PIM Workshop Beyond Email discussion –

Steve Whittaker, Jacek Gwizdka, Tom Erickson, Jonathan Grudin, David Levy.

The discussion was organised around debate of what we hoped were controversial statements about email, each of which have implications for its future.

1. Email is the killer app and should the general focus for PIM

The argument is that email is already a multi-functional application. People use it as a file system, communication manager, todo list, contact manager… So let's acknowledge this, and explicitly integrate these other functions directly into email.

Problems:

(a) **Integrating more functions into email would mean a disaster.** Email isn't that efficient at its 'own' functions anyway. People complain about email overload, as well as not being able to find information, contacts or tasks in their email. So to overload email still further with more applications would be to court disaster.

(b) **People prefer a variety of applications**. Empirical studies suggest that people want to use an ecology of applications, where they focus on the 'core competence' of each of a suite of applications. It follows from this that the best strategy might be to encourage users to migrate some of email's usages to other applications, that are specifically designed for the purpose, e.g. using a dedicated contact manager rather than the email address tool. Actually a more nuanced position might be to provide data level integration of these functions and to allow users access to that data via multiple interfaces.

(c) **Population differences**. Other empirical studies suggest that email isn't the killer app for all populations anyway, and that students for example express a strong preference for using Instant Messenger as their key application. So integration around email clearly wouldn't help the student population. This also implies that there may not be a single killer app for all populations. A conclusion might be drawn. If we follow the killer app integration strategy then the integrating app will differ across populations, making the strategy harder to implement, undermining the unifying nature of the killer app, and possibly the whole integrative strategy.

2. Email is completely broken and we need alternate models for managing information delivered through it

The argument here is that people despair of email, because: they get too much information because of spam and careless broadcasting behaviour, they can't find working information relevant to their current tasks, they find it hard to file email information in such a way that they can find it again. Other studies indicate that email traffic and overload is contributing to work-related stress. Furthermore, these problems can only get worse as the number of email messages sent over the last 3 years has increased 8 fold. Although we noted that email is a legacy application which may now be part of the communication cultural (and hence hard to change) we nevertheless discussed several alternatives to email:

(a) Separating transient versus longer-term communications:

Formatted: Bullets and Numbering

IM plus blog, using IM for communication plus the blog for publishing. It remains to be seen how successful this approach is.

(b) Collaborative project centric information management:
the problem here is the classic workflow problem that not enough of email can be organised into projects to make this a useful unifying organisational principle. In other words, not enough tasks can be organised into collaborative projects and too many messages are singletons that have to be processed in isolation.

This last topic led to two related discussions about task inference and management (3) and workflow (4).

3. Task inference and management is the key to improving email

The argument here is that many of the problems that arise in managing email result from the fact that it's hard to access and organise information relating to the same work task. It follows that if we could infer such tasks then we could make email better within either of above approaches.

(a) several of the group participants were sceptical that we will be able to successfully infer tasks. This has been a classic problem in both HCI and psychology for many years now, and not much progress has been made into task inference. Having said this, much progress has been made recently in areas such as machine learning and text processing, both of which may allow email messages to be analysed in promising new ways.

(b) Again, however, one worry with these techniques is that tasks don't account for enough of email's complexity. In other words even if we could successfully identify a large proportion of email tasks this would still leave a large residue of messages to process that are singletons and not part of any task. However some empirical work might be useful here to determine what proportion of people's email concerns sets of messages related to specific tasks, as opposed to unrelated messages.

4. Email is workflow in disguise (similar to task management)

Most work is collaborative. If we own up to this we could incorporate ideas from workflow, including

(a) better tools to track collaborative tasks

(b) lightweight features that would help people to manage collaborative tasks

(c) but the problem is that this approach has been tried multiple times (e.g. Malone et al., Lotus Notes) with little success. One problem here is that workflow doesn't seem to cover a large enough proportion of users' tasks. Again there seem to be too many messages of other kinds.

5. Redressing the balance between senders and recipients

One major problem for email recipients is their lack of control of the volume of messages that they receive. While spam is a major contributing factor, the issue

applies also to messages from "legitimate" sources. We talked about several approaches to deal with this:

(a) filtering – using AI techniques to build user profiles that would allow intelligent filtering of messages. These techniques could be used to deal both with spam as with non-spam emails. These techniques are improving, but work is still needed to improve the programming interface to these.
(b) We talked about experimenting with other methods to control spam, one might be to charge people for sending messages (pay-to-send). Another idea we talked about was to use reputation systems (or similar techniques) to identify important senders of email, so that their messages might receive precedence (or at least not be deleted).
(c) We also talked about educating people about email sending habits, but we were somewhat sceptical about whether such methods were likely to succeed.
(d) Another thing that we discussed with respect to dealing with non-spam messages, was changing senders' expectations by reducing their expectations that every message will be read. We talked about how if people became used to recipients use of filters that they might start to have decreased expectations that every message they send might be read – which might in turn modify their sending behaviour.
(e) A final approach might be to try and have senders do more work to provide information about messages (e.g. semi-structured messaging). But again part experience suggests that this approach may not work.

6. Searching will solve the email problem

Argument here is that offered by Google's gmail, that the main problem with email is *finding* messages,

(a) but this ignores the fact that many of the problems with email are in deciding what action to take with new incoming messages, and in tracking the status of undischarged messages. Part of the function of the inbox is to serve as a reminder about undischarged messages and it isn't clear that a search only model can address this.

7. Email needs to incorporate 'pull' type components.

Instead of having all information sent directly to users we need to experiment with techniques whereby information that is not directed at a single individual is published at a public location rather than being sent to an individual.

(a) now several tools, blogs, combine blogs with documents stores to address some of the problems of version tracking
(b) problem of deciding what should be published rather than pushed to people
(c) if information is published do we have some form of alerting to tell people where that information is located (otherwise they may not know of its existence)
(d) Problem with alerting is that this may be almost as distracting as the original message
(e) Also users have to know *where* information will be published. If people don't know this then they may not be able to find the information. Even though email may be overloaded at least people know that the information they require is located in their system.