The MOS Multimedia E-Mail System

Ming Ouhyoung, Wen-Chin Chen, Yuong-Wei Lei, Keh-Ning Chang,
Ching-Lun Liang, Shwu-Fen Wang, Yung-Huei Yan, Jiann-Rong Wu,
Herng-Yow Chen, Nien-Bao Liu, Yin-Lai Wang, Tzong-Yin Hwu,
Wei-Ming Su, Rung-Heui Liang, Kuo-Chang Fu, Yah Chen, Tzong-Jer Yang

Communications & Multimedia Laboratory
Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan, R.O.C.

Abstract

In this paper, we present the architecture of a new multimedia E-mail system, which has been successfully developed in the Communication and Multimedia Laboratory of National Taiwan University. This prototype E-mail system is fully compatible with traditional Internet E-mails in that the multimedia emails can be sent through Internet. For allowing one to compose multimedia emails, the system provides various editors for editing video, audio, image, graphics, and multilingual (English and Chinese) text. It is currently implemented on SUN workstations with X-windows and OpenLook as its standard graphical user interface. In order to display video, the workstation are equipped with either an add-on video board or a software-based Codecs. Since its development, the system have been widely used in the National Taiwan University. Experience of users showed that the system is convenient to use and is indeed fully compatible with the traditional Internet E-mails.

Keywords: Multimedia E-mail, Multimedia Applications, Graphical User Interface, Computer Networks, Office Automation

1 Introduction

Electronic mail (E-mail) is used mainly as messages being sent between people. It can also be used for communication between people and processes (virtual users) or even among processes themselves. The number of E-mail users in the U.S. is expected to climb 60 percent in 1993 over 1992; by 1995, the users base could reach 38 millions (source: The Yankee Group [6, 7].) According to [1], E-mail by the mid-'90s will take off in the consumer realm. Portable notebook computers with built-in wireless modems will enable users to send and receive E-mail anywhere. According to Larry H. Landweber [1], most of the world, 108 countries so far, is connected to electronic networks (including E-mail services.)

In 1992, we have launched at National Taiwan University a collective research project named MOS, short for the Multimedia Office Systems. MOS includes two subsystems. One is a multimedia E-mail system for off-line communication, and the other is a video conference system for on-line communication. After ten months' development, the prototype MOS multimedia E-mail system is under experiment use. The system possesses the following distinct features.

- It is fully compatible with Internet E-mails.
- It has various media editors such as audio/video editors, image editor, sound editor, and six different types of text editors, including Emacs and an editor for Chinese.

2 A system overview

The integration of audio, video, images, and graphics data into a single letter for electronic mail provides multimedia information many new media in addition to the traditional text. The purpose of our new multimedia MOS E-mail system is to provide an integrated environment for multimedia document authoring, sending, receiving, and displaying. As shown in Figure 1, a typical session of the MOS E-mail system can be described in two parts: the sending site and the receiving site.

0-8186-5530-5/94 $3.00

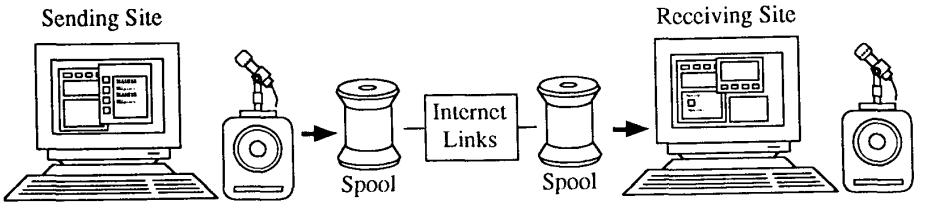

Figure 1: A typical session of the MOS E-mail system.

Sending site

To send an MOS multimedia E-mail, one should prepare a document, either by creating a new document or by modifying a saved or received one (in the case of forwarding.) The entire document consists of pure text and intermixed with certain TEX style control keywords to stand for including certain video, audio, and images messages. One can use media editors such as audio/video editors, image editor, and text editors to compose a medium message and then import the medium message into the document with the help of a dialogue box. One must provide the necessary information of the medium message such as the medium type, the location to access the message data, and the caption of the medium. When the document is ready for delivery, the system will parse the document and construct an ASCII format message. Then the mailing message is delivered through Internet links.

