Data Management in MARRS

by

Kitty A. Monk

B.S. Miami University, 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 :

[illegible]
Major Professor

CONTENTS

Acknowledgements

I would like to dedicate this work to my family and friends. Their love and understanding allowed all this to be possible. I would especially like to thank my advisor, Dr. Rich McBride, for the many hours of time spent in suggesting ideas and guiding this work.

1. Introduction

This paper discusses a computer teleconferencing system that is designed for overseeing the completion of a software design project. The software design has been focused toward a specific type of project. In particular, this conferencing system is to help automate the process of the collecting, reviewing, and presenting of master's reports. This paper does not describe the computer teleconferencing system in detail. Rather it focuses on the data management aspects of the computer teleconferencing system.

Computer teleconferencing can be defined as an interactive group-communication through a computer where individuals in the group can be at different locations [CROS84]. Computer teleconferencing attempts to eliminate the problems associated with people attending meetings. Some typical problems include traveling costs, time and schedule conflicts, thinking on the spot, interest in only one particular item on the agenda. Computer teleconferencing provides many of the features found in a face-to-face conference to its participants. Some of the features found in a computer teleconference include conference and discussion areas, note pads, bulletin boards, status and tracking functions, on-line search operations, voting and data gathering. These features are described in detail in [JANN86].

Some advantages of computer teleconferencing are :

- No travel time and cost (frees a person from the necessity of attending a meeting),
- No time constraints involving time zone changes and conflicting schedules,
- The conference is self-documenting. Any part of the document/discussion can be retrieved at will,
- Users can take time to think before making a comment,
- Users can select items of the conference/discussions in which they want to participate. This is different from a face-to-face conference where an attendee might have to sit through the conference and listen to all the speakers when the interest is only in one speaker,
- Users of a computer conference do not have to 'be' at the conference at the same time. They can 'attend' the conference at their convenience.

Some disadvantages associated with computer teleconferencing are :

- No face-to-face meeting,
- The ability of managers to maintain closer contact and control may make users feel restricted,
- Lack of face-to-face meetings could reduce professional development and contacts.

This paper consists of four chapters.

Computer teleconferencing systems create, manage, and communicate text among the participants. The participants can read, access and respond to this text [CROS83]. The second chapter looks at the relationship of data bases to computer teleconferencing systems. It also looks at some existing computer teleconferencing systems and how they manage their data.

The third chapter is an overview of the system and it looks in-depth at how our computer teleconferencing system will manage the data.

The final chapter contains a summarization and suggests possible extensions to this work.

2. Review of the Literature

2.1 Introduction

A computer teleconferencing system, as defined in the previous chapter, is an interactive group-communication where individuals in the group can be at different locations. The users of the computer teleconferencing system will be accessing, updating, and creating data. How the computer teleconferencing system manages this data is of concern here. We now proceed to investigate what advantages or disadvantages file processing systems have versus data base management systems.

In file processing systems, each file is considered to exist independently. A program directly accesses files in order to retrieve the information requested. The program must contain knowledge about the structure of each file which it can potentially access. Problems can occur when the user needs information that is contained in a number of files. The files might have different formats. For example, a file might adhere to a binary format which is allowed by one language (e.g., COBOL), but not by another (e.g., PL/I). One solution to this problem could be to change the files to one format. However, the application programs accessing these files will now have to be changed to be compatible with the new formats. This conversion could be costly and time-consuming. Another solution could be to duplicate the data.

Data duplication could lead to problems in that the data in one file might be changed while the same data in another file is not. This problem could cause a lack of data integrity. This lack of data integrity would lead to conflicting reports being generated. Another problem deals with program/data independence. Each program contains the file structures of the files it uses. Therefore, if a data structure in a file is changed, all programs using this file must be changed.

Data base management systems have advantages over file processing systems with respect to the preceding areas. In a data base management system there is data integration. All of the data is stored in a data base. Any program must go through a data base management system in order to get at the data in the data base. Thus, the retrieval of information from more than one file is not a problem. The data base management system has not only stored the data in the files, but also has stored a description of the storage format of the data. The programs can now access data from any of the files without concern for the format of the data.

Another advantage of data base management systems is that more information is available from a given amount of data. The programs by going through the data base management system can access the data and also utilize the relationships that exist between the files. Also, there is no great need to duplicate the data in any of the files. This reduction or elimination of data duplication is another advantage of data base management systems. Less data duplication promotes data integrity. A further advantage is that of program/data independence. Since the programs go through the data base management system to access data, any data structure change affects only the data base management system instead of each program using this data.

Other advantages of a data base management system are related to data base utility programs and portability. The data base utility programs provide administrative actions such as creating, maintaining, and restoring the data base. A query/update facility allows the user to access the data without using an application program. Data base management systems are typically not designed to be used with just one particular machine architecture. Typically machine-sensitive information is isolated in one or two modules supplied to the data base management system. Also, with little or no modification, application programs can be ported to different machines. With these advantages in mind, a data base management system could be an integral part of computer teleconferencing systems.

A major part of computer teleconferencing systems is the sending and receiving of messages. A message could be a comment, a response, or even a broadcast message. An important distinction should be made here between computer message systems and computer teleconferencing systems. Computer message systems merely send and receive messages. They are not concerned with managing the information within a message. Computer teleconferencing systems are concerned with both the management of the information in the message and sending and receiving messages among users of the system. Computer teleconferencing systems must provide an on-going electronic filing system for archival and retrieval of information [CROS84].

This chapter will explore the relationship of data bases to computer teleconferencing systems. It will continue with a list of terms and their definitions which are used in this paper. Also, it will provide a brief overview of the computer teleconferencing system which we are implementing. And finally, it will look into how some existing computer teleconferencing systems use files and file structures.

2.2 Terms and Definitions

A list of some common terms used throughout this paper and their definitions are the following :

1. computer teleconferencing system - a computer software system that allows interactive communications among people in different geographical locations [CROS84].

2. asynchronous communication - store and forward communication in which the recipient of a message need not be present when the message is sent. The message can be picked up when it is convenient.

3. data base - collection of data associated with a conference. This collection will also contain a description of the stored data.

4. data base management system - allows users to access and update data in the data base.

5. data base administrator - responsible for managing the data base. Taking care of user conflicts and planning for future requirements [KROE83].

2.3 Brief System Overview

Our system is an asynchronous computer teleconferencing system. It is designed to automate the collection, review, and presentation processes of the Master's project reports.

The user types found in our computer teleconferencing system and their definitions are :

1. conference system administrator - the initiator of the conferencing system. Some responsibilities are to initiate the conferencing system, to establish a list of faculty members, and to maintain and upgrade the conferencing system.

2. major professor - the coordinator of a master's project. The major professor has the capability to establish participants in a conference, assign and delete committee members, assign and delete authors, and assign and delete audience members. The major professor can change the status of the paper associated with the project. The major professor can review, comment, and vote on the paper. Also, the major professor can check the status of the paper and if comments made have been read.

3. author - the writer of a master's project paper. The author can

enter and edit a paper. The functions this person can perform include submitting the paper to the major professor, reviewing and responding to comments, and requesting the status of the paper and participants' names.

4. committee member - a judge of the master's project paper. The functions the committee member can perform include reviewing, making comments, and voting on the paper. The committee member can check the status of the paper and see if comments made have been read.

5. audience member - a participant in the conference. The audience member can review and comment on the paper only when it is in final review. This person can check the status of the paper and determine if comments made have been read.

Some of the major actions which occur in our teleconferencing system are described in the following scenario. An author submits the paper to the major professor. When it is acceptable, the major professor sends the paper to committee review. After this, the paper goes to final review where all participants in the conference can review and comment on the paper.

The users of our computer teleconferencing systems will want to access and use the conference's data in many ways. Some general user functions include the following :

- to access and change data,
- to retrieve documents/files and edit,
- to search and list information,
- to create data,
- to search by keywords, date, or another defined field,
- to send and receive messages,
- to list biographies,
- to reorganize existing data into a report,
- to accumulate or summarize data.

2.4 Data bases and Computer Teleconferencing Systems

Today there is research into the integration of data base management systems and computer message systems [TSIC81]. Data base management systems are concerned with the electronic filing of information. Communication of this stored information is implicit in the concept of data base management systems. Stored information can be retrieved, or communicated, to a user at a future date.

Communication is usually thought of as the primary facility provided by computer message systems. Messages are sent and received between users. [TSIC81] explores the idea that "there is no substantial conceptual difference between message systems and transaction oriented data base systems". One major difference between data base management systems and computer message systems lies in the structure of the data objects used. In data base management systems, records have a fixed format to follow. Messages in computer message systems are relatively format-free. A message usually consist merely of a header and body of text. [TSIC81] and [TSIC82] indicate that data base management systems are trying to get away from the fixed format and that computer message systems will have to start having some kind of fixed format.

Another difference between data base management systems and computer message systems deals with the ownership of data. In computer message systems, a sender owns the message and sends it to a recipient. Ownership of the message is transferred from the sender to the receiver. One could say that sometime between the sender owning it and then the receiver owning it, the system temporarily owns the message. In data base management systems, the data base is thought of as the owner of the data. A user accesses this data and temporarily owns it. The user can modify the data then return it to the data base and also return ownership. At a later time, another data base user accessing this modified data temporarily owns it. So ownership does not appear to be a difference between the two since in both systems, one user temporarily owns it, then the system owns it, and finally another user owns it.

Tsichritzis [TSIC81] also discusses such differences as notification, recording, interpretation, and implementation, and shows that they have no real differences. The main emphasis in each system is what differentiates the two. In data base management systems, the main concern is management of data, while in computer message systems, the main concern is communicating messages.

