SAND--89-1715

DE89 014353

SAND89-1715
Unlimited Release
Printed July 1989

An Integrated Network Management Tool

Marc M. Miller
Computer Communications Design Division
Sandia National Laboratories
Albuquerque, NM 87185

Abstract

This report describes an integrated network management tool designed to minimize the time and effort required to diagnose and resolve data communications problems at Sandia National Laboratories, Albuquerque. A workstation is utilized to consolidate several functions necessary for diagnosing these problems, thus reducing the amount of hardware required for troubleshooting. Functions include managing data PBX and PACX switches, accessing circuit database information on a VAX, and monitoring alarms. In addition to consolidating functions, enhanced capability is provided for simplifying the complex procedure of troubleshooting data PBX problems. The first two sections of this report give an introduction and overview of the network management applications and the last section provides details of operation.

[illegible]

Acknowledgment

The original idea for an integrated network management tool that led to the efforts described in this paper was put forth by Larry Tolendino, 2647.

TABLE OF CONTENTS

GLOSSARY OF ACRONYMS

AMM	-	Apollo Man/Machine application
bps	-	bits per second
DTR	-	Data Terminal Ready
ISV	-	In SerVice
Mbyte	-	megabyte (1,000,000 bytes)
Mhz	-	megahertz (1,000,000 cycles per second)
OSV	-	out of SerVice
PACX	-	Private Automatic Computer eXchange
PBX	-	Private Branch eXchange
PC	-	Personal Computer
TCC	-	Tech Control Center
<cr>	-	carriage return

1.0 Introduction

This report describes an integrated network management tool designed to minimize the time and effort required to diagnose and resolve data communications problems at Sandia National Laboratories, Albuquerque. When users experience a data communications problem, they call a central location called the Trouble Desk. The operator diagnoses the problem and attempts to correct it. Normally, the problem is quickly resolved and the user is able to resume working.

Data communications at Sandia includes switched asynchronous terminals, synchronous terminals, local area networks, and intersite networks. Two systems currently exist for switching asynchronous terminals. Gandalf PACX port contenders have been in place for several years but are being phased out and replaced by an Intecom PBX. These two systems require operator maintenance and control from the Trouble Desk. Due to the fact that Gandalf PACXs are distributed throughout Sandia, a network of workstations was developed for the purpose of supervising and monitoring. The Trouble Desk operator uses one of these workstations when diagnosing and resolving a problem encountered by a PACX user. Each workstation communicates with up to four sets of PACX command and statistics ports for issuing commands and receiving statistics on PACX port activity. The workstations are networked together to allow commands at one station to be submitted to a PACX at another station and to collect statistics from all PACXs at one node. These workstations were dedicated to this application. The Intecom PBX, being a single point-of-control switch, does not require such an elaborate control and monitoring scheme. A simple terminal is the console used by the Trouble Desk operator for resolving user problems.

Circuit information for Sandia data communications circuits is kept in a database on a VAX computer. Applications have been designed to enter and retrieve this data and to log problems that users experience. The Trouble Desk operator will log each trouble call and retrieve information on the caller such as their location and telephone number using a terminal dedicated to this application.

Terminals which are located outside the secure areas are switched off at night and on in the morning to prevent unauthorized access when unattended. Software is used for controlling these terminals by instructing the PACX or PBX to disable or enable the terminal. The Trouble Desk operator enters the on or off time on either a workstation controlling a PACX or the terminal controlling the PBX. Also, an additional terminal is used for monitoring alarms originating from the software controlling the PBX terminals.

In addition to these primary functions, the Trouble desk also included an IBM PC which provided a simplified interface to the PACX workstation application and contained information such as whom to call for circuit repair.

The Trouble Desk was originally arranged with the workstation, PC, and PBX terminal on the desk and three terminals located on a shelf above the desk. One of the terminals on the shelf was used for monitoring errors from the software that turned terminals on and off on the

PBX. The other two terminals were used for the circuit database, trouble log, and testing host connections. When the operator received a trouble call from a PACX user, terminal status was first checked using the workstation (or PC) and any necessary action taken such as disconnecting the terminal from a bad host port. The operator would then have to move to one of the terminals on the shelf to log the call. When a trouble call was received from a PBX user, the operator would have to move to the PBX terminal, attempt to resolve the problem, and again move to the terminal on the shelf to log the call. Also, during the troubleshooting process, the operator might have to call someone to assist and might refer to the PC to find the necessary telephone number.

In order to improve the Trouble Desk operation and provide additional features, all the functions described above were moved to a single workstation. The multitasking capability of the workstation allows each application to run in its own window. Changing applications is a matter of pressing a key until the desired window appears. Thus the Trouble Desk operator can sit at one keyboard and screen to do all troubleshooting and data entry. Figure 1 shows a view of the Trouble Desk workstation screen. Three applications are running in windows and others are in icon form along the upper right side of the screen. In addition to focusing the operator on one screen, applications were developed which improve the troubleshooting of problems and in determining when problems have occurred. A side benefit of the single workstation is the reduction in the amount of hardware and space required to do the job.

The original workstation being used to control the PACXs proved to be inadequate for consolidating the Trouble Desk functions. It had insufficient disk space and memory, and processing speed was too slow for running many applications simultaneously. An Apollo DN3500 was chosen with 4 Mbytes of main memory, a 155 Mbyte hard disk, a 1.2 Mbyte floppy disk, color monitor, and a 25 Mhz 68030 processor.

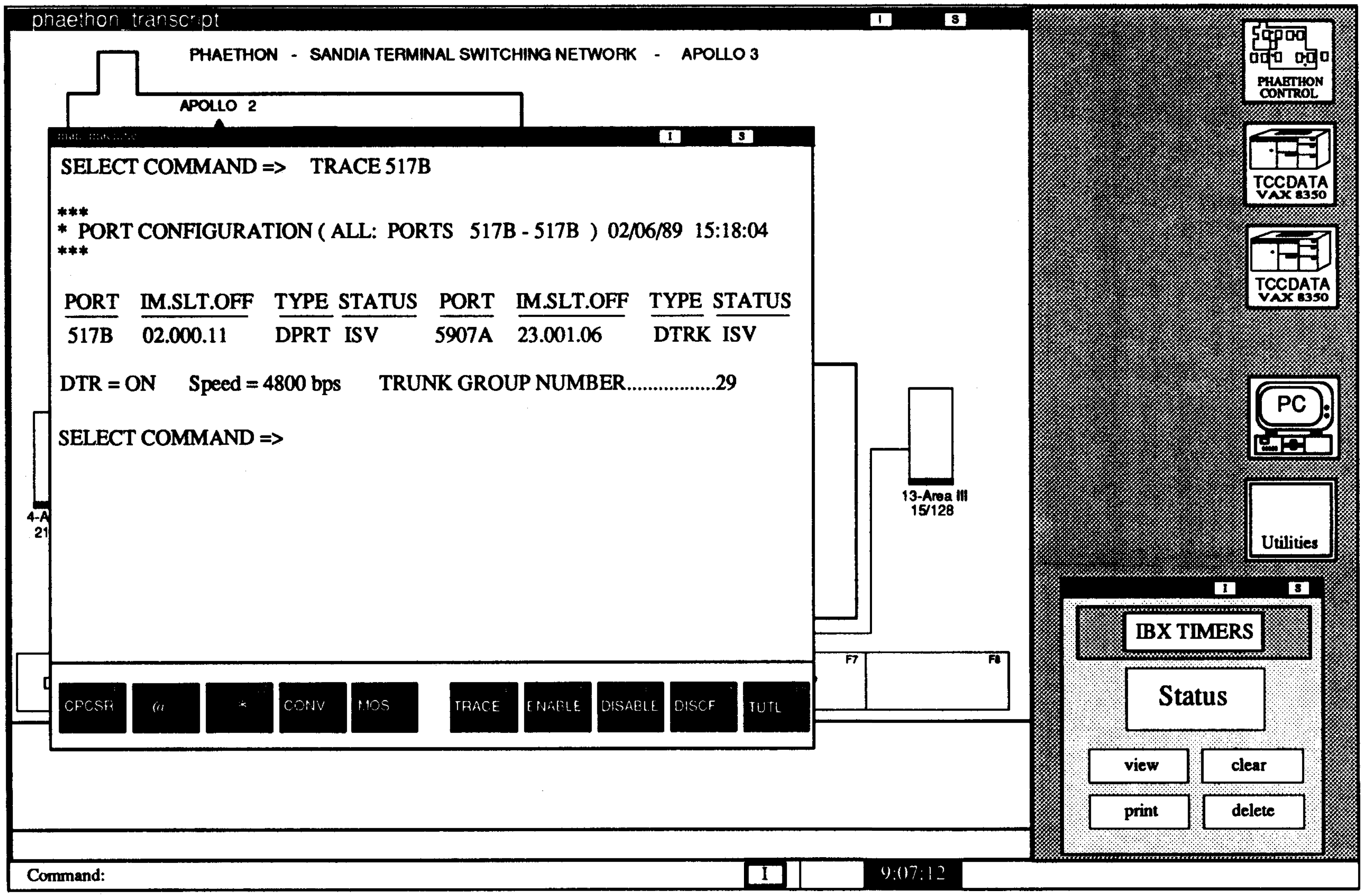
2.0 Description of Applications

This section provides a general view of each of the applications that run on the Trouble Desk workstation and shows examples of some of the screens associated with the applications. Note that the example windows are not drawn to scale.

