FEATURES OF THE MARRS COMPUTER CONFERENCING SYSTEM

by

RONALD M. JANNING

B. S. University of Dayton, 1980

A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Computer Science

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1986

Approved by:

R A McBride

Major Professor

CONTENTS

THIS BOOK WAS BOUND WITHOUT PAGE A2-52.

THIS IS AS RECEIVED FROM CUSTOMER.

LIST OF FIGURES

Acknowledgements

I would like to dedicate this work to my wife, Janet, and my children, Christina and David. Without their love and understanding, this work would not have been possible. Also, I would like to thank my advisor, Dr. Rich McBride, for all of his help and direction.

"Writing is the more personal form
of communication, the one which permits
the most natural expression of feeling.
The message, once detached, can cross
time and space, acquiring objectivity,
permanence and mobility."

by Andrew Feeberg
Western Behavioral Sciences Institute

1. INTRODUCTION

This paper presents MARRS (MAster's Report Reviewing System), a conferencing system, designed to provide a mechanism for tracking Master's Reports through the various phases or revision. The system was implemented on a UNIX 1 based system. The features of the MARRS computer conferencing system are discussed in this paper.

The services provided by MARRS are simple and easy to use by a community of users. Papers are the major element in the system. They are grouped into conferences. The major professor controls the papers within the conference from the time of submission until the time that they are deemed acceptable for publication. During this process, the major professor has the opportunity to name a list of committee

---

1. *UNIX is a Trademark of AT&T Bell Laboratories

and audience members to act as reviewers for the paper. During the "committee review" phase committee members, which automatically includes the major professor, and the author of the paper have an opportunity to comment on the paper. During the "final review" phase audience members are permitted to also comment on the paper. Access to the paper and reviewers' comments are controlled by the major proressor and MARRS.

There are several economic and personal benefits as a result of using a computer conferencing systems. Some of the advantages include:

- Asynchronous computer conrerences messages can be sent at any time. A user therefore, can choose the period when rates are the lowest,
- Computer conrerencing impacts significantly on the conciseness or clarity of answers. It allows users to carefully consider their responses to questions,
- By using a computer conferencing system, the individual can focus on certain topics or discussion. Allowing thus, the user to avoid topics that might not hold any interest to them,
- In a research project conducted by Institute for the Future, it concluded that 80 percent of all computer

conferencing sessions for a user lasted less than 10 minutes. Therefore, allowing the conference participant more time for other problems,

- Computer Conferencing structures a meeting in such a way that its purpose and content are not lost,

- In the context or a group meeting everyone is given the opportunity to offer their 'two-cents' worth of response. More information is then available for the discussion.

For a computer conferencing system to be a replacement for face to face meetings it must support various forms of communication and various types or conferences. The second chapter discusses the available features in a computer conferencing system along with several examples from existing systems.

A computer conrerence emulates features which are thought to be necessary from an actual conrerence, and also provides additional computer-based tools. This paper is geared towards the features used for the application of the procedures surrounding a Master's Report Reviewing System (MARRS). The use of MARRS can aid the collection, the review, and the presentation of a Master's Report. The third cnapter discusses the features or MARRS.

The conclusion, chapter four, presents suggestions for future enhancements to the MARRS application. It also discusses ways for introducing the package to potential users.

There are several terms that the user should become familiar with before preceding.

- Teleconferencing system - "interactive group communication through any electronic medium", this system can be either asynchronous or synchronous.
- Asynchronous teleconferencing system - a "store and forward" conferencing system in which the participants need not all be present at the same time. (The electronic medium is usually a computer where the information can be stored and retrieved at the convenience of the user).
- Conference system administrator - the initiator of the system, and also the person responsible for the overall performance of the system.
- Major professor - the coordinator of the Master's Reports.
- Author - the writer of a paper dealing with a Master's Report.

- Committee member - the judges of the paper.
- audience - the participants of a conference
- permissions - determination if the user can access commands or files within a conrerence according to their user's login name.

The following assumptions have been made and will be carried into the design pnase or this project:

1. The choosing of the committee members is not an application that needs to be part of this system,
2. The end product will run on the Kansas State University VAX system, which is operating with Berkeley UNIX 4.2,
3. It is not a requirement of the design or coding of this project that the final product be portable,
4. The end product will be accessible for the public,
5. An individual VAX user login name is mandatory in order to use the conrerencing system,
6. The project is geared toward a self-documented application,
7. The implementation or this project is geared towards the use of a terminal that is supported by the

Berkeley curses library routines.

2. FEATURES: THE WHAT OF COMPUTER CONFERENCING

2.1 Basic Conferencing Features

For a computer conferencing system to be a replacement for face to face meetings it must support various forms of communication and various types of conferences. In other words, the computer conferencing system must be adaptable to the user's needs. The various forms of communication include video and audio, and the various types of meetings include sales, design, reports, requirements, reviews, etc. There is no standard format for live conferences and the same can also be said for computer conferences.

Computer conferences are flexible. An example of this is the VMSHARE system designed by an IBM user's group in 1976. In an article written by Charles Daney he reported that in 1981 there were 300 participants on the system from North America and Western Europe. At that time there were more than 1100 sessions a month, along with 800 different files in the database. The developers of VMSHARE oriented their system around the collection of various types of files. The types of files that were created include; MEMO, PROB, REPORT, NOTE, LOC, SCRIPT, LISTING, PROC, ADDRESS, HELP, RES, and MAIL. The type of a file determines the scope, the physical characteristics, and which commands can operate on

it. By using various types of files the VMSHARE user is able to conduct a conference. [DANE81-117]

2.1.1 The Paper

No matter what the form of communication or the type of meeting, conferences all have one thing in common and that is the papers. The presentation of papers is the major element of any conference. One of the major facilities of the computer conference is the gathering feature. With the gathering feature you can collect all the personal memos, mail, and conference discussion comments and organize them to make a complete conference. [CROS84-39] Often the order of papers is important. Sometimes for a clear definition of a project, one paper is a predecessor of another and must be presented prior to the release of its successor. This feature can be done in computer conferencing by controlling access to a conference's files.

2.1.2 The Peripherals

Along with the presentation of the papers there are many peripheral features that co-exist at a conference. These features range from meeting notices to the final meeting notes.

2.1.2.1 Meeting Notices

Meeting notices inform possible participants of the date, time, title, and contents or a conference. In case the user forgets or a conference the reminder services provide a means or informing computer conrerence users of a conference in progress. By reviewing the user's status in a conrerence, the system can determine if the user is active or idle. The determination of whether or not a participant is idle can be used to awaken the user who has forgotten the conference is in review or has never received the initial notice or the conference.

2.1.2.2 Participants List

A list or participants is a useful way to allow conference attendees to determine exactly who is present. Such a list often includes the biographies of the people. The computer can use the list of participants and the biographies to tailor the user's capabilities. The initial user's profile value can be set by the coordinator for basic usage of the conferencing system. As the user's mode of interaction increases the user can change some of their own profile values. [WILL81]

2.1.2.3 Bulletin Boards

Bulletin board messages provide notices to conference participants. UNIX supports two type of bulletin board

facilities for the Computer Buffered Information Exchange (CBIE) conferencing system. The CBIE system allows users to organize communication into a network of items. The bulletin board facilities supported in CBIE are the message of the day (motd) and news. The /etc/motd file is updated by the administrator of the UNIX system and is displayed automatically whenever the user logs in. The news facility or UNIX provides a reminder to the user concerning the presence or items. As the user reviews the news, the facility keeps track of the articles that the user has seen and doesn't display the item a second time. In the CBIE system this method is used on a per conference basis. [STR082-290]

2.1.2.4 Personal Note Pad

Scratch paper provides a conference participant with an area in which to take notes. These notes are for personal information, or to ask questions or the speaker or another participant later. The Electronic Information Exchange System (EIES) provides a sophisticated notebook feature. In the system, the user can enter a note on any line of a multiple page private notebook area. The notebook is stored in the same format as a conference item and the pages can easily be converted from a notebook page to a conference item. Along with this feature are one-line reminders. These provide the means of informing the user about the

presence of notes. AUTONOTE is another conferencing system that provides a sophisticated notebook feature. AUTONOTE's specialty is the facility to organize the notes in an hierarchical order. By storing paragraph length notes in an hierarchical form, the system provides a history of conference notes in a network format. [STR082-290]

2.1.2.5 Mail

Mailboxes are used at conferences to provide access to conference attendees. The Engel office prototype system includes a mail facility which has the ability to define mailing address distribution lists for users, allow special handling of messages (e.g., encryption of messages), and display a subject line for users to determine if they need to read an item. A "hold" queue is available for users to place messages in memory to be displayed at a later date. Also, the user can pass mail items off to other users after comments have been added. [STR082-291]

Often the users or a computer system will misunderstand the difference between electronic mail system and computer conferencing system. The difference lies in the handling of the data/text. The mailing system is a private, single transmission of a message and receipt of a reply. Typically, if the recipient does not save the message after reading it, the message disappears. However, the computer conference system keeps a current account of all

transactions. The computer conference system keeps all entries as a summary of the proceedings. Computer conference systems are group oriented while mail is person to person. [JOHA84-168]

2.1.2.6 Discussions

Discussions allow conference participants the chance to exchange ideas and to express viewpoints openly. Darrell Icenoge, Director of Educational Resources at Western Behavioral Science Institute had this to say about computer conferencing and the ability to discuss:

> "A computer conferencing system provides for complex interactions among a group of people by storing the communications on a system in one place. Any part of the discussion can be retrieved at will, enabling an individual to reconstruct the meeting at a convenient time and direct comments to the specific part of the discussion that is of interest or important at the moment." [CROT83-30]

Therefore, enhancing the discussion by creating interaction of conference participants where it is needed.

As computer conferencing begins to involve more multi-line, synchronous discussions an interesting piece of information for both face to face meetings and teleconferencing is the equal-time resolution rule. This rule seeks to promote a balance in participation. On the Apple II Plus computer, the DIALOG program limits turns of a speaker to values

entered at the beginning of the conrerence. The equal-time resolution rule resolves conflicting request for access in favor of a person who has the current least time. The responsibility of cutting off a person who would talk a long time or one who would interrupt often would be controlled by the group or participants rather than either the conference moderator or the next speaker. [STOD84-411]

One common occurrence at most conferences is conflicts. Conflict resolution is the process by which opposing sides try to resolve their differences by discussing various solutions to a problem. In a study done by Levi and Benjamin in 1977, they developed a process by which the opposing sides would rate solutions to conflicts on a scale from -10 to +10. This rating system provides opposing sides a good look at the progress or negotiations. Levi suggests that a computerized conferencing system be used to implement the model. He feels that by taking the human intervention out or the conflict, the computer assures the reliability of the model regardless of the skill of the participant. More attention can therefore be paid to resolving conflicts as they arise. [TUR080-411]

2.1.2.7 Voting

Voting is used at reviews where the reviewers are asked to rate the quality of work. On a computer conferencing system

these votes can be anonymous and when management is involved the votes could be weighted.

2.1.2.8 Meeting Notes

Meeting notes give a history of discussions held at the conference. The conference's note taker might miss key points made at a meeting. The computer conferencing system remembers all interactions. For EIES, a conference secretary function was implemented in the form of HAL Zilog, a Zilog Z-80 development system. HAL was used to access remote databases, perform interpretive structural modeling, update various bulletin boards, remember meeting notices, and keep all participants informed. HAL was considered by all the EIES users to be just like a regular member of EIES. [STR082-304]

2.2 Computer Conferencing features

Computer conferencing systems can offer additional facilities to assist in a conference's various forms of communication.

2.2.1 Data Facilities

Data facilities provide for efficiency of access, and a means for querying on keywords. As data access and modeling tools become more efficient, participants can consult a computer-based model to evaluate the alternative proposals.

[JOHA84-170]

2.2.2 Online Documents

Online training documents provide users with the ability to learn at their own pace. Online documents also provide a means of communication while using the conferencing system. In one of EIES's applications, Peter and Judy Johnson-Lenz designed a system for the Joint Electronic Devices Engineering Council to provide competing companies with a tool to jointly agree on industry-wide standards for products. The application is entitled TERMS, and it provides the following online document:

> +TERMS has been designed to increase the ease and efficiency by which glossaries of terms and definitions may be developed by a geographically dispersed group or people working together over EIES. This special sortware has the following features of particular interest:
>
> - any glossary member may add a term or an alternative proposed definition for a term.
>
> - any glossary member may enter a written comment about a proposed definition.
>
> - participants may vote on the alternative definitions; immediate feedback of tabulations of their preferences is available using a single command.
>
> - all information related to a specific term may be retrieved, including all definitions, comments,

and results of voting, using brief, concise commands.

The TERMS package provides for many of the facilities discussed in this paper; text editing, scratch pad note taking, voting, and data file access. The online document not only informs but can also lead the participant through the use of the system by prompting for inputs. [TUR080-409]

2.2.3 Transcript Generation

The capability to generate a transcript using a computer conferencing system has its definite advantages. An example of this is the work of the International Institute for Applied Systems Analysis (IIASA). This is a company that works with seventeen nations which comprise the National Member Organizations. Much of IIASA's research is done in cooperation with other research organizations around the world. In one of IIASA's projects, a series of books had to be written on international communication tools. Mike Pearson, an employee of IIASA, was chosen as the editor of the book. He said his first experience with computer conferencing was "computerized manuscript conferencing." This feature is the facility of joint authorship of manuscripts.[PEAR81-130] By enabling the participation of several nations in the creation of the books, the end product is usually a more thorough design.

2.2.4 Anonymity

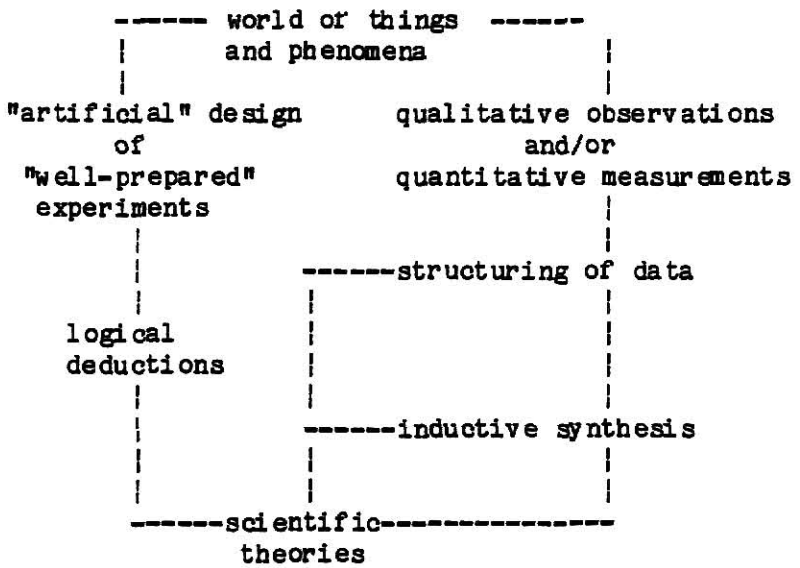
Providing unidentified input allows anonymity of questions and comments. Examples of conferencing systems that used anonymity are OEP and FORUM. OEP, Office of Emergency Preparedness, was one of the first computer conferencing systems developed in 1970. The other conferencing system FORUM/PLANET, developed at the Institute for the Future, was an aovanced system designed for persons with no previous computer experience. Both were systems designed to facilitate Delphi conferences, in which all of the participants of the conrerence enter anonymous entries. This mode or operation is helpful when there are group discussion on sensitive topics. [STR080]

2.2.5 Typesetting Facilities

Typesetting facilities are used in preparing conference papers for publication. This feature provides the means for the simplified inputing of equations or tabular material to be presented at a conference. Reformatting output to the characteristics of different users' terminals is another much needed feature of the typesetting facility since the computer conferencing system must compensate for different page widths, etc. [STR080]

2.2.6 Video

Graphics allows the art diagrams to be finished on time. In a review of the EIES Teleconferencing Network, Lucien Gerardin, the Research Director of Thomson-CSF, describes how the capability of graphs is possible. By combining letters and signs on a teletype he gives the following concrete example:



Gerardin goes on to describe in the article the need for advancement in the displaying of graphics. As new advancements are discovered they will enhance the usage of computer conferencing by giving it the capability to allow the author or a paper to describe in a picture, as well as in words, the ideas or the paper. [GERA81-625]

2.3 The Disadvantages or Computer Conferencing

Teleconferencing has both negative and positive aspects. To ignore the negative aspects of any new tool could be dangerous.

2.3.1 Social Interface Problems

The computer conference cannot replace the social aspect and camaraderie that the face to face conference provides.[CROS84-38] Lack of personal contact leads to lower morale. A large fear or the non-scientific community is the dependency of our society on the new technology and how it makes a community vulnerable to breakdown and sabotage.

2.3.2 Unproductivity of the Project

Sometimes a computer conference will increase unproductivity. This is due to time spent in unnecessary conferences in which items could have been resolved quickly in a face to face meeting.

2.3.3 Harness or Control

Another negative aspect or computer conferencing is due to management's new awareness of meetings. They can easily watch the progress or conferences within their group. Therefore, there is a decrease or freedom among the group because of too much control. Since management is

"watching", comments may become overspecialized and narrow so that a participant looks "good" in the sight of management. This brings only harm to the review of papers.[JCHA84-370]

2.4 The Advantages or Computer Conferencing

The aavantages or the computer conferencing system can best be summarized by the statement of futurist Robert Theobald. He made this comment to the participants of the World Futures conference in Berlin Germany via EIES:

> "I suppose the most dramatic result of EIES for those who have not used it before is that the person living in a rural area of the state of Arizona is able to do as well in communication terms as anyone living in a large city. I was born in Madras, India and I see EIES as being compatible with the communications traditions of this part of the world far more than it is with those or the rich countries... I believe that EIES is part of what I call the shift to the communications era." [TUR080-415]

3. THE APPLICATION OF A COMPUTER CONFERENCING SYSTEM

3.1 The Overview

The computer conference emulates features, which are thought to be necessary, from an actual conference and also provides additional computer-based tools. Consider the situation of overseeing the completion of a software design project. A conferencing system can be used to help in the phases of the software design after the project has been committed and funded. Figure 1 shows a hierarchy of the users that are involved in such a project. The conferencing system can oversee the completion of individual assignments of the design project. The final result created from the conferencing process is added to the conference system's data base in the form of a published paper and accompanying platrorm discussions.