Data base management systems provide security by allowing users to have different permissions, or authorizations for data access. Users of the data base management system share the same data, and potentially might try to update the same data item. To prevent data inconsistencies from simultaneous updates, data base management systems provide locking. The level of locking can range from individual data items to the entire data base. If a user's update request involves a small amount of processing time, then a lock of the entire data base might be acceptable. Locks can also be placed on subsets of the data base, files, and records. The duration of the lock should also be considered. It could be for the processing of an entire job or for a single user request.

There are different views of the data in a data base system. The conceptual schema is the complete, logical view of all the data in the data base. The subschema is a subset of the schema, and it reflects the way in which data in the data base is viewed by individual users. The subschema defines the data an application program would use. The physical view is how the computer sees the data. The data base management system has application programs that satisfy specific user needs. There are also data base management utility programs. These provide a generalized interface to the data base.

The same ideas can be found in computer teleconferencing systems. Utility programs provide administrative actions on the data as well as providing a query and update facility. Application programs can be written to satisfy the specific needs. The physical view is how the computer sees the data. If the computer teleconferencing system has a data base management system, the schema and subschema will be as defined above. If a computer teleconferencing system uses a file processing system, the application programs using the files must contain the file structure and will provide the subschema. The conceptual view of the data will be all the files.

Can a data base management system be used to create a computer teleconferencing system? Data base management systems manage data, but do not necessarily communicate this information to users. When an update takes place, users are not notified. They would find out when they next access this data. Data base management systems provide for user queries. However, the status and tracking facilities found in a computer teleconferencing system let the user know about new comments, new mail, and status changes. A data base management system does not automatically provide these items; a user has to access the data to know of any change. Data base management systems do provide for data management, which is also a major function of computer teleconferencing systems. Communicating messages is also a major function, and data base management systems do not emphasize this. However, an application program could be written for the data base management system to provide such a communication management function.

The user roles in a data base management system can be compared to the user roles in our teleconferencing system. In a data base management system, there is a data base administrator who is responsible for managing the data base. In a conferencing system, there must be a conference system administrator that is responsible for managing the conferences. The data base manager is responsible for the user permissions for accessing data. The major professor has this responsibility in our conferencing system. Other types of users that are common between the two systems are the application programmers and the naive users. The application programmers are responsible for the programs which access the data. The naive users just type the available commands without knowing the details of the programs which these commands invoke.

2.5 Existing Computer Teleconferencing Systems

In this section existing computer teleconferencing systems will be examined. They will be looked at in terms of their files and file structures.

2.5.1 CBIE

The Computer Buffered Information Exchange (CBIE) is discussed in detail in [STRO80] and [STRO82]. It was developed at Columbia University and is currently implemented under the UNIX*

*UNIX is a Trademark of AT&T Bell Laboratories operating system. The CBIE commands are similar to the UNIX shell in the structure of the command. The command verb appears first followed by the arguments.

CBIE is an asynchronous conferencing system and organizes its conference items into a network of items. The conference items can have predecessors and successors. New items added become the successor of the item currently being viewed. Conference items contain a header and a pointer to the text. Associated with each conference item is an item number. This is used in establishing the links between items. Conferees enter their items into a hierarchical file system where they are stored here for future retrieval. In CBIE, "the structuring of the interrelationships among items is of major importance" [STRO80].

The overall file organization of CBIE is like that of UNIX in that files are organized hierarchically. Each directory contains either files or directories. For the storage of data, CBIE uses a flat-file concept. Individual conference items are maintained in a single file which is pointed to by an index file. In the conference item file, there are four separate attributes maintained for each conference item. The attributes of the item include the header, successor list, predecessor list, and actual text of the conference item. CBIE maintains an index file which contains file pointers. There are four pointers associated with each indexed item, namely its header, successor, predecessor and text. This index file also contains the actual length of the conference item and the total length allocated for this conference item. The header contains the author's initials, creation date, title, date and time this item was last modified, flags to indicate the presence of text and/or successors and the item number. The successor list and the predecessor list can be used to move around the network of conference items.

Minimal locking is utilized in CBIE. A lock is placed on text being edited by the user. Since most updates in CBIE are fast, a lock on the entire data base is used.

In CBIE there are two types of users, inside users and outside users. The inside users are known to the UNIX operating system and use the 'CBIE' command to gain access to the system. The outside users must first log in to 'CBIE' and then CBIE will request their login identification(id) and password. The CBIE administrator must set-up a login id and password for the outside users.

Also, there are two kinds of conferences, unrestricted and restricted. Any user can automatically participate in the unrestricted conferences. To participate in a restricted conference, the user must check with the conference administrator and become a member of the conference.

In CBIE there is an CBIE administrator who is responsible for setting up new conferences and maintaining the conferences. The CBIE administrator can be compared to the conference system administrator in our conferencing system. They both are responsible for maintaining the conferences and can create new conferences. The conference administrator in CBIE would be like our major professor. They both are responsible for adding the participants to a conference. In CBIE, any participant of a conference can view any item for that conference. In the MARRS conferencing system, it is only when the paper is in final review that it is possible for all participants to view it.

2.5.2 Telecenter

Telecenter was developed at the International Institute for Applied Systems Analysis and is discussed in detail in [PEAR81]. Telecenter also runs under the UNIX operating system. Telecenter was designed to use many of the UNIX utility programs. An important aspect here is that a user can execute Telecenter's utility programs without the need to understand the details of these utility programs. This can save the user both time and effort.

Telecenter provides for a modular design. If a new tool comes along, such as a better editor or a new data base system, the implementation would be relatively quick and easy. [PEAR81] states that "UNIX's directory/file structure provides a simple database management system for implementors and users of Telecenter."

The primary objective in designing Telecenter was to facilitate research. The users of Telecenter were to be the IIASA researchers. These users can create, review, and comment on conferences. When users submit conference comments, a function is called that creates and links files in the appropriate directories. There is a directory hierarchy for users and also one for conferences. To keep track of who has seen what item, links are established between a user and the items that user has not seen. Then when the user views the item, the link is removed.

2.5.3 VMSHARE

VMSHARE was developed for Share, Inc. and is discussed in detail in [DANE81]. It was designed to overcome the problems of continuity, planning, and communication between meetings. VMSHARE is implemented on an IBM VM/370 operating system and is written in the Exec language which is a standard macro language of CMS. It is an interpreted language.

VMSHARE uses a general purpose file system and orients its system around a collection of files of various types. A file is the basic object in the system. Commands operate upon the files, and the user can display files, create files, modify files, move files, and identify subsets of files. Access to specific files can be restricted to subsets of the conference participants. Each file possesses a certain type which indicates the physical characteristics and the allowable operations upon it. Users can only append comments to accessible files, and only the owner of the file may edit and change the file.

For each conference, one participant is designated as the system administrator. The system administrator's role is to handle complaints and to add new participants to the conference. This system administrator is similar to our major professor in that they both add participants to the conference. An owner of a file in VMSHARE is similar to an author of our conferencing system in that they are the only ones who can change the file. The author is the only one who can edit and change the paper.

2.5.4 ARTHER

ARTHER was designed under the UNIX operating system and is discussed in detail in [DOCK84]. ARTHER is a computer conferencing system to manage and track articles. ARTHER utilizes the UNIX file and directory structure. Files are the basic elements and are referred to as articles. When new subjects are created the appropriate directories are created. A file with a valid list of reviewers is created. Comments are kept in a file named 'comments' under the author directory for each paper. The author can tell who made a comment. Files are copied to appropriate directories when needed. When an article is submitted to the owner of a subject, the article file is copied to the owner's directory. There are also directories for the reviewers.

The users of the system include the ARTHER administrator, the original author, the owner of a subject, and reviewers. The ARTHER administrator is like our conference system administrator. They both create the conferences and maintain them. The owner of the subject is similar to our major professor in that they both control the flow of the document. The original author is the equivalent of our author since in both cases, an author is the only one who can edit and change the document. Also, the reviewers are like our committee reviewers for they both make comments on the document. In ARTHER, the comments are only seen by the original author and the person making the comment. In our conferencing system, the comments can be targeted. There are public comments in both systems which can be accessed by all participants in the conference.

The users of the system can create, review, track, and comment on articles. ARTHER provides tracking in order to obtain the current status of an article. Comments can be created and reviewed and must be deleted by either the original author or the owner of the subject.

The commands available in ARTHER are like the UNIX shell commands. The first item must be the command verb. This can be followed by arguments also. ARTHER utilizes the electronic mail of UNIX in order to send messages to users about new subjects, when articles have been submitted, new comments, and status changes.

2.6 Conclusions

All of the existing computer teleconferencing systems discussed are file processing systems. Our computer teleconferencing system is different in that we utilize both a data base management system and files. The data base management system will allow our users to make queries and updates on the conference. They can list the available conferences, check the status of a conference, list the participants in a conference, and search using keywords. The advantages of a data base management system, as discussed previously, lie in the reduction of data duplication, data integrity, and the program/data independence. The programs go through the data base management system instead of directly accessing the data. Not all our data will be kept in the data base. File processing can be more efficient and faster and because of the variable size of some data, part of our data will be kept in files.

3. System Overview

3.1 Introduction

Computer teleconferencing systems provide interactive group-communication among individuals in different geographical locations. Computer teleconferencing systems are also self-documenting. Participants in a computer teleconferencing system have the capability of accessing, updating, and creating data. A user's capability to perform each of these operations is dependent upon the role the user assumes in the conference.