Receiving site

When one starts the MOS E-mail, the system will detect whether he has received multimedia mails. If so, the system will retrieve the incoming mails from E-mail spools and then decode the mail and display the content depending on the media types. The text portion, including Chinese characters, is displayed on the window directly, and the other media are represented by proper icons. When one selects an icon, the system will invoke the associated medium display agent to display the medium data. One can save part of or the entire letter or modify and forward the received letter to someone else.

Figure 2 shows a screen snapshot of the MOS E-mail system. The left window is the primary window that contains a control area, a mail header pane, and a viewing pane. The control area contain five buttons: File, View, Edit, Font, and Compose for processing the mails and a text field showing the name of the current mail file. Each line in the mail header pane represents a single multimedia document (header.) The viewing pane shows the contents of the mail being selected. In Figure 2 the selected mail containing three icons for audio, video, and image, respectively. The image display agent is activated to show part of the image 'The Simpsons' in upper right of the window.

The right window is a composition window. When one chooses an item from the Compose menu, a composition window is popped out. The four icons, Text, Image, Video, and Audio, in the composition window are used to invoke corresponding media editors. The figure shows that there is a Chinese editor in lower right corner.

In section 3 we will discuss some design issue of the MOS E-mail system, including authoring style, formats, and functional modules definition. Section 4 contains detailed descriptions of the implementation. Section 5 discusses future work and conclusions.

3 Design issues and specification

The first main goal of the MOS E-mail system, is to develop a multimedia system that allows users to compose and deliver multimedia letters. The first principle is that the system must be compatible with the traditional Internet E-mail environment, which means the letters must be delivered in real ASCII, namely, printable characters, and the system does not need to get too much involved with the E-mail module of the operating system. The latter goal of the principle means, saying in another way, the MOS E-mail system will produce multimedia letters in text format which can be delivered by the conventional mail command. This strategy simplifies the interaction between the multimedia E-mail system and the OS. The second goal is that this MOS system is not only a multimedia file composer merely with basic functions (read, deliver, carbon copy, blind carbon copy), but also a powerful media editor which provides a user with some special effect functions to edit.

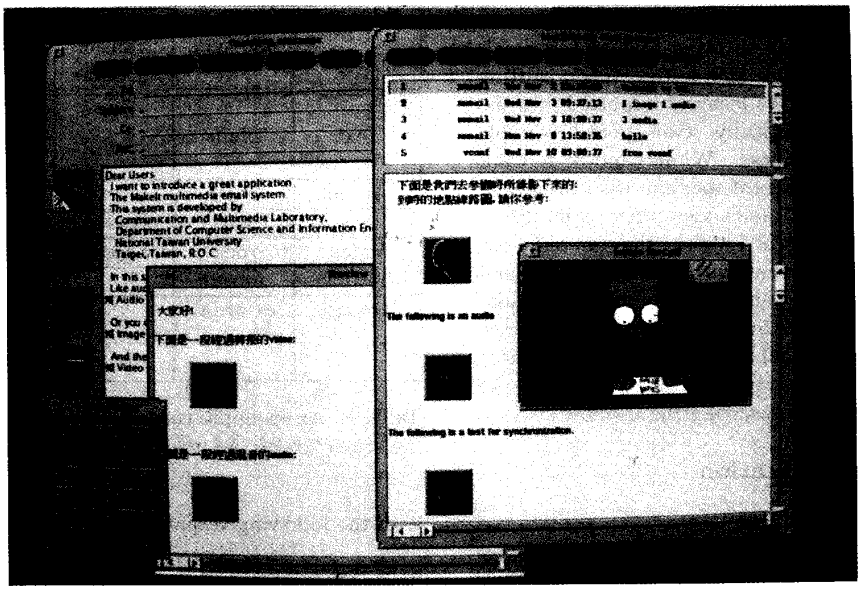

Figure 2: The MOS multimedia E-mail system

3.1 Terminology

The following four terminologies are used throughout this paper:

Authoring document the letter that is used for authoring, i.e. we say that the user is authoring a document.