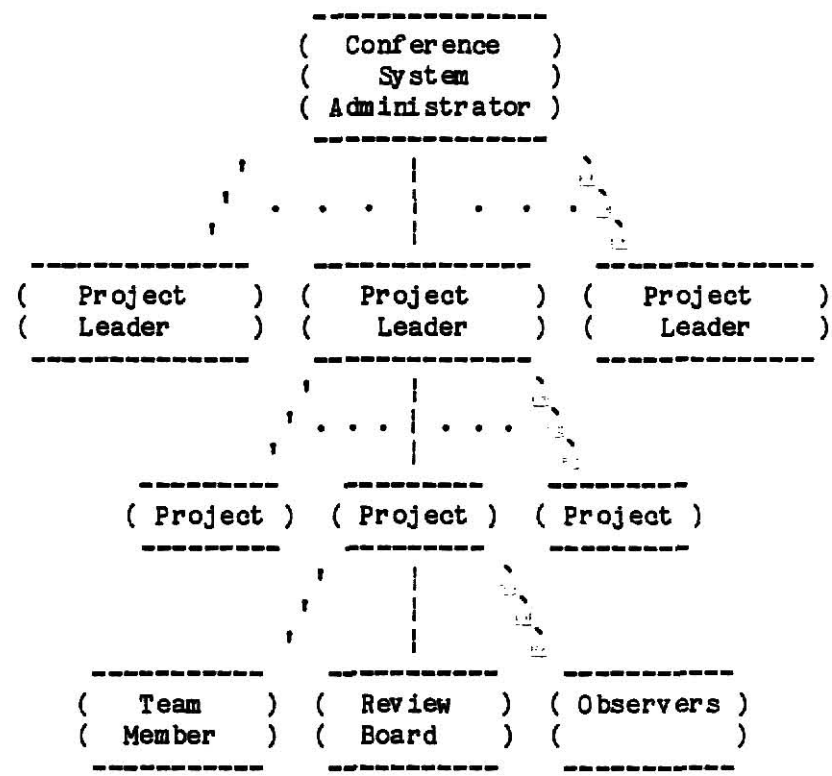


Figure 1. Hierarchy of the users within the final stages of the design process

In particular, this application is geared towards the automation of the procedures surrounding a Master's Report Reviewing System (MARRS). The use of MARRS, which is a computer conferencing system, can aid the collection, the review, and the presentation of a Master's Report. Figure 1 more clearly reflects the roles of a Master's Report reviewing system by relabeling: Project Leader to Major

Professor, Project to Master's Project, Team Member to Author(s), Review Board to Committee, and Observers to Audience.

3.2 Features Used in MARRS

Table 1 is a summarization or the features presented in the previous chapter or this paper.

- Gathering of data	- Sequencing of Presentations
+ Conference Notices	+ Reminder Services
+ List or Participants	+ Biographies
+ Environment Variables	+ Bulletin Board
+ Target Comments	+ Scratch Pad Area
+ Note Reminders	+ Mail
+ Platform Discussion	- Equal Time Resolution
- Conflict Resolution	+ Voting
- Remote DB Access	+ Keyword Querying
+ Online Documents	- TERMS
- Joint Authorship	+ Typesetting
+ Graphics	+ Text Editing

Table 1

The features that have been chosen for MARRS include the items with a "+" in front of the element. These elements were chosen due to the nature of the project. These elements provide the different roles, as depicted in Figure 1, with the capability of presenting and reviewing ideas. Additional features have been created for this application which focuses on Master's Reports. These new features include operations to: establish and update a list of professors, change the status of the paper, and create a

conference.

3.3 The Advantages of Using UNIX

The UNIX system has been chosen for the implementation. There are many reasons for choosing this system. The operating system provides a responsive system designed for interactive applications. To increase the response time, the system administrators can have the operating system limit the number of executable processes, and the number of I/O ports open. The shell routine of UNIX can be tailored to interface to users according to their needs and restrict their access to files. The shell also provides iteration, parameter substitution, I/O redirection, and pipelining. The C programming language, which is supported with UNIX, is easy to maintain, modify, and port between different systems. Additional tools that UNIX supports which the conferencing system can use are:

- Monitor File Changes (sccs),
- Typesetting Facilities (nroff, troff, eqn, tbl),
- Text Editing (edit, sed, vi, emacs, ex, spell),
- Enviroment Commands (date, stty, who, finger),
- Calculation Facilities (dc, bc),
- Office Aids (calendar, leave).

The hierarchical naming structure of UNIX with directory

nodes containing either further directories or files led to the organization or the computer conferencing system as shown in Figure 2. For details on the description or the files and directories of this organization refer to [MON86].

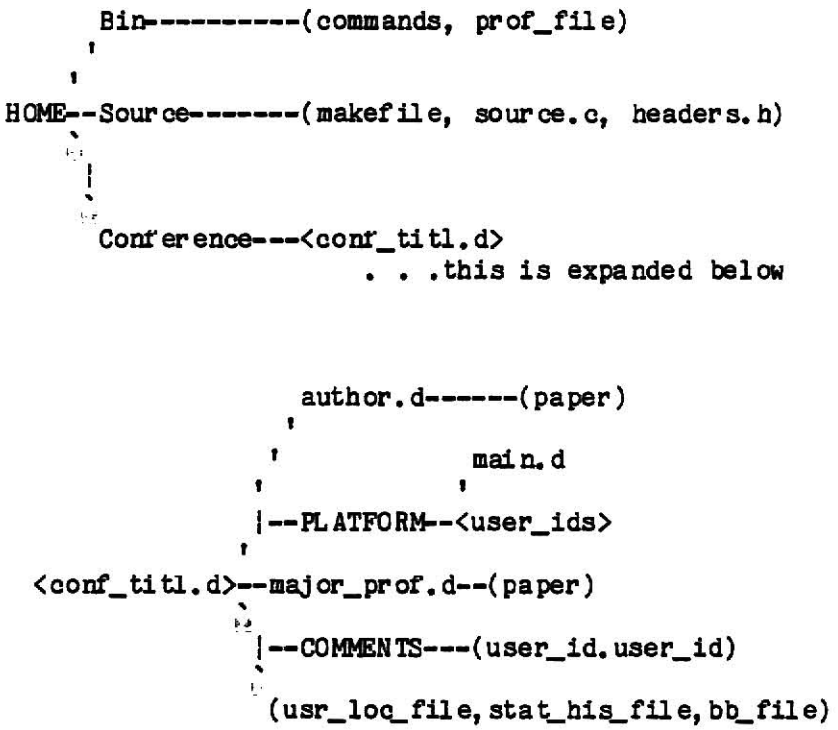


Figure 2. System Organization

3.4 Using the Mail Facility

The UNIX mail feature provides each user of the system with the capability to talk to other participants and to stay current with conference system developments. In this application, mail is used in providing five different

features. These five features include:

1) status changes of the paper,

2) the ability to wake up idle reviewers,

3) private discussions,

4) notification or comments, and

5) voting.

In the case where the paper's status is being changed, the mail provides a means for MARRS to inform the users who have a need to know. For example, consider the case where a paper is changing from committee review to final review. At that point, all participants of the conference are informed through the system's mail facility that the paper is being presented for final review and platform discussion.

Mail is also sent by a routine which acts as the time keeper routine. The purpose of the time keeper routine is to detect when a user has been idle in a conference. An idle reviewer is defined by their inactivity in reviewing a paper. After a predetermined time all idle reviewers are reminded that there is a conference undergoing review. In the case of the final review all participants of the conference are also informed of platform discussion items that are present that have not been reviewed by the user.

The mail facility also is used to support side conferences. This ability is when private sideline discussions occur

between groups or two or more conference participants. This is not a feature of the conferencing system but is automatically provided by the UNIX system. By using the mail facility conference participants save on comments entered into the platform discussion of a conference.

After reviewing a paper the user is provided the opportunity to change the comments that they have placed in their private scratch pad area. After reviewing one's comments the user then submits them to MARRS to be delivered. The recipient of the comments is notified via mail that comments have been made on a paper.

Another application of the mail facility is in the voting process of the committee. After a paper has gone through the final review, the committee members vote on the paper. A committee member has the following choices when voting:

1. Reject it,
2. Major re-work needed,
3. Minor re-work needed,
4. Acceptable for publication.

As a result of the committee's voting, the paper can go to various states. See Appendix 1 for a description of the state changes of a paper.

3.5 MARRS's Online Documentation

Online documentation gives the conference system user the ability to learn or refamiliarize oneself with the commands and procedures or the computer conference system. There are five areas that online documentation helps the user in using MARRS. These five are:

1) prompting,
2) help,
3) listings,
4) bulletin boards, and
5) error messages.

The prompting of the system provides the user with a step by step account of the valid options. However, not all valid input options are accepted. The input that the user enters is validated against their login name to see if they have access to the routine or the data files. As an example of the prompting capabilities, when the user enters the system the following display is painted on the terminal:

```
Welcome to the Kansas State University Conferencing System

     Valid options are:

          b - browse and comment a paper
          c - create a conference
          d - discuss a paper
          l - look at current conferences
          m - bulletin for conferences
          p - create and edit a paper
          q - quit the conferencing system
          s - change a paper's status
          u - update a conference's member list

      Which option? :
```

To use the options "c", "s", or "u" the user's login name must exist in the list of valid professors. The input has been created so the user only has to enter one letter for the routine to distinguish the input.

Further aid is provided through a help routine. The help feature allows a user to see what functions are accessible to them. While in the prompting mode of the system the user can enter a '?' and the routine provides any additional information that the user may need.

Listings of accessible conferences provide the user with a growing document of the status of the system. These listings not only provide the name of the conference, but also provide its status, participants, author(s), committee members, and major professor. Biographies of participants are accessible through the use of the UNIX finger command.

Another type of growing document is the bulletin board

feature. The bulletin board provides users with an area to broadcast a message to other conference members. In this application, the bulletin board remains as a flat file where any valid conference member can add or delete messages. This design decision was made in conjunction with the motd (message of the day) feature that the UNIX system provides.

3.6 Conference Status

The conference status feature provides any user with the ability to list the participants of a conference and the status of the associated paper. Along with the above mentioned listings of accessible conferences is a keyword querying feature. This feature allows the user to go through MARRS's data base looking for information related to certain keywords. Keywords include the paper's title and a list of topics that are presented in the paper. Two states of papers are provided in the data base; the first is for those conferences that have been published, and the second consists of conferences that are in the development stages.

3.7 Conference Scratch Pad

A review and scratch pad feature enables the conference participant to store notes. While reading a paper, the review command of MARRS provides a user with the ability to comment on a specific line of text. A '>' in the left hand

column and a '<' in the right hand column identify the current line or the file that the user can comment on. By entering the comment mode, the terminal screen is placed into a split screen environment for the user to enter any desired comments. When the user is finished creating the note, the comment routine prompts the user for the destination or the comment. The destination of the comment allows comments to be targeted for the appropriate conference participant(s). Any line of text that has a note attached to it will have an '*' in the right hand column, providing a reminder facility of notes.

At the conclusion or reviewing the document the user can review the entered comments. When finished with reviewing the comments, and making any changes that are needed the user can then submit the comments. As a result of the user submitting the comments, individual files are sent to other conference users as determined by the user's preference.

3.8 Conference Discussions

Platrorm discussions provide a question and answer session during the final review of the paper. The platform discussion differs from the review and comment feature. As a reviewer reads a paper, comments can be made. However, in the final review when the user is finished entering a note, the comment routine prompts the user as to whether the item

is to be targeted for a user or for the platform discussion. When an item is for the platform discussion, the routine automatically forwards the item to the Conference's PLATFORM directory. The routine sets up links to the item for any other user on the conference. Therefore, during the final review of papers all of the participants in the conference can view the global comments to the paper by reviewing the paper and can also follow a discussion that may or may not be related to the paper.

3.9 Conference Papers

The EDITOR environment parameter informs the conferencing system of the user's preference. In the case where the user has not set the environment parameter, the routine defaults to the ed editor. This facility is used for when the conference participant enters the bulletin board facility or when the author wants to edit the paper.

There must be specifications for paper entry. Nroff, a UNIX typesetting feature, has many variables and parameter settings that enhance the format of the paper. However, for MARRS the paper must be geared towards a different group of readers. A hard copy of the paper will therefore not appear in a "nice" format but is only needed to aid the author during the revision process.

To create a stable environment for reviewers, a submitted paper from an author is placed in the major professor's directory within the conference. Once the paper is finished being reviewed and there are changes that the author needs to make, then the author has the ability of resubmitting the paper. After being resubmitted, the paper is copied to the major professor's directory for more comments.

3.10 Additional features

The major professor is provided with two additional features. These features are the ability to create a conference and to change a paper's status. When the conference is created, the major professor enters the author(s) of the conference, the committee members, the audience members, and the conference name. As a result of the creation or the conference many files in the system are updated. The author sets the wheels of the system in motion by submitting the paper to the major professor. It is the major professor's responsibility to guide the paper through the other states. This is done by the feature to alter the status of the paper. A validation routine makes sure that the state change is proper.

The MARRS administrator is also provided with two additional features. MARRS is created by an initialization routine. As a result of the initialization routine the Bin, User, and

Conference directories are created from the conferencing system's HOME directory. The conference system administrator is also responsible for updating the professor list. This list is used to determine the validation of users who need to create conferences. This list is also used to validate the committee members id entered by the major professor during the creation of the conference.

3.11 MARRS: The System

This chapter presented the design of a mechanism for tracking Master's Reports through the various stages of revision. To summarize, the following list includes the features that were included in MARRS.

- Keyword querying is provided to search the data base for topics presented in papers and titles.
- Creating a conference allows the major professor to fill in the information needed to set up the files and directories of the conference.
- Conference notices provide all MARRS participants access to the list or open conferences.
- Reminder Services are used as the routine diagnoses all participants usage or the system.

- A list of participants show the users that are established by the major professor and the role of the user.
- Biographies provided by the UNIX utility finger routine.
- Environment variables to control the user's text editing sessions.
- Text editing is used to enter and change comments, papers, and bulletin board entries.
- Typesetting as provided by the UNIX systems nroff utility.
- Graphics provides a means or communicating through the use or pictures.
- Bulletin board provides a community information controlled area.
- Targeted comments provide reviewers the ability to send comments to the necessary people.
- Scratch pad area provides a place where comments are stored while the reviewer reads the paper.
- Note reminders inform reviewers of the presence of comments attached to a line while reviewing a paper.

- Mail provides the facility to inform users of the user's status and events in the MARRS environment.
- Platform discussion areas provide the users with an area to further discuss issues that might arise as the result or a paper.
- Voting is the final step for a committee member before the paper can be published.
- Online documents aid the user in the use of the system.
- Professors list is used to validate users who want to create a conference or else are to be appointed committee members.
- Paper status snows the various stages that the paper can be in during the reviewing process.

4. CONCLUSION

In order to introduce this Master's Report reviewing system to potential users, the advantages must be stressed. These advantages are:

- It has no geographical or time restrictions,
- It can be provided at a low cost,
- It has online documentation capabilities and filing features,
- It does not require the presenter of a topic to have any acting or performing skills,
- It allows the user to go at ones own pace,
- It provides the means for a user to participate in many conferences at one time.

There are some features that could be added to MARRS to enhance the system.

VORTEX, the use of voice synthesis, provides users the ability to call into a computer and receive messages via the phone without the aid of a terminal.

If this project were to tie into the library network it could provide additional resources to the author.

Currently, at Kansas State University, there is a graduate study data base that this project could interface with in order to provide additional biography background on the author.

One item that was hinted at in the body of this paper, but not dealt with is the presentation of the paper, namely how to standardize the formatting of papers for use with both the line printer and the terminal screen.

A more sophisticated bulletin board facility could be added to the MARRS application to promote intergroup communication.

Synchronous computer conferencing was not covered in this paper but could be used to enhance the side line or platform discussions.

In conclusion, computer conferencing is a relatively new tool developed to give aid in the work environment. Systems that are developed to handle conferences must be adaptable to its users' needs. If introduced correctly they will provide both personal and economic relief.

REFERENCES

1. [CROS83] Cross, Thomas B., "The Grapefruit Diet", Journal of Micrographics, April, 1983, pp. 14-18.

2. [CROS84] Cross, Thomas B., "Computer Conferencing", Computerworld on Communications, August 1, 1984, pp. 37-39.

3. [CROT83] Cross, Thomas B., "Computer Teleconferencing and Education", Educational Technology, April, 1983, pp. 29-31.

4. [DANE81] Daney, Charles, "The VMSHARE Computer Conferencing Facility", Computer Message System, Uhlig, R. P. (editor). North-Holland Publishing Company. IFIP, 1981.

5. [DOCK84] Dock, Patricia, "Designs and Implementing a Computer Conferencing System to Manage and Track Articles Through the Revision Process", Master's Report, Department of Computer Science, Kansas State University, 1984.

6. [GERA81] Gerardin, Lucien, "The E.I.E.S. Teleconferencing Network: A User's Viewpoint", Networks From the User's Point of View, L. Csaba, T. Szentivanyi, K. Tamay (eds.) North-Holland Publishing Company, IFIP, 1981, pp. 617-627.

7. [JOHA84] Johansen, Robert, Bullen, Christine, "What to expect from teleconferencing", Harvard Business Review, March-April 1984, pp. 164-174.

8. [MONK86] Monk, Kitty, "Data Management in MARRS", Master's Report, Department of Computer Science, Kansas State Univerisity, 1986.

9. [OLGR83] Olgren, Christine H., Parker, Lorne A., "Teleconferencing: Technology and Applications", Artech House Inc., 1983

10. [PEAR81] Pearson, Michael M. L., Kulp, James E., "Creating An Adaptive Computerized Conferencing System On UNIX", Computer Message System, Uhlig, R. P. (editor). North-Holland Publishing Company. IFIP, 1981.

11. [STOD84] Stodolsky, David Sanders, "Equal-Time Resolution Program For Dialog Management", Behavior Research Methods, Instruments and Computers,

Psychonomic Society, Inc., 1984, p. 411.

12. [STR080] Strom, Bernard Ivan, "A Multi-Copy Structured Database Computer Conferencing System", PhD Dissertation, Columbia University, 1980.

13. [STR082] Strom, B. Ivan, "Computer Conferencing - Past, Present, And Future", Office Information Systems, Naffah, N. (editor). INRIA/North-Holland Publishing Company, 1982.