The computer teleconferencing system which we are implementing is an asynchronous teleconferencing system. The computer teleconferencing system is designed to oversee the completion of a software design project as functionally shown in figure 1. The conferencing system is to be utilized for the later phases of a software design project after it has already been committed and funded. The conferencing system can be used to oversee the completion of individual assignments of the design project.

```
                         ---------------
                        (  Conference   )
                        (    System     )
                        ( Administrator )
                         ---------------
                   /            |            \
                  /  .  .  .    |    .  .  .  \
                 /              |              \
     -------------      -------------      -------------
    (  Project    )    (  Project    )    (  Project    )
    (  Leader     )    (   Leader    )    (   Leader    )
     -------------      -------------      -------------
                         /      |      \
                        / . . . | . . . \
                       /        |        \
           ---------      ---------      ---------
          ( Project )    ( Project )    ( Project )
           ---------      ---------      ---------
                         /      |      \
                        /       |       \
                       /        |        \
          -----------     -----------     -----------
         (   Team    )   (  Review   )   ( Observers )
         (  Member   )   (  Board    )   (           )
          -----------     -----------     -----------
```

Figure 1. Hierarchical View of Participants.

Our software design has been focused toward one specific type of project. In particular, our system should help automate the process of collecting, reviewing, and presenting of Master's projects. This is functionally shown in figure 2.

```
                     ---------------
                    (  Conference   )
                    (    System     )
                    ( Administrator )
                     ---------------
                  /         |         \
                 /  . . .   |   . . .  \
                /           |           \
     ------------     ------------     ------------
    (   Major    )   (   Major    )   (   Major    )
    ( Professor  )   ( Professor  )   ( Professor  )
     ------------     ------------     ------------
                    /       |       \
                   / . . .  | . . .  \
                  /         |         \
          ---------     ---------     ---------
         ( Masters )   ( Masters )   ( Masters )
         ( Project )   ( Project )   ( Project )
          ---------     ---------     ---------
                    /       |       \
                   /        |        \
                  /         |         \
      -----------     -----------     ----------
     ( Author(s) )   ( Committee )   ( Audience )
      -----------     -----------     ----------
```

Figure 2. Hierarchy of the Master's Report Review System.

Our conferencing system will run on Kansas State University's VAX system, which is operating with Berkeley UNIX 4.2. This system will utilize both the INGRES data base management system and separate files for storing data. Our conferencing system uses the 'mail' and 'finger' UNIX utilities and also the environment variables for text editing. The 'mail' routine provides for communication among the participants and reminders to idle participants. The 'finger' routine provides the biographical information of the participants.

The remainder of this chapter describes in detail the management of the data in our conferencing system.

3.2 Directory and File Structures

The "conference system administrator" is responsible for creating, maintaining, and upgrading the conference system. To initiate the conferencing system, the conference system administrator must set up the starting (referred to as HOME) directory for the conferencing system. Once established, a source directory must be created and the system's source and make files must be read into the source directory before the 'make' program is executed.

The system will have a hierarchical directory organization, as shown in figure 3, as a result from executing the make files.

```
      Bin----------(commands, prof_file)
     /
    /
HOME--Source-------(makefile, source.c, headers.h)
    \
     |
     \
      Conference---<conf_titl.d>
                        . . .this is expanded below


                    author.d------(paper)
                   /
                  /             main.d--<item_#>
                 /             /
                 |--PLATFORM--<user_ids>--<item_#>
                /
  <conf_titl.d>--major_prof.d--(paper)
                \
                 |--COMMENTS---(user_id.user_id)
                 \
                  (usr_loc_file,stat_his_file,bb_file)
```

Figure 3. Master's Report Review System Directory and File Structure.

Under the HOME directory will be three directories: the Bin directory, the Source directory, and the Conference directory. These directories and their contents are explained in the following paragraphs.

Under the Bin directory, the commands and the 'prof_file' are stored. Access to the commands will be provided to a conference participant in a menu. The participant can select a function displayed in the menu. Prof_file is a file that contains the valid list of login IDs for the faculty members; each of these faculty members is capable of assuming a role of major professor or committee member. The format of the file is the faculty member's login ID followed by a newline; then the next faculty member's login ID followed by a newline and so on for all the faculty members. A user who wants to create a conference or become a committee member must have their login ID in this file. The conference system administrator is responsible for maintaining this file.

Under the Source directory, the makefile is stored. The makefile will call an initialization routine that is used to set-up the directory structure. Also under the Source directory is the source code for the conferencing system. The routines in the conferencing system are the following :

- marrs.c (the main command parser routine)
- several subprograms to support the marrs.c
- library ingres
- library cursor
- library conference

The global definitions and file structures used by the source code are stored in a header file. Some definitions assumed for our conferencing system are the following :

- maximum number of papers within one conference is ten,
- maximum number of participants in a conference is thirty,
- maximum number of status changes for a paper is twenty,
- maximum length of a paper's filename is fifteen characters,
- there are four possible participant roles in a conference which are :

Role	Defined as
major professor	1
committee member	2
author	3
audience member	4

- there are six possible status states for a paper. These are :

State	Defined as
not submitted	1
paper submitted	2
being rewritten	3
committee review	4
final review	5
to be published	6

Under the Conference directory will be one or more occurrences of the 'conf_titl' directory. The 'conf_titl' directory name is to be an abbreviated name of a master's project. The directory name can consists of letters, numbers, the underscore or the dot. It can be up to but no more than fourteen characters in length. It must begin with a letter or number.

Under each conf_titl directory there is one 'author' directory for each author in the conference up to a maximum of ten. Also in the conf_titl directory, there is the 'major_prof', 'COMMENTS', and 'PLATFORM' directories and the 'usr_loc_file', 'stats_his_file' and 'bb_file' files. The 'creating a conference' routine allows the major professor to input the information needed to set-up these files and directories.

The 'author' directory will contain the paper under consideration in this conference. The name of 'author' directory will be the same as the author's login ID. The 'major_prof' directory will contain the master copy of the paper for review. The directory name will be the major professor's login ID. The paper under the author's directory is the only version of the paper that can be changed. The major professor and committee members can only review their version of the paper and make comments to it. They cannot make changes to the paper. The author is the only one who can make changes to the paper. The author can submit the paper only when it is in a status state of 'Being re-written' (see Appendix 1).

The 'PLATFORM' directory will be used for the platform discussion of a paper. Under this directory will be a main directory and one 'user_id' directory for each conference participant. The 'user_id' directories correspond to individual conference participants' login IDs. Under the main directory, all the platform discussion comments are kept. A link to the comments in the main directory is created to each 'user_id' directory. When a participant views a comment, the link will be removed. The participant can still go to the main directory to re-read any comment. The link is used to inform participants of comments that they have not read. Each comment will have a comment number associated with it. Participants can respond to a comment and associate this new comment to a previous one by using the comment number. This 'PLATFORM' directory provides a participant with the ability to discuss side issues.

The 'COMMENTS' directory provides an area to store comments while a participant reviews a paper. Participants can target comments to other participants. When reviewing a paper, a note reminder feature informs the participant that a comment is associated with this line. Under the 'COMMENTS' directory, several files are stored, each of which contains one conference participant's comments on the paper being presented. A comment file's name is composed of the commenter's login ID followed by the login ID of the participant to whom the comment is directed. This is then followed by the author number (0-9). These items are separated by a dot. For example, if participant 'janning' wants to comment to 'stachowi' on author number 3's paper, the file name would be 'janning.stachowi.3'. This file contains the line number in the paper, a flag to indicate the status of the comment, the number of bytes in the comment and the comment. A comment is followed by the next comment. The status of a comment has the following possibilities :

in progress	- currently working on comment
available	- comment has been submitted
read	- comment has been read

While the comment is 'in progress', it is not available to whom it is targeted. The commenter can review and change the comment. Once the comment has been changed to 'available', it cannot be changed by the commenter and is now available to whom it was targeted. When the comment is changed to 'read', the commenter can see that the comment has been read. When the author submits a paper to the major professor, the comment files are cleared out.

The 'usr_loc_file' is a file that contains the user's login ID, the user's role in the conference, the participant number and the user's location in each author's paper (can be up to ten papers). The participant number is only of interest if the participant is an author or a committee member. The possible user roles are major professor, committee member, author and audience member. Also, preserving a user's location in a paper allows a user to interrupt one's reading and then later to automatically resume at that same point. This file is used by the time-keeper routine to make reminders to idle participants so that a paper does not stagnate in the system. This file is also used to list the participants in a conference and their roles. This file is updated when new participants are included in the conference and also when participants review a paper.

The 'stats_his_file' is a file that contains the author's login ID, the paper filename and a history of the status changes. This history contains the status state and date associated with each status change. In this way, a history of the status changes for a paper will be created. Possible status states are the following :

1. Paper not submitted,

2. Paper submitted to Major Professor -- awaiting new status,

3. Being rewritten,

4. Ready for committee review,

5. Ready for final review,

6. Paper ready to be published.

The initial status for any paper is unsubmitted. The state diagram covering the changes in the status of a paper, and its description are in Appendix 1. Whenever the status of the paper is changed, the stat_his_file will be updated. There is a validation routine to check that the status state is allowed. This file is also used by the time-keeper routine.

The 'bb_file' is a file that contains the bulletin board messages. There is one 'bb_file' per conference. The bulletin board provides a participant with the ability to broadcast a message to other conference participants. 'Bb_file' is a flat-file and all participants can add, read and delete messages.

Not all of the data in our conferencing system will be stored in files. The INGRES data base management system will also store part of our data.

3.3 INGRES Data Base Management System

The INGRES data base management system is a relational data base management system. It runs under the UNIX operating system and is primarily coded in the 'C' programming language. One reason for using the INGRES data base management system is because it is an available tool. The INGRES commands can be embedded in the 'C' programming language. Our conferencing system is coded in 'C' and the UNIX shell. An advantage of using the INGRES data base management system for our conferencing system is the relations provided by INGRES. INGRES is a relational data base system and our conferencing system has relations between items that the participants will query.

Our conferencing system will use the INGRES data base management system to store information about conferences. Included in this information are such things as a list of the conferences available and the status of each conference.

When the conference system administrator creates a conference, an entry will be made into the data base. This entry will indicate the conference name and the status of the conference. Possible status states for conferences are open and closed. An open conference is currently in progress, while a closed conference is finished.