Mailing message the letter that is used for mailing, i.e. the letter sent in the Internet links. In fact, the mailing message can be treated as a multimedia document with features that the media data are coded by pure ASCII.

Document format the predefined format used on authoring documents. A user must follow this format when authoring.

Message format the format used in mailing messages. MOS E-mail composes mailing messages according to the formats in section 3.2.

3.2 Document authoring

The authoring style often determines a multimedia system's look and feel. In the current version the MOS E-mail does not allow users to position the media at arbitrary locations. When a message is shown in the viewing pane, the media objects, consisted of text paragraphs and icons, can be arranged in arbitrary order but must in linear sequence (an example is in Figure 2.) Our goal is to allow one to write letters in free style and feel no restriction and limitation, thus even a draft can be a letter. So the system provides a media section editor that lets a user import the media sections interactively, step by step. When the media section editor is activated, a dialogue box will pop out, and let a user fill in the information of one specific medium, such as what the medium is, where the medium is, what is the caption about the medium, and finally a media section would be inserted into where the cursor is located. For not already prepared media data, the authoring is more straightforward. The user only has to use the corresponding media editor to prepare data, just like recording a live video, drawing a picture, or recording speech, and when the preparation is finished, the media section will be inserted into the document automatically. Of course, a user can edit a document and insert media sections manually, with the aid of the system. The only problem the user has to face to is the format of the media sections must be confined to a predefined syntax.

Users are free to use any preferred text editor to prepare the documents. In fact, the editor the MOS E-mail system supports is the simplest form of a text editor. We don't want to pay too much effort to develop yet another text editor, instead, a user can keep using his personally favored text editor. When writ-

ing a letter, a user can use any text editor that he likes to use, and shifting to other text editors is allowed even in writing the same letter. When shifting to another text editor, the letter is imported into the new text editor automatically. One special example is the authoring of Chinese. While Chinese is commonly recognized as different medium with respect to English, in the MOS E-mail's view there is no difference between Chinese and English, they are all text. One can write a document with mixed Chinese and English and does not need to define medium section. Thus when a user wants to insert Chinese text in the document but the original text editor is not a Chinese editor, he can shift to a Chinese editor and back to the original editor after inserting the Chinese.

3.3 Format specification

One of the main issues of developing such a system, in our experience, is the difficulty to design a proper letter format. Before the letter format is defined, what we can do is to develop the media tools and can not do anything about the authoring/viewing system. This is because the module definition is tightly coupled with the letter format.

There are two types of letter format. The first, called message format, is an intermediate form used just for mailing. The second, called document format, is used for authoring documents. The major difference between the two is that the media data must be put into a message for delivery. Therefore how to encode the data is the main issue. As for an authoring document, how and where to get the media data is the main concern.

Mailing message format

The mail message is used for delivering a mail. For compatible reason, MOS E-mail follows MIME (Multiple Internet Mail Extension) [8, 9, 10, 11] standard as its mailing message format. Our current system doesn't include all content types so far, but have included necessary ones. One issue is that we don't restrict a user in reading a mail following a fixed sequence of steps, for example: first read a paragraph of text, next hear a audio, third another paragraph of text, etc. We hope our user can glance and survey the whole text and its appearance first, and play media in a user controlled situation, and he can play them more than ones if he chooses. Therefore the user can peruse this document with more understanding.

The mail message first contains a MIME-Version header field to specify the version of MIME standard, with the following verbatim text:

```
MIME-Version : 1.0
```

Next, the system uses a Multipart/mixed content-type to bundle the whole top level body part, and all other bodies with private header and body respectively are embed in it, and every parts are separated by a boundary. Following this content type field, the system ignores any data until the first boundary.

```
Content-Type : Multipart /mixed ;
boundary=ThisIsBoundaryString
        Here  are  not useful data
--ThisIsBoundaryString
```

Content type	Sub-type	Content transfer encoding	Comment
Text	plain		char set: US-ASCII, x-BIG5
Audio	basic	base64	
Image	gif	base64	
Video	jpeg	base64	
Multipart	mixed		
Message	external body		local-file, ANON-FTP, or FTP