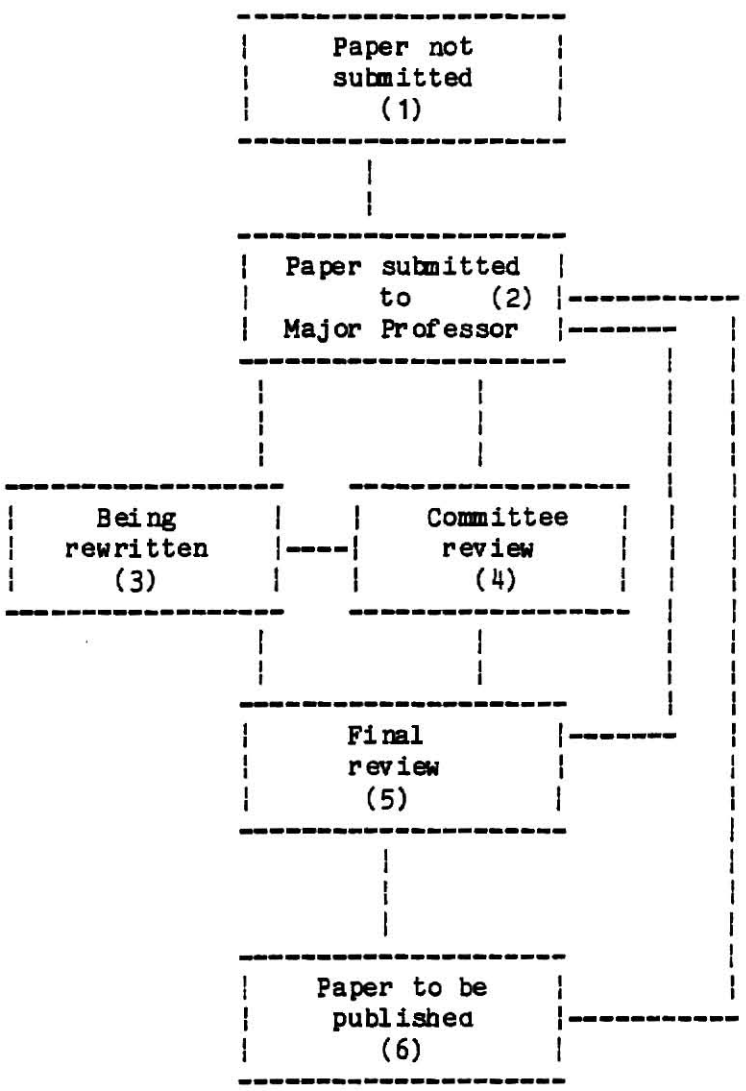
14. [TUR080] Turoff, Murray, Hiltz, Starr Roxanne, "International Potentials of Computerized Conferencing", The Teleconferencing Resource Book: A Guide to Applications and Planning, Lorne A. Parker and Christine H. Olgren (eds.) Elsevier Science Publishers B.V. (North-Holland) IFIP., 1984, pp. 402-417.

15. [WILL81] Williamson, Hilary, Rohlfs, Sabine, "The User Interface Design Process", Computer Message System, Unlig, R. P. (editor). North-Holland Publishing Company. IFIP, 1981.

Appendix 1

State Diagram

Appendix 1 - State Diagram



Appendix 1 (con't)

State Diagram Description

A paper is initialized to "Paper not submitted" (1). When ready, the author can submit the paper to the Major Professor, causing the state to change to (2). After reviewing the submitted paper, the Major Professor can send it back to the author for revisions (3) or to the committee for review (4). In addition, the paper could be sent to the final review (5) or sent to be published (6) if it is ready and it has previously been in states (4) and (5). After being revised (3), an author must submit it back to the Major Professor (2).

After committee review (4), the Major Professor can send it back for revisions (3) or schedule a final review (5). Following the final review the Major Professor can reject it by sending it back to be rewritten (3) or accept it for publication (6).

On the diagram, arrow 1 -> 2 and arrow 3 -> 2 represent state changes due an author's action. All other arrows are a result or a Major Professor's action.

As status changes occur for each paper, the time, date, and new status is recorded in a file. These files show the current status and a concise history of each paper.