When a conference is finished, the conference system administrator will be responsible for archiving the finished report along with the platform discussion. These files will not be kept in the data base. They are stored under the conf_titl directory once the conference is finished. An entry into the INGRES data base will be made that indicates the author name, the conference title, date of approval for the report, the major professor and the topics for the report.

The INGRES data base will contain a conference relation. This relation will contain the conference title, major professor, and conference status.

3.4 User Queries

The participants of a conferencing system want to access, update, and create data. They also want to make queries on the data. Keywords will also be used for data base queries. The participants can look for information related to certain keywords. The keywords include the titles of the papers and a list of topics that are presented in the paper. Some of the query/update functions that a participant in our conferencing system can perform are the following :

1. One can list all the conferences available. A selection on the conferences can be performed which is based upon the major

professor, author or keywords.

2. One can find out the current status of a conference. If authorized, one can change the status of a conference.

3. If authorized, one can add/delete conferences.

4. One can list all the participants in a conference. A selection on the participants can also be performed based upon the role of a participant. For example, one could list all the committee members of a particular conference.

5. If authorized, one can add/delete participants.

6. One can list the valid faculty members.

7. If authorized, one can add/delete faculty members.

8. One can list a participant's biography. The biography will be obtained from the password file.

9. One can list a final paper and the platform discussion.

10. If authorized, one can add/delete a final paper and the platform discussion.

3.5 Conclusions

Our conferencing system uses a mixture of a data base management system and file processing system. The data base management system allows use to be made of INGRES' query/update facilities so that the conferencing system does not have to re-implement them. The use of a data base management system also allows the conferencing system to be more portable than if it just consisted of a file processing system.

The file processing system portion of the conferencing system easily allows communication-oriented activities. The file processing system can use 'mail' to provide notification of any changes in a conference. The file processing portion of the conferencing system can be used during the active stages of a conference when a user needs tools, such as an editor and mail. Thus, in the active stages, a file processing system is more suitable than a data base management system for conferencing activities.

The data used for user queries and tracking in our conferencing system are kept in the data base. All of this data slowly or seldomly changes. The platform discussion and the final paper will be archived when a conference is finished. Files are to be used for the user's location in a paper, the status history of a paper and the valid professor list.

4. Conclusions

Computer teleconferencing systems provide many advantages to their users. Not only is there the money savings associated with the reduction in travel, but the computer conference systems are more efficient than face-to-face conferences. The users have recorded minutes of the conference in the commenters' own words. Computer conferencing systems also provide for interactive communication among a group of people. Computer conferencing systems are available to its participants at their convenience, thus they can 'attend' the conference according to their own time schedules.

Our computer teleconferencing system attempts to help alleviate some of the problems associated with a Master's project. The major professor can monitor the paper and the committee members can review and comment on the paper with the aid of a computer. This would be a great help to people who cannot attend normal university sessions because of work. Also, it would be more convenient to work through the computer than tracking down the major professor and committee members.

This paper examined the relationship of data bases to computer teleconferencing systems. Data base management systems have advantages over file processing systems with respect to reduction of data duplication, data integrity and program/data independence. Other advantages of data base management systems are that more information is available from a given amount of data and data base utility programs are provided.

The similarities/differences between data base management systems and computer teleconferencing systems in the areas of communication, ownership of data, views of the data and user roles were also examined.

Our computer teleconferencing system utilizes both a data base management system and files. Doing this cuts down on the implementation effort by utilizing the existing INGRES query/update facilities. The data base management system allows the users to make queries and updates on the conference. The file processing can be more efficient and faster. The file processing portion of our conferencing system allows communication-oriented activities. The file processing portion is used during the active stages of a conference.

An issue not explored is the possible impact of changing the conferencing system to run using a different data base management system. This could arise if the conferencing system were ported to a system not having the INGRES data base management system or if another data base management system was bought to replace INGRES.

Security is an issue that could be expanded in our computer conferencing system. Our design has assumed a 'user-friendly' environment. There is protection provided on the permissions associated with a particular role in the conference. Also, there are validation routines to insure that a requested state change is allowed from the current state according to the state diagram in Appendix 1.

Our implementation has not addressed the issue of networking. The use of the UNIX 'cu' and 'uucp' commands would make our computer conferencing system available to others.

REFERENCES

[CROS83] Cross, Thomas B., "THE GRAPEFRUIT DIET", Journal of Micrographics, April, 1983, pp. 14-18.

[CROS84] Cross, Thomas B., "COMPUTER CONFERENCING", Computerworld on Communications, August 1, 1984, pp. 37-39.

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

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

[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.

[JANN86] Janning, Ronald M., "Features of the MARRS Computer Conferencing System", Master's Report, Department of Computer Science, Kansas State University, 1986.

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

[KROE83] Kroenke, David, "DATABASE PROCESSING : Fundamentals, Design, Implementation", Second Edition, Science Research Associates, Inc., Chicago, Henley-on-Thames, Sydney, Toronto, 1983.

[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.

[STRO80] Strom, Bernard Ivan, "A MULTI-COPY STRUCTURED DATABASE COMPUTER CONFERENCING SYSTEM", PhD Dissertation, Columbia University, 1980.

[STRO82] Strom, B. Ivan, "COMPUTER CONFERENCING - PAST, PRESENT, AND FUTURE", Office Information Systems, Naffah, N. (editor). INRIA/North-Holland Publishing Company, 1982.

[TSIC81] Tsichritzis, D., "INTEGRATING DATA BASE AND MESSAGE SYSTEMS", IEEE Transactions on Communications, 1981.

[TSIC82] Tsichritzis, Dennis, Rabitti, Fausto A., Gibbs, Simon, Nierstrasz, Oscar, Hogg, John, "A SYSTEM FOR MANAGING STRUCTURED MESSAGES", IEEE Transactions on Communications, Vol. COM-30, No. 1, January, 1982.

[ULLM82] Ullman, Jeffrey D., "Principles of DATABASE SYSTEMS", Second Edition, Computer Science Press, Inc., Rockville, Maryland, 20850, 1982.

[YAO84] Yao, S. Bing, Hevner, Alan R., Shi, Zhongzhi, Luo, Dawei, "FORMANAGER : An Office Forms Management System", ACM Transactions on Office Information Systems, Vol. 2, No. 3, July, 1984, pp. 235-262.

Appendix 1

State Diagram

Appendix 1 - State Diagram

```
                   --------------------
                   |    Paper not     |
                   |    submitted     |
                   |       (1)        |
                   --------------------
                           |
                           v
                   --------------------
                   | Paper submitted  |
                   |       to     (2) |-----------
                   | Major Professor  |------    |
                   --------------------     |    |
                      ^           |         |    |
                      |           |         |    |
                      v           v         |    |
   -----------------     -----------------  |    |
   |     Being     |     |   Committee   |  |    |
   |   rewritten   |<----|    review     |  |    |
   |      (3)      |     |      (4)      |  |    |
   -----------------     -----------------  |    |
                      ^          |          |    |
                      |          v          |    |
                   --------------------     |    |
                   |      Final       |<-----    |
                   |      review      |          |
                   |       (5)        |          |
                   --------------------          |
                           |                     |
                           |                     |
                           v                     |
                   --------------------          |
                   |   Paper to be    |          |
                   |    published     |<----------
                   |       (6)        |
                   --------------------
```

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 of 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 in the conference.

Appendix 2

Application Code

```
creatdb -q marrsd

sysmod marrsd
```

```
equel add_conf.q    ;  cc -c add_conf.c
equel add_paper.q   ;  cc -c add_paper.c
equel add_title.q   ;  cc -c add_title.c
equel add_topic.q   ;  cc -c add_topic.c
equel l_confau.q    ;  cc -c l_confau.c
equel l_confmp.q    ;  cc -c l_confmp.c
equel l_confpa.q    ;  cc -c l_confpa.c
equel l_confst.q    ;  cc -c l_confst.c
equel l_ptitle.q    ;  cc -c l_ptitle.c
equel upd_paper.q   ;  cc -c upd_paper.c
equel upd_status.q  ;  cc -c upd_status.c

ar rv libing add_conf.o
ar rv libing add_paper.o
ar rv libing add_title.o
ar rv libing add_topic.o
ar rv libing l_confau.o
ar rv libing l_confmp.o
ar rv libing l_confpa.o
ar rv libing l_confst.o
ar rv libing l_ptitle.o
ar rv libing upd_paper.o
ar rv libing upd_status.o

ranlib libing

rm *.o
```

```
equel creat_rl.q
cc creat_rl.c -lq
mv a.out /usrb/att/marrs/Bin/creat_rl
```

```
cc platform.c libmarrs
mv a.out /usrb/att/marrs/Bin/platform
```

```
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

ranlib libmarrs

rm *.o
```

```
cc query.c libing -lq
mv a.out /usrb/att/marrs/Bin/query
```

```
#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)
```

```
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 */
```

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


/************************************************
 *                                              *
 * add_item is a routine to add an item to the  *
 * platform directory. This routine creates the *
 * item file under main.d and links it to all   *
 * platform users in the conference.            *
 *                                              *
 ************************************************/

add_item(itptr, itpred, fromid)
char *itptr;
short itpred, fromid;
{
    DIR *opendir();
    DIR *dirp;
    struct direct *readdir();
    struct direct *dp;
    struct stat buf;
    extern char conf_path[120];
    extern char login_name[MAX_LOGIN_SIZE];
    extern int errno;
    char path[120];
    char tmp_path[120];
    char tp2_path[120];
    char pla_path[120];
    char usr_dir[FILENM_MAX];
    char lk_path[120];
    char *index();
    char new_item;
    struct item_info add_list[MAX_ITEMS], *alptr;
    int rc, fdes, fdes1, i, item_num;