2.1 Phaethon

The original workstation used at the trouble Desk ran a single application for controlling the PACX data switches (figure 2.). This application, called PHAETHON, provided a graphical representation of the state of the network of PACXs in addition to providing a means of managing PACXs. The workstation was networked to four other workstations located in other Tech Control Centers (TCCs). Each workstation could communicate with up to four PACXs. Commands could be submitted to any PACX from any node. Thus the Trouble Desk operator was able to command any PACX from the Trouble Desk.

PHAETHON was moved from its original workstation to the new workstation. No changes were required because the two workstations are compatible.



```
SELECT COMMAND =>   TRACE 517B

***
* PORT CONFIGURATION ( ALL:  PORTS  517B - 517B  ) 02/06/89  15:18:04
***

PORT   IM.SLT.OFF   TYPE  STATUS   PORT    IM.SLT.OFF   TYPE  STATUS
517B   02.000.11    DPRT  ISV      5907A   23.001.06    DTRK  ISV

DTR = ON    Speed = 4800 bps    TRUNK GROUP NUMBER.................29

SELECT COMMAND =>
```

Figure 1. Full View of Workstation Display

2.2 Phaethon Control

Because the PHAETHON application is distributed among five nodes and access to a remote PACX from a local Apollo node requires the remote Apollo node to be alive, it became a problem when a remote node application would fail and no one was at the remote location. The remote PACX would then be inaccessible until someone could be dispatched to correct the problem (this usually involved restarting the application). The downtime made operation of the Trouble Desk difficult as users of the affected PACXs could not be helped until the problem was corrected.

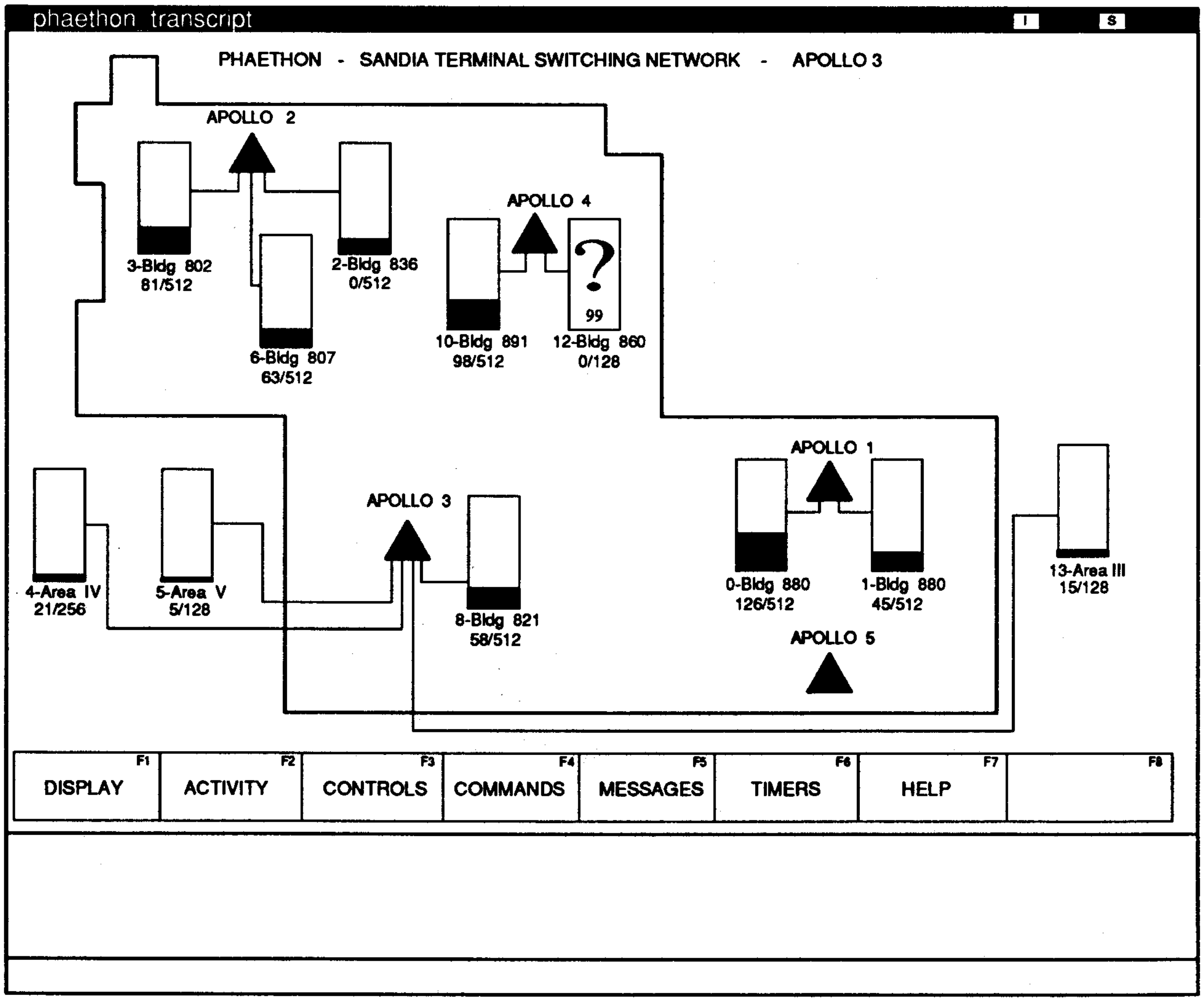
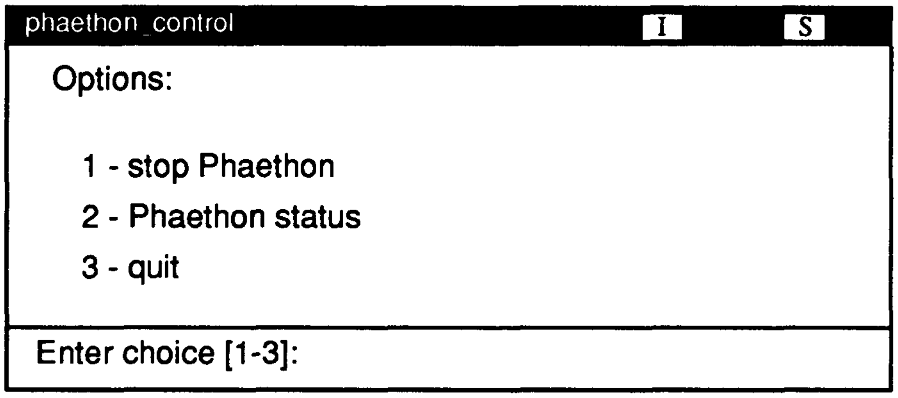


Figure 2. Phaethon_Transcript

A remedy for this problem was to take advantage of the network of the Apollo workstations which allows nodes to submit system commands to other nodes. In this case, a command can be sent to another node to command it to kill a particular process. If PHAETHON is commanded to quit, a separate background task will detect that PHAETHON has aborted, and it will restart PHAETHON. This technique of restarting PHAETHON was built into an application called PHAETHON_CONTROL (figure 3) so that the operator could restart any PHAETHON without knowledge of system commands. The application would issue the appropriate command to signal the remote PHAETHON to halt. The remote background process that oversees PHAETHON would then detect the loss and restart PHAETHON. The operator also had the option of determining the status of a remote PHAETHON to determine if a problem actually existed.



