Architectural Foundations of PINE: A Multimedia Email and News Client for DOS and Unix

Terence E. Gray[1]

Abstract

Pine is a character-based mail and news program emphasizing ease-of-use. Other key design goals were: full Internet mail interoperability including support for multipart, multimedia MIME messages, support for a robust client-server email architecture using the Interactive Mail Access Protocol (IMAP), and integration of mail and network news. A PC version of Pine was developed to improve integration of the mail system and the desktop file system for DOS users. Pine(tm) is copyrighted but freely available, and is in use at more than 2000 sites in over 30 countries. The paper provides background on Pine, but focuses primarily on the requirements for a global email system and their design implications.

I. Introduction

Pine is an electronic mail and news program designed for ease-of-use, while strictly adhering to Internet standards and sound distributed system architecture principles. Although originally developed to support a relatively small group of people, it is now in use at thousands of sites around the world. Pine has several attributes which are of general interest:

- Pine provides a non-intimidating user interface.
- Pine implements the MIME multimedia email standard.
- Pine supports a gateway-free client-server architecture.
- Pine integrates mail and news handling.
- Pine is available for both DOS and Unix systems.

This paper includes an overview of Pine and its history, but not a detailed description of it. Instead, the majority of the paper is devoted to considering the overall requirements for a global email system, and the architecture and protocols needed to support those requirements. Pine is one instance of a "mail user agent" or MUA, and MUAs constitute one element of the overall email architecture. Particular attention is given to two cornerstone protocols: Multipurpose Internet Mail Extensions (MIME) and the Interactive Mail Access Protocol (IMAP). MIME provides a standard for global multimedia and multipart email interoperability, and IMAP provides the basis for a robust and flexible distributed email architecture.

Competing approaches are considered throughout.

II. Email System Requirements

From the end-user's perspective, the following email system requirements hold:

- Communicate with anyone, anywhere, using multimedia email.
- Access mail from anywhere.
- Access mail from multiple computers, and multiple computer types.
- Access multiple mailboxes, including shared mailboxes.
- Easily attach files to messages and save messages or attachments.
- Send and receive authenticated and confidential mail.
- An excellent user interface.
- Same program reads news as well as mail.
- Accommodate large numbers of possibly large messages.
- System is available at all times.

From a system manager's perspective, there are several additional, albeit self-evident, requirements:

- Configuration flexibility.
- Ease of configuration.
- Ease of user support.
- Robustness (high availability and fault tolerance).
- Scalability to large numbers of users.
- Use of standard transport protocols, even for dialup.

III. Market Forces

The technology to support virtually all of these system requirements exists today, but it is not yet pervasively deployed. Battles continue to rage over

1. Dr. Gray is with the University of Washington. He may be reached at *gray@cac.washington.edu*.

which of several conflicting approaches to global, interoperable email should become the One True Path. The key forces are:

- Mainframe-centric communities.
- LAN-centric communities.
- ISO-centric communities.
- The Internet community.

The technology one advocates depends on a variety of factors, but there does seem to be a troubling tendency among many departmental system administrators to focus almost exclusively on communication within one's immediate workgroup at the expense of global email interoperability. The explosion of connectivity and interconnected information resources around the world suggests that building a departmental system where full email functionality (e.g. multimedia) is only available locally is terribly short-sighted.

IV. An Internet Solution

This paper introduces one particular mail program called Pine, but perhaps more importantly, it describes the distributed email architecture and protocols upon which Pine is based. This architecture offers an approach to achieving the stated email system requirements that is unabashedly Internet-centric. While there are significant forces tending to fragment the Internet community (e.g., restrictive acceptable use policies, multiprotocolism, etc.), it is both the natural affinity group for a Carnegie I research university such as ours, and the leading contender for "largest operational electronic mail system" honors. With the number of Internet hosts now well over a million and growing exponentially, it is a safe bet that the Internet mail transport infrastructure will figure prominently in any solution to the "global electronic village" problem.

V. System Architecture

A major design goal of Pine was to provide an excellent, albeit character-based, user interface for the computer novice. It goes without saying that mailers with excellent user-interfaces are crucial to achieving the goal of pervasive electronic mail, but the user-visible aspects of a mailer are only part of the solution. Let's now examine each of the system requirements identified previously and try to educe the system architecture needed to support them.