Table 1: An example content type used in a mailing message (* x-BIG5 for identifying the Chinese text)

Table 1 defines content types and sub-types, each medium or text is packed with a necessary header (Content-Type, Content-transfer-encoding,...etc) and embeded into the mailing message. In addition to encoding a normal medium file into mailing message, the system also support external-body content type, that is, the actual bodies are not included, but merely referred by remote site referencing, either by anonymous FTP or FTP.[12] This implies that a user can reduce the mail size by obtaining some data from the other site.

Authoring document format

The structure of an authoring document is similar to that of a mailing message except that each medium section is shrunk into a coded string indicating the medium type and the location of medium data. The format of each medium section in a authoring document is:

%{Media Type [Audio or Image or Video]: [directory] filename — Access Type [Normal, Local-file, ANON-FTP, FTP] — IP-address — Caption}

3.4 Conceptual software architecture

The absence of a well-defined application development methodology is a major constraint on the growth of multimedia technology [4]. Traditional computer application developer tend to think in very structured ways, whereas professionals from creative services, such as audio and video, tend to have an unstructured or semi-structured approach. The first of the difficulties in developing a multimedia system is to minimize the correlation among modules. We tried to define the modules in logical level, so the following module specification may not be the same as the implementation codes. The MOS E-mail system is divided into five subsystems.

Authoring/Viewing module One module is used to build, modify, and maintain a letter by adding, deleting, and replacing information in the letter, the other is used to extract information from the received letter. This module is the user front end and its style of authoring determines the MOS E-mail's look and feel.

Interface module The user interface of the MOS E-mail is constructed atop the OPEN LOOK Intrinsic Toolkit (OLIT) and the X window system [2, 3]. We build the interface on an existing toolkit in order to make it easy to port to another GUI toolkit.

Media editor module A media editor is needed for creating and editing the media. Each medium requires specific hardware and software support. Each medium tool is not only a module of the system, but also can be an independent program. For example, the image editor is not only one of the tools when authoring a letter, but also is a desktop utility when used alone. The media tools in MOS E-mail include editors for sound, audio/video, image, and text.

Compose/Decompose module The functions of integrating media into a letter and converting non-text data to ASCII format are combined in this module. Similarly the corresponding inverse function, converting an ASCII letter to a multimedia letter and extracting each item of medium, is also realized in this module.

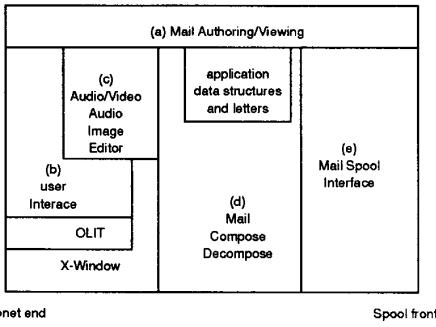

Figure 3: Functional diagram of the MOS multimedia E-mail system

Mail spool interface module Routines for manipulating the mail spool, including mailing letter, deleting letter, and detecting a received multimedia letter.

4 Implementation details

One of the most important issues of implementation is how to integrate multiple media into a single letter, both in mailing messages and internal representation. Our implementation is realized in such a way that the system can change or add any media tools and text editors without affecting the whole system. The newly added medium will not induce too much modification to the system; for instance all we need to do is to define a new medium section type and can add a Chinese character viewer. The following two kernel modules, mail compose/decompose module and media editor modules of the MOS multimedia E-mail system will be discussed respectively.