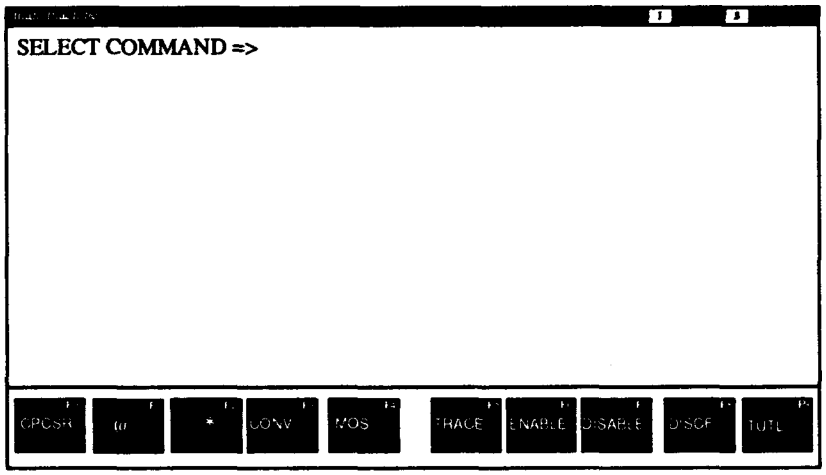
```
Options:

  1 - stop Phaethon
  2 - Phaethon status
  3 - quit

Enter choice [1-3]:
```

Figure 3. **Phaethon_Control**

To start Phaethon_Control, the operator moves the cursor to the phaethon_control icon and presses the right mouse button. A window will open containing the PHAETHON_CONTROL application. The operator then moves the cursor to the input window and enters a '1','2', or '3'. If the operator wishes to determine the status of Phaethon on a node, a '2' is entered. The application will then prompt for a node number to enter. The operator would enter a single digit ranging from 1 to 5 to select the node. The application would then query the selected node to determine the current Phaethon status on that node and then display the results: **PHAETHON is running** or **PHAETHON is not running**. If the operator wanted to restart a Phaethon, '1' would be entered. The application would then prompt for a node number. The operator would enter a digit ranging from 1 to 5 to select a node and the application would then signal the selected node to halt Phaethon. The remote background process that oversees PHAETHON would then detect the loss and restart PHAETHON. To exit the application, the operator would enter a '3'.

2.3 PBX Terminal (Man/Machine Interface)

PBX control was moved to the APOLLO workstation by running an application in a window which connects to one of the APOLLO asynchronous communications ports. A PBX control port was connected to this APOLLO communications port. This application is essentially a terminal emulator in that characters typed are sent out the communications port and characters received are sent to the screen. Thus the operator sees a regular PBX control terminal operating in a workstation window. However, additional capabilities where added to the application by taking advantage of the power of the workstation. The PBX control application is called AMM for APOLLO MAN/MACHINE (figure 4). In addition to sending and receiving characters, AMM also contains programmed keys and custom commands for simplifying the troubleshooting procedure.



```
SELECT COMMAND =>
```

Figure 4. Man_Machine Application Screen

2.3.1 Troubleshooting Technique

With a standard man/machine terminal (ie: dumb terminal connected directly to a PBX control port), the Trouble Desk operator may use several commands to determine the status of a user port when a problem is reported. For example, the operator might want to know if the port is enabled or disabled, or if the port is connected to a host. The operator might start by entering the MOS command to determine the enable status. The operator would enter **MOS<cr>**, then **D<cr>** to display the desired port, then **<cr>** to display all possible configurations (in service, out of service, assigned, etc.), and finally would enter the desired PBX port number. The PBX would then output the status of the user port and the operator would then decide to enable the port if it was disabled. To enable the port, while still in the

MOS command, the operator would enter ***<cr>** to back the MOS command one prompt, **I<cr>** to set the command to 'in service' mode, and then enter the port number. The operator then aborts the command by entering **@<cr>**.

If the user port was connected to a host or in some way hung in the PBX, the operator would have to determine to what port the user is connected. The CONV command is used for this case. The operator would enter **CONV<cr>**, then **F<cr>** for facility, **P<cr>** for port, and finally the port number. The PBX will then display a series of hexadecimal numbers representing different aspects of the port's status. The operator will pick out a certain number, the 'facility' to which the user port is connected, then enter the sequence **F<cr>** and facility number and **@<cr>** to abort the command. The PBX will respond with a similar display for the connected port, including the standard decimal based port number. The operator then knows the connected port and can then determine, if necessary, to what host this port belongs. This is done using the PORT command. The operator enters **PORT<cr>**, **D<cr>** to display port information, the desired port number and **@<cr>** to abort. The PBX will display information including the host group number (known as the trunk group). The operator can then determine, if necessary, what this host group is by using the GRPS command. The operator enters **GRPS<cr>**, **D<cr>** to display trunk group information, and the host number followed by **@<cr>**. The PBX will output the name of the host group to which the user port is connected.

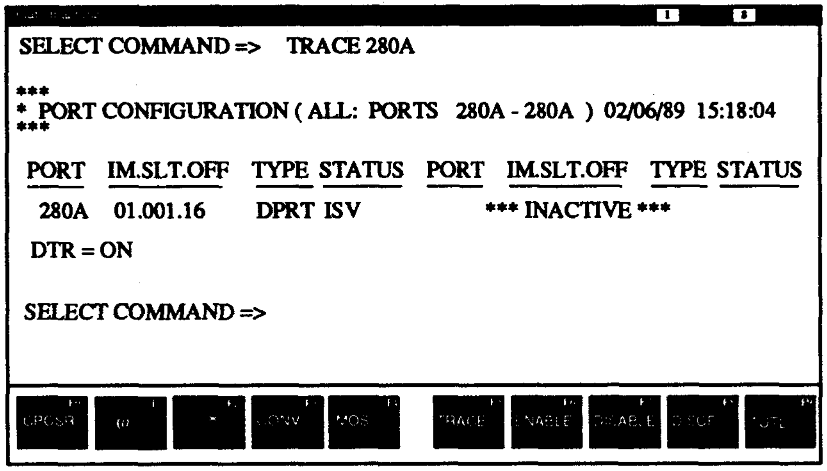
When a user port is hung, the operator disconnects the port by either disabling the user port or the connected port and then enables the port using the MOS command. The operator enters **MOS<cr>**, **O<cr>** to take the port out of service, and the port number. The PBX will output the MOS status as OSV (out of service). The port would then be enabled by entering ***<cr>** to back up one prompt, **I<cr>** to put the port back in service, and the port number. The operator would finish with **@<cr>** to abort the MOS command.

2.3.2 Troubleshooting Enhancements

These awkward and time consuming strings of commands were replaced in the AMM application with custom commands which execute the PBX commands for the operator. These commands are entered by typing the command in full or by pressing programmed keys. For example, to determine the status of a port, the operator presses the F5 key followed by the desired port number:

TRACE 280A<cr>

Note that the port number is entered on the same line as the command. The AMM application will then execute the same series of commands that the operator did as described previously. However, the output will be different than what would be generated if the operator executed each command. The results of all the individual PBX commands are condensed, translated, and presented to the operator. An example is shown in figure 5 where the operator has requested information



```
SELECT COMMAND =>   TRACE 280A

***
* PORT CONFIGURATION ( ALL: PORTS  280A - 280A ) 02/06/89 15:18:04
***

PORT  IM.SLT.OFF  TYPE STATUS  PORT  IM.SLT.OFF  TYPE STATUS
 280A  01.001.16   DPRT ISV          *** INACTIVE ***
 DTR = ON

SELECT COMMAND =>
```

Figure 5. Trace of Inactive Port

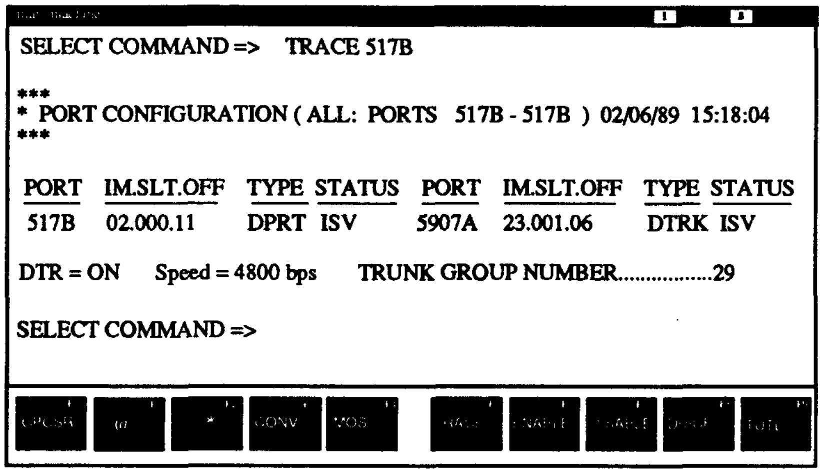
on port 280A. AMM responds by showing characteristics of port 280A and determines, in this case, that it is not currently connected to any other port (inactive). These characteristics include: the physical location of the port in the system or IM.SLT.OFF (Interface Multiplexer 1, Slot 1, Offset 16), type of port (in this case a data port or DPRT), status (ISV or in service) and DTR on. The TRACE command gathers this information by using the MOS command to get the port, IM.SLT.OFF, type, and status, and the CONV command to find connection and DTR status.