Appendix 2

Application Code

```
#
#
#    M A K E B
#
#
cc browse.c libmarrs -lcurses -ltermcap
mv a.out /usrb/att/marrs/Bin/review
```

```
#
#
#       M A K E R
#
#
cc remind.c libmarrs
mv a.out /usrb/att/marrs/Bin/remind
```

```
#
#
#     M A K E L I B
#
#
cc -c inform.c      ; ar rv libmarrs inform.o
cc -c is_comm.c     ; ar rv libmarrs is_comm.o
cc -c list_conf.c   ; ar rv libmarrs list_conf.o
cc -c list_pitem.c  ; ar rv libmarrs list_pitem.o
cc -c type_item.c   ; ar rv libmarrs type_item.o
cc -c valid_item.c  ; ar rv libmarrs valid_item.o
cc -c add_item.c    ; ar rv libmarrs add_item.o
cc -c subs.c        ; ar rv libmarrs subs.o
cc -c usr_conf.c    ; ar rv libmarrs usr_conf.o
rm *.o
```

```
/*
 *
 *
 *      D I S P L A Y . C
 *
 *
 */

#include <sys/file.h>
#include <stdio.h>
#include "/usrb/att/marrs/Source/conf.h"
char flag;
char prof_flg;
char *str, string[10];

/*****************************************************************
 *                                                               *
 * Display.c is the main menu display routine that handles       *
 * the verification that the user is a professor or just         *
 * another user, and then according to their status display      *
 * a menu and when an option is entered create the system        *
 * call for the proper module.                                   *
 *                                                               *
 *****************************************************************/

main()
{
    char c,login_name[FILENM_MAX];
    int old_mask;

    old_mask = umask( 000 );

    /* get the user's login name */

    /*strcpy(login_name, getlogin());*/
    /*temporary code to be removed at kansas live system*/

    CLEAR
    printf("Ore you a major professor ( y or n ) ? ");
    gets(login_name);

    /*if( validate_prof( login_name ) == 0 )
        prof_flg = TRUE;
    else
        prof_flg = FALSE;*/

    if( login_name[0] == 'y' ) prof_flg = TRUE;

    flag=FALSE;

    while( TRUE )
```

```
    {
        header();
        gets( string );
        if( routines( ) ) break;
        sleep(1);
    }
    CLEAR
}

    umask( old_mask );

header()
{

    /* this is the display of the front page of the system */

    CLEAR
    if( !flag )
    {
        printf( "        Welcome to the Kansas State ");
        printf( "University Conferencing System");
        flag = TRUE;
    }

    printf("        Valid options are:0);
    printf("                b - browse and ");
    printf("comment a paper0);
    if( prof_flg )
        printf("                    c - create a conference0);
    printf("                d - discuss a paper0);
    printf("                g - generate a hard copy0);
    printf("                k - keyword search0);
    printf("                l - look at current ");
    printf("conferences0);
    printf("                m - bulletin for conference0);
    printf("                p - create and edit a paper0);
    printf("                q - quit the conferencing ");
    printf("system0);
    if( prof_flg )
    {
        printf("                    r - remind idle users0);
        printf("                    s - change a paper's status0);
        printf("                    u - update a ");
        printf("conference's member list0);
    }
    printf("    Which Option? --  ");
}

routines( )
{
    char outstr[ 50 ];
```

```
char err_flg;

err_flg = FALSE;
switch( string[ 0 ] )
{
    case 'b':
        CLEAR
        system( "/usrb/att/marrs/Bin/review" );
        break;

    case 'c':
        if( !prof_flg )
        {
            err_flg = TRUE;
            break;
        }
        CLEAR
        system( "/usrb/att/marrs/Bin/crcon" );
        break;

    case 'd':
        CLEAR
        system( "/usrb/att/marrs/Bin/platform" );
        break;

    case 'g':
        CLEAR
        system( "/usrb/att/marrs/Bin/pr_com" );
        break;

    case 'k':
        CLEAR
        system( "/usrb/att/marrs/Bin/query" );
        break;

     case 'l':
        CLEAR
        system( "/usrb/att/marrs/Bin/lookcon" );
        break;

     case 'm':
        CLEAR
        system( "/usrb/att/marrs/Bin/bb" );
        break;

    case 'p':
        CLEAR
        system( "/usrb/att/marrs/Bin/paper" );
        break;

     case 'q':
```

```
            break;

        case 'r':
            if( !prof_flg )
            {
                err_flg = TRUE;
                break;
            }
            CLEAR
            system( "/usrb/att/marrs/Bin/remind" );
            break;

        case 's':
            if( !prof_flg )
            {
                err_flg = TRUE;
                break;
            }
            CLEAR
            system( "/usrb/att/marrs/Bin/status" );
            break;

        case 'u':
            if( !prof_flg )
            {
                err_flg = TRUE;
                break;
            }
            CLEAR
            system( "/usrb/att/marrs/Bin/updmem" );
            break;

        default:
            err_flg = TRUE;
            break;
    }
    if( err_flg )
    {
        printf("0ou entered an invalid character; ");
        printf("try again");
        printf("0);
        sleep(1);
    }
    if( string[0] == 'q' ) return( TRUE );
    return( FALSE );
}
```

```
/*
 *
 *
 *       R E M I N D . C
 *
 *
 */

#include <stdio.h>
#include <sys/time.h>
#include <sys/file.h>
#include <pwd.h>                /* for getpwnam call */
#include "/usrb/att/marrs/Source/conf.h"

char login_name[MAX_LOGIN_SIZE];
struct usr_loc_file  usr_loc [ CON_MAX ];
struct stat_his_file stat_his[ PAPER_MAX ];
struct loc_file *usr_pos;

/*********************************************************
 *                                                       *
 *  The remind routine is used to tell idle users that*
 *  they have a limited time to review the paper till *
 *  the paper will change status.                        *
 *                                                       *
 *  An idle reviewer = reviewers location in file = 0 *
 *  Design decisions: reviewers different for differen*
 *  paper status.  Only allow the major professor to  *
 *  remind committee members during committee review  *
 *  and then remind audience and committee members    *
 *  during the final review.                             *
 *                                                       *
 *********************************************************/

main(argc, argv)
int argc;
char *argv[];
{
    int fdes, i, j, ii;
    int num_usrs, num_papers;
    char conf_path[150];        /* path name to conference */
    char conf_name[100];        /* name of conference */
    char auth_string[100];
    char text[20];
    char auth_name[LGN_SZ ];            /* name of author */
    struct save
    {
        char auth_nm[ LGN_SZ ];
        char paper;
        short status;
    } store[ PAPER_MAX ];
```

```
/*First get the user's login name so it can be checked out */
/*The person running this program must be in the professor */
/* file because the person becomes the Major Professor  */

    strcpy(login_name, getlogin());

    printf("1440ho do you want to execute this routine as? ");
    gets(login_name);

    switch ( validate_prof(login_name) )
    {
        case -2:
            fprintf(stderr,
                "ERROR: Could not open professor file0);
            exit(-1);

        case -1:
            fprintf(stderr,
                "Could not find user in professor file0);
            exit(-1);
    }

    CLEAR

    printf("                        REMIND IDLE USERS0);

      /* list all conferences for which the
         user is the MAJOR PROFESSOR */

    switch ( usr_conf(login_name, MAJ_PROF, conf_name) )
    {
        case -1:
            exit(-1);
            break;

        case 0:
            exit(-1);
            break;

        case 1:
            break;

        default :
            fprintf(stderr,
                "Bad return from routine usr_conf0);
            exit(-1);
            break;
    }


    /* read in the conference's usr_loc_file */
```

```
#
#
#       M A K E D
#
#
cc display.c
mv a.out /usrb/att/marrs/Bin/marrs
```

```
sprintf(conf_path, "%s%s.d/%s", CONF_HOME,
                                conf_name, USR_FILE);
fdes = open( conf_path, O_RDONLY, 0 );

if(fdes == EOF)
{
    fprintf(stderr, "Can't open %s0, conf_path);
    close( fdes );
    exit(-1);
}
if(( num_usrs = read( fdes, usr_loc, sizeof( usr_loc ) ) ) <= 0 )
{
    printf("Error in reading %s0, conf_path );
    close( fdes );
    exit( 0 );
}
close( fdes );

num_usrs = num_usrs / ( sizeof( struct usr_loc_file ) );

/* read in the conference's stat_his_file */

sprintf(conf_path, "%s%s.d/%s", CONF_HOME,
                                conf_name, STAT_FILE);
fdes = open( conf_path, O_RDONLY, 0 );

if(fdes == EOF)
{
    fprintf(stderr, "Can't open %s0, conf_path);
    close( fdes );
    exit(-1);
}
if( ( num_papers = read( fdes, stat_his,
                                sizeof( stat_his ) )) <= 0 )
{
    fprintf(stderr, "Error in reading %s0, conf_path );
    close( fdes );
    exit( 0 );
}
close( fdes );

num_papers = num_papers /(sizeof( struct stat_his_file ) );

if (num_papers <= 0)
{
    printf("No papers in conference0);
    exit(-1);
}
for( ;; )
{
  CLEAR
```

```
for( i = 0, ii = 0; i < num_papers; i++ )
{
  /* look backwards for current status */
  for( j = ( STAT_MAX - 1 ); j >= 0; j-- )
      if( stat_his[ i ].status[ j ].paper_stat > 0 )
          break;

  /* didn't find a current status set j to zero */
  if ( j == -1 ) j = 0;

  if((stat_his[i].status[j].paper_stat != FIN_REVU) &&
      (stat_his[i].status[j].paper_stat != COM_REVU))
      continue;

  /* get the author's name */
  get_part_name(usr_loc,stat_his[i].art_num,auth_name);

  /* get the status number's text */
  get_stat_text(stat_his[i].status[j].paper_stat,text);

  if( ii == 0 )
  {
      printf("0Conference: %s0, conf_name);
      printf("Author        Paper 0);
      printf("Status           Status Date0);
      printf("------        n");
      printf("------------           ----------0);
  }
  printf("%-8s      %-16s       %s", auth_name, text,
          ctime( &stat_his[ i ].status[ j ].date ) );
  /* store author's name and status of paper */
  sprintf( store[ ii ].auth_nm, "%s", auth_name );
  store[ ii ].paper = stat_his[ i ].art_num;
  store[ii++].status=stat_his[i].status[j].paper_stat;
}

if( ii == 0 )
{
  printf( "0here are no papers in committee or" );
  printf( "final review within 0 );
  printf( "conference <%s>0, conf_name );
  sleep( 3 );
  exit( 0 );
}

for( ;; )
{
  printf( "0nter author's name " );
  printf( "
912   printf( " 'q' to quit ):  " );
```

```
  gets( auth_string );

  if( strcmp( auth_string, "q" ) == NULL )
  {
      ii = -1;
      break;n
  }

  /* look through all the authors */

  for( i = 0; i < ii; i++ )
      if( strcmp( auth_string, store[ i ].auth_nm )
                                    == NULL ) break;
  if( i == ii )
  {
      printf("Invalid author's name0);
      continue;
  }
  else
      break;
}
CLEAR
if( ii == -1 ) break;

printf( "All idle reviewers will be sent mail!0 );

for( j = 0; j < num_usrs; j++ )
{
  if( ( usr_loc[ j ].role == AUD_MEM )  &&
      ( store[ i ].status == COM_REVU ) )
      continue;

  for( usr_pos = &( usr_loc[ j ].location[ 0 ] );
       usr_pos < &( usr_loc[ j ].location[PAPER_MAX]);
                                          usr_pos++ )
       if( usr_pos->auth_num == store[ i ].paper )
          break;

  if((usr_pos == &( usr_loc[j].location[PAPER_MAX]))||
                          ( usr_pos->usr_locate == 0 ) )
  {
      printf("idle reviewer - %s0,usr_loc[j].usr_id);
      /* TEMPORARY */
      /* if( store[ i ].status == COM_REVU )
          inform( store[ i ].auth_nm,
                          usr_loc[ j ].usr_id, IDLE_C,
                                          conf_name );
      else
          inform( store[ i ].auth_nm,
                          usr_loc[ j ].usr_id, IDLE_F,
                                          conf_name);*/
```

```
        }
      }
      if( store[ i ].status == COM_REVU )
        inform( store[ i ].auth_nm,
                        "janning", IDLE_C, conf_name );
      else
        inform( store[ i ].auth_nm,
                        "janning", IDLE_F, conf_name );
      sleep( 3 );
      if( ii == 1 ) break;
    }
}
```

```
/*
 *
 *          B R O W S E . C
 *
 */

#include <sys/file.h>
#include <stdio.h>
#include "/usrb/att/marrs/Source/conf.h"
#include <curses.h>

WINDOW *win1, *win2, *win3, *win4, *win5;

FILE *fd, *fopen();

int status, num_usrs, num_cmnts, cur_line, end_line;

char login_name[ FILENM_MAX ];
char conf_name[ FILENM_MAX ];
char ppr_name[ FILENM_MAX ];
char conf_dir[ FILENM_MAX ];
char conf_path[ 100 ];
struct usr_loc_file usr_loc[ CON_MAX ],
                    sysadmn, *user, *maj_prof;
struct loc_file *usr_pos;
struct stat_his_file stat_his[ PAPER_MAX ], *paper;
struct cmnt_hdr comm_hdr;
char comment[ MAX_CMNT ];
struct cmnt_hdr *cur_comm;
        /* assume max of 300 comments per paper
           10 commnts / participant */
struct cmnt_hdr comm_lst[ 300 ];
        /* should I send mail to user of comments */
char send_comm[ CON_MAX ];

/*************************************************************
 *                                                           *
 * MAIN - this is the routine that allows the user to        *
 * scan a text file and place comments into a separate       *
 *  file for to be sent to other participants within a       *
 * conference within MARRS.  However there are some limits   *
 * that I (rmj) have placed into the routine, although I     *
 *  do use the variable LINES as given to me from the        *
 *  curses routine I've only system tested and designed      *
 *  this package on a terminal with 24 lines, it's up to     *
 *  you to improve the performance if it is placed on a      *
 *  different screen size.  Please familiarize yourself      *
 *  with the curses library routines before trying to        *
 * maintain these routines, it may help you in understand-   *
 * ing the logic of updating the screen. Enjoy!!!            *
 *                                                           *
```

```
**********************************************************/

main()
{
    bool flag;
    int motion, sign, fdes, i, num_papers;
    char c, path[ MAX_ED_PATH ], string[ FILENM_MAX ];
    struct usr_loc_file usr_tmp, *tmp_usr;

    /* set all elements of send_comm to false */
    for( i = 0; i < CON_MAX; i++ )
        send_comm[ i ] = FALSE;

    /*initialize the sysadmn structure for error reporting*/

    sysadmn.role = SYS_ADMIN;
    sprintf( sysadmn.usr_id, SYS_ADMN );

    /* get the conference the user wants to be in */

    for(;;)
    {
        CLEAR
        printf( "The following conferences " );
        printf( "are available.0 );
        if( list_conf( NULL ) == 0 )
        {
            CLEAR
            printf( "There are no conferences created " );
            printf( "within marrs.0 );
            printf( "Therefore there are no papers yet " );
            printf( "to review. Sorry!!0 );

            sleep( 2 );
            exit( 0 );
        }
        printf( "0nter a conference name: " );
        gets( conf_name );
        sprintf( conf_dir, "%s.d", conf_name );
        if( list_conf( conf_dir ) ) break;
        printf( "0ncorrect input: try again.0 );
        sleep(1);
    }


    /* get the user's login name */

    /*strcpy(login_name, getlogin());*/
    /*temporary code to be removed at kansas live system*/

    CLEAR
```

```
printf("Oho do you want to execute the routine as?");
gets(login_name);

/* get the user's role from the usr_loc_file */

/* read in the conference's usr_loc_file */

sprintf( path, "%s%s/%s/%s", HOME, CONF_DIR,
                                        conf_dir, USR_FILE );
fdes = open( path, O_RDONLY, 0 );
if((num_usrs = read(fdes,usr_loc,sizeof(usr_loc))) <= 0)
{
    CLEAR
    printf("Error in the creation of the conference ");
    printf("%s.", conf_name );
    printf( " Sorry!!Ohe system administrator " );
    printf( " of the conferencing" );
    printf(" system will be informed.O);
    sprintf( usr_tmp.usr_id, "%s", login_name );
    inform( &usr_tmp, &sysadmn, BAD_USF, conf_name );
    close( fdes );
    exit( 0 );
}
close( fdes );

num_usrs = num_usrs / ( sizeof( struct usr_loc_file ) );
/*scan all the conference's usr_locs for the login_name*/

for( user = &usr_loc[ 0 ];
                    user < &usr_loc[ num_usrs ]; user++)
    if( strcmp( login_name, user->usr_id ) == 0 ) break;

if( user == &usr_loc[ num_usrs ] )
{
    CLEAR
    printf("You do not yet exist within conference ");
    printf("%s.O, conf_name );
    printf("Do you want to become a member of ");
    printf("the conference? ( y or n ): ");
    gets( string );
    if( string[0] == 'y' )
    {
        printf("Ohe major professor will be ");
        printf("informed.O);
        printf("You will be informed later via mail.O);
        sprintf( usr_tmp.usr_id, "%s", login_name );
        for( user = &usr_loc[ 0 ];
                    user < &usr_loc[ num_usrs ]; user++)
            if( user->role == MAJ_PROF ) break;
        if( user == &usr_loc[ num_usrs ] )
            inform( &usr_tmp, &sysadmn, BAD_USF,
```

```
                                                conf_name );
        else
            inform( &usr_tmp, user, NEW_MEM, conf_name );
    }
    exit( 0 );
}

/* read in the conference's stat_his_file */

sprintf( path, "%s%s/%s/%s", HOME, CONF_DIR,
                             conf_dir, STAT_FILE );
fdes = open( path, O_RDONLY, 0 );
if( ( num_papers = read( fdes, stat_his,
                             sizeof( stat_his ) ) ) <= 0)
{
    CLEAR
    printf("Error in the creation of the " );
    printf("conference %s.", conf_name );
    printf( " Sorry!!0he system administrator " );
    printf( " of the conferencing" );
    printf(" system will be informed.0);
    sprintf( usr_tmp.usr_id, "%s", login_name );
    inform( &usr_tmp, &sysadmn, BAD_SHF, conf_name );
    close( fdes );
    exit( 0 );
}

num_papers = num_papers /(sizeof(struct stat_his_file));

/* determine the major professor of the conference */

for( maj_prof = &usr_loc[ 0 ];
            maj_prof < &usr_loc[ num_usrs ]; maj_prof++)
    if( maj_prof->role == MAJ_PROF ) break;

if( maj_prof == &usr_loc[ num_usrs ] )
{
    CLEAR
    printf("Error in the creation of the " );
    printf("conference %s.", conf_name );
    printf( " Sorry!!0he system administrator " );
    printf( " of the conferencing" );
    printf(" system will be informed.0);
    sprintf( usr_tmp.usr_id, "%s", login_name );
    inform( &usr_tmp, &sysadmn, NO_MPF, conf_name );
    exit( 0 );
}

/* get the paper the user wants to browse */

for(;;)
```

```
{
    if( num_papers == 0 )
    {
        CLEAR
        printf( "There are no papers created within ");
        printf( "conference %s.0, conf_name );
        printf( "Therefore there are no papers " );
        printf( "yet to review. Sorry!!0 );
        sleep( 2 );
        exit( 0 );
    }
    CLEAR
    printf( "The following papers are available " );
    printf( "within conference %s.0, conf_name );
    for( i = 1, paper = &stat_his[ 0 ];
                    paper < &stat_his[ num_papers ];
                                    paper++, i++ )
    {
        printf("    %-s",paper->paper_file);
        if( !( i % 4 ) )printf("0);
    }
    printf( "0nter a paper name: " );
    gets( ppr_name );
    for( paper = &stat_his[ 0 ];
            paper < &stat_his[ num_papers ]; paper++ )
         if( strcmp( ppr_name, paper->paper_file )
                                            == 0 ) break;
    if( paper < &stat_his[ num_papers ] ) break;
    printf( "0ncorrect input: try again.0 );
    sleep(1);
}

/* validate that the user is allowed to view the paper
   the major professor can only
   review during paper_submitted,
   committee_review, and final_review
   status's.  The committee
   members can only review during
   committee_review and final
   review.  The author can review
   the paper when the paper is
   submitted, paper is in re_work,
   paper is in committee_review
   and paper is in final_review.
   The audience can only review
   the paper during the final review.
   See conf.h for the various
   states of a paper and users of the
   conferencing system. */
   /*  Note - a designers decision was
       made here that if the
```

```
        user is an author within the
        conference then they are
        allowed the same priveleges of
        any other author of the
        conference */

for( i = 0; i < STAT_MAX; i++ )
    if( paper->status[ i ].paper_stat == 0 ) break;

i--;
status = paper->status[ i ].paper_stat;

if( ! ( ( ( user->role == MAJ_PROF ) &&
                          ( ( status == PAPER_SUB ) ||
                            ( status == COM_REVU  ) ||
                            ( status == FIN_REVU ))) ||
        ( ( user->role == COM_MEM  ) &&
                          ( ( status == COM_REVU  ) ||
                            ( status == FIN_REVU ))) ||
        ( ( user->role == AUTHOR   ) &&
                          ( ( status == PAPER_SUB ) ||
                            ( status == RE_WORK   ) ||
                            ( status == COM_REVU  ) ||
                            ( status == FIN_REVU ))) ||
        ( ( user->role == AUD_MEM  ) &&
                          ( ( status == FIN_REVU )))))
{
    CLEAR
    if( status == NOT_SUB )
        printf( "The paper is not yet submitted." );
    if( status == PAPER_SUB )
        printf("The paper is in the submitted status." );
    if( status == RE_WORK )
        printf( "The paper is in the rework status." );
    if( status == COM_REVU )
        printf(
          "The paper is in committee review status.");
    if( status == PAPER_PUB )
        printf( "The paper has been published." );
    printf( "0nd as a" );
    if( user->role == MAJ_PROF )
        printf( " major professor " );
    if( user->role == COM_MEM )
        printf( " committee member " );
    if( user->role == AUTHOR )
        printf( "n author " );
    if( user->role == AUD_MEM )
        printf( "n audience member " );
    printf("you do not have permission0);
    printf("at this time to " );
    printf("browse the paper. Sorry!!0 );
```

```
    sleep( 4 );
    exit( 0 );
}

/* get the comments that have
   been attached to this paper for
   this user. */

gath_comm();

/* set the user loc file
    to the role of the people
    to be non reviewers depending
    upon the status of the paper,
    this will be used later
    on when the reviewer wants
    to target comments. */

for( tmp_usr = usr_loc;
            tmp_usr < &usr_loc[num_usrs]; tmp_usr++ )
{
    if( ! ( ( ( tmp_usr->role == MAJ_PROF ) &&
                              ( ( status == PAPER_SUB ) ||
                                ( status == COM_REVU  ) ||
                                ( status == FIN_REVU ))) ||
            ( ( tmp_usr->role == COM_MEM  ) &&
                              ( ( status == COM_REVU  ) ||
                                ( status == FIN_REVU ))) ||
            ( ( tmp_usr->role == AUTHOR   ) &&
                              ( ( status == PAPER_SUB ) ||
                                ( status == RE_WORK   ) ||
                                ( status == COM_REVU  ) ||
                                ( status == FIN_REVU ))) ||
            ( ( tmp_usr->role == AUD_MEM  ) &&
                              ( ( status == FIN_REVU ))))
    {
        tmp_usr->role = NON_REVUR;
    }
}

/* check to see if the user
    wants to continue where they left off from */
/* the end result is to ensure
    that the cur_line is initialized correctly */

comment[0] = 0;

for( usr_pos = &(user->location[0]);
            usr_pos < &(user->location[PAPER_MAX]);
                                        usr_pos++)
    if( usr_pos->auth_num == paper->art_num ) break;
```

```
if((usr_pos < &(user->location[PAPER_MAX])) &&
                              (usr_pos->usr_locate != 0))
{
  printf( "Oould you like to pick up ");
  printf( "where you left off from? ( y or n ) ");
  gets( comment );
}

sprintf(path,"%s%s/%s/%s.d/%s",HOME,CONF_DIR,
                              conf_dir,maj_prof->usr_id,
                                                  ppr_name);
fd = fopen( path, "r" );

/* initialize the terminal io setting for curses */
initscr();
            /* this window is to box in paper */
win1 = newwin(LINES-1,COLS-1,0,0);
            /* this window is to display paper */
win2 = newwin(LINES-3,COLS-4,1,2);
            /* this window is to box in comments */
win3 = newwin(LINES-9,COLS-1,9,0);
            /*this window is to edit comments in*/
win4 = newwin(LINES-11,COLS-4,10,2);
            /*this window is for the option display*/
win5 = newwin(1,COLS-2,LINES-1,0);

/*accept user input without echoeing it to the screen*/
noecho();
/*accept user input in raw mode without waiting for nl*/
crmode();
/*leave the cursor at the place of last updating */
leaveok(win1,TRUE);
leaveok(win2,TRUE);
leaveok(win3,TRUE);
leaveok(win4,TRUE);
leaveok(win5,TRUE);
leaveok(stdscr,TRUE);

/* this will accumulate the user's numeric input */
motion = 0;
/* this will be either -1 of negate
    or 1 for positive motion */
sign = 1;

end_line = -1;
cur_line = -(LINES - 7);

clear();
refresh();
if( comment[0] == 'y')
{
```

```
    i = LINES +  usr_pos->usr_locate - 7 ;
    display_ppr( i );
}
else
    display_ppr( LINES - 3 );

for(flag=TRUE;flag;)
{
    switch( ( c = getch() ) )
    {
        case '1': case '2': case '3': case '4': case '5':
        case '6': case '7': case '8': case '9': case '0':

        /* NUMBER INPUT: if the user wishes to move so
           many pages or lines then allow the user
           to enter a number before the 'p' or 's'
            option */

            motion = (motion*10) + (int) (c - '0');
            break;

        case '+':

        /* PLUS SIGN: allow the user to make
            the number input positive. */

            sign = 1;
            break;

        case '-':

        /* NEGATIVE SIGN: allow the user to make
            the number input negative. */

            sign = -1;
            break;

        case 'c':

        /*  COMMENTS MODE - in this mode the user
            will be able to browse current existing
            comments and also place their
            own comments tagged to the current
            line along with the option to target
            the comments to specific users within
            the conference */

            display_cmnts();

            /* once done then return the user to
               the current page */
```

```
    clear();
    refresh();
    display_ppr( 0 );
    motion = 0;
    sign = 1;
    break;

case 'p':

/* PAPER MODE - after all the other
    modes are done this command
    will advance the displayed page
    to the specified area in
    increments of page size jumps */

    if( motion == 0 ) motion = 1;
    if(!((cur_line == end_line) &&
            (sign*motion > 0)) &&
       !((cur_line == 4) && (sign*motion < 0)))
    {
        clear();
        refresh();
        display_ppr(sign*motion*(LINES-3));
    }
    motion = 0;
    sign = 1;
    break;

case 'q':

/* QUIT MODE - let the user get out
    of this infinite loop!!! */

    /* send mail to all users that
    received comments from user */

    sprintf( path, "%s within the %s",
                    conf_name, ppr_name );
    for( i = 0; i < num_usrs; i++ )
    {
        if( send_comm[ i ] )
        {
            inform( user, &(usr_loc[ i ]),
                            ATTCH, path );
            break;
        }
    }
    /* read in the conference's usr_loc_file */

    sprintf(path,"%s%s/%s/%s",HOME,CONF_DIR,
                            conf_dir,USR_FILE);
```

```
    fdes = open( path, O_RDWR, 0 );
    num_usrs =
            read(fdes,usr_loc,sizeof(usr_loc));

    num_usrs = num_usrs /
            ( sizeof( struct usr_loc_file ) );
    /* scan all the conference's usr_locs
            for the login_name */

    for( user = &usr_loc[ 0 ];
            user < &usr_loc[ num_usrs ]; user++)
        if( strcmp( login_name, user->usr_id )
                                    == 0 ) break;

    for( usr_pos = &(user->location[0]);
         usr_pos < &(user->location[PAPER_MAX]);
                                    usr_pos++)
        if( usr_pos->auth_num ==
                        paper->art_num ) break;

    if( usr_pos < &(user->location[PAPER_MAX] ))
    {
        usr_pos->usr_locate = cur_line;
    }
    else
    {
        for( usr_pos = &(user->location[0]);
             usr_pos < &(user->location[PAPER_MAX]);
                                    usr_pos++ )
            if( usr_pos->auth_num == 0 ) break;
        usr_pos->auth_num = paper->art_num;
        usr_pos->usr_locate = cur_line;
    }
    /* write out new usr_loc information */
    lseek( fdes, 0, 0 );
    num_usrs = num_usrs *
            ( sizeof( struct usr_loc_file ) );
    write(fdes, usr_loc, num_usrs);
    close( fdes );
    flag=FALSE;
    break;

case 'l':

/* LINE MODE - while browsing if the
    user would like to move the lines
    either up or down so many places this
    mode will move the page in the
    direction desired */

    if( motion == 0 ) motion = 1;
```

```
            if(!((cur_line == end_line) &&
                    (sign*motion > 0)) &&
               !((cur_line == 4) && (sign*motion < 0)))
            {
                clear();
                refresh();
                display_ppr( sign * motion );
            }
            motion = 0;
            sign = 1;
            break;

        case '?':

        /* HELP MODE - display the options
            that the user has available and then
            wait for a character to refresh
            the page */

            display_hlp();
            getch();

            /* once done then return the user
               to the current page */

            clear();
            refresh();
            display_ppr( 0 );
            motion = 0;
            sign = 1;
            break;

        default:

        /* ERROR MODE - the user entered an
            invalid character inform
            them of it by flashing the screen */

            /*flash();*/
            motion = 0;
            sign = 1;
            break;

    }  /* end of switch */
} /* end of for */

/*  return the terminal to a cleared screen
    before returning to the user */

clear();
refresh();
```

```
    endwin();
    fclose( fd );
}

/*************************************************************
 *                                                           *
 *  Gather_Comm will gather the comments that the user       *
 *  has had attached to the paper.                           *
 *                                                           *
 *************************************************************/

gath_comm()
{
    int is_cmnts;
    int i, fdes;
    char path[100];
    struct cmnt_hdr *cmnts_in;
    void qsort();
    int cmpr();

    if( is_comm( conf_dir, user->part_num, paper->art_num))
    {
       cmnts_in = comm_lst;
       num_cmnts = 0;

       sprintf( path, "%s%s/%s/t%02d.%02d",CONF_HOME,
                      conf_dir, COMMENTS, user->part_num,
                                          paper->art_num );
       fdes = open( path, O_RDWR, 0 );

       for(;;)
       {
          if( read( fdes, cmnts_in, 8 ) != 8 ) break;
          lseek( fdes, cmnts_in->msg_len, 1 );
          cmnts_in++;
          num_cmnts++;
       }
       close( fdes );
       cmnts_in->line_num = -1;
       qsort( comm_lst, num_cmnts, 8, cmpr );
    }
}

/*************************************************************
 *                                                           *
 * DISPLAY PAPER - iff called with a zero this routine       *
 * repaints the page with the current line as is.  This      *
 * routine is the heart of the browser it is what figures    *
 * out how far to read into the file stream for the data     *
 * I considered using the curses scrolling page routines     *
 * but I decided to hard code it.  It is left as an          *
```

```
 * exercise to someone more brave than I.  I've put alot    *
 * of things in to pad for the beginning and the end of     *
 * the document.                                            *
 *                                                          *
 ************************************************************/

display_ppr( motion )
int motion; /* what type of motion is this display to have */
{
    char *bufptr, buffer[100];
    int i,j;

    if( motion != 0 )  /* check if there is motion */
    {
        wmove(win2,0,0);
        wclear( win2 );

        /* this code that follows is to
            postion the file stream pointer in the proper
            position in the file. Two conditions
            exists : 1) the condition that the user
            wants to display some text that is not a
            page forward and, 2) the condition where the
            user wants to display something further than or
            equal to a page. ( a page is LINES - 3 ). */

        if(  motion < LINES-3 )
        {