Communicate with anyone, anywhere, using multimedia email. The key to global multimedia email interoperability is pervasive deployment of the relevant standards. However, there are many alternative email standards and many definitions of what constitutes a standard. The relevant standards in the Internet mail world are:

- RFC-821 Simple Mail Transport Protocol (SMTP).
- RFC-822 Header Definition.
- RFC-1341 Multipurpose Internet Mail Extensions (MIME).
- RFC-1421 Privacy Enhanced Mail (PEM).

There are recent extensions to SMTP (e.g. 8bit transfer) to improve efficiency and robustness of the Internet mail transport infrastructure, but those extensions are optional, whereas the standards above are mandatory if full functional interoperability is to be achieved. Of course, TCP/IP transport is assumed.

Access mail from anywhere. For purposes of this discussion, "from anywhere" will be defined as "from any Internet host or via dialup."Since one may not always have access to an IP-connected Graphical User Interface (GUI) mailer, the architecture should permit the existence of character-based mailers that can be used via *telnet* or any async dialup connection.

Access mail from multiple computers and multiple computer types. It should be possible for individuals who use more than one computer (especially those who use one type of computer in their office and a different type in a lab or at home or while traveling) to have full access to at least their primary "inbox." Although most computers can *telnet* to a host with a character-based mailer, for full functionality the model must allow for platform-specific mailers, each exploiting the native GUI and accessing a common mail store.

Since delivery of one's mail to a desktop computer is generally inconsistent with the goal of having access to one's mail from multiple computers at different times, the architecture must be based on a client-server approach. Messages and their state information must reside on a mail store remotely accessible from multiple computers, and indeed, multiple types of computers.

In our case, the important types of computers are: PCs, Macintoshes, Unix systems, X Window System terminals (plus any of these machines emulating a character terminal). NT and OS/2 may grow in importance, but they have not yet become a factor at our site.

Access multiple mailboxes, including shared mailboxes. We see increasing interest in the use of "email delivery filters" which separate incoming mail traffic into different mailboxes according to subject, source, recipient, etc. Thus, there will be a need to allow access to more than one "inbox." It is even possible that the multiple inboxes could reside on different computers.

Another important use of email is transaction processing by a group of people. Messages sent to a shared mailbox are handled by the next available person. This requires a shared mailbox supporting concurrent updates of at least message state information (i.e. whether a particular message is answered, flagged, marked deleted, etc.)

Easily attach files to messages and save messages or attachments. At many sites, desktop computers are used as little more than dumb terminals when it comes to network information services such as email. Yet the same machines are used extensively for personal productivity, e.g. word processing, spreadsheets. In such cases, it is very cumbersome to send a document or spreadsheet to a colleague.

It should be possible to directly save into one's primary file system a message or message attachment stored on the mail server. Likewise, it should be possible to directly incorporate into an outgoing message any file in one's primary file system. In particular, it should not be necessary to transfer files from one computer to another in order to attach them to a message. This goal is supported by using a client-server architecture with the MUA running on one's primary computer.

Send and receive authenticated and confidential mail. Most email systems are not very secure. The emerging Internet standard for Privacy Enhanced Mail provides a sound basis for authenticated and (optionally) confidential messaging. As always, pervasive deployment of this standard is required for global interoperability.

An excellent user interface. "Standards vary." Some users prefer command-line interfaces, others menus, perhaps most now favor GUIs emphasizing "point-and-shoot" operations. When selecting a mailer, the user-interface should certainly be a prime consideration. However, there seems to be a tendency to let user-interface considerations totally dominate the mailer decision, to the exclusion of system architecture considerations. Without the proper foundation, some aspects of which will not be apparent in the user-interface, it may be impossible to achieve the system goals. Caveat emptor.

Pine is but one mailer among many. Because use via *telnet* was a requirement, Pine is character, not GUI-based. Its strength is in providing a low-stress environment for reading and sending mail. Whether Pine qualifies as a mailer with "an excellent user interface" is left to others to decide. It will certainly not please everyone, yet its success indicates that it fills an important niche. Whether its popularity is due more to user-interface design, or the underlying protocols and architecture it embraces (MIME and IMAP), or its integration of mail and news... is unclear.

In any case, the architecture must allow for platform-specific MUAs. It should be possible to develop mailers that take full advantage of the characteristics of the personal computer being used, especially its GUI.

