Bolt Beranek and Newman Inc.

AD A112887

Report No. 4930

Remote Site Maintenance: Final Report

B. Woznick

DTIC ELECTE APR 1 1982

March 1982

Prepared for:
Naval Electronic Systems Command

DTIC FILE COPY

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

82 04 01 060

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
	AD-A112 887	

4. TITLE (and Subtitle)

REMOTE SITE MAINTENANCE: FINAL REPORT

5. TYPE OF REPORT & PERIOD COVERED

9/1/78 - 9/3/81

6. PERFORMING ORG. REPORT NUMBER

4930

7. AUTHOR(s)

B. Woznick

8. CONTRACT OR GRANT NUMBER(s)

N00039-78-C-0405

9. PERFORMING ORGANIZATION NAME AND ADDRESS

Bolt Beranek and Newman Inc.
10 Moulton Street
Cambridge, MA 02238

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

ARPA Order 3175.17

11. CONTROLLING OFFICE NAME AND ADDRESS

Advanced Research Projects Agency
1400 Wilson Boulevard
Arlington, VA 22209

12. REPORT DATE

March 1982

13. NUMBER OF PAGES

47

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

NAVALEX
Washington, DC 20360

15. SECURITY CLASS. (of this report)

Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution Unlimited

DTIC ELECTE APR 1 1982 S H D

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Remote Site Module, Private Line Interfaces, Network Operations Center, UNIX, Remote Site Maintenance, Computer Networks, Network Operations Center.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This Final Report covers work performed on the Remote Site Module (RSM) subsystems of the Advanced Command and Control Architectural Testbed (ACCAT). The project had two major objectives: to acquire and install the hardware at the NPS, CINCPACFLT, and BBN RSMs, and to perform research into the remote maintenance of network hosts.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Report No. 4930

REMOTE SITE MAINTENANCE: FINAL REPORT

B. Woznick

March 1982

Prepared for:

Naval Electronic Systems Command

DTIC COPY INSPECTED 2

Accession For
NTIS GRA&I
DTIC TAB
Unannounced
Justification
By
Distribution/
Availability Codes
Dist | Avail and/or Special
A

Table of Contents

1 Introduction

1.1 Summary of Objectives and Results

The Remote Site Module (RSM) subsystems of the Advanced Command and Control Architectural Testbed (ACCAT) are currently located at the Naval Ocean Systems Center (NOSC), the Naval Postgraduate School (NPS), Commander-in-Chief Pacific Fleet (CINCPACFLT), and at Bolt Beranek and Newman Inc. (BBN) in Cambridge, Massachusetts. The RSM is a Digital Equipment Corporation PDP-11 running the UNIX operating system (UNIX is a trademark of Bell Laboratories), which is used to access the central processors of the ACCAT system at NOSC. The current project had two major objectives:

- to acquire and install the hardware at the NPS, CINCPACFLT, and BBN RSMs; and
- to perform research into the remote maintenance of network hosts.

System maintenance, in general, is a difficult problem; the maintenance of a distributed system like the ACCAT with its RSMs is even more complex. The Remote Maintenance Experiment is an attempt to extend the concepts developed in the maintenance of the ARPANET to the maintenance of network hosts. In the ARPANET model, remote maintenance consists of three major elements: monitoring, analysis, and correction of system components. This study has shown that the balance of activities shifts as one

moves to host maintenance where the central issues are support and control of host software.

1.2 System Configuration

1.2.1 ACCAT/RSM Remote Maintenance Network

The Remote Site Module System has three types of sites: the Central Operational Site (at NOSC), the Remote Operational Sites (at NPS and CINCPACFLT), and the Central Maintenance Site (at BBN in Cambridge). The RSM is designed to provide access to the ACCAT central facility from a number of locations, by use of a secure subnet of the ARPANET.

The Central Operational Site (COS) has the ACCAT central facility and an RSM. The Central Maintenance Site (CMS) PDP-11/70 is used for a variety of purposes in addition to those associated with remote site maintenance. During normal operations this facility is in an unclassified mode and is not part of the secure subnet. It is, however, designed to allow easy conversion to classified operations on the secure subnet. While it was originally expected that this conversion would be performed on both a scheduled and a demand basis, almost all Remote Maintenance activities are scheduled; currently, the CMS system joins the classified subnet about once each week.

1.2.2 RSM Hardware Configuration Summary

The standard RSM hardware configuration consists of a DEC PDP-11/70 minicomputer with a variety of peripheral devices. The major subsystems of the RSM hardware are:

- o the central processor, memory, and interface units;
- o disk and magnetic tape units;
- o alphanumeric displays;
- o color graphic displays;
- o a high resolution graphic display; and
- o an interface to the ARPANET through a Private Line Interface Unit.

The standard configuration may be found in Appendix A. The various installations are not identical. Some have additional peripherals, such as line printers, while others have more memory, terminals, or disk storage than the standard configuration does.

1.2.3 RSM Software Configuration Summary

The RSM is essentially an intelligent terminal handler which is used to support the ACCAT system at various locations away from the host (for example, at NOSC). A list of the standard software components may be found in Appendix B. In summary, this list includes:

- o the UNIX operating system;
- o UNIX utilities for file management, program development, and routine operation of the system;
- o text processing utilities, including the Ned Editor and the phototypesetting software;
- o the RITA software used for interacting with the NOSC applications;
- o graphics software for the Genisco and Tektronix display units; and
- o network control software.