    /* Check if item already has a predecessor */
    if(valid_item(itpred) == TRUE)
    {
        /* Read item_list_file into add_list */
        sprintf(tmp_path,"%s/%s/%s/%s",conf_path,PLATFORM,PLAT_MAIN,
                ITEM_FILE);
        fdes = open(tmp_path, O_RDWR | O_CREAT, 0666);

        if((item_num = read(fdes, add_list, sizeof (add_list))) < 0)
```

```
{
  printf("\nRead error on ITEM_FILE");
  close(fdes);
  return(FALSE);
}

new_item = 1;
if(item_num != 0)
{
  item_num = item_num/ITEM_LENGTH;
  for(alptr = &add_list[0]; alptr < &add_list[item_num];
        alptr++)
    new_item++;
}
else
{
  alptr = &add_list[0];
}

/* Create item under main.d. */
sprintf(tmp_path,"%s/%s/%s/%d",conf_path,PLATFORM,PLAT_MAIN,
        new_item);

fdes1 = open(tmp_path, O_CREAT | O_WRONLY, 0666);

i = strlen(itptr);

/* Write new item into new file under main.d */
if(write(fdes1, itptr, i) < 0)
{
  close(fdes1);
  printf("\nWrite on new item file failed. errno = %d",errno);
  sleep(1);
  return(FALSE);
}

close(fdes1);

/* Want to link new item to all user directories *
 * under PLATFORM except for the user creating   *
 * the new item.                                 */

sprintf(usr_dir,"%s%s", login_name, DIR_ENDING);
sprintf(pla_path,"%s/%s",conf_path,PLATFORM);
sprintf(tp2_path,"%s/.",pla_path);

if((dirp = opendir(tp2_path)) == NULL)
{
  printf("\nopendir failed");
  sleep(1);
  return(FALSE);
```

```
}

/* Find all user directories except user creating item */
for(dp = readdir(dirp); dp != NULL; dp = readdir(dirp))
{
   sprintf(path, "%s/%s", pla_path, dp->d_name);
   stat(path, &buf);

   if((buf.st_mode & S_IFDIR) &&
    (index(dp->d_name, '.') != &(dp->d_name[0])) &&
    (strcmp(dp->d_name, PLAT_MAIN)) &&
    (strcmp(dp->d_name, usr_dir)))
   {
      /* Link new item */
      sprintf(lk_path,"%s/%d",path,new_item);
      rc = link(tmp_path, lk_path);
      if(rc != 0)
      {
        printf("Link failed rc = %d  errno = %d\n",rc,errno);
        printf("tmp_path = %s",tmp_path);
        printf("\nlkpath = %s\n",lk_path);
        sleep(2);
        continue;
      }
   }
}

closedir(dirp);

/* Update item_list_file */
alptr->name = new_item;
alptr->pred = itpred;
alptr->succ = NULL;
alptr->from_who = fromid;

lseek(fdes, 0, 0);

/* If item is in response to another item, *
 * then have to update successor entry for *
 * old item in the item_list_file.         */

if(itpred != NULL)
{
  i = 0;
  for(alptr = &add_list[0]; alptr < &add_list[item_num];
        alptr++)
  {
    if(alptr->name == itpred)
    {
      alptr->succ = new_item;
      break;
```

```
          }
        }

        if(alptr == &add_list[item_num])
        {
          printf("\nRead error on item file after lseek");
          close(fdes);
          sleep(1);
          return(FALSE);
        }
      }