4.1 Compose/decompose module

The compose/decompose module is one of the most complicated code segments for control logic and data structures. The function of the module is straightforward — it controls the data flow pipeline from authoring document to internal representation, then to mailing message, and vice versa. As stated above, the module performs two functions in the pipeline, integrating media data and converting non-text data. The module parses the authoring document to get the global information of a letter and local information of the medium objects and constructs a internal database for the authoring document. In this stage the media

data are not yet imported to the system, the database contains the information about the global statistics of the document together with local information of each medium object such as the location of data. The database can then be fed into the pipeline to construct a message or be retrieved back to the viewing stage for preview. The major advantage of the internal representation is that it is suitable for both authoring and viewing, because the media data are isolated from the database and stored alone. This strategy abstracts the medium objects and is good at the usage of memory. To prepare a mail message means the module will convert the non-text data and prepare a fixed format multimedia message for delivery. The module encodes the non-text media by the base64 codec function, so that all media data are transformed into ASCII form. Then each medium object is inserted as a medium section in predefined format — control part first and followed by converted data part with a pair of delimiters to define the scope. Finally the message is passed to spool manipulation module for mailing.

Compose module

The compose module is activated when the mail is edited well and is ready to be delivered. The basic work is to scan the authoring document and convert every parts (English text, Chinese text, Video, Audio or Image) into a corresponding body part and finally assemble them in a whole mailing message. In order to add a media format more easily when a user is authoring, a user can press the "Media Edit" button and fill in medium attributes, then system will automatically convert the information into a corresponding message to authoring document. But this special message line may be wrong since the media file may not exist or this message line is modified and destroyed by a user somehow. The compose module must have the ability to warn a user's mistake before delivering. And this module should make sure every delivering mail is guaranteed correct.

We have developed a preview agent to supplement our pure text authoring document which cannot provide a user with a complete information of the receiver's (maybe his boss) view of the mail. This preview agent also let a user to guarantee this mail is indeed what he want, not only the layout but also every media file. A user can use the preview agent as if the receiver are using this system to read this mail. One problem of our preview agent is that the media encoding is time consuming, and a user doesn't want to take a lot of time in encoding again just for modifying a word. So our preview agent doesn't actually encode anyway, and offers a near real-time service. We illustrate this compose module in Figure 4.

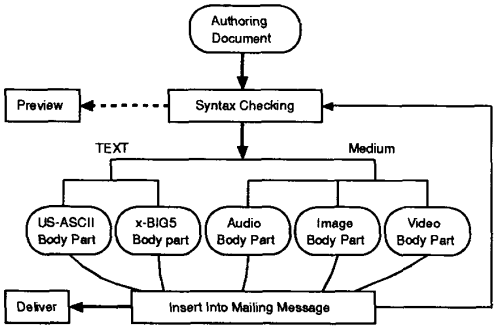

Figure 4: Compose Module Data Flow

Decompose module

The decompose module is just the inverse of compose module. But in order to offer more convenient usage and time-saving, we have the following implementation:

- Decoding on demand
- Simple players for media display
- Forward function

Decoding on Demand: As stated above, decoding is often time consuming (especially for video). A user maybe want to just survey the text and play a few media which he is interested in. So the module first extracts all text and records all media data attributes, and decodes them only when a user selects the medium he wants. This issue is not only for normal base64 decoding demand but also for ftp demand.

Simple players for media display: We invoke simple players, instead of complete editors, for a user to play media. These simple players provide some functions including re-play, pause, stop, volume, eject.

Forward function: One can forward and modify a multimedia email to someone else. One issue of forward function concerns how to be more intelligent to save time by avoiding unnecessary decoding. So one can choose a arbitrary email in mail message pane, forward, and modify it on the compose text window.

4.2 Media editor modules

Audio/video editor

This Audio/Video editor (abbreviate A/V editor) [14, 15] is used for live video editing with audio and

