

# Taskmaster: recasting email as task management

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. During this workshop, we will describe the field research that inspired Taskmaster and the principles behind its design. We will also show 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 [11] detailed how it supports a variety of time and task management activities. Whittaker and Sidner [17] extended her findings to show how the email inbox is a repository of working information containing "to-dos", "to-reads" and items of "indeterminate status" 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 [7]. But despite 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 [10, 17].

Acknowledging this problem, a few recent projects have tried to overhaul email's interface [4, 5, 9, 12, 14, 15, 16]. 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 is, for instance, disconnected from the inbox, while we have found that messages and to-dos are often indistinguishable. Outgoing messages, which frequently convey important task-related information (such as an outstanding action for someone

else) 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" [8] by recognizing upfront that this technology is no longer simply concerned with messaging, but instead that *dealing with email and managing tasks and projects are indistinguishable*. We accomplish this goal purely through a redesign of email's user experience without changing its fundamental technical infrastructure.

## MANAGING TASKS IN EMAIL

Our field studies [3, 7, 8] revealed that a significant number of tasks managed in email are more complex than simple "fire and forget" responses to messages. 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.

It became quickly apparent that 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 person whose email we analyzed [3] 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. 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.

More generally, we identified seven specific problems that participants in our studies 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 importantly, getting a task oriented overview, at a glance, rather than scrolling around and 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 [1, 2] to integrate fieldwork findings with design.

### **THE TASKMASTER DESIGN PHILOSOPHY**

Taskmaster goes much further than previous efforts to address the seven problems above by repositioning email as task management. A number of principles combine to distinguish Taskmaster from an ordinary mail tool.

#### **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 [13], and like to see messages before moving them anywhere. Categorization can also defeat the use of the inbox as a primary to-do list [15].

Taskmaster maintains the to-do function of thrasks by keeping them in its main list view (the top pane in fig. 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 fig. 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 fig. 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 [8] 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 [5] 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. We

have seen messages get deleted while an attached document 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).

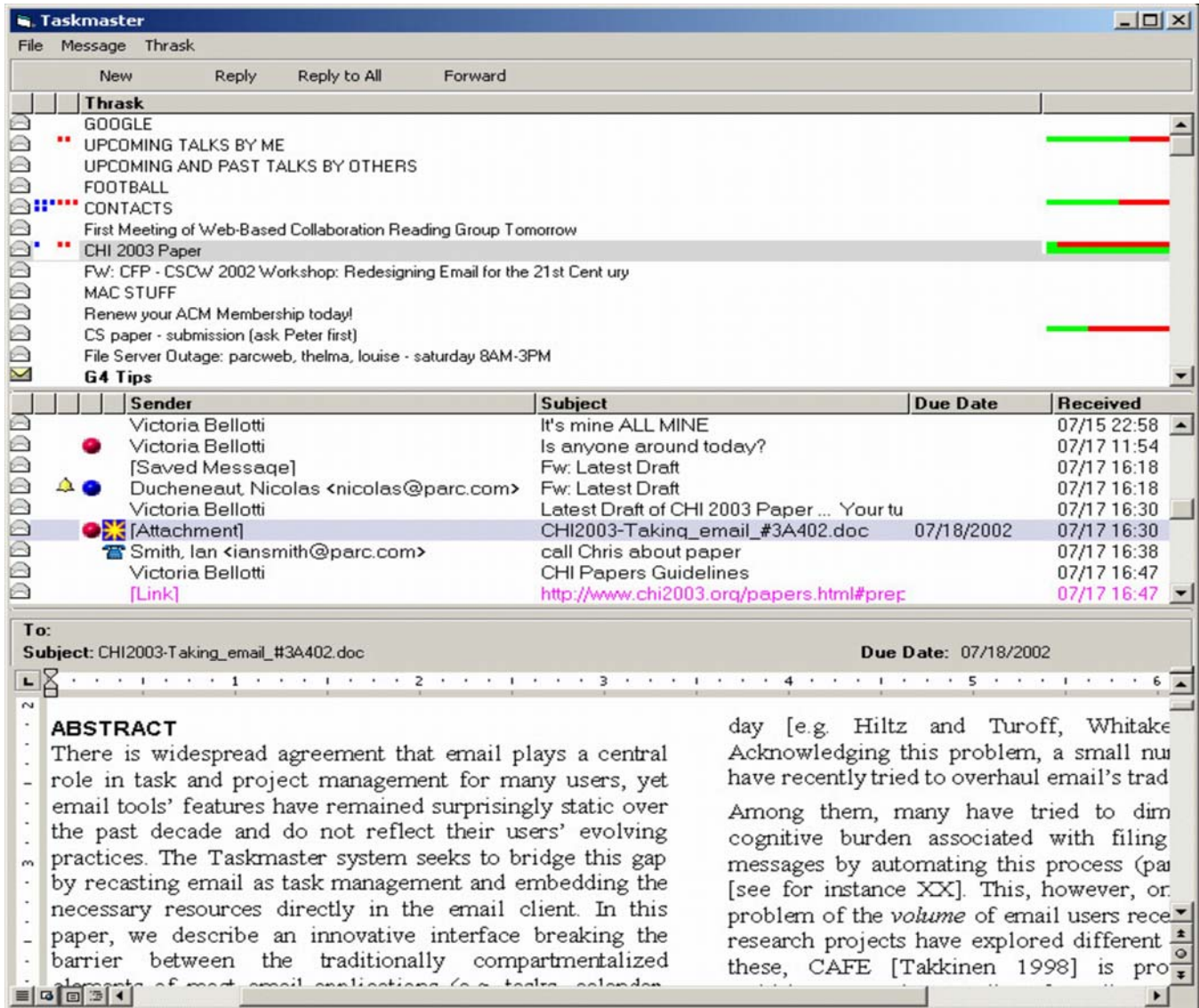


Figure 1. A screenshot of Taskmaster. The top pane is the thrask list viewer; the middle pane is the message and other thrask member items list viewer, and the bottom pane is the content preview.

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.

#### **EVALUATING EMAIL AS TASK MANAGEMENT**

Overall, our evaluation of Taskmaster showed that positioning email as task management is something that users do find compelling. Our prototype suffered from some 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 could be refined in future systems. We plan to discuss our findings in more detail during the workshop.

#### **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.

#### **REFERENCES**

1. Beck, K. (2000). *Extreme Programming Explained*. Addison-Wesley.
2. 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.
3. Bellotti, V., Ducheneaut, N., Howard, M., & Smith, I. (2002). Email-centric Task Management and its Relationship with Overload (Working paper): PARC, Inc.
4. 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.
5. Cadiz, J. J., Dabbish, L., Gupta, A., & Venolia, G. D. (2001). Supporting Email Workflow. MSR-TR-2001-88: Microsoft Research.
6. Denning, P. (1982). Electronic Junk. *Communications of the ACM*, 25(3). 163-165.
7. Ducheneaut, N., & Bellotti, V. (2001a). Email as Habitat: An Exploration of Embedded Personal Information Management. *Interactions*, 8(5), 30-38.
8. Ducheneaut, N., & Bellotti, V. (2001b). A Study of Email Work Activities in Three Organizations (Working paper): PARC, Inc.
9. 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.
10. Hiltz, R. S., & Turoff, M. (1985). Structuring Computer-Mediated Communication Systems to Avoid Information Overload. *Communications of the ACM*, 28(7), 680-689.
11. 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.
12. 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.
13. 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.

14. 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.

15. 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.

16. 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.

17. 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.