      /* Write updated item_list_file */
      item_num = new_item * ITEM_LENGTH;
      write(fdes, add_list, item_num);
      close(fdes);
      return(TRUE);
    }
    return(FALSE);
}
```

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

/***********************************************
 *                                             *
 * list_pitem is a routine to list the contents *
 * of a specified user platform directory.     *
 * It is used to list the open items for a user *
 * in the conference.                          *
 *                                             *
 ***********************************************/

list_pitem(upath)
char *upath;
{
    DIR *opendir();
    DIR *dirp;
    struct direct *readdir();
    struct direct *dp;
    struct stat buf;
    char path[120];
    char tmp_path[120];
    char *index();
    int count;

    count = 0;
    sprintf(tmp_path,"%s/.",upath);
    dirp = opendir(tmp_path);

    /* if open fails then user not part of this conference  */

    /* List all files under user's directory except temp,   *
     * item_list_file, and those beginning with a '.'       */

    for(dp = readdir(dirp); dp != NULL; dp = readdir(dirp))
    {
        sprintf(path, "%s/%s", upath, dp->d_name);
        stat(path, &buf);
        if((buf.st_mode & S_IFREG) &&
            (index(dp->d_name, '.') != &(dp->d_name[0])) &&
            (strcmp(dp->d_name, TEMP_FILE)) &&
            (strcmp(dp->d_name, ITEM_FILE)))
        {
            count++;
            printf(" %s  ",dp->d_name);
            if(!( count % 8 ))
                printf("\n      ");
        }

    }

    /* If no files are found, then there are no new items  */

    if(count == 0)
        printf(" none ");

    closedir(dirp);
    return(TRUE);

}
```

```
#include <stdio.h>
#include <sys/types.h>          /* for stat call */
#include <sys/file.h>
#include <sys/dir.h>
#include <sys/errno.h>          /* for stat call */
#include "/usrb/att/marrs/Source/stat.h"       /* for stat call */
#include "/usrb/att/marrs/Source/conf.h"
#include <strings.h>

int errno;
char conf_path[120];                    /* path name to conference      */
char conf_name[FILENM_MAX];             /* conference name              */
char ed_command[SYSTEM_CALL_SIZE];      /* holds system call for editor */
char more_command[SYSTEM_CALL_SIZE];    /* holds system call for "more" */
char *pptr, Pitem_path[120];            /* path name to Platform items  */
char *tptr, Titem_path[120];            /* temp path to Platform items  */

char *str, string[20];
char keep_going;
short item_pred, item_from;
char *getenv(), *editor, conf_editor[MAX_ED_PATH];
char login_name[MAX_LOGIN_SIZE];
char item_buf[800];
int(fdes);
int i;

/****************************************
 *                                      *
 * platform is the main routine for     *
 * the platform discussion.  It prints  *
 * the conferences available and has    *
 * the user request a conference.       *
 * The user must be a member of the     *
 * conference and at least one paper    *
 * in the conference must be in final   *
 * review.  Any new items for the user  *
 * are listed under 'Open items:".      *
 *                                      *
 ****************************************/

main(argc, argv)
int argc;
char *argv[];
{

struct usr_loc_file users_file[CON_MAX], *uptr;
struct stat_his_file stat_file[PAPER_MAX], *paptr;
char valid;
int rc, num_users, num_papers;
```

```
pptr = Pitem_path;
tptr = Titem_path;
keep_going = TRUE;

/* First get the user's login name so it can be checked out */
strcpy(login_name, getlogin());

CLEAR

/* List all conferences that the user is in */
switch(usr_conf(login_name, ANY_ROLE, conf_name))
{
    case -1:          /* No conferences found for user */
      return(-1);
      break;

    case 0:           /* User wanted to quit */
      return(-1);
      break;

    case 1:           /* OK, got conference name */
      break;

    default:          /* Bad return from usr_conf */
      printf("\nBad return from usr_conf\n");
      return(-1);
      break;
}


while(keep_going)
{
    CLEAR
    valid = 1;
    printf("\n\nConference : %s",conf_name);

    /* User must be member of the conference */
    sprintf(conf_path,"%s%s%s",CONF_HOME,conf_name,DIR_ENDING);
    sprintf(Pitem_path,"%s/%s",conf_path,USR_FILE);

    fdes = open(Pitem_path, O_RDONLY, 0);
    if((num_users = read(fdes,users_file,sizeof(users_file))) <= 0)
    {
      printf("\nRead error on usr_loc_file for conference %s",
                    conf_name);
      close(fdes);
      sleep(1);
      break;
    }

    close(fdes);
```

```
num_users = num_users/(sizeof (struct usr_loc_file));

/* Find user's participant number */
for(uptr = &users_file[0]; uptr < &users_file[num_users];
        uptr++)
{
  if(!strcmp(login_name, uptr->usr_id))
  {
    item_from = uptr->part_num;
    break;
  }
}

if(uptr == &users_file[num_users])
{
  printf("\n\nSorry, you are not a member of conference %s\n",
        conf_name);
  sleep(2);
  break;
}

/* At least one paper in this conference must be
   in final review */

sprintf(Pitem_path,"%s/%s",conf_path,STAT_FILE);
fdes = open(Pitem_path, O_RDONLY, 0);
if((num_papers = read(fdes, stat_file, sizeof (stat_file)))
        <= 0)
{
  printf("\nCannot read stat_his_file for conference %s",
        conf_name);
  close(fdes);
  sleep(1);
  break;
}

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

for(paptr = &stat_file[0]; paptr < &stat_file[num_papers];
        paptr++)
{
  for(i = 0; (i < STAT_MAX) &&
        (paptr->status[i].paper_stat != 0); i++);

  i--;
  if((paptr->status[i].paper_stat) == FIN_REVU)
    break;
}

if(paptr == &stat_file[num_papers])
```

```
      {
        printf("\n\nSorry, there are no papers in final review");
        printf(" for conference %s\n",conf_name);
        sleep(2);
        break;
      }

      /* List open items for this user */
      sprintf(Pitem_path,"%s/%s/%s%s",conf_path,PLATFORM,login_name,
              DIR_ENDING);

      printf("\nOpen items : ");

      list_pitem(pptr);

      while(valid == 1)
      {
         printf("\n\nEnter Valid Option ( t, a, q, ? ) : ");
         gets(string);
         switch(string[0])
         {
              case 't' :
                 valid = 0;
                 type_opt();
                 sleep(2);
                 break;

              case 'a' :
                 valid = 0;
                 add_opt();
                 break;

              case 'q' :
                 valid = 0;
                 keep_going = FALSE;
                 break;

              case '?' :
                 valid = 0;
                 help_opt();
                 break;

              default :
                 printf("\n** Invalid option ");
                 break;

           }  /* end of switch */

         }  /* end of while */

  }  /* end of while */
```

```
}  /* end of main  */


/**************************************************
 *                                                *
 * type_opt calls type_item providing there are *
 * items in the platform discussion for this     *
 * conference.                                    *
 *                                                *
 **************************************************/

type_opt()
{
  /* If there are platform items */
  if(check_main() == TRUE)
  {
    /* List all items in the platform */
    sprintf(Titem_path,"%s/%s/%s",conf_path,PLATFORM,PLAT_MAIN);
    printf("\n\nItems available:\n");
    list_pitem(Titem_path);
    type_item();
  }

  else
  {
    printf("\n\nSorry, there are no items in the platform ");
    printf("\ndiscussion for conference %s\n",conf_name);
    sleep(2);
  }
}


/**************************************
 *                                    *
 * add_opt allows the user to add an *
 * item to the platform discussion.  *
 * This new item can be in response  *
 * to another item.                  *
 *                                    *
 **************************************/

add_opt()
{
   struct stat buf;

   item_pred = NULL;

   /* If there are items in the platform, *
    * check if this is in response to one */

   if(check_main() == TRUE)
```

```
{
     printf("\n\nResponse to item # (0 for none) : ");
     gets(string);
     if(string[0] == '0')
     {
        item_pred = NULL;
     }

     else
     {
        item_pred = atoi(string);
     }

}

/* Check to see if item # already has a response to it */
if(valid_item(item_pred) == TRUE)
{

  /* Create a temporary file to work in */
  sprintf(Titem_path,"%s/%s/%s/%s",conf_path,PLATFORM,PLAT_MAIN,
            TEMP_FILE);

  if((fdes = open(Titem_path,O_CREAT | O_TRUNC, 0666)) < 0)
  {
       printf("\nOpen failed on temp_path %s",Titem_path);
       close(fdes);
       return(FALSE);
  }

  close(fdes);

  /* Edit the temporary file */
  if((editor = getenv("EDITOR")) == NULL)
  {
        strcpy( conf_editor, DEFAULT_EDITOR );
        printf("\nYou are in the 'ed' editor\n\n");
  }
  else
  {
        strncpy( conf_editor, editor, MAX_ED_PATH - 1);
  }

  sprintf(ed_command,"%s %s", conf_editor, Titem_path);
  system(ed_command);

  /* Check to see if file is not empty */
  stat(Titem_path, &buf);
  if(buf.st_size > 0)
  {
    /* Read file into item_buf */
```

```
        fdes = open(Titem_path, O_RDONLY, 0);
        i = read(fdes, item_buf, sizeof (item_buf));
        item_buf[i] = NULL;
        close(fdes);
        add_item(item_buf,item_pred, item_from);
      }
      }
}


/******************************************
 *                                        *
 * help_opt displays a help menu to the   *
 * user that describes each option.       *
 *                                        *
 ******************************************/

help_opt()
{
    CLEAR
    printf("\n\n     Valid options are:\n\n");
    printf("                a - add an item to the ");
    printf("platform discussion\n");
    printf("                t - type an item in the ");
    printf("platform discussion\n");
    printf("                ? - this display\n");
    printf("                q - leave the platform discussion\n");
    printf("\n\n\n     Hit return to continue ");
    gets(string);
}


/*************************************
 *                                   *
 * check_main checks to see if there *
 * are any items in the platform.    *
 *                                   *
 *************************************/

check_main()
{
  DIR *opendir();
  DIR *dirp;
  struct direct *readdir();
  struct direct *dp;
  struct stat buf;
  char path[120];
  char tmp_path[120];
  char main_path[120];
  char *index();
  int count;
```

```
   count = 0;
   sprintf(main_path,"%s/%s/%s/.",conf_path,PLATFORM,PLAT_MAIN);
   sprintf(tmp_path,"%s/.",main_path);
   dirp = opendir(tmp_path);

   /* Check for any files in main.d that do not begin with  *
    * a '.'  and is not temp or item_list_file.             */

   for(dp = readdir(dirp); ((dp != NULL) && (count == 0));
         dp = readdir(dirp))
   {
      sprintf(path,"%s/%s", main_path, dp->d_name);
      stat(path, &buf);
      if((buf.st_mode & S_IFREG) &&
         (index(dp->d_name, '.') != &(dp->d_name[0])) &&
         (strcmp(dp->d_name, TEMP_FILE)) &&
         (strcmp(dp->d_name, ITEM_FILE)))
      {
         count++;
      }
   }
   closedir(dirp);
   if(count != 0)
   {
      return(TRUE);
   }

   else
   {
      return(FALSE);
   }
}
```

```
#include <stdio.h>
#include <sys/types.h>          /* for stat call */
#include <sys/file.h>
#include <sys/dir.h>
#include <sys/errno.h>          /* for stat call */
#include "/usrb/att/marrs/Source/stat.h"
#include "/usrb/att/marrs/Source/conf.h"
#include <strings.h>

int errno;

/*****************************************
 *                                       *
 * query is the main routine for keyword *
 * searching. The user can list all the  *
 * conferences and paper file names that *
 * are in the data base.  Keywords are   *
 * entered when the paper is in the      *
 * state of ready to be published.       *
 *                                       *
 *****************************************/

main(argc, argv)
int argc;
char *argv[];
{
   char look_topic;
   char *str, string[20];
   char conf_name[15];
   char paper_name[14];
   int i;

   look_topic = TRUE;

   while(look_topic)
   {
     CLEAR
     printf("\nEnter topic (? for help, q to quit) : ");
     gets(string);

     if(string[0] == '?')
     {
        CLEAR
        printf("\n\nThe keyword search provides the capability to \n");
        printf("list the conference names and paper file names \n");
        printf("associated with a requested topic.\n");
        printf("\nThe topic can be at most 20 characters.\n");
        printf("\n\nHit return to continue ");
        gets(string);
        continue;
     }
```

```
      if(string[0] == 'q')  /* User wants to quit */
         look_topic = FALSE;

      else
      {
         i = strlen(string);
         if(i > 0)   /* topic has been entered */
         {
           l_confpa(string);  /* List conferences on this topic */
           printf("\n\nEnter paper file to display ");
           printf("paper title (0 for none) : ");
           gets(string);
           strcpy(paper_name,string);

           i = strlen(string);
           if((string[0] == '0') || (i == 0))
              continue;

           while(TRUE)
           {
              printf("Enter conference name (q to quit) : ");
              gets(string);

              if(string[0] == 'q')
                break;

              i = strlen(string);
              if(i > 0)          /* List paper title */
              {
                 strcpy(conf_name,string);
                 l_ptitle(paper_name,conf_name);
                 break;
              }
           }
         } /* end of if */
      }  /* end of else */
   }  /* end of while */
}  /* end of main */
```

```
#include <stdio.h>
#include <sys/file.h>
#include <sys/types.h>          /* for stat call */
#include <sys/errno.h>          /* for stat call */
#include "/usrb/att/marrs/Source/stat.h"        /* for stat call */
#include "/usrb/att/marrs/Source/conf.h"
#include <strings.h>

/****************************************
 *                                      *
 * type_item prints out an item in the *
 * platform discussion.  If the user is*
 * reading it for the first time, then *
 * the file is unlinked.               *
 *                                      *
 ****************************************/

type_item()
{
   extern int errno;
   extern char conf_path[120];          /* path name to conference */
   extern *pptr, Pitem_path[120];
   struct stat buf;
   struct item_info item_list[MAX_ITEMS], *item_ptr;
   struct usr_loc_file users_list[CON_MAX], *uptr;
   char *mptr, msg_path[120];
   char msg_buf[800];
   char *index;
   char *str, string[20];
   char *iptr, item_path[120];
   char usr_part, from_login[MAX_LOGIN_SIZE];
   int item_name, from_user, num_user;
   int item_size, item_num;
   int fdes, i, rc, read_item;

   str = string;
   iptr = item_path;
   mptr = msg_path;
