MANAGEMENT

Using the Tool for Real

We used the XP development approach to focus our engineering resources in particular on a prototype that was optimized for evaluation during actual work. Our main concern was not aesthetics (our prototype is admittedly not glamorous), but rather testing the hypothesis that one can treat email more seriously as task management. In particular, the features we wanted to test could not show their value in a laboratory experiment. We needed people to use the prototype in their *real email work*.

In [4] we report on the many obstacles that had to be overcome to accomplish this demanding goal. We used the tool for months ourselves. This helped us refine the design ideas (and get rid of bad ones), and to iron out bugs and usability problems. We then addressed the challenge of

getting people to switch their mail program (a mission-critical application with much legacy data and structure invested in it). This required finding participants who could run the prototype (which depended on a particular configuration of Microsoft Office™ and Outlook). Six participants from the previous phase of our research were able to run the prototype (see table 1). Three more volunteers signed up to make up the numbers (see table 1).

The evaluation involved three phases:

1. Training on all the features of the prototype and how to use it as a task management tool, and giving the participant documentation with guidelines to remind them how to use it.
2. Testing each participant a couple of days later to make sure they understood how to use all the features and instructing them to use the tool to manage all their email work.
3. A final brief video record and transcription of use of and discussion of the tool, followed by a structured debriefing interview in which we asked the participant to assess each of the features and rate how well the tool improved their ability to manage tasks in email using Likert scales and freeform explanations.

Participants were given the option of terminating the study at any time if they found the tool was hindering them.

FINDINGS

Extent of Taskmaster Use

All participants used Taskmaster to manage their email for at least one entire day after the test phase. The reasons for ceasing use of the prototype were as follows:

- R1 had to install a new version of Office for a research project one day after starting to use Taskmaster, which caused the tool to stop working on her machine.
- BD1 experienced so many crashes on her machine that she stopped using Taskmaster after two days.
- PR experienced some addressing problems, which may have been an interaction between her style of use and poor feedback when real names instead of addresses were typed into To: lines. She reluctantly stopped using the tool after about a week although she liked many features.
- BD2 did not commit to using Taskmaster exclusively (he went back to Outlook when he felt overwhelmed by work) and unread messages in Taskmaster built up over a few days, without any user organization, making it too time consuming for him to recover after one week.
- R2 and R5 continued to use Taskmaster for the full two weeks and then returned to Outlook.
- GM3, R3 and R4 are still using Taskmaster at the time of writing this paper, 4 months later. This is in spite of the fact that no technical support has been provided to them at any time since the end of the study.

In general, it was technical limitations of Taskmaster (e.g. sensitivity to variations in PC configurations) and missing features (e.g. printing, feedback about address completion, omitted due to limited engineering time resources) rather

than problems with the design concepts that led to people giving up before the full two weeks. Indeed Taskmaster (built in ten person-months) has only a tiny fraction of the feature enhancements that Outlook boasts. However, even those that gave up using the prototype found the design concepts compelling; those who continued to use it, loved them. The average overall approval rating for Taskmaster was 4.11 (on a Likert scale of 1: Hate it, to 5: Love it).

Experience of Email as Task Management

All or Nothing Use

We found that inability to use Taskmaster exclusively, for whatever reason, limited its success. Email users invest heavily in the features of their client and the structures of folders that they create over time, relying on them to systematize and keep track of work. Using two tools simultaneously that accumulate the same messages (as Taskmaster and Outlook were doing) requires keeping both sets of email up to date; double the investment. BD1 who gave up using the tool (more due to instability of the prototype) summed this point up most clearly:

"I continued to use Outlook for global addressing contacts, calendar those sort of things Not having these things really made writing email cumbersome. [...] I would give it a higher rating, but I had to go back and folder everything in Outlook. [...] I felt I had to do the work twice."

Similarly, PR experienced severe addressing problems and BD2 reported that a lack of key features (printing and formatting) caused him to keep going back to Outlook. For these people, it became difficult to continue with Taskmaster not so much because the concept of email as task management failed, but because Taskmaster lacked standard features they wanted and because switching back and forth to Outlook to use those features was impractical.