Same program reads news as well as mail. Mail and news (bulletin boards) are related technologies with overlapping functionality. Why not offer a consistent user-interface and similar capabilities for both worlds? Indeed, why not use the same program for dealing with both kinds of messages? Sounds like a good idea.

Accommodate large numbers of possibly large messages. This goal requires explicit consideration in all aspects of the architecture, but especially in the design of the mail servers, and MUAs designed to run on limited resource desktop computers. Ideally, limits on numbers and sizes of messages should be a function of system management policy, not inherit limits in the system design.

System is available at all times. As I grow older, there are few characteristics that I value more than dependability. Unpredictable and sporadic electronic communication is almost worse than no communication. It is essential to have an architecture that emphasizes robustness.

There are plenty of ways distributed computing systems can fail. But there are also plenty of things system designers can do to improve the odds. For example, a clever idea might be to deliver mail to an always-up host that is regularly backed-up, as opposed to a loosely managed desktop machine. There is also the question of whether or not to make email gateways a part of the solution. A case can (and in a later section, will) be made that email gateways are part of the problem, not the solution.

VI. Client-server Approach

Several of the objectives discussed above lend themselves to, if not dictate, a client-server approach to the email architecture. A client-server email system is one where the mail (at least some of it) is stored on one machine (the "mail server", or "mail store") and is accessed by "client" MUA programs running on different machines. This strategy is in contrast to delivering mail to and storing it on the same machine that hosts the MUA program.

There are actually three common models:

- Mail is delivered to the machine where it is read.
- Mail is delivered to a "mail drop" machine. An MUA on a different machine later pulls all of the pending mail to the local machine where it is then read by the MUA as a local file

or set of files.

- Mail is delivered to a mail server, and an MUA on a different (client) machine interacts with the server on a per-message basis. The mail remains on the server until explicitly deleted by the user.

Independent of where new mail is delivered and stored is the question of where saved messages are stored. When an email user saves a message for future use, it is typically stored in the user's primary file system or home directory. However, use of a client-server mail protocol adds the possibility of storing saved messages on the mail server along with the user's inbox or on a completely separate server. This can be useful where client machines have minimal resources or are configured without personal file storage, as might be the case in a shared computer lab.

Each of the three models listed above has its place, but only the client-server design meets our requirements. In particular:

- Multiple, platform-specific, MUAs accessing the same inbox.
- Concurrent access to shared mailboxes.
- Tight coupling of mail system and personal (desktop) file system.
- Choice of where to store saved messages.

In some cases the models can be profitably combined. For example, a timesharing system could be configured to act concurrently as a mail server. A user could login and use a mail user agent accessing the local mail files, or use an MUA from a different machine, perhaps a desktop machine, and access that same mail via the client-server protocol. However, this idea has limits. For example, pretending that the average desktop PC can serve as an adequate mail server for even occasional access by other computers requires a leap of faith and finds little support from available software. In order to be a proper server, the desktop machine must be able to export the mailbox via a client-server protocol that permits use of a full-featured MUA on another machine.

The client-server approach could be implemented using a generic remote file access protocol such as NFS. However, there are integrity issues (due to remote file locking and scaling problems) and efficiency concerns that argue for an application-specific client-server protocol. One proven solution, IMAP, will be described in a subsequent section.

Finally, in order to offer flexibility and configurability to the system administrators, the overall architecture must allow for per-user configuration information and saved messages to be stored either on the personal computer or a shared computer. A client-server model supports this goal.

VII. Implementation Strategy

Distilling the original system requirements with some top-level system design decisions, we have the following objectives:

- Global multimedia email interoperability using Internet standards.
- A client-server architecture to support remote, multiplatform access.
- Good mail client software for PCs, Macs, Unix systems and X terminals.
- Robustness.
- Scalability.