AMM translates the TRACE command into the long sequence of commands:

```
MOS<cr>,D<cr>,<cr>,port#<cr>,@<cr>
CONV<cr>,F<cr>,P<cr>,port#<cr>,F<cr>,facility#<cr>,@<cr>
```

This is executed automatically for the operator. Because AMM determined that the port is not connected to another port, there is no need to execute the PORT command for determining the trunk group of the host.

If the traced port is connected to another port, the operator would see a screen as shown in figure 6:



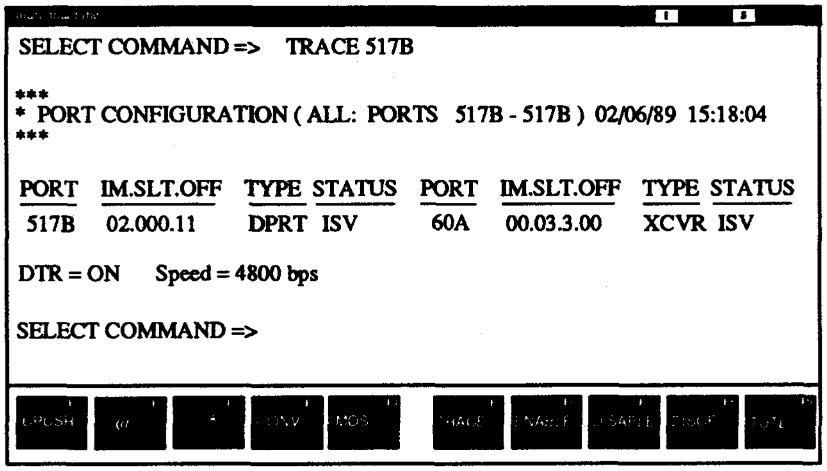
```
SELECT COMMAND =>   TRACE 517B

***
* PORT CONFIGURATION ( ALL:  PORTS  517B - 517B  )  02/06/89  15:18:04
***

PORT   IM.SLT.OFF   TYPE  STATUS   PORT    IM.SLT.OFF    TYPE  STATUS
517B   02.000.11    DPRT  ISV      5907A   23.001.06     DTRK  ISV

DTR = ON     Speed = 4800 bps     TRUNK GROUP NUMBER.................29

SELECT COMMAND =>
```

Figure 6. Trace of Active Port

Again, the TRACE command shows the status of the port in question, but also shows the system location, type, and status of the connected port. In this case, port 517B is connected to port 5907A. Port 5907A is a host port (DTRK or data trunk), is in service (ISV), and is located at 23.001.06 (Interface Multiplexer 23, Slot 1, Offset 6). The TRACE also shows that the originating port has DTR, is communicating at a data speed of 4800 bps and that the trunk group number of the host is 29. The sequence of commands executed in this case is the same as shown above except that, in addition, the PORT command is needed for determining the trunk group of the host as shown:

```
PORT<cr>,D<cr>,port#<cr>,@<cr>
```

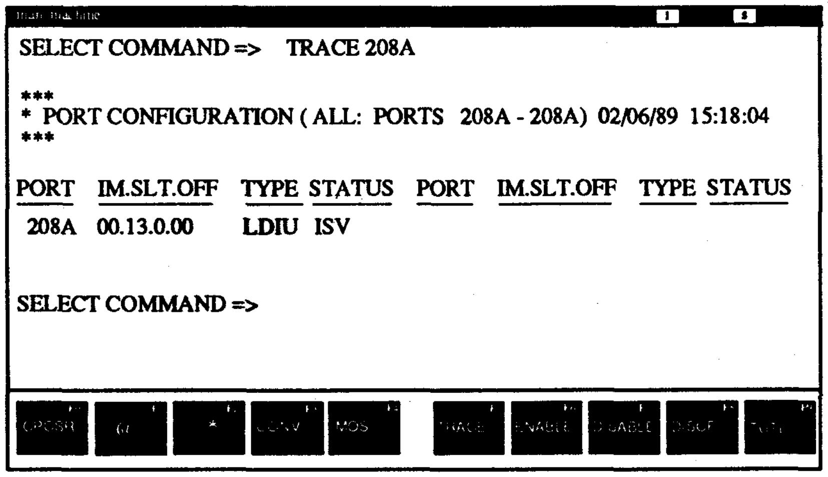
A port connected to a transceiver would appear in the TRACE command as shown in figure 7. A transceiver is a port with which a user communicates during the PBX login dialog sequence. The results of this TRACE are similar to a trace to a host port except that the connected port is shown as XCVR.



```
SELECT COMMAND =>   TRACE 517B

***
* PORT CONFIGURATION ( ALL:  PORTS  517B - 517B ) 02/06/89  15:18:04
***

PORT   IM.SLT.OFF   TYPE  STATUS   PORT   IM.SLT.OFF   TYPE  STATUS
 517B   02.000.11    DPRT  ISV       60A    00.03.3.00   XCVR  ISV

DTR = ON     Speed = 4800 bps

SELECT COMMAND =>
```

Figure 7. Port Connected to Transceiver

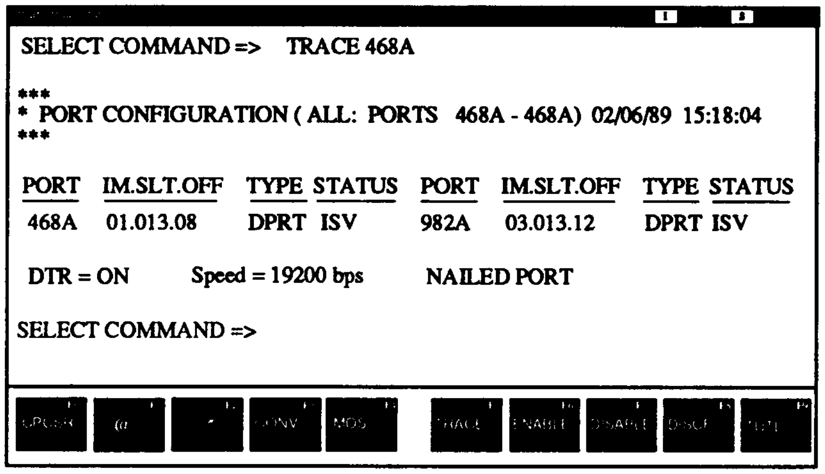
Ports that are configured for Ethernet appear differently when traced. This is due to the fact that an Ethernet port does not connect directly to another port, but communicates with many ports simultaneously over a common data path. These ports also do not have communication signals associated with asynchronous ports. Thus the TRACE will show the port, its status, and system location, but will not show DTR status, speed, or any connected port information. This is shown in figure 8.



```
SELECT COMMAND =>   TRACE 208A

***
* PORT CONFIGURATION ( ALL:  PORTS  208A - 208A) 02/06/89  15:18:04
***

PORT   IM.SLT.OFF   TYPE  STATUS   PORT   IM.SLT.OFF   TYPE  STATUS
 208A   00.13.0.00   LDIU  ISV

SELECT COMMAND =>
```

Figure 8. Trace of Ethernet Port

Ports that are permanently connected to other ports are called 'nailed ports'. Nailed ports are typically used for connections such as a printer remotely connected to a host or for a terminal that does not require switching to other hosts. When a user calls in to the Trouble Desk with a nailed port problem, they typically do not know that their port is nailed or that the operator needs to know this for proper debugging. Therefore, the TRACE command includes this information when tracing a nailed port as shown in figure 9:



```
SELECT COMMAND =>   TRACE 468A

***
* PORT CONFIGURATION ( ALL:  PORTS  468A - 468A) 02/06/89  15:18:04
***

PORT   IM.SLT.OFF   TYPE  STATUS   PORT   IM.SLT.OFF   TYPE  STATUS
468A   01.013.08    DPRT  ISV      982A   03.013.12    DPRT  ISV

DTR = ON       Speed = 19200 bps       NAILED PORT

SELECT COMMAND =>
```

Figure 9. Trace of Nailed Port

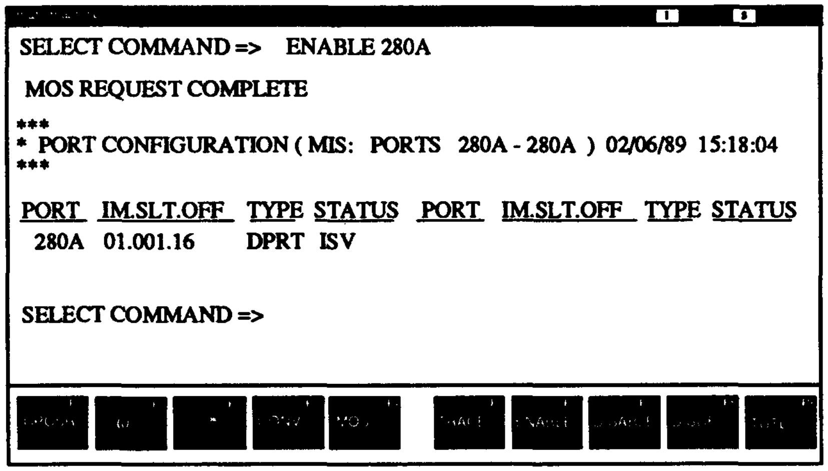
Once the operator has traced a port, action can be taken to correct the problem. As described above, the port can be placed in service, taken out of service, or disconnected depending on the problem. For example, if a port is not working and the trace shows that the port is out of service (OSV), the operator can issue the **ENABLE** command as follows:

ENABLE 280A<cr>

When the operator presses the F6 function key, the command **ENABLE** appears on the screen followed automatically by the most recently traced port number, in this case 280A. The operator can then choose to accept this port number and enter a carriage return to execute the command, or can edit the port number if a different port is desired. Normally the default port will be accepted and the operator has only to enter F6 followed by a carriage return to execute the **ENABLE** command as shown in figure 10. AMM will translate the enable command into the following man/machine command sequence :

MOS<cr>,I<cr>,port#<cr>,@<cr>

and execute this sequence for the operator.



```
SELECT COMMAND => ENABLE 280A

MOS REQUEST COMPLETE

***
* PORT CONFIGURATION ( MIS: PORTS 280A - 280A ) 02/06/89 15:18:04
***

PORT  IM.SLT.OFF  TYPE STATUS  PORT  IM.SLT.OFF  TYPE STATUS
 280A  01.001.16   DPRT  ISV

SELECT COMMAND =>
```

Figure 10. Example of Port Enable

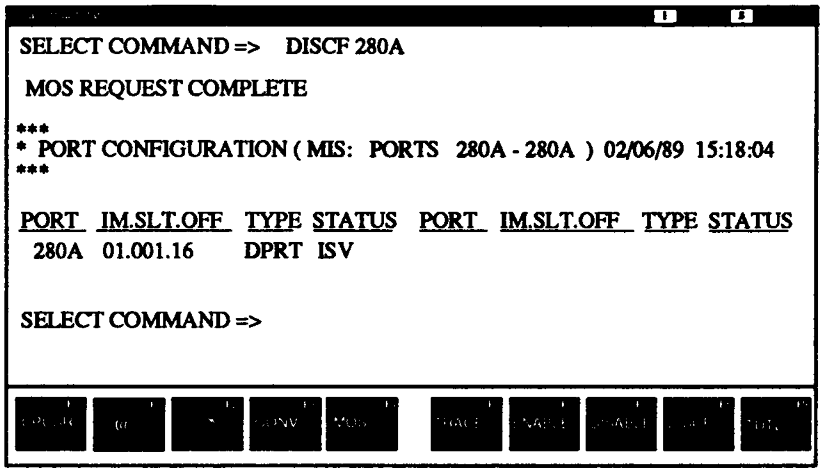
If the operator finds from the trace that the port is in service and connected to a host port, but the user is unable to communicate with the host, the port can be disconnected by taking it out of service and restoring it to service. This is done using the **DISCF** command:

DISCF 280A<cr>

When the operator presses the F8 function key, the command **DISCF** appears on the screen followed automatically by the port number, in this case 280A. The operator can then choose to accept this port number and enter a carriage return to execute the command, or can edit the port number if a different port is desired. Normally the default port will be accepted and the operator has only to enter F8 followed by a carriage return to execute the **DISCF** command as shown in figure 11. The AMM application will translate the disconnect command into the following man/machine command sequence:

MOS<cr>,F<cr>,port#<cr>,*<cr>,I<cr>,port#<cr>,@<cr>

and execute this sequence for the operator. The result of the disconnect command shows that the port has been restored to service (ISV).



```
SELECT COMMAND =>   DISCF 280A
 MOS REQUEST COMPLETE
***
* PORT CONFIGURATION ( MIS:  PORTS  280A - 280A ) 02/06/89 15:18:04
***

PORT  IM.SLT.OFF  TYPE STATUS  PORT  IM.SLT.OFF  TYPE STATUS
 280A  01.001.16   DPRT  ISV

SELECT COMMAND =>
```

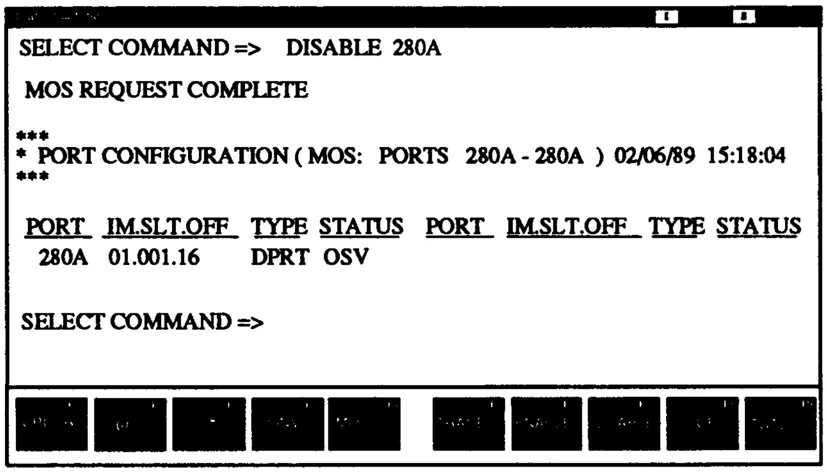
Figure 11. Example of Port Disconnect

The 'F' in the **DISCF** command refers to a forced disconnect where the connected ports are disconnected immediately by using the 'F' option of the MOS command. This is required when disconnecting a user port from a host port in order to affect the disconnect immediately. The other MOS option is to take a port out of service after the port disconnects itself, typically when a user logs off a host. This option is available to the Trouble Desk operator by typing **DISC** followed by a space. AMM will then display the port number that was last traced and the operator can then enter a carriage return to execute the command. This command has not been defined as a programmed key due to the limited number of programmable keys and the limited usefulness of the command.

When the operator determines that a port should be taken out of service and remain so for some time, the command **DISABLE** is used:

DISABLE 280A<cr>

The operator presses the F7 key and enters the desired port number as shown in figure 12. Note that the port number is not automatically entered for the operator. This is to allow the operator to choose to disable either the user port or the host port, depending on the situation.



```
SELECT COMMAND =>  DISABLE 280A

 MOS REQUEST COMPLETE

***
* PORT CONFIGURATION ( MOS:  PORTS  280A - 280A ) 02/06/89 15:18:04
***

 PORT  IM.SLT.OFF  TYPE STATUS  PORT  IM.SLT.OFF  TYPE STATUS
  280A  01.001.16   DPRT  OSV

SELECT COMMAND =>
```

Figure 12. Example of Port Disable

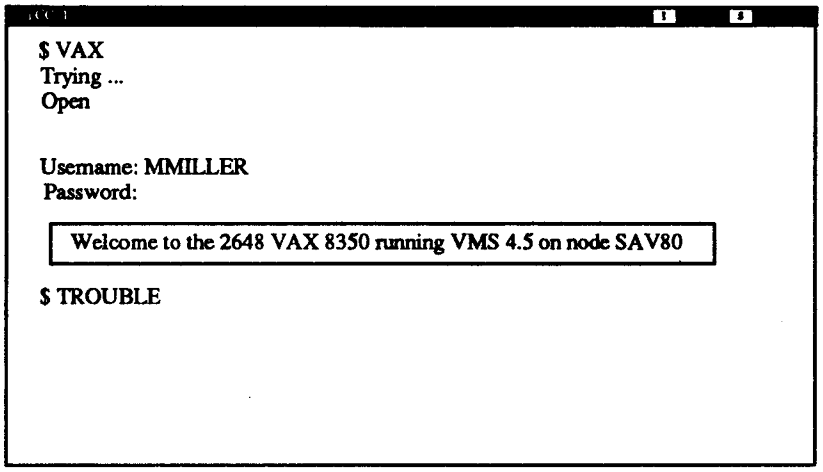
Typically, when a user experiences a problem with a host, it is due to a malfunctioning host port. In this case, the operator would want to disable that host port until the problem could be resolved. Therefore, when the operator presses the F7 key for the **DISABLE** command, no port number is displayed, allowing the operator to choose to enter the user port or the host port. The AMM application will translate the DISABLE command into the following man/machine sequence:

MOS<cr>,O<cr>,port#<cr>,@<cr>

The result of the DISABLE command shows that the port has been take out of service (OSV).