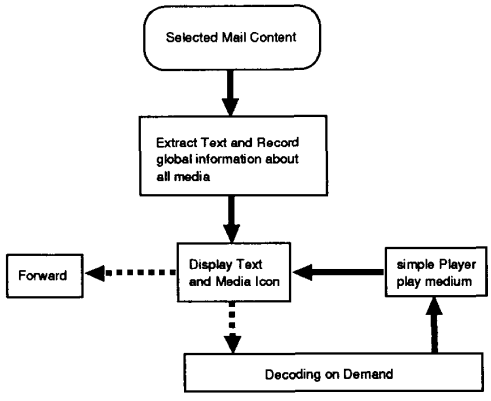

Figure 5: Decompose Module Data Flow

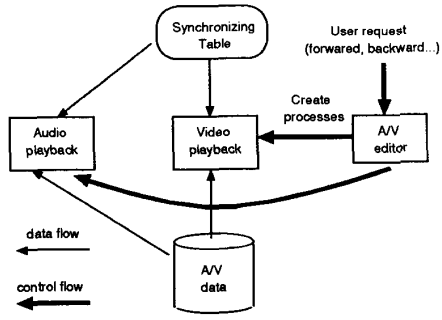

Figure 6: Audio/Video Synchronization

video synchronization. The two main functions are as following: **RECORD & PLAYBACK**. The tool accept audio/video signal from video camera and display it on the screen. User can record a period of video into a file and play it with forwarding, backwarding, pausing, frame tracking, arbitrary speed playing operation. **EDIT**: it support a user to edit multi-channel audio/video with basic operations such as copy, cut, and paste and some special effect operations, for examples fade in fade out...etc.

The most difficult job is to provide the synchronization between these media. Our two basic principles are: **(1)** It must keep the quality of audio, that is, the audio cannot break on playing. **(2)** Audio/Video must always synchronize. So we give the highest priority to audio playing process, and drop some video frames if CPU cannot keep synchronizing on playing. In Fig 6, every media generate processes after playing, and system create a synchronizing table before playing, then all media play follow this synchronizing table as well as the system timer. The result of our implementation keeps the synchronization of audio and video even the CPU or I/O were highly loaded. If the computer is equipped with special hardware for true color display and real time compression/decompression, such as the Parallax board in our implementation, a hardware based A/V editor is invoked and a peak performance of more than 25 frames per second is achieved. Otherwise, a software based video compression/decompression module is used.[13] We can modify the synchronizing table further to change the result of playing and add additional media, for instance TEXT, in it, and still keep synchronization.

Sound editor

This editor is used to process pure audio data that is generated and stored as PCM standard. One can not only record, play, fast forward, rewind, retrieve, and save audio files, but also copy, cut, and paste a portion of sound between multi-channel. Certain special effects such as fade in, fade out, mixed, and muffle are also supported by processing the sampling pulse data. One design difficulty is to keep the audio quality. The system solves this problem by advancing queuing audio data to avoid voice breaking.

Image/graphics editor

This is an X-window based tool that can manipulate image files in various formats such as GIF, TIFF, SunRaster file format, X11 Bitmap, PostScript, and JPEG. Currently the drawing and painting functions include drawing line, text, circle, rectangle, dashed line, color selection, line width, filled area etc. Resizing, dithering, smoothing are also features of it. In this editor, color manipulation on the same image is quite straightforward, but manipulation on multiple images concurrently causes problems. Because most workstations are equipped with 8-bit plane frame buffers, the colormap manipulation for cutting and pasting color images do have some constraints, i.e., at most 256 colors in use. At present we use the strategy that all images share the same colormap and the difference between original images and the new ones is dealt with by image processing techniques.

Multilingual text editors

As we stated above, a user is not restricted to use a specific text editor, and is free to use his personally preferred editor. In general, we include popular and existing text editors such as Emacs, View, Sun