            rewind( fd );
            for( i=0; i < ( cur_line - 4  + motion ); i++ )
            {
                if( fgets( buffer, 90, fd ) == NULL )
                {
                    if( end_line == -1) end_line = i++;
                    rewind( fd );
                    for( i = 0; i < end_line - 4; i++ )
                        fgets( buffer, 90, fd );
                    break;
                }
            }
        }
        else
        {
            for( i = 0; i < ( motion - ( LINES - 3 ) ); i++ )
            {
                if( fgets( buffer, 90, fd ) == NULL )
                {
                    if( end_line == -1 )
                        end_line = cur_line + LINES - 7 + i;
                    rewind( fd );
                    for( i = 0; i < end_line - 4; i++ )
```

```
                fgets( buffer, 90, fd );
            break;
        }
    }
}

/* this code that follows is to
   display the page to the second
   window.  Notice that alot of
   care has been placed here so that
   when the user hits the end of
   the document that the routine
   knows how to handle the
   displaying of the last couple of lines */

for(i=0;i<LINES-3;i++)
{
    bufptr = buffer;
    for( ; bufptr<=&buffer[99] ; ) *bufptr++ = ' ';
    bufptr = buffer;
    if( fgets( bufptr, 90, fd ) == NULL )
    {
/* less than the top four lines were displayed */
      if( i < 4 )
      {
       /* the end_line has not been determined */
        if( end_line == -1)
        {
            end_line = 0;
            rewind( fd );
            for(;;)
            {
                if( fgets( bufptr, 90, fd ) ==
                                    NULL ) break;
                end_line++;
            }
        }
        rewind( fd );
        for(i=0;i<end_line-4;i++)
                fgets( bufptr, 90, fd );
        wmove(win2,0,0);
        wclear( win2 );
        for(i=0;i<4;i++)
        {
            bufptr = buffer;
            for( ; bufptr<=&buffer[99] ; )
                *bufptr++ = ' ';
            bufptr = buffer;
            fgets( bufptr,90,fd );
            if(i==3)
            {
```

```
                    for(j=0;j<78;j++)
                        if( buffer[j] == ' ')break;
                    buffer[j-1] = ' ';
                    buffer[j] = ' ';
                    buffer[76] = NULL;
                }
                wprintw( win2, "%s", bufptr );
            }
          }
          break;  /* break out of page display loop */
        }
        if(i==3)
        {
            for(j=0;j<78;j++)
                if( buffer[j] == ' ')break;
            buffer[j-1] = ' ';
            buffer[j] = ' ';
            buffer[76] = NULL;
        }
        wprintw( win2, "%s", bufptr );
    }
}
else
{
    touchwin( win2 );
}

cur_line += motion;
if( cur_line < 4 ) cur_line = 4;
if( ( end_line != -1 ) &&
    (cur_line > end_line ) ) cur_line = end_line;

box(win1,'|','-');
mvwprintw(win1,4,0,">");
mvwprintw(win1,4,COLS-2,"<");

cur_comm = ( struct cmnt_hdr * ) NULL;

/* calculate the commented lines and
    place a star in the position */

for( i = cur_line - 3;
     i < ( cur_line + LINES - 6 ); i++)
{
    for( j = 0; j < 300; j++ )
    {
        if( i < comm_lst[j].line_num ) break;
        if( i > comm_lst[j].line_num ) continue;
        if( i == cur_line ) cur_comm = &comm_lst[j];
        mvwprintw(win1,( i - cur_line + 4 ),COLS-2,"*");
        break;
```

```
        }
    }
    wclear( win5 );
    mvwprintw(win5,0,0,
       "These options are available: c, p, l, q, ?");
    wprintw(win5, "Current line = %-6d  ",cur_line);

    wrefresh( win1 );
    wrefresh( win2 );
    wrefresh( win5 );
}

/*********************************************************
 *                                                       *
 * DISPLAY HELP - will show the options available to     *
 * the user during the browsing mode.                    *
 *                                                       *
 *********************************************************/

display_hlp()
{
    clear();
    printw("                You are currently ");
    printw("in the BROWSER routine");
    printw("             The options available ");
    printw("within this routine are:");
    printw("0);
    printw( "           c (comment)  - ");
    printw( "places you into the comment mode.");
    printw( " [[+|-]n]  l (lines)    - ");
    printw( "advances the display frwd or bkwrd <n> lines.");
    printw( " [[+|-]n]  p (page)     - ");
    printw( "advances the display frwd or bkwrd <n> pages.");
    printw( "           q (quit)     - ");
    printw( "back up to the conferencing routine.");
    printw( "           ? (help)     - ");
    printw( "to display this page.");
    printw("0);
    printw(
"                    Hit return to continue the browser.");
    refresh();
}

/*********************************************************
 *                                                       *
 * DISPLAY COMMENTS PAGE - box in the area and then      *
 * display the help page and wait for the user to        *
 * respond, if the user enters a "v" and then if         *
 * there are any comments for the current line display   *
 * them and then if the user enters a "q" quit the mode  *
 * if the user enters an "i" place then into the input   *
```

```
 * mode where everthing typed is entered into a file    *
 * once in the input mode the user must enter a "." at  *
 * the beginning of a line by itself to leave the mode  *
 * but before leaving prompt the user for the target of *
 * the comment.                                         *
 *                                                      *
 ********************************************************/

display_cmnts()
{
    bool flag;

    wclear(win3);
    box(win3,'.','.');
    wrefresh(win3);
    cmnts_hlp();
    for(flag=TRUE;flag;)
    {
        switch( getch() )
        {
            case 'i':

            /* INPUT MODE - place a comment
                on the current line
                first - allow input of comment
                with editting capabilities
                second - prompt the user for
                target of comment
                third - format comment and
                send off to user */

                cmnts_inp();
                cmnts_hlp();
                break;

            case 'v':

            /* VIEW MODE - iff there are comments
                that have been attached to the current
                line are they to be shown */

                 if( cur_comm ==
                        ( struct cmnt_hdr * ) NULL ) break;
                 cmnts_vw();
                 cmnts_hlp();
                 break;

            case 'q':

            /* QUIT MODE - let the user get out
                of this loop */
```

```
                flag = FALSE;
                break;

            default:

            /* ERROR MODE - the user entered an
                invalid character inform them of it
                by flashing the screen */

                /*flash();*/
                break;

        }   /* end of switch loop */
    }       /* end of for loop */
}

/*********************************************************
 *                                                       *
 * COMMENTS HELP display - will show the available       *
 * to the user during the browsing mode.                 *
 *                                                       *
 *********************************************************/

cmnts_hlp()
{
   wmove(win4,0,0);
   wclear(win4);
   wprintw( win4, "0 );
   wprintw( win4, "            You are currently ");
   wprintw( win4, "in the Browser's Comment routine.0);
   wprintw( win4, "              The options available ");
   wprintw( win4, "within this routine are:0);
   wprintw( win4, "            i (input) - ");
   wprintw(win4,"to attach a comment to the current line.0);
   wprintw( win4, "            q (quit)  - to ");
   wprintw( win4, "return to the browsing mode.0);
   if( cur_comm != ( struct cmnt_hdr * ) NULL )
       wprintw( win4, "            v (view)  - to view ");
       wprintw( win4, "any current line comments.0);
   wprintw( win4, "                       Enter ");
   wprintw( win4, "one of the options.");
   wrefresh(win4);
}

/*********************************************************
 *                                                       *
 * COMMENTS INPUT routine - accept within window 4       *
 * user input along with editting moves of the           *
 * cursor, but first just accept straight input          *
 *                                                       *
 *********************************************************/
```

```
cmnts_inp()
{
    char str[90],*loc;
    int len,line,i,j,num_in;
    struct usr_loc_file *tmp_usr;
    char target[10];
    int ids[ 10 ];

    wmove(win4,0,0);
    wclear( win4 );
    wprintw(win4, "Enter a single '.' on a newline ");
    wprintw(win4, "to terminate.  Erase char is Backspace");
    leaveok( win4,FALSE );
    for(line=0,loc=comment;;line++,loc++)
    {
        wmove(win4,line+1,0);
        wrefresh( win4 );
        for(i=0;i<COLS-4;i++)
        {
            str[i] = wgetch(win4);

            if( str[i] == '0 ) break;

            if( str[i] == ' 10' )
            {
                str[i-1] = ' ';
                mvwprintw(win4,line+1,0,"%s",str);
                i -= 2;
            }
            else
                wprintw(win4,"%c",str[i]);
            wrefresh( win4 );
        }
        str[i]=' ';

        if( (str[0] == '.' && str[1] == ' ')
                                || ( line > LINES-14 ) )
        {
            if( line > LINES-14 )
            {
              mvwprintw(win4,LINES-12,0,
             "Maximum of %d comment lines allowed",LINES-13);
              wrefresh( win4 );
              sleep( 2 );
            }
            *loc = NULL;
            comm_hdr.line_num = cur_line;
            comm_hdr.msg_len  = strlen( comment );
            comm_hdr.from_id = user->part_num;
            break;
        }
```

```
    if( ( len = strlen( str ) ) > CMNT_LL )
        len = CMNT_LL - 2;
    strncpy( loc, str, len );
    loc += len;
    *loc = '0;
}
leaveok( win4,TRUE );

/* prompt the user for the target of the comment */

wmove( win4, 0, 0 );
wclear( win4 );
wprintw( win4,
"   Who do you wish to target this comment for?0);
for(i=0,tmp_usr=usr_loc;
            tmp_usr < &usr_loc[num_usrs];i++,tmp_usr++)
{
    if( ( tmp_usr->part_num == user->part_num ) ||
        ( ( tmp_usr->role == AUTHOR ) &&
          ( tmp_usr->part_num != paper->art_num ) ) ||
        ( tmp_usr->role == AUD_MEM ) ||
        ( tmp_usr->role == NON_REVUR ) )
    {
         i--;
         continue;
    }
    if( tmp_usr->role == MAJ_PROF )
         wprintw( win4,
            "       %1d - major professor ( %s )0,
                                  i, tmp_usr->usr_id );
    if( tmp_usr->role == AUTHOR )
         wprintw( win4, "       %1d - author ( %s )0,
                                    i, tmp_usr->usr_id );
    if( tmp_usr->role == COM_MEM )
         wprintw( win4,
            "       %1d - committee member ( %s )0,
                                  i, tmp_usr->usr_id );
         ids[ i ] = tmp_usr->part_num;
}
ids[ i ] = user->part_num;
wprintw( win4,
    "       %1d - self ( %s )0, i++, user->usr_id );
if( status == FIN_REVU )
{
    wprintw( win4, "       %1d - audience0, i++ );
    wprintw( win4,
          "       %1d - platform discussion0, i );
}
else
   i--;
```

```
mvwprintw( win4, LINES-12, 0, "   To target the " );
mvwprintw( win4, LINES-12, 0,
    "comment enter either a number or 'q' to quit.   " );
wrefresh( win4 );

str[1] = NULL;

for( j = 0 ; j <= i ; j++ )
    target[ j ] = FALSE;

for( ;; )
{
    str[0] = getch();
    if( str[0] == 'q' ) break;
    if( str[0] < '0' || str[0] > '9' ) continue;
    num_in = atoi( str );
    if( num_in > i ) continue;
    if( target[ num_in ] == FALSE )
    {
        target[ num_in ] = TRUE;
        mvwprintw( win4, num_in + 2, 5, "*" );
    }
    else
    {
        target[ num_in ] = FALSE;
        mvwprintw( win4, num_in + 2, 5, " " );
    }
    mvwprintw( win4, LINES-12, 0,
    "   To target the comment enter either " );
    wprintw( win4, "a number or 'q' to quit.   " );
    wrefresh( win4 );
}

for( j = 0 ; j <= i ; j++ )
    if( target[ j ] == TRUE ) break;

if( j > i )
{
    mvwprintw( win4, LINES-12, 0,
    "   Since you did not target the comment ");
    wprintw( win4, "no one will see this input. ");
    wrefresh( win4 );
}
else
{
    mvwprintw( win4, LINES-12, 0,
    "    Your comment is currently being ");
    wprintw( win4, "to the targetted person.  ");
    wrefresh( win4 );
}
```

```
    if( status == FIN_REVU )
    {
        if( target[ i-- ] == TRUE )
        {
            /* platform discussion */
            sprintf( conf_path, "%s%s.d",
                                CONF_HOME, conf_name );
            add_item( comment, NULL, user->part_num );
            mvwprintw( win4, i + 3, 5, " " );
            wrefresh( win4 );
        }
        if( target[ i-- ] == TRUE )
        {
            for( tmp_usr = usr_loc;
                 tmp_usr < &usr_loc[ num_usrs ]; tmp_usr++ )
            {
                if( tmp_usr->role == AUD_MEM )
                {
                    if( targ_comm( tmp_usr->part_num )
                                                == FAIL )
                    {
                        inform( maj_prof, &sysadmn,
                                        COMM, conf_name );
                    }
                    send_comm[ tmp_usr->part_num ] = TRUE;
                }
            }
            mvwprintw( win4, i + 3, 5, " " );
            wrefresh( win4 );
        }
    }
    for( ; i >= 0; i-- )
    {
        if( target[ i ] == TRUE )
        {
            if( targ_comm( ids[ i ] ) == FAIL )
            {
                inform( maj_prof, &sysadmn, COMM,
                                                conf_name );
            }
            else
            {
                send_comm[ ids[ i ] ] = TRUE;
                mvwprintw( win4, i + 2, 5, " " );
                wrefresh( win4 );
            }
        }
    }
}

targ_comm( to_id )
```

```
int to_id;
{
    /* in this routine I will take the to_id and create
       a file of the format t##.## the first one
        number is obvious the last is the paper
        number since the comments are stored in the
        same COMMENTS directory per each conference.
        If the file exist then the comments will be
        appended to the file.  The comment
        header is written in also for later
        referencing the comment. */

    char path[100];
    int fdes, nbytes;

    sprintf( path,"%s%s/%s/t%02d.%02d",CONF_HOME,
                                conf_dir,COMMENTS,
                                to_id, paper->art_num );
    if( ( fdes =
          open( path, O_WRONLY|O_CREAT|O_APPEND, 0666 ) )
                                                  == -1 )
        return( FAIL );
    nbytes = strlen( comment ) + sizeof( comm_hdr );
    if( write( fdes, &comm_hdr, nbytes ) != nbytes )
        return( FAIL );
    close( fdes );
    if( user->part_num == to_id ) gath_comm();
    return( SUCCESS );
}

int cmpr( p1, p2 )
struct cmnt_hdr *p1, *p2;
{
    int ret;

    ret = 0;
    if( p1->line_num < p2->line_num ) ret = -1;
    if( p1->line_num > p2->line_num ) ret =  1;
    return( ret );
}

/*****************************************************
 *                                                   *
 *  Comments view is the routine that will enable    *
 *  the user to look at comments attached to the     *
 *  current line.  Options will be available to      *
 *  delete a comment or to read the next comment     *
 *  if there are multiple comments attached to       *
 *  the same line.                                   *
 *                                                   *
 *****************************************************/
```

```
cmnts_vw()
{
    bool flag;
    int occur, last_fid;

    occur = 0;
    last_fid = cur_comm->from_id;

    read_comm( occur );

    for( flag=TRUE; flag; )
    {
        switch( getch() )
        {
            case 'b':

            /* BACKWARDS MODE - should only happen
                if cur_comm-1.line_num is equal to
                the cur_line, display the comment */

                if( (cur_comm-1)->line_num != cur_line )
                    break;
                cur_comm--;
                occur--;
                read_comm( occur );

                break;

            case 'd':

            /* DELETE MODE - many decisions to make
                here, first delete the comment out of
                the file. If there are no forward or
                backward comments then bring up then
                set the flag to false. Must also
                fix the cmnts_list that was set up
                by the gath_comm routine */

                break;

            case 'f':

            /* FORWARDS MODE - should only happen
                if cur_comm+1.line_num is equal to
                the cur_line, display the comment */

                if( (cur_comm+1)->line_num != cur_line )
                    break;
                cur_comm++;
                occur++;
                read_comm( occur );
```

```
                break;

            case 'q':

            /* QUIT MODE - let the user get out
                of this loop */

                flag = FALSE;
                break;

            default:

            /* ERROR CASE - the user entered an
                invalid character */

                break;
        }
    }
}

read_comm( occur )
int occur;
{
    char path[100];
    int fdes, i;

    wmove( win4,0,0 );
    wclear( win4 );

    sprintf( path,"%s%s/%s/t%02d.%02d",
                CONF_HOME,conf_dir,COMMENTS,
                user->part_num, paper->art_num );
    comment[0] = NULL;

    fdes = open( path, O_RDWR, 0 );
    for(;;)
    {
        if( read( fdes, &comm_hdr, 8 ) != 8 ) break;
        if( comm_hdr.line_num != cur_line )
        {
            lseek( fdes, comm_hdr.msg_len, 1 );
            continue;
        }
        if( occur > 0 )
        {
            occur--;
            lseek( fdes, comm_hdr.msg_len, 1 );
            continue;
        }
        read( fdes, comment, comm_hdr.msg_len );
        comment[ comm_hdr.msg_len ] = NULL;
```

```
        break;
    }
    close( fdes );

    wprintw( win4, "%s", comment );

    mvwprintw( win4, LINES-12, 0, "From: " );
    if( cur_comm->from_id == user->part_num )
    {
        mvwprintw( win4, LINES-12, 8, " self " );
    }
    else
    {
        for( i = 0; i < num_usrs; i++ )
            if( cur_comm->from_id == usr_loc[ i ].part_num ) break;
        mvwprintw( win4, LINES-12, 8,
                " %s ", usr_loc[i].usr_id );
    }

    if( ( (cur_comm+1)->line_num == cur_line )  &&
        ( (cur_comm-1)->line_num == cur_line ) )
        mvwprintw( win4, LINES-12, 20,
                "These options are available: b, f, q" );
    else
        if( (cur_comm+1)->line_num == cur_line )
            mvwprintw(win4, LINES-12, 20,
                "These options are available: f, q" );
        else
            if( (cur_comm-1)->line_num == cur_line )
                mvwprintw(win4,LINES-12,20,
                "These options are available: b, q");
            else
                mvwprintw(win4,LINES-12,20,
                "This option is available: q");
    wrefresh( win4 );
}
```

```
/*
 *
 *
 *       I N F O R M . C
 *
 *
 */

#include "/usrb/att/marrs/Source/conf.h"
#include <stdio.h>
#include <sys/file.h>

/*************************************************************
 *                                                           *
 * Inform routine will be the contact for routines to the    *
 * other participants within the conferencing system, but    *
 * before calling the routine the destination user           *
 * ( des_usr ) must be validated as the appropriate          *
 * target for the mail message.                              *
 *                                                           *
 * called with: src_usr ( source user of message )           *
 *              des_usr ( destination user for message )     *
 *              error   ( type of error )                    *
 *              conf    ( conference name )                  *
 *                                                           *
 *************************************************************/

inform( src_usr, des_usr, error, conf )
struct usr_loc_file *des_usr, *src_usr;
int error;
char *conf;
{
    char call[ 100 ];
    char file[ 100 ];
    char abbr_msg[ 40 ];
    FILE *tmp_fl;
    int i;

    if( error == ATTCH )
    {
        for( i = 0 ;; i++ )
            if( ( call[ i ] = *(conf+i) ) == ' ' ) break;
        call[ i ] = ' ';
        sprintf( file, "%s%s/%s.d/.m%05d",
                        HOME, CONF_DIR, call, getpid() );
    }
    else
        sprintf( file, "%s%s/%s.d/.m%05d",
                        HOME, CONF_DIR, conf, getpid() );

    tmp_fl = fopen( file, "w" );
```

```
switch( error )
{
    case BAD_USF:
        sprintf( abbr_msg,
            "marrs - bad user location file", 0 );
        fprintf( tmp_fl,
    "1440hile marrs user <%s> was trying to browse0,
                            srq_usr->usr_id );
        fprintf(tmp_fl,
    "the conference <%s>, the user_loc_file ",conf);
        fprintf (tmp_fl,
    "could 0ot be read.  Please fix and inform ");
        fprintf (tmp_fl, "the user.");
        break;

    case BAD_SHF:
        sprintf( abbr_msg,
            "marrs - bad status history file", 0 );
        fprintf( tmp_fl,
    "1440hile marrs user <%s> was trying to browse0,
                            srq_usr->usr_id );
        fprintf(tmp_fl,
    "the conference <%s>, the stat_his_file ",conf);
        fprintf (tmp_fl,
     "could 0ot be read.  Please fix and inform ");
        fprintf (tmp_fl, "the user.");
        break;

    case NEW_MEM:
        sprintf( abbr_msg,
            "marrs - request for new user", 0 );
        fprintf( tmp_fl,
    "0ser <%s> wants to become a ", srq_usr->usr_id );
        fprintf( tmp_fl,
    "new member of the conference <%s>.0, conf );
        ftmp_fl, "Please inform user.0 );
        break;

    case NO_MPF:
        sprintf( abbr_msg,
            "marrs - no major professor", 0 );
        fprintf( tmp_fl,
    "1440hile marrs user <%s> was trying to browse0,
                            srq_usr->usr_id );
        fprintf(tmp_fl,
    "the conference <%s>, the user_loc_file ",conf);
        fprintf (tmp_fl,
     "did 0ot contain a major professor record.0);
        fprintf (tmp_fl,
     " Please fix and inform the user.");
        break;
```

```
    case COMM:
        sprintf( abbr_msg,
            "marrs - can't target comments", 0 );
        fprintf( tmp_fl,
    "1440hile a user of the conference <%s> was0,
                            conf );
        fprintf(tmp_fl,
    "trying to target a comment to a participant0);
        fprintf (tmp_fl,
    "an error occurred.  Please fix and inform the0);
        fprintf (tmp_fl,
    " major professor <%s>.", src_usr->usr_id );
        break;

    case IDLE_C:
    case IDLE_F:
        sprintf( abbr_msg,
            "marrs - idle reviewer notice", 0 );
        fprintf( tmp_fl,
    "1440ithin the conference <%s>,0,conf );
        fprintf( tmp_fl,
    "the Master's Report from <%s> ",src_usr->usr_id );
        if( error == IDLE_C )
            fprintf( tmp_fl,
            "is0ow in committee review.0 );
        else
            fprintf( tmp_fl,
            "is0ow in final review.0 );
        fprintf( tmp_fl,
    "According to MARRS's records you have0 );
        fprintf( tmp_fl,"been idle.0 );
        break;

    case ATTCH:
        sprintf( abbr_msg,
    "marrs - comments attached", 0 );
        fprintf( tmp_fl,
    "0or the conference %s paper,0,conf );
        fprintf( tmp_fl,
    "you received new comments from <%s>.0,
            src_usr->usr_id );
        break;

    default:
        fclose( tmp_fl );
        return;
}

fclose( tmp_fl );
sprintf( call, "mail -s
    abbr_msg,/* des_usr->usr_id,*/ file );
```

```
    system( call );
    unlink( file );
}
```

```
/*
 *
 *
 *       L I S T _ C O N F . C
 *
 *
 */

#include <sys/types.h>
#include "/usrb/att/marrs/Source/stat.h"
#include <sys/dir.h>
#include "/usrb/att/marrs/Source/conf.h"
#include <curses.h>
#include <strings.h>

/************************************************
 *                                              *
 * list_conf is a routine to list the contents  *
 * of a specified conference directory.  It can *
 * also be used to validate an entered name     *
 * to ensure the proper input.                  *
 *                                              *
 * value is the only input to the routine.  If  *
 * the user wants to know how many conferences  *
 * exist then enter Null for value, else to val-*
 * idate a name pass the address to the string  *
 * and this routine will answer with TRUE or    *
 * FALSE depending on the permission and the    *
 * availability withing the conference directory*
 *                                              *
 ************************************************/

list_conf( value )
char *value;
{
    DIR *opendir();
    DIR *dirp;
    struct direct *readdir();
    struct direct *dp;
    struct stat buf;
    char path[100];
    char *rindex(),*index();
    int count;

    count = 0;
    sprintf(path, "%s%s/.", HOME, CONF_DIR );
    dirp = opendir( path );
    for( dp = readdir( dirp );
                dp != NULL; dp = readdir( dirp ) )
    {
        sprintf( path, "%s%s/%s", HOME, CONF_DIR,
```

```
                                        dp->d_name );
        stat( path, &buf );
        if( ( buf.st_mode & S_IFDIR ) &&
                    ( value == NULL ) &&
            ( index( dp->d_name, '.' ) != &(dp->d_name[0])))
        {
            count++;
            *( rindex( dp->d_name, '.' ) ) = NULL;
            printf("%18s",dp->d_name,buf.st_mode);
            if( !( count % 4 ) ) printf("0);
        }
        else
        {
            if( ( buf.st_mode & S_IFDIR ) &&
                ( !strcmp( dp->d_name, value ) ) )
            {
                closedir( dirp );
                return( TRUE );
            }
        }
    }
    closedir( dirp );
    if( value == NULL )
    {
        printf("0);
        return( count );
    }
    return( FALSE );
}
```

```
/*
 *
 *
 *    I S _ C O M M . C
 *
 *
 */

#include <sys/types.h>
#include <sys/dir.h>
#include "/usrb/att/marrs/Source/conf.h"
#include <curses.h>
#include <strings.h>

/*************************************************
 *                                               *
 * is_comm   is a routine to find out if the     *
 * specified conference user has for the paper   *
 * a comment file.  TRUE is returned if yes else*
 * FALSE is returned.                            *
 *                                               *
 *************************************************/

is_comm( conf_dir, usr_id, paper_num )
char *conf_dir;
int usr_id, paper_num;
{
    DIR *opendir();
    DIR *dirp;
    struct direct *readdir();
    struct direct *dp;
    char path[100];
    char cmp_str[6];

    sprintf(cmp_str,"%02d.%02d", usr_id, paper_num );
    sprintf(path, "%s%s/%s/.", CONF_HOME, conf_dir,
                                        COMMENTS );
    dirp = opendir( path );
    for( dp = readdir( dirp );
                        dp != NULL; dp = readdir( dirp ) )
    {
        if( strcmp( &(dp->d_name[1]), cmp_str ) == 0 )
        {
            closedir( dirp );
            return( TRUE );
        }
    }
    closedir( dirp );
    return( FALSE );
}
```

```
/*********************************************
 *                                           *
 *              C O N F . H                  *
 *                                           *
 *********************************************/

#ifdef  bug
#define         DBUG    1

#else

#define         DBUG    0
#endif


#define ANY_ROLE  0
#define MAJ_PROF  1     /* Possible participant roles within a  */
#define COM_MEM   2     /* conference.                          */
#define AUTHOR    3
#define AUD_MEM   4

#define SYS_ADMIN -1    /* Other roles in the conference        */
#define NON_REVUR -2

#define LGN_SZ     9    /* Maximum user login id size + 1       */

#define MAX_LOGIN_SIZE          10
#define MAX_AUTHORS             10
#define MAX_PAPER_LENGTH        30
#define MAX_ED_PATH             50
#define SYSTEM_CALL_SIZE        200
#define MAX_ITEMS               200

#define PAPER_MAX  10   /* Maximum number of papers in a conference */
#define CON_MAX    30   /* Maximum number of participants in a conference */
#define STAT_MAX   20   /* Maximum number of status changes for a paper */
#define FILENM_MAX 15   /* Maximum length for a paper's filename */
#define CMNT_LL    80   /* Maximum length of a comments line length */
#define MAX_CMNT   880  /* Maximum comment CMNT_LL * ( LINES - 14 ) */

#define NOT_SUB    1    /* Possible status states for the paper */
#define PAPER_SUB  2
#define RE_WORK    3
#define COM_REVU   4
#define FIN_REVU   5
#define PAPER_PUB  6
#define MAX_STATES 6

#define YES 1
#define NO  0
```

```
# define        TRUE    (1)
# define        FALSE   (0)

#define FAIL -1
#define SUCCESS 0

/* possible errors within the review code */

#define BAD_USF  1   /* bad usr_loc_file of the conference */
#define BAD_SHF  2   /* bad stat_his_file of the conference */
#define NEW_MEM  3   /* user wants to become a new member of conference */
#define NO_MPF   4   /* no major professor found in usr_loc_file */
#define COMM     5   /* browser had trouble targetting a comment */
#define IDLE_C   6   /* used to remind idle user's in comm_r through mail */
#define IDLE_F   7   /* used to remind idle user's in finl_r through mail */
#define ATTCH    8   /* used to inform users of attached comments */

#define CLEAR   system("/usr/ucb/clear");


/*******************************************************
#define BIN         "/usrb/att/marrs/Bin"
#define SOURCE      "/usrb/att/marrs/Source"

*******************************************************/

#define HOME        "/usrb/att/marrs/"
#define CONF_HOME   "/usrb/att/marrs/Conference/"
#define PROF_FILE   "/usrb/att/marrs/Bin/prof_file"
#define CONF_DIR    "Conference"
#define COMMENTS    "COMMENTS"
#define PLATFORM    "PLATFORM"
#define STAT_FILE   "stat_his_file"
#define USR_FILE    "usr_loc_file"
#define DIR_ENDING  ".d"
#define PLAT_MAIN   "main.d"
#define ITEM_FILE   "item_list_file"
#define TEMP_FILE   "temp"

#define SYS_ADMN "janning"

#define DEFAULT_EDITOR  "/bin/ed"
#define DEFAULT_PRTR    "dorm"
#define DEFAULT_NROFF   "/usr/bin/nroff -mm"
#define PAPER_NAME      "PAPER"
#define BB_NAME         "bb_file"

#define CONF_DIR_MODE 00777     /* umask will take away from these modes, */
                                /*          WATCH OUT                     */

struct loc_file {
```

```
        char auth_num;
        short usr_locate;
        };

struct usr_loc_file {
        char usr_id[LGN_SZ];
        char part_num;
        short role;
        struct loc_file location[PAPER_MAX];
        };


struct stat_file {
        short paper_stat;
        long date;
        };

struct stat_his_file {
        char art_num;
        char paper_file[FILENM_MAX];
        struct stat_file status[STAT_MAX];
        };

struct cmnt_hdr {
        short line_num;
        short msg_len;
        short from_id;
        short spare;
        };

struct item_info {
        short pred;
        short succ;
        short from_who;
        short name;
        };

char fac_id[LGN_SZ];

#define ITEM_LENGTH sizeof (struct item_info)
```

```
/*************************************************
 *                                               *
 *             S T A T . H                       *
 *                                               *
 *************************************************/

struct  stat    /* this is called stat in /usr/include/sys */
{
        dev_t   st_dev;
        ino_t   st_ino;
        unsigned short st_mode;
        short   st_nlink;
        short   st_uid;
        short   st_gid;
        dev_t   st_rdev;
        off_t   st_size;
        time_t  st_atime;
        int     st_spare1;
        time_t  st_mtime;
        int     st_spare2;
        time_t  st_ctime;
        int     st_spare3;
        long    st_blksize;
        long    st_blocks;
        long    sp_spare4[2];
};

#define S_IFMT  0170000         /* type of file */
#define         S_IFDIR 0040000 /* directory */
#define         S_IFCHR 0020000 /* character special */
#define         S_IFBLK 0060000 /* block special */
#define         S_IFREG 0100000 /* regular */
#define         S_IFLNK 0120000 /* symbolic link */
#define S_ISUID 0004000         /* set user id on execution */
#define S_ISGID 0002000         /* set group id on execution */
#define S_ISVTX 0001000         /* save swapped text even after use */
#define S_IREAD 0000400         /* read permission, owner */
#define S_IWRITE 0000200        /* write permission, owner */
#define S_IEXEC 0000100         /* execute/search permission, owner */
```