   read_item = 1;

   while(TRUE)
   {
     printf("\n\nEnter Item Number (q to quit) : ");
     gets(str);

     if(string[0] == 'q')
       break;

     item_size = strlen(str);
     if(item_size == 0)
        continue;
```

```
    /* Check for item under the user directory */
    item_name = atoi(str);
    sprintf(item_path,"%s/%d",Pitem_path,item_name);
    rc = stat(item_path, &buf);

    /* If item not under user directory, check under main.d */
    if(rc != 0)
    {
       sprintf(item_path,"%s/%s/%s/%d",conf_path,PLATFORM,PLAT_MAIN,
               item_name);

       rc = stat(item_path, &buf);

       /* If item not under main.d, then item does not exist */
       if(rc != 0)
       {
          printf("Item does not exist\n");
          continue;
       }

       /* If item under main.d, then don't want to unlink file */
       read_item = 0;
    }

   sprintf(msg_path,"%s",item_path);
   sprintf(item_path,"%s/%s/%s/%s",conf_path,PLATFORM,PLAT_MAIN,
       ITEM_FILE);

   fdes = open(item_path, O_RDONLY, 0);

   /* read item_list_file */
   if((item_num = read(fdes, item_list, sizeof(item_list))) <= 0)
   {
     printf("\nCannot read item file in type option");
     close(fdes);
     sleep(1);
rn(FALSE);
   }

   close(fdes);
   item_num = item_num/ITEM_LENGTH;

   for(item_ptr = &item_list[0]; item_ptr < &item_list[item_num];
               item_ptr++)
   {
     if((item_ptr->name) == item_name)
     {
       /* Get user participant number */
       from_user = item_ptr->from_who;
       break;
     }
}

if(item_ptr == &item_list[item_num])
{
  printf("\nCannot find item");
  sleep(1);
  return(FALSE);
}

/* Find login name */
sprintf(item_path, "%s/%s",conf_path,USR_FILE);
fdes = open(item_path, O_RDONLY, 0);
if((num_user = read(fdes, users_list, sizeof (users_list)))
            <= 0 )
{
  printf("\nCannot read user list");
  close(fdes);
  sleep(1);
  return(FALSE);
}

close(fdes);
num_user = num_user/(sizeof (struct usr_loc_file));

for(uptr = &users_list[0]; uptr < &users_list[num_user]; uptr++)
{
  if(uptr->part_num == from_user)
  {
    strcpy(from_login,uptr->usr_id);
    break;
  }
}

if(uptr == &users_list[num_user])
{
  printf("\nCould not find participant");
  sleep(1);
  return(FALSE);
}

/* Print header for item */
printf("\n\nItem %d: ( %s )\n",item_name,from_login);
if((item_ptr->pred) != NULL)
  printf("    response to item: #%d",item_ptr->pred);

if((item_ptr->succ) != NULL)
  printf("    followed by item: #%d",item_ptr->succ);

/* Read item file into msg_path */
fdes = open(msg_path, O_RDONLY, 0);
rc = read(fdes, msg_buf, sizeof(msg_buf));
```

```
    if(rc <= 0)
    {
      printf("\nRead error on msg_buf");
      close(fdes);
      sleep(1);
      return(FALSE);
    }

    /* Add NULL to end of item, then print item */
    msg_buf[rc] = NULL;
    printf("\n\n%s",msg_buf);
    close(fdes);

    /* If item under user directory, unlink item */
    if(read_item == 1)
    {
        if((rc = unlink(mptr)) < 0)
        {
          printf("unlink failed rc = %d  errno = %d\n",rc,errno);
        }
    }

    printf("\n\n\nEnd of item.   Hit return to continue  ");
    gets(str);
    return(TRUE);

  }
}
```

```
#include <stdio.h>
#include <sys/file.h>
#include <sys/types.h>          /* for stat call */
#include <sys/errno.h>          /* for stat call */
#include "/usrb/att/marrs/Source/stat.h"        /* for stat call */
#include "/usrb/att/marrs/Source/conf.h"
#include <strings.h>

/********************************
 *                              *
 * valid_item checks to see if  *
 * the item being requested     *
 * already has a predeccessor.  *
 *                              *
 ********************************/

valid_item(value)
short value;
{

   int errno;
   extern char conf_path[120];          /* path name to conference       */
   struct item_info vadd_list[MAX_ITEMS], *vlptr;
   struct stat buf;
   char path[120];
   char info_file[120];
   int fdes, i, num_items;

   /* If item is in response to another item */
   if(value != NULL)
   {
     sprintf(path,"%s/%s/%s/%d",conf_path,PLATFORM,PLAT_MAIN,value);

     /* Check to see if item exists */
     if(stat(path, &buf) < 0)
     {
        printf("\nItem #%d does not exist",value);
        sleep(1);
        return(FALSE);
     }

     sprintf(info_file,"%s/%s/%s/%s",conf_path,PLATFORM,PLAT_MAIN,
        ITEM_FILE);

     if((fdes = open(info_file, O_RDONLY, 0)) < 0)
     {
        printf("\nOpen error on ITEM_FILE");
        close(fdes);
        return(FALSE);
     }
```

```
      /* Read item_list_file into vadd_list */
      if((num_items = read(fdes, vadd_list, sizeof(vadd_list))) <= 0)
      {
         close(fdes);
         printf("\nError with ITEM FILE");
         return(FALSE);
      }

      close(fdes);
      num_items =  num_items/ITEM_LENGTH;

      /* Check if item already has a predeccessor */
      for(vlptr = &vadd_list[0]; vlptr < &vadd_list[num_items]; vlptr++)
      {
         if((vlptr->name == value) && (vlptr->succ != NULL))
         {
           printf("\nItem already has a response");
           sleep(1);
           return(FALSE);
         }
      }

      return(TRUE);
    }

   return(TRUE);
}
```

```
/****************************************
 *                                      *
 * add_paper adds an author, the paper *
 * file name, and conference name to   *
 * the 'papers' relation in the marrsd *
 * data base.                           *
 *                                      *
 ****************************************/

add_paper(auth_name,pf_name,cf_name)
char *auth_name, *pf_name, *cf_name;
{

## char a_name[9];
## char p_name[15];
## char conf_n[16];

## ingres marrsd
   sprintf(a_name,"%s",auth_name);
   sprintf(p_name,"%s",pf_name);
   sprintf(conf_n,"%s",cf_name);

## append to papers(author=a_name,pfname=p_name,
##     confname=conf_n)

## exit

}
```

```
#include <stdio.h>
#include <strings.h>


/*****************************************
 *                                       *
 * add_title adds the title for a paper  *
 * in the 'pfile' relation in the marrsd *
 * database.                             *
 *                                       *
 *****************************************/

add_title(paper_nm,conf_nm)
char *paper_nm, *conf_nm;
{

## char pa_name[15];
## char pa_title[126];
## char pa_cname[16];

   int i, total_char, try_again;
   char line[80];
   char title_buf[125];

## ingres marrsd

   try_again = 1;

   sprintf(pa_name,"%s",paper_nm);
   sprintf(pa_cname,"%s",conf_nm);

   while(try_again)
   {
     try_again = 0;
     total_char = 0;
     title_buf[0] = NULL;

     printf("\nEnter title for %s (. on line by itself to quit) :\n",
             paper_nm);

     gets(line);
     i = strlen(line);

     if(i == 0)
     {
       try_again = 1;
       continue;
     }

     while((line[0] != '.') || (i > 1))
     {
          total_char += i;
          if(total_char > 125)
          {
            printf("\nPaper title can be at most 125 characters.");
            printf("\nPlease enter title again.\n\n");
            try_again = 1;
            break;
          }

          else
          {
            strcat(title_buf,line);

            gets(line);
            i = strlen(line);
          }
        }  /* end of while */

      }  /* end of while */

      if(total_char > 0)
      {
       sprintf(pa_title,"%s",title_buf);
##     append to pfile(pname=pa_name,cname=pa_cname,ptitle=pa_title)
      }

## exit

}
```

```
#include <strings.h>
#include "/usrb/att/marrs/Source/conf.h"

/*****************************************
 *                                       *
 * add_topic adds a maximum of 10 topics *
 * for a paper to the 'kwords' relation  *
 * in the marrsd data base.              *
 *                                       *
 *****************************************/

add_topic(paper_name,con_name)
char *paper_name, *con_name;
{

## char p_name[15];
## char t_name[21];
## char c_name[16];

   char string[20];
   int i, count;

## ingres marrsd

   count = 0;
   sprintf(p_name,"%s",paper_name);
   sprintf(c_name,"%s",con_name);

   for(;;)
   {
     if(count == 10)
     {
       printf("\nSorry, you can only have a maximum of 10 topics. ");
       sleep(1);
       break;
     }

     while(TRUE)
     {
      printf("\nEnter topic (? for help, q to quit) : ");
      gets(t_name);

      if(t_name[0] == '?')
      {
        CLEAR
        printf("\n\nThe keyword search provides the capability to \n");
        printf("list the conference names and paper file names \n");
        printf("associated with a requested topic.\n");
        printf("\nThe topic can be at most 20 characters.");
        printf("\nOnly one topic can be entered at a time.\n");
        printf("\n\nHit return to continue ");
```

```
          gets(string);
          continue;
        }

        else
          break;

      }


      i = strlen(t_name);
      if(i > 20)
      {
        printf("\nTopic can be at most 20 characters.\n");
        sleep(1);
        continue;
      }

      if(i == 0)
      {
        continue;
      }

      if(t_name[0] == 'q')
         break;


      else
      {
##     append to kwords(keywrd=t_name,cfname=c_name,paptitle=p_name)
       count++;
      }
    }

## exit

}
```

```
/*****************************************
 *                                       *
 * creat_rl creates the relations for the *
 * marrsd data base. It then saves them   *
 * until dec 31, 1990.                    *
 *                                       *
 *****************************************/

main()
{

## ingres marrsd

## create confdata(name=c15,status=c6,majprof=c8)
## create papers(author=c8,pfname=c14,date=i4,confname=c15)
## create kwords(keywrd=c20,paptitle=c14,cfname=c15)
## create pfile(pname=c14,cname=c15,ptitle=c125)

## save confdata until dec 31 1990
## save papers until dec 31 1990
## save kwords until dec 31 1990
## save pfile until dec 31 1990

## exit

}
```

```
/***********************************
 *                                 *
 * l_confau lists the conferences  *
 * for an author.                  *
 *                                 *
 ***********************************/

l_confau(auname)
char *auname;
{

## char lau_name[9];
## char lau_conf[16];

   int count;

## ingres marrsd

   count = 0;

   sprintf(lau_name,"%s",auname);

   printf("\nConferences with author %s: \n\n",auname);

## range of p is papers
## retrieve (lau_conf = p.confname)
##   where p.author = lau_name
## {
     count++;
     printf("%s",lau_conf);

     if(!(count % 4))
       printf("\n");

## }

   if(count == 0)
   {
     printf("\nSorry, there are no conferences for %s\n",auname);
     sleep(1);
   }

   printf("\n");

## exit

}
```

```
/************************************
 *                                  *
 * l_confmp lists the conferences  *
 * that a major professor has.      *
 *                                  *
 ************************************/

l_confmp(mpname)
char *mpname;
{

## char lmp_name[9];
## char lmp_conf[16];
   int count,i;

## ingres marrsd
   sprintf(lmp_name,"%s",mpname);
   count = 0;
   i = 0;

## range of c is confdata
## retrieve(lmp_conf = c.name)
##   where c.majprof = lmp_name
## {
     if(i == 0)
     {
       printf("\nConferences with major professor %s:\n\n",mpname);
       i++;
     }

     count++;
     printf("%s",lmp_conf);

     if(!(count % 4))
       printf("\n");
## }

  if(count == 0)
  {
    printf("\nSorry, there are no conferences for %s\n",mpname);
    sleep(1);
  }

  printf("\n");

## exit

}
```

```
/****************************************
 *                                      *
 * l_confpa lists the conferences and   *
 * paper names that are associated      *
 * with a topic.                        *
 *                                      *
 ****************************************/

l_confpa(topic)
char *topic;
{

## char kyword[21];
## char lpa_conf[16];
## char lpa_papr[15];

   int count;

## ingres marrsd

   count = 0;
   sprintf(kyword,"%s*",topic);

## range of k is kwords

## retrieve(lpa_conf = k.cfname, lpa_papr = k.paptitle)
##   where k.keywrd = kyword
## {
     count++;
     printf("\nConference: %s\n",lpa_conf);
     printf("Paper file: %s\n",lpa_papr);
## }

   if(count == 0)
   {
     printf("\nSorry, there are no conferences for %s\n",topic);
     sleep(1);
   }

## exit

}
```

```
/**********************************
 *                                *
 * l_confst lists all conferences *
 * and there status.              *
 *                                *
 **********************************/

l_confst()
{

## char l_name[16];
## char l_status[7];

   int count;

## ingres marrsd

   count = 0;

## range of c is confdata
## retrieve (l_name=c.name,l_status=c.status)
## {
     count++;
     printf("\nConference: %s   Status: %s",l_name,l_status);
## }

   if(count == 0)
   {
     printf("\nSorry, there are no conferences ");
     sleep(1);
   }

   printf("\n");

## exit

}
```

```
/***********************************
 *                                 *
 * l_ptitle lists the paper title *
 * for a paper.                    *
 *                                 *
 ***********************************/

#include <strings.h>

l_ptitle(paper,conf)
char *paper, *conf;
{

## char lt_title[126];
## char lt_fname[15];
## char lt_cname[16];

   char string[10];
   int count;

## ingres marrsd

   count = 0;
   sprintf(lt_fname,"%s",paper);
   sprintf(lt_cname,"%s",conf);

   printf("\nPaper title: \n\n");
## range of t is pfile
## retrieve(lt_title=t.ptitle)
##   where t.pname=lt_fname and t.cname=lt_cname
## {
     count++;
     printf("%s",lt_title);
## }

   if(count == 0)
   {
     printf("\nSorry there is no paper %s ",paper);
     sleep(1);
   }

   else
   {
     printf("\n\n\nHit return to continue ");
     gets(string);
   }

## exit

}
```

```
/****************************************
 *                                      *
 * upd_paper puts the published date    *
 * of a paper in the 'papers' relation *
 * in the marrsd data base.             *
 *                                      *
 ****************************************/

upd_paper(art_name,cnf_name,f_date)
char *art_name, *cnf_name;
long f_date;
{

## char fin_aname[9];
## char fin_cname[15];
## long fin_date;

## ingres marrsd
   sprintf(fin_aname,"%s",art_name);
   sprintf(fin_cname,"%s",cnf_name);
   fin_date = f_date;

## range of p is papers
## replace p(date = fin_date)
##   where p.author=fin_aname and p.confname=fin_cname

## exit

}
```

```
/****************************************
 *                                      *
 * upd_status changes the status of     *
 * a conerence from 'open' to 'closed'  *
 * in the confdata relation in the      *
 * marrsd data base.                    *
 *                                      *
 ****************************************/

upd_status(cname)
char *cname;
{

## char cl_conf[16];

## ingres marrsd
   sprintf(cl_conf,"%s",cname);

## range of c is confdata
## replace c (status = "closed")
##   where c.name = cl_conf

## exit

}
```