2.4 VAX Access

The Trouble Desk workstation is connected to a VAX 8350 host via Ethernet to allow the operator to access the circuit database that resides on the VAX. A VT100 emulator is run on the workstation and TCP/IP communications software is used to enable the workstation to appear as a terminal to the host. An example connection is shown in figure 13. The operator opens the connection to the VAX by typing 'VAX'. This command file will start the VT100 emulator and start a TELNET session to SAV80. The TCP/IP software then attempts to open a telnet session to the host and is able to complete the connection (this usually takes only a few seconds). The window in which the VT100 emulator is running now appears and behaves as a VT100 terminal. The operator logs on to the VAX and receives the welcome message and the system prompt. In this example, the operator then types 'TROUBLE' to start the database trouble log application.



```
$ VAX
Trying ...
Open

Username: MMILLER
Password:

 Welcome to the 2648 VAX 8350 running VMS 4.5 on node SAV80

$ TROUBLE
```

Figure 13. VAX **TELNET** Session

Due to the multitasking and multiwindowing capability of the workstation and the multisession capability of TCP/IP, the operator can have multiple simultaneous VAX sessions. Two windows are automatically provided for the operator, and additional windows can easily be created if additional sessions are desired. The operator changes between windows by simply pressing a key on the keypad or mouse until the desired window is on top of the others.

2.5 Timer_Status

As mentioned previously, ports on the PBX which are outside the secure area are enabled and disabled daily to prevent unauthorized access during off hours. The software that controls the ports is called 'timers', referring to the fact that the ports are controlled based on time of day. This software runs on a VAX and communicates with the PBX through a man/machine port. It is very important for the Trouble Desk operator to be aware of any problems that might arise with the Timers because of the necessity of insuring that ports are indeed being enabled and disabled as expected. Consequently, an application was developed for monitoring the activities of Timers and to inform the Trouble Desk operator when something has gone wrong. This application is called Timer_Status (figure 14). It receives and displays error messages from the VAX when the VAX determines that a problem has developed.

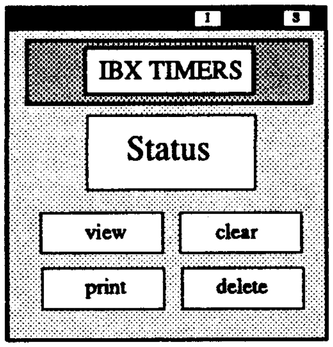


Figure 14. **Timer_Status**

When an error has occurred, Timer_status will flash the status box red to draw the attention of the operator. The operator can then clear the alarm and view the alarm message(s) as received from the VAX. The operator also has the option to print the messages and delete the current message file. To clear the alarm, the cursor is moved to the Timer_status window and into the '**clear**' box which turns color to indicate the cursor is in the box. The operator pushes the left mouse button to then restore the status box to the non-blinking state. To view the alarms, the cursor is moved to the '**view**' box and the left mouse button is pushed. A window will open which contains the alarm messages. An example is shown below:

```
10-MAY-1989 14:30:00.33 LOG-8  ERROR: timeout from port disable: 2870A
10-MAY-1989 14:30:00.33 LOG-8  ERROR: port 2870A will not disable
10-MAY-1989 17:30:00.33 LOG-9  ERROR: timeout waiting for response to @<cr>
10-MAY-1989 17:30:00.33 LOG-9  ERROR: timeout waiting for response to @<cr>
10-MAY-1989 17:30:00.33 LOG-9  ERROR: timeout waiting for response to @<cr>
10-MAY-1989 17:30:00.33 LOG-9  ERROR: timeout waiting for response to @<cr>
10-MAY-1989 17:30:00.33 LOG-9  ERROR: UNABLE to contact man/machine
```

Each line shows the time and date of the error, the number of the VAX log file that contains the error, and a summary of the error. In this case, on May 10 at 2:30 p.m., an attempt was made to take port 2870A out of service, but the port remained in service for some reason. The operator could then attempt to learn what is wrong with port 2870A and correct the problem. Later the same day, the VAX was unable to contact the PBX and gave up trying. This is a serious problem because under these conditions, no ports can be enabled or disabled. The problem should be diagnosed and corrected as soon as possible. If the problem continues, it will be reported every 30 minutes on the half hour until the problem is resolved.

2.6 Utilities

There are several system operations which the Trouble Desk operator must be aware of in order to maintain the workstation. These operations include shutting down the workstation, creating windows, and establishing areas on the screen where the cursor should not enter. To make the job easier and quicker for the operator, an application called **UTILITIES** was developed which will execute these functions by simply selecting the desired function and pressing a button. Figure 15 shows the main UTILITIES menu. The operator activates the application by moving the cursor to the UTILITY icon and pressing the right mouse button. A window will open containing the main UTILITIES menu. The operator moves the cursor to this window and into the box containing the desired function. The box will then change color to indicate that the function has be selected. Pushing the left mouse button will either select a second menu or activate the function depending on what is chosen.

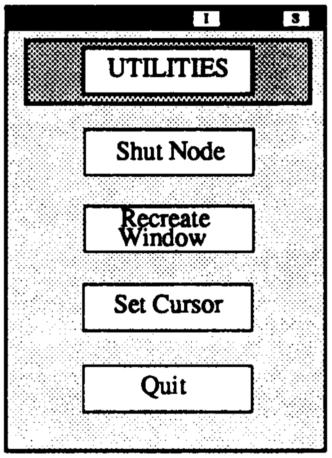


Figure 15. **Utilities** Main Menu

To shut down the node, the cursor is moved into the 'Shut Node' box and the left mouse button is pressed. A new menu will appear as shown in figure 16. The operator can then cancel the command by again pressing the left mouse button while the cursor is in the 'Cancel' box or move the cursor down to the 'OK' box and then press the left mouse button. If is it desired to shut down the node, UTILITIES will issue a command to the system and an orderly shutdown will occur. After the shutdown, the node can be reset and restarted, or left off if needed.

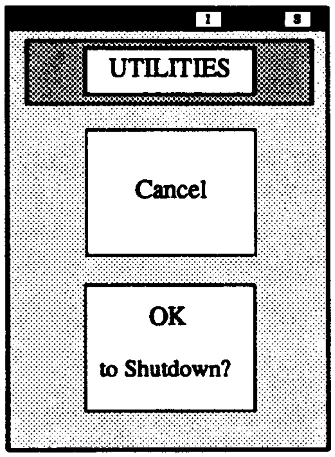


Figure 16. **Utilities** Shutdown Menu

Occasionally, a problem develops with one of the workstation windows. Sometimes an application will become stuck or hang for some reason and the only recourse is to kill the window in which the application is running and restart the window and application. This can be done easily using UTILITIES by moving the cursor to the 'Recreate Window' box and pressing the left mouse button. A new menu will appear as shown in figure 17. The cursor is then moved to the box containing the name of the desired window and the left mouse button is pressed. UTILITIES will then kill and delete the window, if it exists, and then recreate it. The operator can the restart the application in the window.

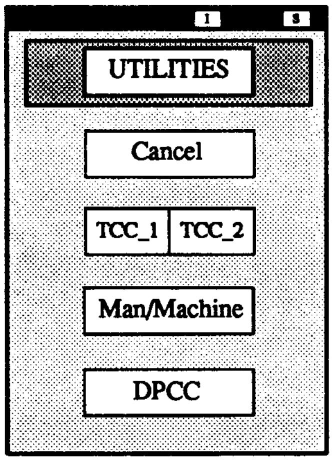


Figure 17. **Utilities** Recreate Menu

There are several regions of the workstation screen where the cursor is not supposed to enter when popping between windows. These regions include the function key menu at the bottom of the AMM application, the digital clock, and the Timer_status window. To prevent the cursor from entering these regions inadvertently, a command can be issued which defines the region as 'off limits' to the cursor. However, because there are several of these regions, executing the command for each such region would be tedious and time-consuming. Instead, UTILITIES has a function for automatically setting each of these regions as 'off limits'. The operator moves the cursor to the 'Set Cursor' box (see figure 15) and presses the left mouse button. The application will then execute the series of commands to set the 'off limits' regions. This does not have to be done very often, usually only after an application has been restarted.

3.0 Application Details

The following sections provide details of each application including directory organization, startup files, and descriptions of software operation.