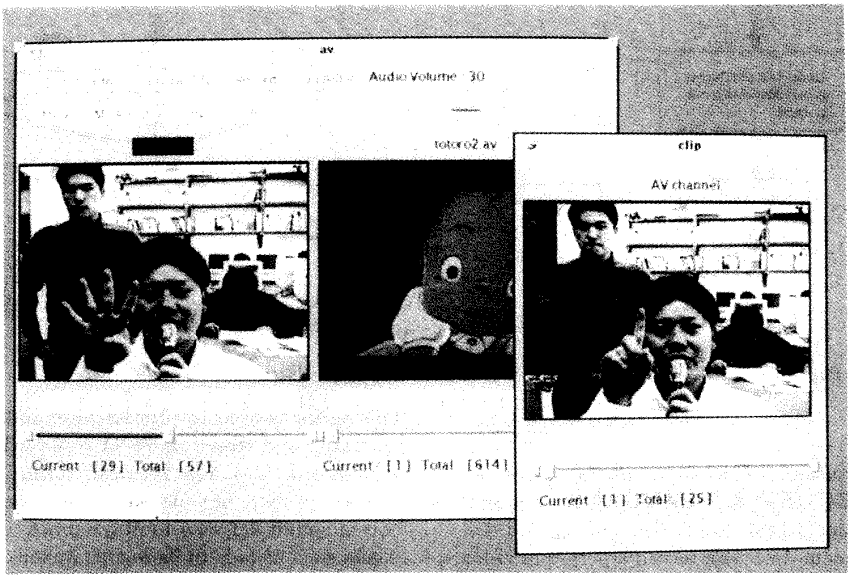

Figure 7: The A/V editor.