Appendix 3

User's Manual

CONTENTS

LIST OF FIGURES

Master's Report
Reviewing System
(MARRS)

Ronald M. Janning
Kitty Monk
Thomas J. Stachowicz

Abstract

This document describes an application routine that can be used within a group of reviewers to create and review documents. This particular application has been geared at the process of presenting a Master's Report. The MARRS system allows the user to:

- review and comment on a paper,
- leave messages on a bulletin board,
- participate within a platform discussion,
- generate a hardcopy of the comments and paper,
- do a keyword search of topics from published papers within the system,
- list the current conferences within the system, and
- create and edit a paper.

If the user is added to the professor list, the system allows the user to:

- create a conference,
- update the audience members,
- remind idle users, and
- change the status or a paper.

Acknowledgements

This package would not exist without the guidance we received from Rich McBride.

1. Overview

This package is the result of the authors' implementation of a computer conferencing system. Our goal here is not to create a general document reviewing system, but rather an application geared at the creation and reviewing processes of a Master's Report. However, this package with minor modifications could be used within other applications.

In order to gain access to MARRS the user's PATH variable should contain the path:

```
/usrb/att/marrs/Bin
```

After this path is appended to the user's paths, then type the command: 'marrs'. The user's terminal will then clear and MARRS's main menu display will fill the screen. Since there are somethings that only a major professor can do, the page display could vary according to whether or not the user's login name is found in the professor file.

A user can set three environment variables in order to customize the operations of MARRS. The first of these is the variable 'EDITOR'. To set EDITOR, the following two statements can be put in the user's '.profile' or entered at the snell prompt:

```
EDITOR=/usr/ucb/vi
  export EDITOR
```

Instead or '/usr/ucb/vi', any other editor can be specified. The derault editor used is '/bin/ed'.

The second of these is the variable 'MARRS_PRTR'. To set MARRS_PRTR, the following two statements can be put in the user's '.prorile' or entered at the shell prompt:

```
MARRS_PRTR=lpr
export MARRS_PRTR
```

Instead or 'lpr', any other printer can be specified. The derault printer used is 'dorm'.

The last of these variables 'MARRS_NROFF'. To set MARRS_NROFF, the following two statements can be put in the user's '.prorile' or entered at the shell prompt:

```
MARRS_NROFF='/usr/bin/nroff -mm -n3'
        export MARRS_NROFF
```

Instead of '/usr/bin/nroff -mm -n3', any other nroff command

can be specified. The default nroff command used is '/usr/bin/nroff -mm'.

1.1 Terminology

In this document, the following words are used:

- conference - a collection of papers that a major proressor presides over and contains author(s), committee member(s), and an audience,
- paper - the base unit within the MARRS system that is created by an author and can be reviewed by participants within the conference,
- status changes - the various steps that a paper within the MARRS system can be in at any particular time. The paper can be in any of the following states:
  1. Not submitted,
  2. Submitted,
  3. Rework,
  4. Committee Review,
  5. Final Review, or
  6. Published,
- major professor - The "owner" and "creator" of a conference,
- committee member - one or the judges that determine whether a paper should be published or reworked,
- author - the "owner" and "creator" of a paper within a conference,
- audience member - a participant within a specific conference that can review a paper that is in final review.

1.2 Ideas or Computer Conferencing

Computer teleconferencing, differs from other forms of teleconferencing in that voice communication is not used. In addition, one's geographic location is does not limit its use.

Teleconferences, in general, can be either synchronous or asynchronous. A synchronous teleconference is one that is conducted in real time -- that is, everyone is present at their own location at the same time. An asynchronous teleconference can be conducted without the restriction of

everyone "meeting" at the same time.

Computer teleconferencing can be of either form, although asynchronous teleconferencing is more popular. With an asynchronous computer teleconference, the inherent data storing nature of a computer makes it ideal for such a "store and forward" system. Therefore asynchronous computer teleconferencing is not restricted by location or time.

Synchronous computer teleconferencing uses a keyboard for information entry and can tie together many computer users. The dialogue of a user can be seen by the intended recipients as it is entered at the keyboard.

Computer teleconferencing's biggest advantage is that it overcomes geographic and time constraints. This type of teleconferencing is self-documenting so meeting notes or minutes do not have to be documented or reinterpreted.

1.3 Features

The remainder of the sections within this manual will discuss the use of the system. It is divided into various options that are accessible from the main menu page.

The MARRS system allows the user to:

- review and comment on a paper (option - b),
- participate in a platrorm discussion (option - d),
- generate a hard copy (option - g),
- do a keyword search or topics from published papers within the system (option - k),
- list the current conferences (option - l),
- leave messages on a bulletin board (option - m), and
- create and edit a paper (option - p).

Ir the user is added to the professor list, the system also allows the user to:

- create a conrerence (option - c),
- remind idle users (option - r),
- change the status or a paper (option - s), and
- update the audience members (option - u).

1.4 The Data Manager

The data in MARRS is in UNIX files and the INGRES data base management system. The INGRES data base management system is a relational data base system and allows use to be made of INGRES' query/update facilities. File processing is used during the active stages or the conrerence.

The INGRES data base contains information about the conferences such as conferences title, status, paper titles , keywords, etc. Some of the query/update function that a participant or a conference can perform are :

1. list all the conferences available,
2. find the current status of a conference,
3. add/delete conferences (if authorized),
4. list participants in a conference, and
5. list a participant's biography.

1.5 Limitations

This system was created to run on the Kansas State University VAX system loaded with Berkeley Unix 4.2. It is not guaranteed that this system will work under any other type of hardware or operating system. The source code for the features discussed herein are attachments to the authors' Master's Report.

2. B - Option 'The browser'

This option gives the user the capability to browse and comment a paper. This option should only be used after the major professor has created a conference.

2.1 Startup

The Marrs system will display to the user a list of available conferences. The following is displayed:

The following conferences are available.

aaaa bbbb cccc ddddd

Enter a conference name:

If there are no conferences in the system to open, the system will print: There are no conferences created within marrs, and then return the user back to the main menu display. If the user types in an incorrect conference name, Incorrect input: try again, will be displayed and then the system will reprompt you for a conference name. If, however, the correct conference name was typed in but there is a problem with one or the conference's data files, the following message will be displayed:

Error in the creation of the conference aaaa Sorry!!
The system administrator or the conferencing system
will be informed.