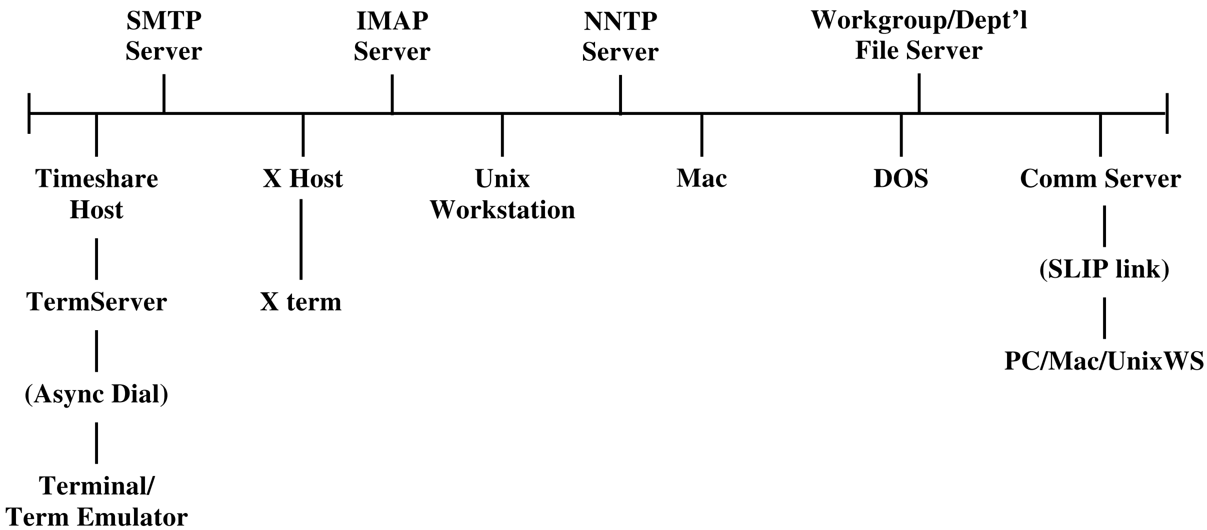
The resulting email architecture contains the following elements:

- A high-availability system for incoming mail storage.
- A high-availability system for outgoing mail dispatch (SMTP).
- A client-server protocol for interactive access to the mail store (IMAP).
- A collection of mail-user-agents for various types of computers.
- TCP/IP connectivity between all elements.
- Compliance with applicable Internet mail standards in all elements.
- An (optional) workgroup or departmental file server.
- An (optional) NNTP or IMAP server for news.

The incoming mail store and the outgoing mail host functions may be on separate machines, or added functions on the departmental file server. Likewise, the News server may be a separate machine or simply another service on the departmental machine. The primary protocol for accessing the mail server is the Interactive Mail Access Protocol (IMAP). News may be accessed by either the Network News Transport Protocol (NNTP) or by IMAP.

The relationship of major system elements is represented in the diagram on the next page.

Notable by their absence in this architecture are email gateways. This is in stark contrast to the conventional wisdom of the day, which assumes a collection of dissimilar LAN-centric and Mainframe-centric mail systems glued together by multiple or multi-lingual gateways. Gateways are obviously necessary to link dissimilar email systems, so



traversing them is sometimes unavoidable. But that doesn't mean you should design *your* email system around a gateway strategy! Gateways tend to reduce overall system reliability and in many cases do not perfectly preserve the semantics or structure of a complex message. And when they misbehave, senders, recipients, and postmasters all suffer.

VIII. MIME: What and Why?

MIME stands for "Multipurpose Internet Mail Extensions" and is described in RFC-1341. It is becoming *the* canonical method for sending any type of file via Internet mail. It allows one to send multipart and multimedia mail, including spreadsheets, WP documents, binary data, programs, images, audio, etc. There can also be alternative representations of the same information sent as a multi-part MIME message. MIME is fully compatible with the huge installed base of Internet mail transport software, and provides a standard way of identifying, encoding, and decoding binary or 8-bit text attachments. MIME also provides for multi-national character sets.

Most contemporary email packages offer a way to send files as message attachments, but they use many different approaches, thus undermining interoperability. It is often necessary for both receiver and sender to use identical computers and mail software for the exchange to be successful. MIME offers the prospect of a standard multi-platform solution. Moreover, MIME includes data typing facilities which can hide much of the attachment and detachment drudgery from the user.

Since MIME is fairly new, not everyone has access to a MIME-capable mailer yet. Fortunately, interest in MIME is spreading rapidly, and it is clearly the best hope for global exchange of multi-part or multimedia messages and binary file attachments. Indeed, it is already being used for non-mail applications such as document delivery, and encoding binary files for Gopher, WAIS, and Usenet news services.

In summary, MIME is:

- A method for sending binary data via Internet mail.
- Capable of sending WP docs, spreadsheets, images, audio, code.
- Capable of sending alternative representations of the same data.
- Capable of carrying data type information to facilitate post processing.
- Growing rapidly.