Those who were able to get by without certain features and who did not experience serious technical problems were all able to complete the study. From the outset, we had modest expectations about the chances of getting people to use Taskmaster for the full two weeks, given its lack of features compared to what our users were used to with Outlook. So we were gratified when we discovered that three of the five participants who completed the study did not want to switch back in to Outlook despite Taskmaster's limitations. Clearly something was making up for its limited feature set. In the following sections we take each of our design principles in turn and discuss how its embodiment in our prototype was received by our users and how its implementation might be refined.

Threaded Task-Centric Collections

Taskmaster's thrasks were both reliable at collecting threaded messages, attachments and links and a successful means of organizing message content. As expected, preserving the context of messages was useful [7, 11, 15].

R2 "The high level good thing is that I think that it is surprisingly useful to me that when something arrives I actually see it in the context of the conversation...."

Our users rated both organizing and deleting content within the thrask model with an average 4.2 (on a scale of 1: Hate it to 5: Love it). As BD1 put it, *"TM does a generally good job of thrasking, so I've only had to merge it a few times, so it's great."* Users combined thrasks more often than they broke them apart, suggesting that topic drift was not as common as we had anticipated from previous fieldwork [9]. However, people still found splitting thrasks useful, even if they did it infrequently.

Providing the ability to intervene in what would otherwise be a simple message threading function clearly confers an advantage over simple threading. Combining documents and links added to that advantage:

GM3 "It's just nice to be able to have the control over mixing [...] related things together, even though they might not be sort of the identical kind of thing. And I've also taken a bunch of things out of my Outlook [...] that would be applicable to a particular thrask."

Users' Proposed Design Refinements

Some Taskmaster users suggested being able to archive thrasks would have been an advantage for the long term. One user stated that being able to create sub-thrasks would be useful too. It seems that, ideally, thrasks could be blended with folders for archival purposes.

Another refinement, which was more a matter of a missing feature, was to have the sender of the most recent message in a thrask be visible in Taskmaster's top pane. We had previously established in interviews that the sender is the critical determinant to the recipient of whether a message is important and should be made immediately visible.

Equality for All Content

The thrask model was made considerably more powerful by elevating documents and links to be first-class citizens and displaying them in the preview pane when selected in the item list view. In Taskmaster, important documents and links are much easier to get at when they are needed:

PR "I liked that a lot because the difference here [In Outlook] is for example, this is an image but... this image is tiny [and] it's not very important. Now I'm clicking and I'm clicking and it's just... there's so many steps, for just this [unimportant image] and I could have seen that in my email [in Taskmaster]."

GM3 "Last week I had this meeting that had, I don't know, about 4 different responses and about 6 different documents and it was really helpful to be able to have them all in one place. And to be able, during the conference call, to bring up everybody's message [...] and have the attachments [...] it's just really easy to click and not have to open this message and then go down and click here and wait for the application to open..."

Content equality in listing and previewing was rated as a 4 on our Likert scale. However, there was clearly a need to refine it as discussed below.

Users' Proposed Design Refinements

The biggest problem with the content management was that the layout of the three panes of Taskmaster was fixed and

the bottom pane, in particular, was not large enough. It was clear that users would prefer to be able to pick a personal window configuration. Double-clicking on items does open them in their own separate window, which can be sized as desired, but this takes us back to wasting time on window management. We need to find a better layout solution for all three of the Taskmaster view panes.

Another proposed refinement was to provide a fully functional application preview pane so that items could be manipulated without ever having to open an application.

Task-centric Meta-information for All Items

While users liked most of the task-centric meta-information such as deadlines, reminders, action balls, color-coding and iconic flags, these items were not quite as successful as we had hoped (rated between 3 and 4 on our Likert scale). The deadlines, reminders and action balls were the most successful, being used occasionally and rated around 4.

GM3 "When I have a lot of thrasks like this that are mainly just reminders, it's helpful for me to quickly look and go oh I'm the one that needs to do something here. [...] it is also very handy for me to keep track of, OK this is here because I'm waiting for someone else to give me the information."

BD2 "There is a [...] bizdev [thread] where somebody asked a question about market size, and in almost all circumstances I would have forgotten ever to reply to that, but it's still flagged, it's overdue, I put a little red dot also, it means I owe something to somebody."