If there is a bad file within the conference the user will be returned to the main menu display.

2.1.1 Becoming an audience member

The user is assigned a specific role within a conference. This role can be one of four roles: major professor, committee member, author, or audience member. All roles except for audience members must be assigned during the creation of the conference. If a user or MARRS types in a conference name that they are not yet a member of, the following is displayed:

You do not yet exist within conference aaaa.

Do you want to become a member of the conference? (y or n):

The user, whom wants to become an audience member, should answer 'y' to the prompt. As a result, the major professor will be informed through UNIX mail. The user, through UNIX mail, will be informed later of the major professor's

decision. MARRS then returns the user to the main menu display.

If the user has entered a correct conference name that they are a member or and there is no problem with the data file the system will check to see if there are any papers in the conference. If there are no papers the system will display: There are no papers created within conference aaaa, and will return the user to the main menu display. On the other hand, if there are papers in the conference the following is displayed:

The following papers are available within conference aaaa

mmmmm nnnnn ooooo pppppp

Enter a paper name:

If the incorrect paper name is entered then the system will display: Incorrect input: try again, and will reprompt for the paper name.

2.1.2 Incorrect paper's status?
The paper's status will be checked to assure the condition of the paper matches with the role of the user. For example; if a user is an audience member the paper can only be reviewed during the final review. If a paper is not in the correct state for reviewing and a reviewer attempts to browse and comment the paper, MARRS will inform the user with one of the following error messages:

The paper is not yet submitted.
The paper is in the submitted status.
The paper is in the rework status.
The paper is in the committee review status.
The paper has been published.
The system is unable at this time to browse the paper. Sorry!!

The user is returned to the main menu display.

2.2 The Page

If the user has been able to make it through all the preliminary setup steps the following is displayed:

Would you like to pick up where you left off from? (y or n)

This is only if the user has previously been reviewing the paper. By entering a 'y' the user is returned to the spot where they left off at, any other response returns them to the beginning of the paper.

The browser uses two pages for its displays. The first page displays the paper. The second display is a page where comments can be viewed or entered. Figures A3-(1 thru 5) show the various states or the page displays. WARNING: If some unexpected error happens and the user is kicked out the browser, it is possible that the terminal setting will be incorrect. Enter 'reset' at the unix shell level to reset the terminal settings.

The symbols '>' (greater than) and '<' (less than) as shown in figure A3-1 point to the current line.

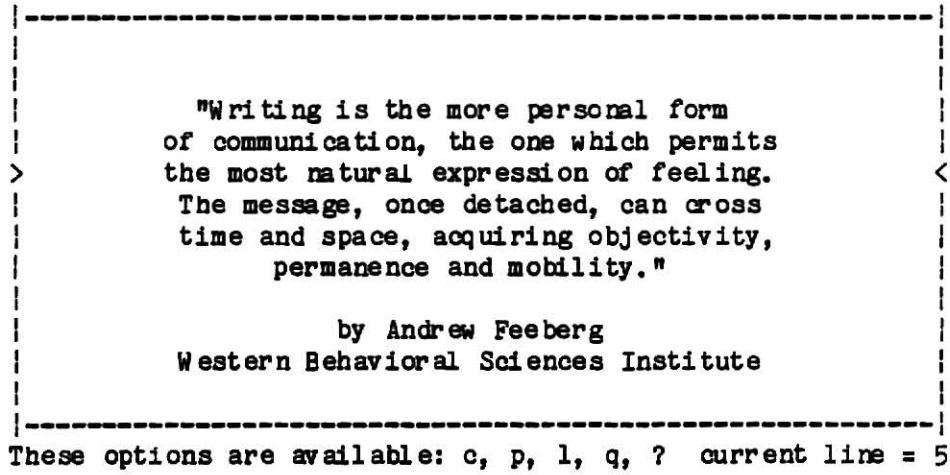


Figure A3-1. The browser page display

The current line is where a comment would be attached if the user were to enter the comment mode to input a comment. When a comment has been attached to the current line, the '<' will be replaced with a '*' (star), as shown in figure A3-2. A comment can only be viewed by moving the '*', found in the outer left hand column, to the current line.

2.2.1 Help

As shown in figure A3-1 a '?' can be entered as one of the options. As a result of entering a '?' the main page display will be cleared and the following information will be displayed on the screen:

The options available within this routine are:

```
          c (comment) - places you into the comment mode.
[[+|-]n]  l (lines)   - advances the display <n> lines.
[[+|-]n]  p (page)    - advances the display <n> pages.
          q (quit)    - exit the browser routine.
          ? (help)    - to display this page.

           Hit return to continue the browser.
```

After reviewing the information the user hits the return key and MARRS returns the user to the main display page.

2.2.2 Going Forwards

The 'p' (page) and 'l' (line) options are to advance the main display forwards 'n' (a number) of pages or lines, respectively. If the user DOES NOT enter a '-' (minus sign) before entering either a 'p' or an 'l' MARRS will step the user forward through the paper. If a '-' is hit and the user decides before entering the 'p' or 'l' that they would rather go forward then a '+' symbol can be used to cancel the '-'. The variable 'n' is constructed of the sequential numbers that the user enters. The following examples show the user's input and the result of the input:

Input	Result
1 5 p	advance 15 pages forward
+ 1 5 l	advance 15 lines forward
- 1 + 6 p	advance 16 pages forward
- 1 2 3 k p	advance 1 page forward

Shown in the last example a 'k' input is an invalid option to the browser, as a result, the previous information put in -123 is cleared out.

2.2.3 Going Backwards

The 'p' (page) and 'l' (line) options are also used to advance the main display backwards 'n' (a number) of pages or lines, respectively. If the user DOES enter a '-' (minus sign) before entering either a 'p' or an 'l' MARRS will step the user backward through the paper. If a '+' is hit and the user decides before entering the 'p' or 'l' that they would rather go backward then a '-' symbol can be used to cancel the '+'. The variable 'n' is constructed of the sequential numbers that the user enters. The following examples show the user's input and the result of the input:

Input	Result
- 1 5 p	advance 15 pages backward
- 1 5 l	advance 15 lines backward
+ 1 - 6 p	advance 16 pages backward
- 1 2 3 k - p	advance 1 page backward

Shown in the last example a 'k' input is an invalid option to the browser, as a result, the previous information put in -123 is cleared out.

2.2.4 Entering the Comments Mode
In order to attach a comment to a specific line or to view a comment from another reviewer the line must be the current line. To enter the comment mode the user uses the 'c' option. The comment menu page display will be placed over the bottom of the browser's main display page.

2.3 Comments Mode

The screen display shown in figure A1-2 shows the condition of a user that has entered the comment mode while a comment had been attached to the line. If no comment had been entered on the current line the '*' would be a '<' and the comment's menu display would not include the 'v' option.

```
|-------------------------------------------------------|
|                                                       |
|                                                       |
|          "Writing is the more personal form           |
|         of communication, the one which permits       |
>         the most natural expression of feeling.       *
|          The message, once detached, can cross        |
.......................................................
.                                                     .
.   You are currently in the Browser's Comment routine.
.                                                     .
.    The options available within this routine are:   .
.                                                     .
.  i (input) - to attach a comment to the current line.
.  q (quit)  - to return to the browsing mode.        .
.  v (view)  - to view any current line comments.     .
.                                                     .
.            Enter one or the options.                .
.                                                     .
.......................................................
```

Figure A3-2. The menu page display while in the comment mode

In order to return to the browsing mode type 'q'. To view

comments, type 'v'. And in order to input a new comment enter the 'i' option.

2.3.1 Input
A simple editor was chosen for the design of entering comments. Once in the comment's input mode the user can enter eleven (11) lines or 76 characters per line. If the user tries to go past the 76 character spot, the input will be truncated. Once the input for a line is entered and the newline (carriage return) has been hit, the line of text just entered cannot be edited. If a comment is typed in incorrectly, it is recommended that the user not target the comment entered and reenter the comment's input mode. In figure A3-3 the page display shows the process for entering input.

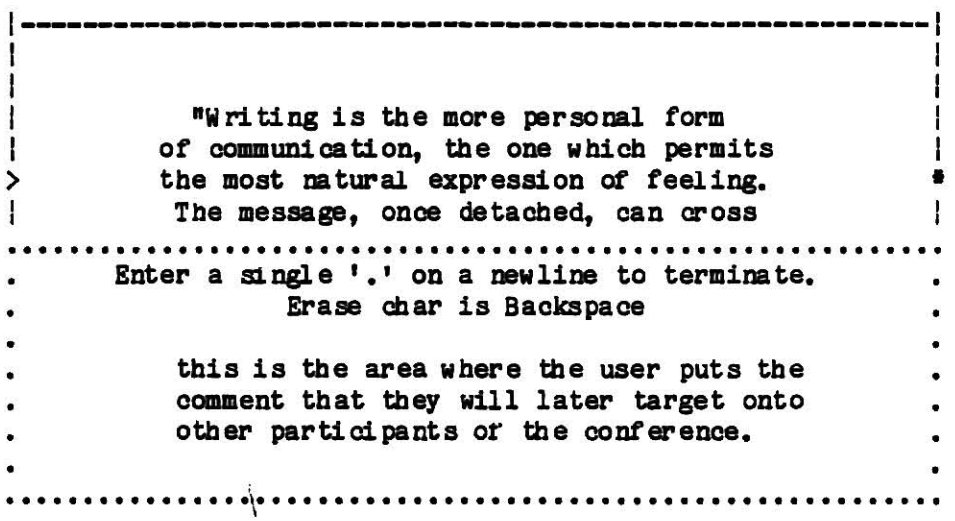


Figure A3-3. The input page display while in the comment mode

The backspace character must be used in order to erase incorrect input. When finished entering the comment, the user must type a '.' followed by a carriage return on a line by itself to terminate the input mode.

2.3.2 Targeting Comments
After the user has entered the comment MARRS will prompt for the target of the comment. The possible targets depend on the status of the paper. In figure A3-4 the example shows the targets or a paper that is in the final review.

```
|-------------------------------------------------------|
|                                                       |
|                                                       |
|          "Writing is the more personal form           |
|         of communication, the one which permits       |
>         the most natural expression of feeling.       *
|          The message, once detached, can cross        |
.......................................................
.                                                     .
.     Who do you wish to target this comment for?     .
.                                                     .
.   * 1 - major professor ( rich )                    .
.     2 - author ( stachowi )                         .
.     3 - committee member ( unger )                  .
.   * 4 - self ( janning )                            .
.     5 - audience                                    .
.     6 - platrorm discussion                         .
.                                                     .
.  To target the comment enter a number, or 'q' to quit.  .
.                                                     .
.......................................................
```

Figure A3-4. The target page display while in the comment mode

As shown in the example the user has typed in a '1' and a '4', which means that the comment will be attached to the major professor's comment file and their own. If the user types in an incorrect target the input can be canceled by retyping in the number. For example, if a '5' is entered and the user decides that the comment should not be sent to all of the audience members after typing in another '5' the '*' will be erased from the display and the comment will not be sent. In the above example, if the user were to type in a '6' the comment would be added to the conference's platform discussion. For more information concerning the platform discussion see the D option. If the user were to type a 'q' the following message would be displayed:

Your comment is currently being sent to the targeted person.

Only if a number has a '*' by it will the comment be sent. If none of the numbers have a '*' by them and a 'q' is entered the following message is displayed:

Since you didn't target the comment no one will see the input

The user is returned to the comment's menu display page.

2.3.3 Viewing Comments

If the current line, as shown in figure A3-5 has a '*' in the left hand column the 'v' option is displayed in the comment's menu display page. After entering a 'v', the comment page is cleared and the first comment is displayed.

```
|-------------------------------------------------------|
|                                                       |
|                                                       |
|          "Writing is the more personal form           |
|        of communication, the one which permits        |
>        the most natural expression of feeling.        *
|         The message, once detached, can cross         |
.........................................................
.                                                       .
.                                                       .
.         this is the area where the comment will       .
.         be displayed that was targeted at the         .
.         reviewer.                                     .
.                                                       .
.                                                       .
.  From: self       These options are available: b, f, q  .
.........................................................
```

Figure A3-5. The viewing page display while in the comment mode

The reviewer's name, who entered the comment, is displayed on the last line. If the comment is one of several comments then the author is given the b or f option. These options will take the user backward ('b') or forward ('f') through the list or attached comments. The 'q' option is used to return the user to the comment's menu display.

2.3.4 Leaving the Comments Mode

To leave the comment mode the user must return to the comment's menu display and enter a 'q'. The user will then be returned to the browser's page display.

3. C - Option 'Create a conference'

This option allows a user to create or initialize a conference. Only a user whose login name is found in MARRS's "professor file" can create a conference.

3.1 Naming the Conference

After the "c" option is selected from the main menu, the following is displayed:

CREATING A CONFERENCE

Enter the conference name you are the major professor of:

A conference name can be at most 15 characters long. Spaces/tabs are not allowed as part of the name. After a name is entered, the following is displayed:

Conference name entered is 'aaaa', OK? (y or n)

If an 'n' is entered, the previous prompt will be displayed again. When a 'y' is entered, further checking is done on the conference name. If the conference name has already been used, the user will be informed by a message "Conference name already exists" and a conference name will again be prompted for.

When a valid conference name is entered and a 'y' is entered, the authors of this new conference are prompted for.

3.2 Adding Authors

The section describes how the authors and their paper names are added to the conference.

After the new conference is named, the following is displayed:

Time to add authors or the conference
NOTE: All authors must be added at this time!

Enter login id of author #1 (Enter '.' when done):

The author's login name must be a valid login name on the system. If it is not, the user will be informed of this and will be prompted for again. When a valid author's name (call it 'stachowi') is entered, the next prompt shows:

```
Enter paper name for stachowi:
```

A paper name with a maximum of 14 characters is allowed. Spaces/tabs are not allowed as part of the name. After a valid paper name (call it 'telesystems') is entered, the following is displayed:

```
Paper name entered is 'telesystems', OK? (y or n)
```

If an 'n' is entered, the previous prompt will be displayed again. When a valid paper name is entered and a 'y' is entered, another author will be prompted for (as described above).

There can be a maximum of 10 authors per conference. An author's name cannot be the same as any other participant already entered into the conference.

When all authors and paper names have been added, enter a '.' at the author's login id prompt. The committee members of the conference will now be prompted for.

3.3 Adding Committee Members

The section describes how the committee members are added to a conference. Since only a major professor can create a conference -- and since a major professor is always a committee member, the user creating a conference is automatically added as a committee member.

After all the authors are added by the major professor (called 'rich') , the following is displayed:

```
Time to add committee members of the conference
NOTE: All committee must be added at this time!

Committee member 'rich' is being added
Enter login id of committee member #2
  (Enter '.' when done):
```

The committee member's login name must be found in MARRS's "professor file". If it is not, the user will be informed of this and will be prompted for again.

A committee member's name cannot be the same as any other participant already entered into the conference. When a valid committee member's name is entered, the next committee member is prompted for.

When all committee members have been added, enter a '.' at the committee member prompt. The audience members of the conference will now be prompted for.

3.4 Adding Audience Members

The section describes how the audience members are added to a conference. Note: All audience members do not have to be added at this point. An option "u" is available at the main MARRS menu to update the audience member list.

After all the committee members are added by tr professor, the following is displayed:

```
Time to add audience members to this conference

Enter login id or audience member #1
 (Enter '.' when done):
```

The audience member's login name must be a valid login name on the system. If it is not, the user will be informed of this and will be prompted for it again.

An audience member's name cannot be the same as any other participant already entered into the conference. When a valid audience member's name is entered, the next audience member is prompted for.

When all audience members have been added, enter a '.' at the audience member's id prompt.

3.5 Conference Structure Setup

Following the entry of the conference participants, appropriate directories are created and files written. The user is then returned to the main MARRS menu.

The conference setup is done automatically. Therefore the

user does not have to take any more specific actions. However, the user should allow several seconds for the program to setup the conrerence. Once the user is returned to the main MARRS menu, the setup is complete.

4. D - Option 'The (Platrorm) discussion'

This option gives the user the capability to view or attach an item in the platform discussion. The items in the platrorm discussion are available to all participants of the conference. There is a restriction in the platform discussion that at least one of the papers in the conference must be in final review. Once in the platform discussion, the user gets a list or conferences that the user is a member of and providing that there is more than one conference, the user will be requested to enter a conference name. The user can enter 'q' to quit at this point and return to the marrs main menu display. The user can also enter a conference name. If the user is a member of only one conference, then the user is not requested to enter conference name. The following screen is display for a user that is a member or more than one conrerence :

Conferences available:

marrs	tjs1	tjs2
star_wars	test_con	dummy_con

Enter conference name :

The user then enters a conrerence name from the list of available conferences. If the user enters a conference and there are no papers in final review for that conference, the following message is printed :

Sorry, there are no papers in final review for conference (conference name).

If the user enters 'marrs' as the conference name and there is a paper in final review, then the following is displayed :

Conference : marrs
Open items : none

Enter Valid Option (t, a, q, ?) :

The 'Open items' field lists any new items that the user has not read. Once the user reads an item, it will no longer be listed as an open item. The user can now select one of the available options. They are the following :

- type(t)
- attach(a)
- quit(q)
- help(?)

4.1 Type(t)

The type option allows the user to print (or type) out an item that is in the platform discussion. If there are no items in the platform discussion and the user selects the type option, the following message is printed :

```
Sorry, there are no items in the platform
discussion for conference (conference name).
```

If there are items in the platform discussion, the following is displayed when the 't' option is entered :

```
Conference : marrs
Open items :  none

Enter Valid Option ( t, a, q, ? ) : t

Items available:
 1  2  3

Enter Item Number (q to quit) : 2

Item 2: ( monk )
   response to item: #1
   followed by item: #3

This is platform item 2.


End or item.   Hit return to continue
```

The 'Items available' is a list of all items in the platform discussion. The user enters an item number followed by a carriage return. In this example, a '2' was entered. The heading gives the item number, login name of the person who made the comment, what item this item is in response to, and

also if this item is followed by another item. If the item is not in response to another item or if it is not followed by another item, then these lines are not printed. The item is then printed followed by a message that says you're at the end of item and to hit return to continue with the platform discussion.

4.2 Attach(a)

The attach option allows the user to enter a new item in the platform discussion and if desired to attach this item to another item in the platform discussion. An item can only have one successor and one predecessor. If the user tries to attach the item to an item that already has a response to it (successor), the following message is printed :

Item already has a response.

