21 September 1995

Andrew

Andrew Consortium
School of Computer Science
Carnegie Mellon

Wishlists: An Agenda for 1996

Fred Hansen
Director, Andrew Consortium

In preparation for the annual meeting, we posted a request for wishlists for 1996. The respondents, included as an Appendix, were unanimous in focusing on electronic mail and the world wide web as areas of concentration. Possibly this means that ez is already meeting the needs of its users.

In the following, all time estimates leave no leeway for additional effort to correct bugs in the remainder of the system.

Web Browser

Tom Neuendorffer has nearly completed his Andrew based web browser, based on Kemmetmueller's HTML viewer. This provides not only a separate browser application, but also the capability of selecting a URL in *any* document and "Warp"ing to view it from with an Andrew application.

Essential additions are mentioned in the wish list from Rochester:

> converters between HTML and Andrew document formats
> widgets to support tables, forms, and other HTML entities

These can be completed in three or four months each.

Widgets

As I will discuss in another paper, we have nearly completed the backbone of a highly extensible widget set. It provides for the creation of new widgets with minimal coding, and the coding can be in either C++ or an extension language.

The "ASlot" mechanism will make it possible to have attributes on widgets that can be accessed/set both with standard C++ function calls and with an extension language. It is even possible to add attributes after compilation of the widget.

Using this technology, we can create new widgets for the crucial cases of PushButton, ToggleButton, RadioButton, TextResponse, Label, and List. Or these can be adapted from existing Andrew widgets. In either case, the implementation of HTML forms seems not so much a question of new widgets, but of parsing the HTML to determine what widgets need be displayed, where to put them, and how to link their functionalities.

Forms and tables for HTML can be completed in four to six months.

Messages

The complexity of messages arises from dealing with a variety of message sources and storage pools:

> i) External mail arrival and departure. Sometimes mail is delivered strung together in a single file (in /usr/spool). In other systems mail is delivered as separate files in a given directory (~/Mailbox).
>
> ii) Newsgroup arrival and departure. The quintessential news source is via "NNTP" from a server. Under the Andrew message delivery system, at CMU, news is stored in a central pool and accessed via files in the a large shared file system. (The Andrew message delivery system is NOT essential to Andrew messages.)
>
> iii) Treating mailing lists as newsgroups. Mailing lists wind up with multiple copies of messages cluttering personal mailboxes all over campus. Instead, CMU subscribes to the list and then users access it as though it were in a newsgroup. Many users do not know the difference.
>
> iv) Storing a user's private mail. There are various storage schemes for a persons mail. Andrew Messages has its own, but this strategy is a problem when users of other mail programs wish to use messages on a trial basis.

This year we added the "batmail" mail reader, written by Miles Bader several years ago. This a package which allows access to Andrew mail from within the GNU "emacs" editor. The emacs portion just displays message bodies which it fetches by conversing with another process called, thematically, "robin". Robin has all the library calls to access mail and news from The Andrew mail delivery system.

The batmail design of separating user interface from delivery system across a process boundary seems to me attractive. By carefully defining the messages allowed to flow across the boundary, a single mail reader can be used with processes designed for several different message sources and storage schemes. The design I have in mind has three processes:

> Mail Reader/Writer - Reads messages from incoming sources and sends messages outbound to those same sources. Separate reader/writer processes would be available for each major delivery system (˜/Mailbox, AMDS, /usr/spool, IMAP, ...).
>
> Newsgroup Reader/Writer - Deals with messages incoming from and outgoing to newsgroups. Versions support various options (AMDS, NNTP, ...)
>
> Storage - Saves, indexes, and regurgitates the user's personal mail. Versions would be available for popular schemes (message's own, MH, /usr/ucb/mail, IMAP, ...).

A "mailstartup" program would scan a user's environment and determine which mail access options are appropriate. Since this will be Greek to most users, the program will have to be fairly careful, but cannot rely on dialog with the user, other than to try to explain what it is doing.

Separating "messages" into separate processes along these lines can be completed in four to six months. This would include design of the server protocol. With luck, this can be adapted from existing standards or the batmail model. (Determining the protocols will take at least half the time.) There would remain the task of building the access servers for protocols not now supported.