IX. IMAP: What and Why?

IMAP is the "Interactive Mail Access Protocol" described in RFC-1176. IMAP defines a standard way for MUAs (mail user-agents) to interact with a mail repository. It allows manipulation of a remote mailbox as if it was local. IMAP clients for different types of computers can access the same mail store, and record changes in message state, e.g. marking a particular message as deleted. IMAP clients can take advantage of the particular characteristics of each computer type, such as graphical user interfaces, and integrate smoothly with the user's personal file store. Recent revisions of IMAP also have specialized support for handling MIME messages.

Multi-platform accessibility. The ability to access one's mailbox from multiple computers is a key advantage for those who either routinely or occasionally use more than one computer. This is even more true for those who use multiple types of computers. Some vendor-specific email packages embrace both Macs and PCs, but rarely any other types of systems. Many do not even support *telnet* access to one's mailbox.

Multi-location accessibility. Since IMAP uses TCP/IP, it works anywhere in the Internet. Also, since there are character-based IMAP clients available, an IMAP mail store can be accessed from anywhere in the Internet via the ubiquitous *telnet* protocol. Interactive access to one's mail via the Internet is particularly useful to anyone who travels to other institutions, but the same approach also embraces local dialup access. Remote access to a LAN-centric mail store is rarely possible without resorting to use of a vendor-specific dialup access protocol.

Concurrent mailbox access. IMAP allows multiple clients to read the same mailbox, although possibly only one of them will have read-write access. If a suitable mailbox format is chosen for the IMAP server, it is possible to share a mailbox with multiple clients concurrently updating state information. This is very useful if several people are processing email requests coming into a single mailbox.

A companion protocol to IMAP called IMSP (Interactive Mail Support Protocol) is currently under development at Carnegie Mellon University. IMSP will provide tools to conveniently manage large email installations with many IMAP servers.

X. IMAP vs. POP

POP ("Post Office Protocol") is a store-and-forward protocol for moving email from a drop point to a single final destination, usually a PC or Mac hard disk. Once delivered to the PC or Mac, the messages are typically deleted from the POP server. POP is not well-suited for the goals of accessing one's inbox from different places and different machines at different times. However, from a system design perspective, POP is still a more robust foundation for a distributed email system than vendor-specific systems requiring gateways to Internet mail.

In comparing POP with IMAP, here are the key points:

- Two different paradigms:
  - POP = store-and-forward
  - IMAP = client-server
- POP advantages:
  - Minimum use of connect time
  - Minimum use of server resources
  - More software currently available
- IMAP advantages:
  - Multiplatform access to remote mailboxes
  - Efficient management of large mailboxes
  - Faster startup
  - Can create and append to remote folders
  - Ability to use server for searching
  - Usable over low-speed links
  - Can deal with very large messages
  - Allows fetching individual MIME segments (or parts)
  - Suitable for non-email info; e.g., NetNews

POP is of interest in situations where the only access to a mail server is via expensive dialup connections and multi-platform access to one's inbox(es) is not needed. It is also useful in environments where client machines are resource-rich and servers are resource-poor. Note however that IMAP offers a superset of POP functionality, so an IMAP client could choose to operate in "POP mode."

XI. IMAP Clients

Interest in IMAP is growing. There are now several commercial companies developing IMAP software, and more freely available IMAP software is also appearing. IMAP servers for Unix and TOPS-20 have been available for many years, and a port to Windows NT is expected soon. An IMAP server for VMS is commercially available.

Here are some of the IMAP clients currently available:

- For Macintosh: MacMS and Mailstrom from Stanford
- For DOS: PC-Pine from UW
- For Unix systems: Pine from UW
- For NEXTSTEP: MailManager and EasyMail from UW
- For X Window System: XIMAP from Stanford

A company in Canada is developing a Windows IMAP client, and there are also IMAP clients for TOPS-20 and various LISP machines.

The X, NEXTSTEP, and Mac clients all use the native GUI of their respective platforms. Pine and PC-Pine are character-based.

XII. Mail vs. News

At most sites, mail and bulletin board (news) access are treated as completely different problems, yet there is a great deal of similarity between the two worlds. A growing number of people have been wondering why the same tools couldn't be used to access both mail messages and news articles. Our conclusion was that there needed to be a few differences in the user interface when reading news vs. mail, but that the commonality was overwhelming, so it made good sense to have a single tool that could do both. Given the distributed email architecture described above, the basic functionality was already in place.