3.1 Directory Organization

The Trouble Desk workstation directories of interest are:

```
/user/apollo5
/user/apollo5/user_data
/user/apollo5/com
/tcd
/tcd/timer_alarms.dir
/tcd/util
/phaethon
/sys/node_data.b795
```

1. /user/apollo5 -- this is the user directory for node 5 (Trouble Desk workstation). It contains only subdirectories. The directory /user/apollo5/user_data contains the login startup file **startup_dm.191** which contains instructions for starting the application windows and the applications as shown:

```
#
#  /user/apollo5/user_data/startup_dm.191
#
#  start the following processes at login
#
# Startup alarm server to monitor network hardware problems
cps /sys/alarm/alarm_server -nets -disk -hw -p 5 -msg -n alarm_server bell1
#
# set screen blanking timeout to none
cpo /com/sh -c /com/scrto -none
#
```

```
# load fonts and icons
FL F16.B
fl /phaethon/phaethon_icons -i
(954,30)dr;(954,103)dr;(884,30)idf
#
# establish tabs
ts 7 -r
#
# start clock
cps /tcd/digclk_tmp '771' -n clock
#
# start applications
(805,600)dr;(1023,773) cp/com/sh -n boss -F -C 'wd /phaethon;boss'
(550,75)dr;(1000,250) cp -i -c 'y' /com/sh -n phaethon_control -F -C 'wd /phaethon; control'
(20,35)dr;(780,685); cp -i -c 'l' /com/sh -prompt1 AP: -n TCC_1 -F -C 'wd /phaethon'
(30,75)dr;(770,725); cp -i -c 'l' /com/sh -prompt1 AP: -n TCC_2 -F -C 'wd /phaethon'
(0,200)dr;(760,760); cp -i -c 'I' /com/sh -prompt1 AP: -n man_machine -f -c 'wd /tcd;amm'
(0,0)dr;(750,650); cp -i -c 'B' /com/sh -n PC_AT -f -c 'wd /user/apollo5;dpcc -c -dev /dev/com1 /dev/lpt2'
(400,180)dr;(610,530); cp -i -c 'W' /com/sh -n utilities -f -c 'wd /tcd/util;utility'
#
wi boss
(822,465)dr;(1023,760) cp/com/sh -n timer_status -f -c 'wd /tcd/timer_alarms.dir; timer_status'
```

2. /user/apollo5/com -- contains some custom and utility commands:

sleep:	causes a process to halt for a specified time interval, used by process BOSS that oversees Phaethon.
lsyserr:	a system utility which lists system errors
dmpf:	dump a file to the screen
vax:	starts the VT100 emulator and TELNETs to the VAX.

3. /tcd -- This is the main directory for the Trouble Desk applications. It contains all the files for AMM.

4. /tcd/timer_alarms.dir -- this directory contains all files for TIMER_STATUS.

5. /tcd/util -- this directory contains all files for UTILITY.

6. /phaethon -- this directory contains all files for PHAETHON and PHAETHON_CONTROL.

7. /sys/node_data.b795 -- this directory contains the startup file STARTUP.SPM for starting the server processor DSP90_TCC:

```
# STARTUP.SPM, /SYS/SPM, default server process manager startup command file, 10/03/86
#
#
# To make sure line 1 listens to XOFFs
#
cps /com/tctl -line 1 -insync
#
# To startup default printer
#
cps /com/sh -c '/com/prsvr' //p5/sys/printer_config.data -n print_server
#
# Initialize temp file container directories for C programs.
# (/tmp and /usr/tmp are well_known and oft-used temp file directories
# for both UNIX and non-UNIX C language programs.)
cpo /com/sh -c /sys/dm/init_tmp_dirs
#
# Start servers for TCP/IP, must wait while tcp_server initializes
cps /sys/tcp/tcp_server &
/com/sleep 20
cps /sys/tcp/ftp_server &
cps /sys/tcp/telnet_server &
```

3.2 Details of Each Application

3.2.1 AMM

The following FORTRAN source files make up the AMM application:

AMM.FTN
CNVRT.FTN
DISABLE.FTN
DISC.FTN
DISPLAY.FTN
ENABLE.FTN
GET.FTN
GET_ATTENTION.FTN
INITIALIZE.FTN
PUT.FTN
READ_CONSOLE.FTN
READ_MM.FTN
TRACE.FTN
TUTL.FTN

The main routine is AMM.FTN. This routine first calls the routing INIT in the file INITIALIZE.FTN. INIT initializes the window for graphics, loads fonts, draws the function key diagram, sets up the asynchronous port SIO1, initializes the event counter and returns to

AMM. AMM then calls the routine GET_ATTENTION which sends an '@' sign to man/machine in order to stimulate a response (ie: wake up man/machine). At this point, AMM arms the event trigger and waits for an event.

The two events that AMM waits for are input from the keyboard and input from man/machine. If input is received from man/machine, event #1 triggers and AMM calls READ_MM. If input is received from the keyboard, event #2 is triggered and AMM calls READ_CONSOLE. The normal sequence of events is for the operator to type something at the keyboard. This will trigger AMM to call READ_CONSOLE which will process the keyboard data and sends it to man/machine. The response from man/machine will trigger AMM to call READ_MM which will receive the data from man/machine and display it on the screen.

AMM is organized into two parts. One part acts as a man/machine emulator in that commands entered at the keyboard are sent to man/machine which processes them and sends back the result. This part looks and acts just as if a dumb terminal were connected to man/machine. The other part of AMM contains custom commands for submitting series of commands to man/machine and formatting the results for display.

AMM makes the distinction between normal mode and custom mode by examining each command and looking for a match. If no match is found, the command is assumed to be a normal man/machine command and is passed on to man/machine. If a match is found, the command is custom. At this point, if the command requires, the current port number is output to the screen. If the operator wants to use this port, a carriage return is entered. AMM will then call the appropriate subroutine. For example, if the operator wishes to trace a port and enters TRACE either by pressing function key F5 or typing the letters, AMM will output the current port number, wait for a carriage return, then call subroutine TRACE. This subroutine then issues the series of man/machine commands necessary to obtain information regarding the desired port. When the subroutine is finished, it returns to the main routine which waits for a trigger.

The workstation and man/machine communicate via an RS232 connection in which asynchronous port sio3 on the workstation is connected to asynchronous man/machine port T4 on the PBX.

3.2.2 VAX Access

Access to the VAX via the workstation is accomplished by interconnecting the two via a gateway as shown in figure 18. The gateway is required because the workstation communicates using a token ring network and the VAX using Ethernet. A node on the token ring acts as the gateway by providing an interface to Ethernet. This node, called DSP90_TCC, is a server processor which has no disk or display of its own, but shares the Trouble Desk workstation disk. When a TELNET session is originated on the Trouble Desk workstation, it sends a message to DSP90_TCC, telling it to make a connection over the Ethernet to the

VAX. Once the connection is established, the server routes packets between the token ring and ethernet.

Nodes on the Token Ring network are assigned TCP/IP network addresses of the type 200.0.0.x where x represents the unique value for each node. For example, the Trouble Desk node has the address 200.0.0.12. The following example is an excerpt from the file /sys/tcp/hostmap/local.txt, which contains a list of known TCP/IP addresses and corresponding host names:

```
NET : 200.0.0.0 : TOKEN RING
NET : 202.0.0.0 : ETHERNET
;
;GATEWAY TO ETHERNET
;
;HOST : 200.0.0.7,202.0.0.1 : DSP90_TCC
;
;P1
;
HOST : 200.0.0.12 : P5 : DN3500 : AEGIS : TCP/TELNET, TCP/FTP :
;
;SAV80
;
HOST : 202.0.0.2 : SAV80 : VAX : VMS : TCP/TELNET, TCP/FTP :
```

This file is used by TCP/IP to translate a host name to an address before attempting a connection. Thus a user can access another node by requesting a connection using the host name and not have to recall an address.

Any node on the '200' network can directly communicate with any other node on the '200' network when using TCP/IP applications such as TELNET. However, to access a node on another network, the gateway must be used. The VAX is assigned the TCP/IP address 202.0.0.2, thus it is on the '202' network and can only be accessed from the '200' network by going through the gateway node.

The gateway node knows that it has two network interfaces and that the '200' network is associated with the Token Ring and that the '202' network is associated with Ethernet. This is established in the file /sys/node_data.b795/networks:

```
200.0.0.7 on dr0  ;gateway to domain ring
202.0.0.1 on eth0 ;gateway to ethernet
```

When TCP/IP packets arrive at the gateway node from the Token Ring and are addressed to a node on the '202' network, it knows to send these packets over the Ethernet. Conversely, if packets arrive from the Ethernet addressed to a node on the '200' network, it knows to send these over the Token Ring.