An example or the attach option follows :

```
Conference : marrs
Open items :  none

Enter Valid Option ( t, a, q, ? ) : a

Response to item # (0 for none) : 0

You are in the 'ed' editor
0
```

The 'Response to item # (0 for none)' line is printed only when there are items in the platform discussion. If the user just wants to enter a new item, a '0' indicates that this in not in response to any other item. The user is then put into an editor. If the user's EDITOR environment variable is set, the user will be put in that editor. If it isn't, then the default editor is the 'ed' editor. When the user finishes entering the new item, the user enters a 'w' to write the new item and then a 'q' (if the 'ed' editor is being used) to quit the editor. The user then returns to the platform discussion display. All other users in this conference will get this new item as an open item when they enter the platform discussion.

4.3 Help(?)

The help (?) option is to help explain to the user what the various options of the platrorm discussion are. The following screen is displayed :

```
Conrerence : marrs
Open items :  none

Enter Valid Option ( t, a, q, ? ) :

     Valid options are:

               a - add an item to the platform discussion
               t - type an item in the platrorm discussion
               ? - this display
               q - leave the platform discussion


     Hit return to continue
```

To continue with the platrorm discussion, the user hits the return.

4.4 Quit

To exit the platform discussion, the user enters a 'q' for quit. The user returns back to the marrs main menu display. An example is :

```
Conrerence : marrs
Open items :  none

Enter Valid Option ( t, a, q, ? ) : q
```

The user is returned to then returned to the marrs main menu display.

5. G - Generate a Hard Copy

This option allows a user to generate a hard copy of a paper. The hard copy includes all the comments on the paper that were directed to the user. A user can only use this option in a conference to which he/she belongs.

5.1 Selecting the Conference

After the "g" option is selected from the main menu, a search is done to see what conferences are available in which the user is a participant.

If no conferences exist for which the user is a participant, "No conferences exist for you" will be displayed and the user is returned to the main MARRS menu.

If only one conference exists for which the user is a participant in, the user will skip this part on "Selecting the Conference" and go directly to the "Selecting the Author/Paper" section.

If, however, more than one conference exists for which the user is the major professor of, the following display is shown:

```
                    PRINT COMMENT MODE

     Conferences available:

          aaaa          bbbb          cccc          dddd


     Enter a conference name ('q' to quit):
```

At this point a valid conference name, from the list given, should be entered. If a 'q' is entered, the user is returned to the main MARRS menu. If an invalid conference name is entered, the user is informed of the "Invalid conference name" and prompted for another conference name.

When a valid conference name is entered, the user is allowed to select the author or the paper to be printed.

5.2 Selecting the Author/Paper

Following the selection or a conference, the next display shows something like:

```
                Conference: aaaa

        Author          Paper Name
        ------          ----------
        stachowi        telesystems
        janning         features
        monk            datamanager

    Enter author's name whose paper you want comments on
     ('q' to quit):
```

The author's login name entered must be from the list presented. If it is not, the user will be informed of this and will be prompted for another name.

When a valid audience member's name is entered, a check is done to see if the paper is available for reading. If it is not, the user will get an appropriate message and be prompted for another name.

If no comments have been directed to the user on the specified author/paper, the user will get an appropriate message and be prompted for another name.

If the paper is present and there are comments for the user, the paper and the comments will be spooled to a printer. Note: The printer can be selected by the environment variable MARRS_PRTR -- See the Introduction.

Following this, the author/paper menu is displayed again. At this point, another author's name can be entered. If a 'q' is entered, the user is returned to the main MARRS menu.

6. K - Option 'The Keyword Search'

This option gives the user the capability to perform keyword searching on published papers. A paper that is in the 'Ready to be published' state has keyword topics associated with it. There can be a maximum of ten keywords per paper. When the user enters the keyword search routine, the following is displayed :

Enter topic (? for help, q to quit) :

The user can enter a topic or a '?' for a help message. If a '?' is entered, the following is displayed :

The Keyword search provides the capability to
list the conference names and paper file names
associated with a requested topic.

The topic can be at most 20 characters.

Hit return to continue

After the user enters a topic, the following is displayed :

```
Conference:
Paper name:

Conference:
Paper name:

Conference:
Paper name:

Enter paper name to see paper title (0 for none) :
```

The user can now enter the paper name to see the paper's title. A '0' can be entered to continue with the keyword search. If a paper name is entered, the following is displayed :

```
Paper name:
Title:

Hit return to continue
```

7. L - List Current Conferences'

This option allows a user to see what conferences currently exist on the system. In addition, the user can select specific conference in order to get additional information on the conference.

7.1 Selecting the Conference

After the "l" option is selected from the main menu, all conference names are displayed. Note: The use of this option does not require that the user belong to the conference.

When conferences exist on the system, a display similar to the following is shown:

```
                        CONFERENCE LISTING

     Conferences available:

          aaaa          bbbb          cccc          dddd
          eeee          ffff

     For more information, enter a conference name
      ('q' to quit):
```

At this point a valid conference name, from the list given, should be entered. If a 'q' is entered, the user is returned to the main MARRS menu. If an invalid conference name is entered, the user is informed of the "Invalid conference name" and prompted for another conference name.

If a valid conference name is entered, the user is given more information on the conference. The following is an example of what might be printed for conference 'bbbb':

```
                        Conference: bbbb

          AUTHOR        PAPER NAME      PAPER STATUS
          ------        ----------      ------------
          stachowi      telesystems     SUBMITTED
          janning       features        SUBMITTED
          monk          datamanager     FINAL REVIEW

     MAJOR PROFESSOR:    rich
```

```
COMMITTEE MEMBERS:  gustoff  unger

AUDIENCE MEMBERS:   humphry  hummel   chance   merrit

Hit return to continue:
```

At this point, a carriage return can be entered to get to the "conference listing" menu previously described. The user can then select another conference name or enter "q" to quit and return to the main MARRS menu.

8. M - Leave Message on Bulletin Board

This option allows a user to read and/or change the bulletin board for a specific conference. An editor is used to add, delete, or modify the bulletin board. A user can only use this option in a conference to which he/she is a participant.

8.1 Selecting the Conference

After the "m" option is selected from the main menu, a search is done to see what conferences are available in which the user is a participant.

If no conferences exist for which the user is a participant, "No conferences exist for you" will be displayed and the user is returned to the main MARRS menu.

If only one conference exists for which the user is a participant in, the user will skip this part on "Selecting the Conference" and go directly to the "Bulletin Board Option Selection" section.

If, however, more than one conference exists for which the user is a participant in, the following display is shown:

```
                    BULLETIN BOARD

     Conferences available:

              aaaa          bbbb          cccc          dddd

     Enter a conference name ('q' to quit):
```

At this point a valid conference name, from the list given, should be entered. If a 'q' is entered, the user is returned to the main MARRS menu. If an invalid conference name is entered, the user is informed of the "Invalid conference name" and prompted for another conference name.

When a valid conference name is entered, the user is given the bulletin board option menu.

8.2 Bulletin Board Option Selection

Following the selection of a conference, the user is given a choice of several options as shown:

```
Valid options are:

        read   - read the bulletin board
        change - change (edit) the bulletin board
        quit   - leave the bulletin board

Function? :
```

If the user selects "q" (quit), control is passed back to the main MARRS menu.

8.2.1 Change Bulletin Board

If a "c" is entered at the bulletin board option display, an editor is invoked so that the user can add to, delete from, or modify the conferences bulletin board.

An editor is used to alter the bulletin board. The bulletin board can be made in any desired format or style. Note: The editor used can be selected by the environment variable EDITOR -- See the Introduction.

When the editor is exited, control is passed back to the bulletin board option menu.

8.2.2 Read Bulletin Board

If a "r" is entered at the bulletin board option display, the "more" command is invoked so that the user can read the board. If no information is present on the bulletin board, a message indicating this will be output and control is passed back to the option menu.

If there is information on the bulletin board, it will be displayed. If more than one page is present, the space bar can be used to advance the board by a page. A <cr> (carriage return) can be used to advance by a single line.

When the end of the board is reached, the user will see:

```
AT BOTTOM OF BULLETIN BOARD
HIT RETURN WHEN YOU ARE PREPARED TO CONTINUE
```

At this point a <cr> can be entered to get back to the following options menu:

```
Valid options are:

        read   - read the bulletin board
        change - change (edit) the bulletin board
        quit   - leave the bulletin board

Function? :
```

Again the user now has the choice of r(eading), c(hanging), or q(uitting).

9. P - Create/Edit a Paper

This option allows a user to edit a paper in a conference. The editor is used to create the paper when it hasn't already been created. A user can only use this option in a conference which he/she is an author.

9.1 Selecting the Conference

After the "p" option is selected from the main menu, a search is done to see what conferences the user is an author in.

If no conferences exist for which the user is an author, "No conferences exist for you" will be displayed and the user is returned to the main MARRS menu.

If only one conference exists for which the user is an author in, the user will skip this part on "Selecting the Conference" and go directly to the "Editting the Paper" section.

If, however, more than one conference exists for which the user is an author in, the following display is shown:

```
                    CREATE/EDIT A PAPER

     Conferences available:

          aaaa          bbbb          cccc          dddd

     Enter a conference name ('q' to quit):
```

At this point a valid conference name, from the list given, should be entered. If a 'q' is entered, the user is returned to the main MARRS menu. If an invalid conference name is entered, the user is informed of the "Invalid conference name" and prompted for another conference name.

When a valid conference name is entered, the user is allowed to edit his/her paper.

9.2 Editing the Paper

Following the selection of a conference, an editor is invoked so that the user can alter the paper. Note: The editor used can be selected by the environment variable EDITOR -- See the Introduction.

When the editor is exited, the user may be given a chance to submit the paper to the major professor.

9.3 Submitting the Paper

Following the editing session, the user is given a chance to submit the paper if it is in the "not submitted" or the "re-work" state. If this is the case, something like the following is displayed:

```
                    Conference: aaaa

    Author     Paper Status        Status Date
    ------     ------------        -----------
    janning    NOT SUBMITTED       Wed Jun 11 13:33:56 1986

    Changing status from NOT SUBMITTED to SUBMITTED.
      OK? ('y' or 'n'):
```

If the user desires to submit the paper, a 'y' should be entered. At this point, the paper is "nroff'd" and comments associated with the paper are removed. The user is then returned to the main MARRS menu. Note: The nroff command used can be selected by the environment variable MARRS_NROFF -- See the Introduction.

If an 'n' is entered in response to the 'change status' question, the user is simply returned to the main MARRS menu.

If following the editing session, the paper is not in a state where it can be changed to "submitted", information on the paper is shown (as above) along with a message indicating that a status change is not allowed. The user is then returned to the main MARRS menu.

10. R - Reminde Idle Users

This option allows a major professor to remind idle users in a conference. A user can only use this option for a conference in which he/she is a *major professor*.

As a conference proceeds, the papers in that conference undergo several state changes. Two of these states are the committee and final review. This option gives the major professor the ability to remind idle users that a specific author's paper within a conrerence is being reviewed. An idle reviewer is defined by the condition of a reviewer that has no previous location stored for the paper.

10.1 Selecting the Conference

After the "r" option is selected from the main menu, a search is done to see what conferences are available in which the user is a major professor.

If no conferences exist for which the user is a major proressor, "No conferences exist for you" will be displayed and the user is returned to the main MARRS menu.

If only one conference exists for which the user is a major professor in, the user will skip this part on "Selecting the Conrerence" and go directly to the "Selecting the Author" section.

If, however, more than one conrerence exists for which the user is the major professor of, the following display is shown:

REMIND IDLE USERS

Conrerences available:

aaaa bbbb cccc dddd

Enter a conrerence name ('q' to quit):

At this point a valid conrerence name, from the list given, should be entered. If a 'q' is entered, the user is returned to the main MARRS menu. If an invalid conference name is entered, the user is informed of the "Invalid conference name" and prompted for another conference name.

When a valid conference name is entered, the user is allowed to select an author of the paper whose participants are to be reminded.

10.2 Selecting the Author

Following the selection of a conference, the next display shows something like:

```
                    Conference: cccc

Author          Paper Status        Status Date
------          ------------        -----------
janning         FINAL REVIEW        Thu Jun 11 07:35:59 1986
stachowi        COMMITTEE REVIEW    Thu Jun 10 07:31:58 1986

Enter author's name whose
reviewers are to be reminded, ('q' to quit) :
```

The author's login name entered must be from the list presented. If it is not, the user will be informed of this and will be prompted for another name. Entering a "q" will return the user to the main MARRS menu.

After a valid author's login name is entered the MARRS routine will issue mail to all idle user's of the paper. The user is then reprompted, if there is more than one paper, for another author's name. If there is only one paper or after the user types in a "q" for the author's prompt the user is returned to the main MARRS menu.

11. S - Change Status or Paper'

This option allows a user to change the status of a paper in a conference. A user can only use this option for a conference in which he/she is a major professor.

As a conference proceeds, the papers in that conference undergo several status changes. Only certain changes are allowed from any given state. The following table shows the allowable state changes:

State	Allowable new state(s)
Not submitted	-> paper submitted
paper submitted	-> re-work, committee review final review, paper published
re-work	-> paper submitted
committee review	-> re-work, final review
final review	-> re-work, paper published
paper published	-> (no allowable state changes)

11.1 Selecting the Conference

After the "s" option is selected from the main menu, a search is done to see what conferences are available in which the user is a major professor.

If no conferences exist for which the user is a major professor, "No conferences exist for you" will be displayed and the user is returned to the main MARRS menu.

If only one conference exists for which the user is a major professor in, the user will skip this part on "Selecting the Conference" and go directly to the "Selecting the Author" section.

If, however, more than one conference exists for which the user is the major professor of, the following display is shown:

PAPER STATUS CHANGE

Conferences available:

aaaa bbbb cccc dddd

Enter a conrerence name ('q' to quit):

At this point a valid conference name, from the list given, should be entered. If a 'q' is entered, the user is returned to the main MARRS menu. If an invalid conference name is entered, the user is informed of the "Invalid conference name" and prompted for another conference name.

When a valid conference name is entered, the user is allowed to select an author of the paper whose status will be changed.

11.2 Selecting the Author

Following the selection of a conference, the next display shows something like:

Conference: cccc

Author	Paper Status	Status Date
janning	SUBMITTED	Thu Jun 11 07:35:59 1986
stachowi	SUBMITTED	Thu Jun 10 07:31:58 1986

Enter author's name whose status
will be changed ('q' to quit):

The author's login name entered must be from the list presented. Ir it is not, the user will be informed of this and will be prompted for another name. Entering a "q" will return the user to the main MARRS menu.

When a valid audience member's name is entered, a new display will show the allowable state changes for the author/paper selected.

11.3 Making the Status Change

Following the selection of aa author, the next display shows something like:

```
Current status for stachowi is SUBMITTED

Allowable state changes are to:
                        re (Re-work Needed)
                        co (Committee Review)
                        fi (Final Review)
                        pa (Paper Published)

Enter new status ('.' for no change):
```

The user can now enter any of the available state changes listed on the display. If an invalid state is entered, the user is informed or this and re-prompted for a new status.

If a valid state is entered, the change is recorded and mail is sent to the appropriate people about this status change. Following this, control is returned to the author's menu. The new menu will look something like (assume "co" was entered as the new state):

```
                        Conference: cccc

Author          Paper Status            Status Date
------          ------------            -----------
janning         SUBMITTED               Thu Jun 11 07:35:59 1986
stachowi        COMMITTEE REVIEW        Thu Jun 12 07:34:11 1986

Enter author's name whose status will be changed ('q' to quit):
```

As before, the user can quit or make additional status changes. Entering a "q" will return the user to the main MARRS menu.

12. U - Option 'Update Audience Member List'

This option allows a user to add audience members to a conference. Only the major professor of an already initialized conference (see the "c" option) can add a audience members to a conference.

The total number of participants (authors, committee members, and audience members) that can be in one conference is limited to 30.

12.1 Selecting the Conference

After the "u" option is selected from the main menu, a search is done to see what conferences are available in which the user is the major professor of.

If no conferences exist for which the user is the major professor, "No conferences exist for you" will be displayed and the user is returned to the main MARRS menu.

If only one conference exists for which the user is the major proressor or, the user will skip this part on "Selecting the Conference" and go directly to the "Adding Audience Members" section.

If, however, more than one conference exists for which the user is the major professor or, the following display is shown:

```
                    ADDING AUDIENCE MEMBER(S)

     Conferences available:

          aaaa          bbbb          cccc          dddd

     Enter a conference name ('q' to quit):
```

At this point a valid conference name, from the list given, should be entered. If a 'q' is entered, the user is returned to the main MARRS menu. If an invalid conference name is entered, the user is informed of the "Invalid conference name" and prompted for another conference name.

When a valid conference name is entered, the user is allowed to add new audience members.

12.2 Adding Audience Members

Following the selection or a conference, the next display shows:

```
Adding new audience members to this conference

Enter login id or audience member #4
 (Enter '.' when done):
```

The audience member's login name must be a valid login name on the system. If it is not, the user will be informed of this and will be prompted for another name. Note: In the above example, it is assumed that three audience members have previously been added.

An audience member's name cannot be the same as any other participant already entered into the conference. When a valid audience member's name is entered, the next audience member is prompted for.

When all audience members have been added, enter a '.' at the audience member's id prompt. At this point, appropriate directories are created and files updated. The user is then returned to the main MARRS menu.

FEATURES OF THE MARRS COMPUTER CONFERENCING SYSTEM

by

RONALD M. JANNING

B. S. University of Dayton, 1980

AN ABSTRACT OF A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department or Computer Science

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1986

AN ABSTRACT OF A MASTER'S REPORT

This paper presents MARRS (MAster's Report Reviewing System), a conferencing system, designed to provide a mechanism for tracking Master's Reports through the various pnases or revision. The system was implemented on a UNIX based system. The features of the MARRS computer conferencing system are discussed in this paper.

The services provided by MARRS are simple and easy to use by a community of users. Papers are the major element in the system. They are grouped into conferences. The major professor controls the papers within the conference from the time or submission until the time that they are deemed acceptable for publication. During this process, the major professor has the opportunity to name a list or committee and audience members to act as reviewers for the paper. During the "committee review" phase committee members, which automatically includes the major professor, and the author or the paper have an opportunity to comment on the paper. During the "final review" phase audience members are permitted to also comment on the paper. Access to the paper and reviewers' comments are controlled by the major professor and MARRS.