News is usually accessed via the Network News Transport Protocol (NNTP), however IMAP can also be used to access a news database. The basic choice is whether personal message-state information (the ".newsrc" file) is stored on the user's primary (local) machine, or on the central news server. NNTP assumes the .newsrc file is on the (local) client machine; IMAP assumes it is on the central server. Both models have their place, and both are currently in use. Just as with the question of where to store saved messages (locally or remotely) it is up to users and system managers to pick the configuration most appropriate for a particular site.

XIII. Roads Not Taken

What has been presented herein is an email architecture that is purported to have numerous desirable properties (amazing restorative powers, the ability to inhibit aging, etc.) But against what alternatives should this approach be compared?

Let's review the options:

- *Mainframe-centric solutions*. These are more-or-less by definition architecturally closed systems. Being able to take advantage of the native GUI of common desktop platforms, and tightly integrate mail with the personal file system are possibilities simply not within the scope of these designs. Also, in the case of vendor-specific systems, it is necessary to traverse a gateway to reach the Internet mail world, and those gateways typically do not support MIME or PEM.
- *LAN-centric solutions*. These are by far the most seductive. They typically offer excellent user-interfaces, and increasingly sophisticated integration of mail with other (personal productivity) applications. However, if one is interested in global interoperability, they come with two serious disadvantages: (1) it usually takes a long time for Internet standards to be integrated into these systems, although a few "mainstream" vendors are now beginning to pay more attention to Internet requirements, and (2) since all of the LAN-centric offerings require some form of gateway for access to the Internet, there is a risk that the semantic mismatches between the LAN email system and the Internet email system will result in loss of information when structured messages traverse the gateway. The result is typically "islands of partial connectivity." In addition, there are architectural disadvantages relating to local interoperability and flexibility. There may be exceptions, but in general:
  - Remote access to one's mail requires use of a vendor-specific protocol.
  - Mail access is via vendor-specific remote file access protocols.
  - Client machines other than PCs and Macs are usually not supported.
  - It is often not even possible to use *telnet* for mail access.
- *ISO-centric solutions.* X.400 is often represented as the wave of the future, the one unifying email solution behind which everyone should rally. Certainly every major vendor and every trade magazine presents this view continuously. Is it true? Or, is it possible that every major vendor and every trade magazine has underestimated both the problems with the X.400 standard and the significance of the large and explosively growing Internet email installed base? Hard to believe, but many people including the author have come to the second conclusion. It is beyond the scope of this paper to argue the point, but suffice it to say that one need look no further than X.400 address syntax to realize that this was the work of a committee.

Another issue concerns scalability. In addition to the obvious issues of limits on mailbox and message sizes—which in some systems can be astoundingly low—there is a question of how many people you can send email to. All of the non-Internet email approaches have at least *some* implementations that assume everyone you ever want to talk to will be in a local address book. This is necessary in X.400 products because of the awkwardness of native X.400 addresses, and the present lack of a global X.500 directory infrastructure. By contrast Internet mail systems, based on simple addressing, are inherently open-ended. They *allow* address books, but are fully usable to reach people who are not in the local database.

Last, but not least, there is the question of gateway *robustness*. By definition, all of the non-Internet email approaches require a gateway to exchange mail with Internet hosts. Unfortunately, a disproportionate number of email problems have their roots in gateway bugs, configuration problems, or low-availability

LAN servers acting as email gateways. Just recently a large email distribution list received nearly a hundred bogus messages due to an email gateway problem. In some cases, gateway software is only available for DOS platforms. A few of these products cannot even sustain multiple simultaneous TCP connections.

In fairness, let's close this section by disclosing the disadvantages of the recommended email architecture. Because both MIME and IMAP are relatively new, the choices for IMAP client software are still rather limited, especially for IMAP clients that support MIME. Although PC-Pine supports both, it is a DOS, rather than a native Windows application. Thus it is character-based, and does not purport to offer a state-of-the-art GUI. Fortunately, MIME already has a critical mass of vendor support, and interest in IMAP is growing. It shouldn't be long before one can have the best of both worlds: contemporary GUIs plus superior interoperability and architecture.