Users' Proposed Design Refinements

The main problem with our meta-information was that, although users liked the ideas, they found that better integration with information management resources would have helped. For example:

R5 "Can you extract all the messages or the thrasks that are color-coded? [...] It would reiterate this kind of color-coding with the bars."

One user proposed being able to attach meta-information such as actions to entire thrasks and then to provide more details about that information at the item level. Another proposed an idea that we did have but never implemented:

R4 "See if I had something here where I could write a note to myself, what I wanted to do, not just put a reminder but write a note to myself."

We concluded that we were heading in the right direction with these features, but many more enhancements were needed to make meta-information really worthwhile for users to exploit extensively. We believe calendar integration, searching by meta-information and additional aggregations will improve the value of meta-information.

Aggregations of Information for an Overview

The aggregations made meta-information much more useful by providing an overview in the top pane. The deadline and reminder aggregations (warning bars) were especially popular (rated at an average of 4.4 on our Likert scale):

R1 "I liked the visualization bars a lot, that's something for which there is no counterpart to in Outlook. Seeing the growing red, just having an idea at a glance..."

R3 "... they're a visual reminder [...] I could see sort of the slack time that I have, the relation of the green to red gave me an indication of how urgent it was."

PR "...this is visually so there I mean, green or red!!! I just thought it was terrific to walk in, in the morning and to see the change over time and be able to know what was due Monday morning, versus Tuesday morning and to be able to see that transition and go yes I did, yes I did, yes I did, no I didn't [pointing at the bars in P1]."

In most thrasks we saw only one action ball, so the aggregation simply served as a reminder that there was an action associated with a thrask. It may be that the number of actions associated with a thrask is not important as an aggregation. Simply knowing there is some action could be enough, since the user only needs have their attention drawn to the thrask to open it and be reminded of any actions by the message headers and action balls inside.

The task-specific contact lists (pop-up lists of all those involved in any messages in a thrask; rated on average at 3.5 on our scale) were difficult to assess due to implementation problems; quirks in Microsoft's email addressing scheme often led to inclusion of multiple strange addresses for each contact. Some users loved this feature, but others did not:

R2 "... when I saw it I knew it was the right thing but it has the usability thing about the addresses I can't read. And the other thing is sometimes the list has duplicates."

It would have taken too much work to resolve this problem in time for our study, but, despite some users' reservations, the idea itself seems promising given at least some encouraging feedback from our users.

Users' Proposed Design Refinements

Our users showed us that aggregations are useful if they are of the right kind. For example, warning bars gave people the ability to see which thrask contained the most urgent item at a glance. This helped in prioritizing email work. However, the action balls did not convey which actions were most important, so while they gave a sense of to-do-ness they did not help much in planning one's work. One still had to examine the contents of a thrask to find out what to do. Perhaps R4's suggestion of adding notes might have been a better aggregation. One could imagine users adding a couple of words to actionable items, which could be propagated up to the thrask level, like the warning bars.

Some users also pointed out that our task-centric address lists lacked the essential property of being able to distinguish between To:, Cc:, and Bcc: fields which made them much less useful than they could have been.

Summary of Findings

Overall, our evaluation of Taskmaster showed that positioning email as task management is something that users do find compelling. Our prototype suffered from many technical implementation problems and a limited

feature set. In spite of this, the fieldwork-driven design principles; Threaded Task-Centric Collections, Equality for all Content, Task-centric Meta-information for All Items and Aggregations of Information for an Overview were compelling enough to outweigh the limitations for some users who still prefer to use the prototype instead of Outlook. Further, all our users' comments have contributed to a number of ideas for how our design principles can be refined in future implementations.

CONCLUSION

Our research shows that it is possible to significantly and positively affect email users' experience by embedding task management resources directly in the inbox, where they are most needed, as well as breaking down the barriers between the various components of contemporary email applications. The small set of features we have built into our prototype and tested appears to be a strong foundation for a radical (and long overdue) overhaul of email's user interface. It is also a clear indication that life in the email habitat should be rethought not in terms of messaging, but rather in terms of the various activities users are trying to accomplish through that activity. As we noted in our users' comments however, there is much work left to do to perfect this vision. Therefore, we are currently working on pushing some of our concepts further.

ACKNOWLEDGMENTS

We thank all our study participants for their time and cooperation, and Christine Neuwirth, Beki Grinter and other colleagues who contributed to this research.