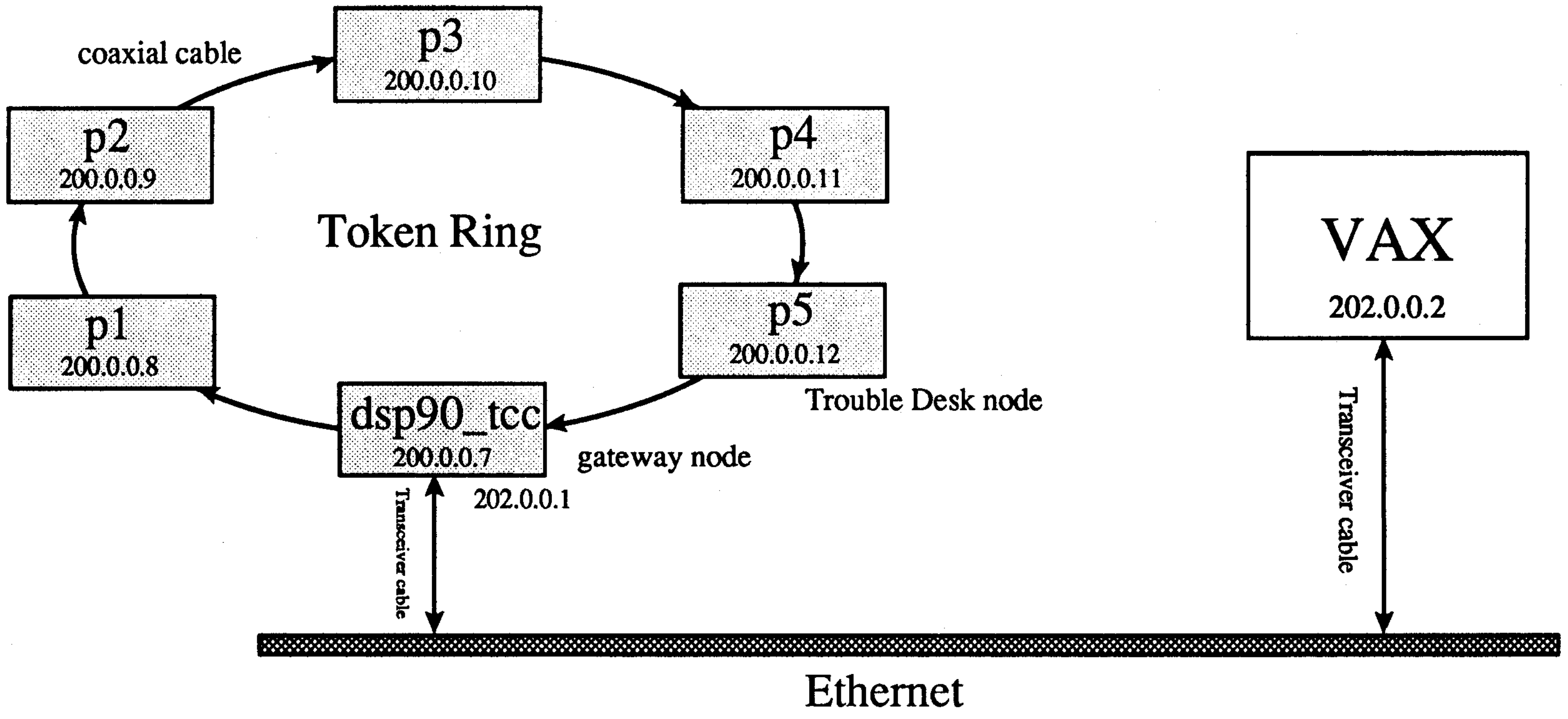

coaxial cable
p3
200.0.0.10
p2
200.0.0.9
p4
200.0.0.11
Token Ring
VAX
202.0.0.2
p1
200.0.0.8
p5
200.0.0.12
Trouble Desk node
dsp90_tcc
200.0.0.7
gateway node
202.0.0.1
Transceiver cable
Transceiver cable
Ethernet

FIGURE 18.

3.2.3 Timer Status

TIMER_STATUS is composed of a single Fortran source file called **TIMER_STATUS.FTN**. It resides is the directory /tcd/timer_alarms.dir.

The process of monitoring errors that occur with the Timers application starts on the VAX. The software on the VAX (called **IBX_TIMERS**) that controls PBX ports will write errors into the log file that is generated each time it runs. This log file is called **IBX_TIMERS.LOG** and resides in the directory **UD:[TCCDATA]**. Once this log file is written, a program called **ERR_PROCESS** starts which examines the log file for errors:

```
$! ERR_PROCESS.COM
$
$
$ DEFINE SYS$PRINT DUMMY
$ SET NOON
$ sear ibx_timers.log "error","%","info:" /output=srch.tmp
$ size=f$file_attribute("srch.tmp","eof")
$ if size .eq. 0 then goto exit
$ file = f$search("ibx_timers.log")
$ TIME=F$TIME()
$ version = f$extract(f$locate(";",file)+1,6,file)
$ open tmp srch.tmp
$ open/wr tmp1 err.tmp
$ start:
$ read/end_of_file=finish tmp record
$ record = f$extract(0,45,record) !purge record to 45 char.
$ wr tmp1 time," LOG-",version," ",record
$ goto start
$ finish:
$ close tmp
$ close tmp1
$ ftp p5
 user apollo5 <password>
 cd /tcd
 put flag.dat flag.dat
 cd /tcd/timer_alarms.dir
 append err.tmp timer_error.dat
 bye
$ del err.tmp;*
$ exit:
$ del srch.tmp;*
$ EXIT
```

If any errors are found, they are extracted and written into the temporary file **ERR.TMP**. This file is then sent to the Trouble Desk workstation by FTP file transfer protocol. The file arrives at the workstation as /tcd/timer_alarms.dir/timer_error.dat. In addition, the VAX sends a control file called **FLAG.DAT** which contains nothing but is used by the workstation as an indicator that an error has occurred. This arrives at the workstation as /tcd/flag.dat.

The application **TIMER_STATUS** on the workstation continuously scans (at 30 second intervals) the directory /tcd for the file **FLAG.DAT**. If the file is found (has therefore just arrived from the VAX), **TIMER_STATUS** will then flash the status window red to indicate to the operator that an error has occurred and the error file has been received. **TIMER_STATUS** will then delete the file /tcd/flag.dat and await the next arrival of the file. At this point, the operator can examine, print, or delete the error file.

TIMER_STATUS begins by initializing its window for graphics and drawing the menu. It then establishes an event that will trigger every 1/4 second. This event is designed for flashing the Status box at a 1/4 second rate when required. At every 120th pass (approximately 30 seconds), **TIMER_STATUS** checks for the existence of the file /tcd/flag.dat. If the file exits, a flag is set for flashing the status box upon returning to the event trigger. The next step is to determine if the cursor has entered the window. If not, the program returns to the event trigger to wait for 1/4 second. If the cursor has entered the window, the program checks to see if the cursor has entered one of the menu boxes. If not, the program returns to the trigger. Otherwise, the cursor has entered one of the box regions and the appropriate action is taken.

If the cursor has entered the 'view' box and the left mouse button is pressed, the following sequence of commands are executed for displaying the error file:

```
(0,0)dr;(730,420)cv /tcd/timer_alarms.dir/timer_error.dat;pb
```

This sequence establishes a window region, creates a view of the file /tcd/timer_alarms.dir/timer_error.dat and displays the bottom portion of the error file. If any errors have been received, the most recent ones will appear at the bottom. To view previous errors, the operator has only to use the 'scroll up' key.

If the cursor has entered the 'clear' box and the left mouse button is pressed, the flag is cleared which will stop the flashing of the status box.

If the cursor has entered the 'delete' box and the left mouse button is pressed, then several events occur. First, the file /tcd/timer_alarms.dir/timer_error.dat is opened, using the FORTRAN OPEN statement. The variable open_stat saves the status of the OPEN command. If the status is 'file not found', then the message 'timer_error.dat does not exist' is displayed and no further action is taken, since the file does not currently exist. If the status is 'concurrency violation', then the file is already open and being viewed in a window. Therefore, it cannot be currently deleted. The message 'timer_error.dat in use' is displayed. If neither of the above conditions are true, the file 'timer_error.dat' is deleted by closing the file with status = 'delete'.

If the cursor has entered the 'print' box and the left mouse button is pressed, the file /tcd/timer_alarms.dir/timer_error.dat is opened as described above. If the file is found, it is queued for printing. If not, the message 'timer_error.dat does not exist' is displayed and the file cannot be queued for printing.

At the conclusion of each of these routines,the program returns to the event trigger and waits. If the file /tcd/flag.dat was found on the previous pass, the status box will now begin flashing.

Distribution:

2643	L. D. Buxton
2645	W. D. Swartz
2647	M. O. Vahle
2647	L. F. Tolendino
2647	M. M. Miller (10)
2648	D. M. Darsey
2648-1	J. P. Sena
3141	S. A. Landenberger (5)
3151	W. I. Klein (3)
3141-1	C. A. Ward (8) for DOE/OSTI
8270	R. C. Dougherty
8524	J. A. Wackerly

DOE-AL/CPID
W. L. Heatherly