As mentioned, our global interoperability requirement can be partially achieved via gateways to vendor-specific systems. However, their architectural foundations make them unsuitable for sites that must accommodate more than just PCs and Macs, or that wish to control support costs by focusing on Internet protocols rather than multiple vendor-specific protocols.

Looking ahead, one wonders whether "mainstream" mail software will interoperate with Internet-standard authentication technology, or whether history will repeat itself and system managers will be forced to support multiple authentication systems, each with its own user database. Can't we do better? This problem has already been solved and the technology is freely available.

XIV. Pine Background

Pine is a mailer designed specifically for the novice computer user, but with optional commands available for "power users." It is based on Internet mail protocols (e.g. RFC-822, SMTP, IMAP, and MIME) and currently runs on DOS and a variety of UNIX platforms.

The guiding principles for achieving ease-of-use in Pine were: careful limitation of features, one-character mnemonic commands, always-present command menus, immediate user feedback, and high tolerance for user mistakes. It is intended that Pine can be learned by exploration rather than reading manuals. Feedback from the University of Washington community and a growing number of Internet sites has been encouraging. A stand-alone version of Pico, Pine's message composition editor, is also available. It is a very simple and easy to use text editor with text justification and a spelling checker.

XV. Pine Features

- Mail index showing a message summary which includes the status, sender, size, date and subject of messages.
- View and process mail with the following commands: forward, reply, save, export, print, delete, capture address and search.
- Address book for saving long complex addresses and personal distribution lists under a nickname.
- Multiple folders and folder management screen for filing messages.
- Message composer with easy-to-use editor and spelling checker. The message composer also assists with entering and formatting addresses and provides direct access to the address book.
- Online help specific to each screen and context.
- Supports access to remote mail repositories via the IMAP2 protocol defined in RFC-1176.
- Support for multipart mail conforming to proposed MIME Internet standard. This allows attachments to mail messages such as graphics (GIF, TIFF...), sounds, and other files such as spreadsheets and binary files.
- User-configurable feature levels.

XVI. Pine History

In the spring of 1989, the University of Washington's Office of Computing & Communications began searching for an easy-to-use character-based mailer that ran on Unix systems. The target audience was University administrators who were then using an MVS program derived from UCLA's BEN mailer. After a few months of searching and evaluating the options available at the time, including several X Window System offerings, we concluded that none of them met our ease-of-use goals. Thus the Friendly Unix Mail Project (FUMP) was begun. It was decided to start with Elm code and incrementally modify it to meet our needs. In keeping with Elm's arborist roots, *"Pine"* was selected as the new mailer's name. Pine was not originally intended to be an acronym, but inevitably it was turned into one. There are several versions, including "Pine Is No-longer Elm," since there is hardly any Elm code left in it anymore, and "Program for Internet News & Email." Laurence Lundblade was given overall responsibility for the project, and Mike Seibel joined soon after to focus (initially) on the Composer module, which later was extracted from Pine to become a stand-alone editor called Pico (for Pine composer).

The earliest versions of Pine were in limited use by the end of 1989. In January 1990, the current Pine version was 0.6. During the rest of 1990, the user base continued to grow as the program continued to be refined. It was not long before the original audience of University administrators was dwarfed by the number of faculty, staff, and students who had voluntarily migrated to Pine from a variety of other Unix, VMS, and VM/CMS mailers.

In February 1991, Pine 1.01—the first version with IMAP support—was operational, using the c-client email tool kit developed by Mark Crispin. However, it took six months to smooth rough edges, so the production release of the first IMAP-capable Pine didn't occur until September 1991.

In August 1991 we were contacted by folks at the Texas Education Network (TENET) about the prospect of using Pine to provide electronic mail service to K-12 teachers throughout Texas. This was the first instance of significant Pine use outside of UW, and the feedback from TENET was most gratifying.

January 1992 brought the release of Pine 2.0, the first public source distribution. In March 1992, a version was available that contained a primitive newsreading capability, using IMAP for accessing the news server. This was done to support another project being developed in parallel: UWIN, the University of Washington Information Navigator. Pine's newsreader mode is used to access the Clarinet news service as well as local information and documents.

In July 1992, Pine 3.0 was released. This was the first general release with support for MIME attachments. Just before this, work began in earnest on a port of Pine to PC-DOS. Getting the code to compile was not so difficult, but restructuring it so that arbitrarily large MIME messages and arbitrarily large mailboxes could be handled in a machine with a 640KB address space took heroic efforts by Mike Seibel and Mark Crispin. At the end of 1992, a test version of PC-Pine was available. By this time, the Unix version was in use at over 2,000 sites in 30 countries.