REFERENCES

1. Bälter, O. (2000). Keystroke level analysis of email message organization. In *Proceedings of CHI 2000, Conference on Human Factors in Computing Systems*, ACM, NY, 105-112.
2. Bälter, O., & Sidner, C. (2000). Bifrost Inbox Organizer: Giving Users Control Over the Inbox. Technical Report #: 00-08: IBM Watson Research Center.
3. Beck, K. (2000). *Extreme Programming Explained*. Addison-Wesley.
4. Bellotti, V., Ducheneaut, N., Howard, M., Neuwirth, C. M., & Smith, I. (2002). Innovation in Extremis: Evolving an Application for the Critical Work of email and Information Management. In *Proceedings of DIS 2002, Designing Interactive Systems*, ACM, NY, 181-192.
5. Bellotti, V., Ducheneaut, N., Howard, M., & Smith, I. (2002). Email-centric Task Management and its Relationship with Overload (Working paper): PARC, Inc.
6. Boone, G. (1998). Concept Features in Re:Agent, an Intelligent Email Agent. In *Proceedings The Second International Conference on Autonomous Agents*. ACM, NY, 141-148.
7. Cadiz, J. J., Dabbish, L., Gupta, A., & Venolia, G. D. (2001). Supporting Email Workflow. MSR-TR-2001-88: Microsoft Research.
8. Denning, P. (1982). Electronic Junk. *Communications of the ACM*, 25(3), 163-165.
9. Ducheneaut, N., & Bellotti, V. (2001a). Email as Habitat: An Exploration of Embedded Personal Information Management. *Interactions*, 8(5), 30-38.
10. Ducheneaut, N., & Bellotti, V. (2001b). A Study of Email Work Activities in Three Organizations (Working paper): PARC, Inc.
11. Eklundh, K., & MacDonald, C. (1994). The Use of Quoting to Preserve Context in Electronic Mail Dialogues. *IEEE Transactions on Professional Communication*, 37(4), 197-202.
12. Gwizdka, J. (2002). Reinventing the Inbox: Supporting Task Management of Pending Tasks in Email. In *Proceedings of CHI 2002 Human Factors in Computing Systems*, ACM, NY, 550-551.
13. Hiltz, R. S., & Turoff, M. (1985). Structuring Computer-Mediated Communication Systems to Avoid Information Overload. *Communications of the ACM*, 28(7), 680-689.
14. Mackay, W. E. (1988). More than Just a Communication System: Diversity in the Use of Electronic Mail. In *Proceedings of CSCW'88, Conference on Computer-Supported Cooperative Work*, ACM NY, 26-28.
15. McDaniel, S. E., Olson, G. M., & Mage, J. C. (1996). Identifying and Analyzing Multiple Threads in Computer-Mediated and Face-to-face Conversations. In *Proceedings of CSCW'96, Conference on Computer-Supported Cooperative Work*, ACM NY, 39-47.
16. Mock, K. (2001). An Experimental Framework for Email Categorization and Management. In *Proceedings of The 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM NY, 392-393.
17. Reder, S., & Schwab, R. (1990). The Temporal Structure of Cooperative Activities. In *Proceedings of CSCW'90, Conference on Computer Supported Cooperative Work*, ACM NY, 303-316.
18. Rohall, S. L., Gruen, D., Moody, P., & Kellerman, S. (2001). Email Visualizations to Aid Communications. In *Proceedings of InfoVis 2001 The IEEE Symposium on Information Visualization*, IEEE, 12-15.
19. Segal, R. B., & Kephart, J. O. (1999). MailCat: An Intelligent Assistant for Organizing E-mail. In *Proceedings of The Third Annual Conference on Autonomous Agents*, ACM NY, 276-282.
20. Takkinen, J., & Shahmehri, N. (1998). CAFE: A Conceptual Model for Managing Information in Electronic Mail. In *Proceedings of HICSS-31, The 31st Hawaii International Conference on System Sciences*, 44-53.
21. Whittaker, S., & Sidner, C. (1996). Email overload: exploring personal information management of email. In *Proceedings of CHI'96, Conference on Human Factors in Computing Systems*, ACM, NY, 276-283.