Scrolling and Styles

Scrolling has been much improved due to Rob Ryan's efforts. There are still cases where there are large jumps, and Rob has been working on these. For really smooth scrolling, however, a map is needed which relates character position to physical position in the entire document. We have some ideas of data structures that will make this possible.

A vaguely related area is that of styles. We have vastly reduced the space required for styles in documents with embedded texts. However, much more can be done to simplify the system, especially in eliminating the 3-2 tree and introducing style deltas. There would be some small saving in the scrolling effort if a style revision effort were mounted first.

Styles revision will take six to eight months. Scrolling improvements will take three to five months.

Appendix

Date: Wed, 20 Sep 1995 13:00:03 -0500 (CDT)
From: Todd Inglett <tinglett@VNET.IBM.COM>, Wayne D Richardson <wdr+@rchland.ibm.com>, Mark Turner <turner+@rchland.ibm.com>, Robert Kemmetmueller <rskm+@rchland.ibm.com>

Fred,

Here is our current wishlist. We've kept it short to keep it to the point. We have been debating whether or not to work on *messages*, but the concensus here is that you don't have nearly enough resources (1.5 people) to get anything really interesting done with *messages* in a short amount of time. Keep in mind that we now have to deal with *Lotus Notes* because we now own the company. Our team obvously needs to learn more about *Notes* before we proceed head first into a big *messages* upgrade.

Meanwhile, I would say that any work you do find to do with *messages* should be focused on integrating it with the web in various ways. We'll probably do that here, too, and I'll try to keep you up-to-date with any progress we might make.

-todd

Andrew Wish List (Fall, 1995)

This contains a list of priority items that need attention in the Andrew toolkit and those applications shipped with it. There will always be ongoing bug fixes and minor enhancements that are not considered here.

Web Integration

We feel it will be next to impossible for the consortium to implement a browser as good as *netscape* with the limited human resources available. However, ATK could be used to build useful WWW authoring tools from a basic HTML editor to online reviewing. To do this ATK should be able to trivially read/write URLs and display HTML. ATK should also be able to handle HTML forms.

The following must be developed or enhanced:

1. The HTML edit view needs much work. Some can be done here, but textview may need some serious overhauling.

2. Inline images still look awful and print slowly. The color support must be reworked to reduce overall color use, dither colors, and rasterize the image for printing.

3. Widgets must be developed to implement HTML forms.

4. Messages should be able to interpret "text/html" as an inline mime type.[1]

5. We need a way to convert ATK to HTML or other data formats. This includes the ability for *figure* to produce an image of itself. This process must not require an X server because it may be run by the web server as a cgi-bin where an X server is not available.

Widgets

For now we can easily survive with the small widget set that must be developed for HTML. In particular, these are the priorities for a widget set:

1. Implement those widgets needed for HTML 2.0

2. Widget look and feel should be as "standard" as possible. Use the new Motif 2.0 design when in doubt (see http://www.osf.org/motif/Motif20.html).

3. The C++ programming interface for the widgets must be intuitive and simple. Do not use atoms, strings or other weird techniques for setting attributes of widgets. Use real C++ method calls.

4. Interactive "building" of the widgets is not important for widgets. The HTML view will fill this role.

5. Keyboard input focus navigation is not important for widgets. The HTML view will fill this role.

6. End user customization is not important for widgets. The HTML view will fill this role.

---

[1] The whole "Lotus Notes" issue is up for debate here at present so we are *not* asking for any additional enhancements for messages. Keep in mind that we have ˜2000 users of messages, though, and we don’t care to do a major mail folder conversion process unless it is trivial and very well tested.

7. An "extension language" is not important for widgets. If anything we would want to use HotJava for this, but it is not important at this time.

8. Widgets *can* be views. At one time a view was thought to be too heavyweight, but we don't have any evidence of that anymore (i.e. hardware has caught up).

With these requirements it should be fairly straightforward to develop a core set of widgets. Use of the widgets within the HTML edit view will verify that the interface is either easy or clumsy to use by another view. We want it to be easy, of course.

Look and Feel

With our new OS/2 port the look and feel of each application is becoming more important. No other toolkit provides weird menus like ATK's. We need to provide a way to enable a fairly "standard" looking menubar, probably with cascading menus, a standard help menu and checkboxes for toggle items. We also need "standard" popup menus. Note that most toolkits treat the popup menu as a context sensitive menu, and it may be useful for ATK to do the same.