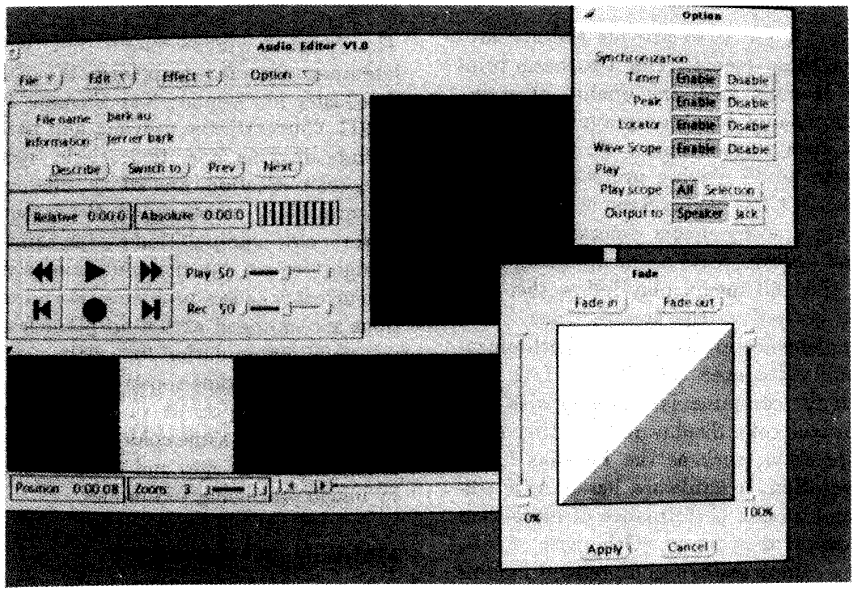

Figure 8: The Sound editor

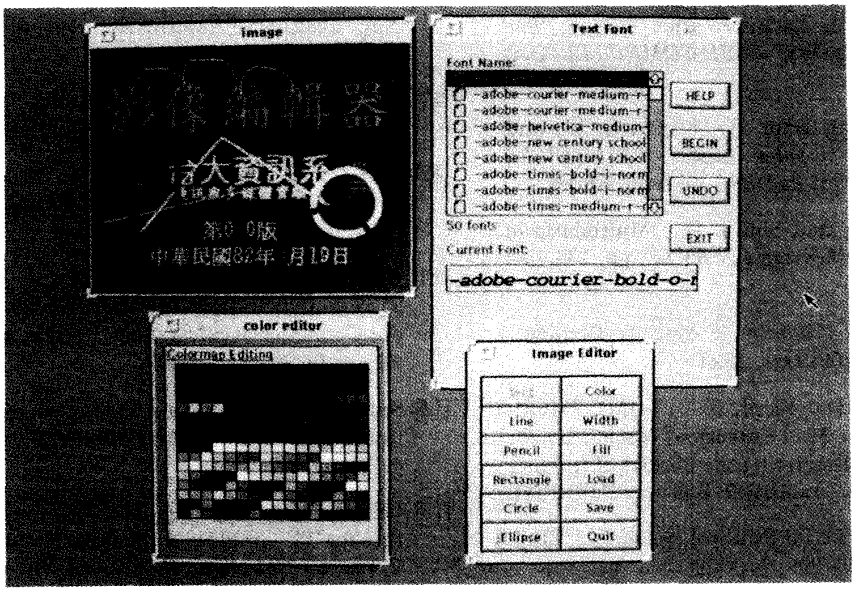

Figure 9: The Image/Graphics Editor

Openwin text editor, and cxterm plus celvis (for Chinese) by forking an Xterm or a process, where cxterm is an xterm for the Chinese display, and celvis is a View like editor for Chinese. Currently, all facilities are provided both under X11R4/R5 and Sun Openwin 2.0/3.0. The forking of the editors is through a pop-up menu which is hard coded in the program. In the near future, we will see to it that the content of the menu be modified by the user in case that the text editor one wants to use is not listed in the menu.

5 Future work

The way we can go one step further, we believe, is the modeling of authoring style which concerns the issues of speed of learning, speed of use, minimizing error rate, rapid recall, and attractiveness [5]. The current constraint on sequential ordering of medium sections will be refined to arbitrary position in a letter. When a user authors a medium, he can either drag the icon of the medium directly into the letter, or input the file name of the medium directly. And the user can edit and read mail with multi-font, multi-size, and multi-color rich-text. When a user reads a mail which includes an image, he will see a reduced size image pasted on an image icon, and he can manipulate it by select the icon and display the original size data. A new content type: multipart/alternate is considered for deciding video hardware or software decoding. We can further include FAX function, and we will develop a multimedia database which is more intelligent in managing media by query language. Currently, we apply this technique to more applications, such as multimedia bulletin board system (BBS).

6 Conclusion

Multimedia E-mail is an emerging application-oriented technology embracing many computer disciplines, including interface design, networks, authoring system, and digital video/audio. In this paper we propose and implemented the MOS E-mail system and we hope this system can serve more and more users.

References

[1] Perry, T. S., Adam, J. A., "E-mail: pervasive and persuasive," *IEEE Spectrum*, pp. 22-29, Oct. 1992.

[2] *Desktop SPARC, DeskSet Reference Guide*, Sun Microsystem Inc.

[3] Barkakati, N., *UNIX DESKTOP GUIDE TO OPEN LOOK*, SAMS, 1992.

[4] Narasimhalu, A. D., Christodoulakis, and Stavros, "Multimedia Information System: The Unfolding of a Reality," *IEEE COMPUTER*, Oct, 1991, pp. 6-8.

[5] Foley, J. D., van Dam, A., Feiner, S. K., and Hughes, J. F., *Computer Graphics PRINCIPLES AND PRACTICE, 2nd Ed.*, Addison Wesley.

[6] Englowstein, H. and Smith, B., "Multiplatform E-Mail: Mixed Messaging," *BYTE*, pp. 136-154, Mar. 1993.

[7] Reinhardt, A., "Smarter E-Mail Is Coming," *BYTE*, pp. 90-109, Mar. 1993.

[8] Borenstein, N. and Freed, N. , "MIME (Multipurpose Internet Mail Extension) Mechanism for Specifying and Describing the Format of Internet Message Bodies," Jun. 1992, Bellcore, RFC-1341.

[9] Sollins, K. R., "TFTP Protocol (version 2)," Jun. 1981, MIT, RFC-783.

[10] Postel, J. B., "Simple Mail Transfer Protocol," Aug. 1982, USC/Information Science Institute, RFC-821.

[11] Sirbu, M.A., "Content-Type header field for Internet message," Mar. 1988, CMU, RFC-1049.

[12] Postel, J. B. and Reynolds, J. K., "File Transfer Protocol," Oct. 1985, USC/Information Science Institute, RFC-959.

[13] Huang, H. C., Huang, J. H., Wu, J. L., "Real-Time Software-Based Video Coder for Multimedia Communication Systems," *ACM Multimedia 93*, pp. 65-73, Aug. 1993.

[14] Schnofr, P., "Integrating Video into an Application Framework," *ACM Multimedia 93*, pp. 411-417, Aug. 1993.

[15] Lei, Y. W., Ouhyoung, M., "A New Architecture for TV Graphics Animation Module," *IEEE Trans. on Consumer Electronics*, Vol. 39, No. 4, pp. 795-799, Nov. 1993.