1993 brought the production release of PC-Pine, and an enhanced Unix version. Major goals for the remainder of 1993 include refinement of Pine's newsreading capabilities, additional MIME support, and aggregate operations on groups of messages.

XVII. Pine Futures

The list of requested enhancements for Pine is long, but there are several significant directions worth highlighting:

Additional newsreading support. By the time this paper is published, Pine should have fairly credible newsreading capabilities, but we will no doubt have more work to do in this area.

Kerberos. Kerberos is a network authentication system originally developed at MIT. It is the planned foundation for many network computing environments. As version 5 of Kerberos stabilizes and is incorporated into OSF'S Distributed Computing Environment product, Kerberos is sure to become the dominant protocol for network authentication. Pine and IMAP server code will be modified to work with Kerberos.

Disconnected use, based on IMAP extensions. It is sometimes useful to be able to download a selection of messages into a portable computer, without deleting them from the server, and process them offline. Later, one would like to connect to the mail server again and resynchronize the state of the server using information from the "cache copy" of messages on the portable. For example, transferring back to the server the list of messages deleted while processing offline. Using lessons learned from a mail protocol that does support this functionality (DMSP), IMAP and Pine will be extended to support disconnected use.

PEM. Privacy Enhanced Mail is not yet widely available, but it is easy to believe that demand for PEM will grow rapidly in the future. Work is continuing on the specification that will define the interaction between PEM and MIME. It is reasonable to expect that a future version of Pine will support PEM in accordance with this specification.

XVII. Pine Availability

Although not in the public domain, Pine and Pico, including source code, are available at no cost via anonymous FTP from *ftp.cac.washington.edu* on the Internet. Other provisions for distribution have not been made. From the Internet, you may try out Pine and leave comments by telneting to demo.cac.washington.edu and logging in as *pinedemo*.

Pine is very portable and runs on a variety of UNIX machines including Suns, RS6000s, DECstations, NEXTs, and Sequents. Pine uses termcap or terminfo files to accommodate a variety of terminal types. The terminal emulator must support cursor positioning, clear screen, and reverse video. PC-Pine is currently available in three versions: Novell LWP, FTP's PC/TCP, and packet driver.

For further information send email to *pine@cac.washington.edu*. To be informed of future releases, send email to *pine-announce-*

request@cac.washington.edu. To join an IMAP discussion list, send email to: *imap-request@cac.washington.edu.* Pine is a trademark of the University of Washington. It is the work of Laurence Lundblade, Mike Seibel, Mark Crispin, Steve Hubert, and Sheryl Erez at the University of Washington's Office of Computing & Communications. (NB: Lundblade is no longer with UW, but continues to contribute to the Project. Upon Laurence's departure, Mike Seibel became Pine's chief programmer and project lead.)

XVIII. Conclusions

Pine has been extremely successful in achieving its goals, but it is only one element of a comprehensive email architecture. The most important advice I can offer others struggling with design tradeoffs for electronic mail systems is: (a) Internet mail protocols offer the best hope for truly global email interoperability; (b) MIME and PEM will become increasingly essential standards; and (c) IMAP is currently the best foundation for a robust client-server distributed email system. With interest in these protocols steadily growing, we can look forward to email systems that combine the best in user-interface technology with sound distributed system architecture principles.

IX. Biography

Terry Gray is currently Director of Networks and Distributed Computing at the University of Washington, and an Affiliate Professor in the UW Computer Science & Engineering Department. Prior to coming to UW, he was Vice President of Engineering for the Bridge Communications Division of 3Com Corporation.

Other work experience includes Bell Laboratories, Ampex Corporation, and many years in the UCLA Computer Science Department, first as a graduate student, later in both faculty and staff roles.

He received a BS in Electronic Engineering from Northrop Institute of Technology in 1967 and a PhD in Computer Science from UCLA in 1978.

For nearly 20 years his principal focus has been on distributed system architecture, but he has also published work in computer security, software engineering, distributed operating system design, and network job control. His primary responsibility at UW is to foster development of a high-availability network computing environment that encompasses all four basic foodgroups of desktop computing (Macs, PCs, Unix workstations, and X terminals), and provides convenient access to both campus and world-wide information resources.