Appendix 3

MARRS 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 of 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 paths:

```
/usrb/att/marrs/Bin
/usr/ingres/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 shell prompt:

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

Instead of '/usr/ucb/vi', any other editor can be specified. The default 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 '.profile' or entered at the shell prompt:

```
MARRS_PRTR=lpr
export MARRS_PRTR
```

Instead of 'lpr', any other printer can be specified. The default 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 '.profile' 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 professor 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 of 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 of 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 platform discussion (option - d),
- generate a hard copy (option - g),
- do a keyword search of 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).

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

- create a conference (option - c),
- remind idle users (option - r),
- change the status of 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 of the conference.

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 of 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 of the conference's data files, the following message will be displayed:

```
Error in the creation of the conference aaaa Sorry!!
The system administrator of 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 of 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 of 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 of 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.

```
|-------------------------------------------------------------|
|                                                             |
|                                                             |
|           "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           |
|                                                             |
|-------------------------------------------------------------|
These options are available: c, p, l, q, ?  current line = 5
```

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 of 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 of 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.

```
|------------------------------------------------------|
|                                                      |
|                                                      |
|            "Writing is the more personal form        |
|          of communication, the one which permits     |
>          the most natural expression of feeling.    *
|           The message, once detached, can cross      |
........................................................
.      Enter a single '.' on a newline to terminate.   .
.                 Erase char is Backspace              .
.                                                      .
.           this is the area where the user puts the   .
.           comment that they will later target onto   .
.           other participants of the conference.      .
.                                                      .
........................................................
```

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 of 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 - platform 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 of 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 of 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 the major professor, the following is displayed:

```
Time to add audience members to this conference

Enter login id of 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 conference. Once the user is returned to the main MARRS menu, the setup is complete.

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

This option gives the user the capability to view or attach an item in the platform discussion. The items in the platform 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 of 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 of more than one conference :

```
Conferences available:

          marrs               tjs1                tjs2
          star_wars           test_con            dummy_con

Enter conference name :
```

The user then enters a conference 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 of 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 of 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 platform discussion are. The following screen is displayed :

```
Conference : 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 platform discussion
               ? - this display
               q - leave the platform discussion


     Hit return to continue
```

To continue with the platform 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 :

```
Conference : 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 of the paper to be printed.

5.2 Selecting the Author/Paper

Following the selection of 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: marrs
Paper name: features

Conference: marrs
Paper name: datamanager

Conference: marrs
Paper name: telesystems


Enter paper file to print paper title (0 for none) :
```

The user can now enter the paper name and then will be prompted for the conference name. The user can enter a 'q' to continue or the conference name. If the user enters the conference name the paper's title is printed. A '0' can be entered to continue with the keyword search. If a paper name is entered, say 'datamanager', the following is displayed :

```
Paper title: datamanager

Data Management in MARRS


Hit return to continue
```

7. L - List Current Conferences

This option allows a user to see what conferences currently exist on the system. A user can choose to look at the status of all conferences or choose conferences that have a specific major professor or author. In addition, the user can select specific conference in order to get additional information on the conference.

7.1 Selecting the Listing Option

After the "l" option is selected from the main menu, a new menu is 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:

```
    Valid options are:

            l - list all conferences
            s - list conferences and their status
            m - list conferences by major professor
            a - list conferences by author's name
            q - quit

    Enter your desired option:
```

The user can enter any of the options listed. Following completion of that option's routine, control is passed back to the previous menu. At this point the user can quit or enter another option.

7.1.1 The 'l' option - 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.

7.1.2 The 's' option - Selecting the Conference
If an 's' is entered at the option menu, all past and present conferences and their statuses are displayed. The following shows what a typical screen would show.

```
    Conference: aaaaaa          Status: open
    Conference: bbbbbbb         Status: closed
    Conference: ccccc           Status: open
    Conference: dddd            Status: open
    Conference: eeeee           Status: closed


    Hit return to continue:
```

At this point the user can hit a carriage return when finished observing the display. Control is then passed back to the options menu.

7.1.3 The 'm' option - Selecting Conferences by Major Professor

If an 'm' is entered at the option menu, the system will display the following:

```
    Enter major professor's login name :
```

The user then can enter a major professor's login name. The following display is the shown (assume "rich" was entered as the major professor's name):

```
    Conferences with major professor rich:

    aaaaaa          cccccccc        ddddd


    Hit return to continue:
```

At this point the user can hit a carriage return when finished observing the display. Control is then passed back to the options menu.

7.1.4 The 'a' option - Selecting Conferences by Author

If an 'a' is entered at the option menu, the system will display the following:

```
    Enter author's login name :
```

The user then can enter an author's login name. The following display is the shown (assume "rich" was entered as the major professor's name):

```
    Conferences with author monk:

    ffffff          gggggggg        hhhhh


    Hit return to continue:
```

At this point the user can hit a carriage return when finished observing the display. Control is then passed back to the options 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 enviroment 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 conference 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 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:

```
                    REMIND IDLE USERS


     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 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 of 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 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 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. 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.

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 of 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 professor of, 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 of, 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 of a conference, the next display shows:

```
Adding new audience members to this conference

Enter login id of 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.

Data Management in MARRS

by

Kitty A. Monk

B.S. Miami University, 1980

An Abstract of 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

An Abstract of a Master's Report

This paper discusses a computer teleconferencing system that is designed for overseeing the completion of a software design project. The software design has been focused on automating the collection, the review, and the presentation of master's reports. It is assumed that this software project is already committed and funded. This paper does not describe the computer teleconferencing system in detail. It looks at the issues surrounding data management in a computer teleconferencing system.

This conferencing system uses a mixture of a data base management system and file processing system for the storage of its data. It is implemented under the UNIX operating system. The INGRES relational data base management system is used to store information about conferences and their participants. Participants in a conference can make queries involving conferences and participants.