The ATK keyboard input model is also quite outdated. We can live with it during the interim, however, since it is only exposed to the end user through initfiles.

We must be very careful implementing a basic widget set to make sure they can behave like their native counterparts on X, OS/2 and perhaps Win95. We must also be careful in allowing X to creep into the implementation. For example, it may be fine for environ::GetProfile() to read X resources for the X implementation--but this must be done generically so they may be read from INI files under OS/2 (and Win95).

Date: Fri, 8 Sep 1995 17:28:55 -0400 (EDT)
From: Bill Cattey <wdc@MIT.EDU>

(I have intentionally replied to the wide list.)

Thank you, Fred, for soliciting wish lists. I think discussion about the direction of Andrew is a very important part of keeping the product alive and moving in the best possible direction.

I feel that the two most important projects that could be undertaken at this time are:

1. A rewrite of messages.
2. An AUIS Web browser.

I am coming to believe that it does not matter what kind of horrible insides a program has as long as it provides an application that people want. For this reason, I am coming to believe that AUIS can only stay viable if people can get it, and run binaries of applications they want from it.

AUIS allows us to provide absolutely the best mail interface and web interface, because, rather than spawning sub-processes, we just snap a dynamic link for whatever new media type comes along.

Even though we like AUIS because of its beautiful architecture, the only people who actually BUY it are people who have an idea of what to use it for. Most of these folks do NOT think of using it to write an application in.

At the beginning of this message I said that I believed people run applications because of what they do, and it doesn't matter how horrible they are inside. Well, messages is an interesting application of this rule. It has stuff inside that make it very hard for sites to get it running. Many more people would use it at MIT if, when they ran it the first time, it didn't core dump, or try to do a bunch of non-obvious things. Messages may die if it doesn't have its superfluous complexity cut away, so that the beautiful mail interface that it has, can function well enough for wide deployment.

Because of my love for messages, I propose that fixing it be done first.

But pragmatically speaking, a powerful (but internally simple) AUIS web browser is probably the most important thing to work on. (Doesn't Tom Neuendorffer have one already under development?)

Thanks, again, for asking.

-wdc

Date: Sat, 9 Sep 1995 21:35:25 PDT
From: Bill Janssen <janssen@parc.xerox.com>

Ah, Bill, you pushed my messages button...

Messages should be completely re-written to be a standard SMTP mail tool. Its current structure, unfortunately, is obnoxiously and needlessly idiosyncratic. In particular,

-- It should use MH-style mail directories, with the indices in additional files. The AUIS v5 mail directories were *almost* compatible, but had just enough pointless differences to sink it.

-- It should use some standard extension language to process mail filter scripts; Scheme (at least a dozen good small free Scheme interpreters are available) or Python would be my pick, but *anything*, just so long as it isn't some randomly bug-full and incomplete implementation of a standard language (FLAMES) or some elegant special language based on wierd abstractions that no one else in the entire world uses (NESS -- Sorry, Fred). I finally scrapped FLAMES filtering and wrote my own filtering program.

-- CUI and VUI suffer from almost willfully user-unfriendly UI design. Scrap them and let people use MH when on a dumb terminal.

Just my two cents as a 6-year messages user...

Bill

Date: Mon, 11 Sep 1995 11:09:27 -0400 (EDT)
From: Bill Schell <bill@allegra.att.com>

I agree with Bill Cattey's Andrew wish-list 100%. Fix up messages first and then do a web browser. Messages is pretty much the dominant (only at most sites) AUIS application used here at Bell Labs. We're seeing people switch to other MIME mailers (Zmail, etc.) that aren't really as nice due to rough edges in messages. For example, in Zmail, most media other than text show up as clunky attachments, rather than in-line.

An AUIS based web browser is also a great idea. Being able to add new media types that show up as part of your document, rather than in a pop up window managed by an external viewer process would be a big win. Netscape is trying to do this via OLE on PC's, but OLE is arbitrarily baroque and hard to program with.

Bill Schell
AT&T Bell Labs, Murray Hill, NJ
bill@allegra.att.com

20 September 1995
Victor Lanier, BellSouth
(by phone)

Scrolling is much improved. It could benefit from additional improvements in smoothness, avoiding the large jumps that currently interrupt smooth scrolling as insets are passed over.