The list of controlled software has evolved throughout the life of this study and continues to evolve.

1.3 Summary of Results

The initial model for the Remote Maintenance function was based on BBN's experience with maintenance in the ARPANET. In this model, remote maintenance consists of three major elements: monitoring, analysis, and correction of system components. All parts of the system are immediately accessible to the individuals responsible for its maintenance. These same individuals could be thought of as the only direct users of the system; certainly no one else can change any of the ARPANET software.

The ACCAT system operates on a secure subnet of the ARPANET, with access through Private Line Interfaces (PLIs) at each site. Access from BBN is quite limited, and this changes the priorities

and characteristics of the system maintenance problem. There is a substantial user community, and, furthermore, many of the users are engaged in writing programs for use on the system.

Remote Maintenance comprises three different activities. The first, which is closest to the original concept of Remote Maintenance, is concerned with system level questions such as: "Is the file system healthy?", or "Why is the system so slow?", or "Why did it crash?" The second contains questions that more directly concern the user, such as: "What is different about the new version of this program?", or "Exactly what does this command do?", or "Why doesn't this work?" Finally, there are those general activities which are needed to maintain the system, such as hardware maintenance, system upgrades, and operational support. The following general observations may be made about host system Remote Maintenance in this kind of environment.

- o When the maintainer's access to the system is limited, which is a likely model for the real-life tactical situation, the RSM must be provided with tools which allow it to proceed whenever possible, asking for maintenance only when absolutely necessary.
- o The interaction of the users with a time-sharing system, even one with a relatively limited range of activities, requires greater emphasis on mechanisms which assist the users, and which allow them to help one another when formal maintenance is unavailable.
- o Monitoring and diagnosis tools which depend on routine examination of results are less important than tools which can take action when a failure is detected.
- o Diagnostic tools and repair tools should be designed in such a way that a relatively unsophisticated local

operator may use them, reporting the results to the maintainer when necessary.

The experience of this study has shown that Remote Maintenance is a viable and important concept in a multi-host enviroment, but that it cannot yet be completely defined in the same way that the ARPANET remote maintenance function can be.

2 Remote Maintenance

2.1 Definitions of Remote Maintenance

The Remote Maintenance Experiment (RME), which is part of the Remote Site Module system, is research in the maintenance of distributed systems. The first requirement is to understand system maintenance, then to extend this understanding to the multiple processor case, where the processors may be a heterogeneous collection of host systems connected by some network. The RSM project therefore has two components: one is the research into the remote maintenance of systems, the other is the actual doing of system maintenance at the various ACCAT sites. These components interact so that the ACCAT RSMs serve as a testbed for the Remote Maintenance Experiment.

In the following sections various aspects of the remote maintenance problem are discussed. These include:

- o a description of system maintenance;
- o a brief analysis of things which make system maintenance easy;
- o a discussion of multi-system and multi-site maintenance in a research or development environment; and
- o a description of facilities which would simplify remote maintenance.

A fuller discussion of these issues may be found in Quarterly Technical Report No. 19 (November 1980).

In some sense, the issues in the design and implemenation of maintainable systems are the detection and control of change in the system or its environment. One wishes to observe: 1) hardware, software, or data structure failures before they cause irreparable damage, and 2) changes in user behavior before it causes damage or causes the system to run out of some critical resource. One also wishes to have a controlled software system to assure that errors can be effectively identified and repaired.

One can identify at least four classes of maintainable systems.

1) Systems like ARPANET nodes which are fully connected at all times and always under the control of the maintenance organization. The system components can be maintained over the network and generally have rigidly defined functions.

2) Traditional imbedded systems which have well-defined functions, are fully controlled, but have no normal path to the maintenance organization.

3) Systems wherein the operating system software is well-defined, but the user program level is less well defined. Generally, the two levels are maintained by different groups; the operating system is often a release of software from a remote organization, while the user programs are maintained by a local group.

4) Systems in which the operating system and the user programs are not well-defined. These will be excluded from the discussion, since they are best described as experimental.

The Remote Site Modules are in Category 3; the operating system software (UNIX) is maintained centrally, along with many of the important utility programs. On the other hand, all sites

have local programs which perform some functions.

2.2 General Maintenance Services

In the ARPANET model, software maintenance functions can be performed from the Network Operations Center (NOC); it is virtually never necessary to go to a site to deal with a software fault. The NOC can examine the contents of any IMP or TIP, and can cause it to be reloaded from a master copy or from an adjacent IMP or TIP. The NOC also monitors the activity of each element of the network, and has a reasonably complete model of normal activity.

As attractive as this model is, it overlooks some important aspects of the remote maintenance problem. In the first place, the communications sub-net of the ARPANET consists entirely of systems containing no moving parts. In the second place, the system is entirely under control of the NOC; no one else introduces software, and any malfunction which exists is entirely due to the hardware and software which the NOC controls, or to the traffic which affects only those tables controlling the network. Finally, the sub-net has no direct human users.

The Remote Maintenance Experiment is concerned with the general operational maintenance problem for a distributed set of (relatively) homogeneous hosts on the network. This means that

all of the following tasks, which are included in the actual operational maintenance of a system, must be considered:

1) correction of program errors in controlled software, especially the operating system kernel;

2) on-going monitoring of software behavior and data system integrity;

3) analysis of hardware and software error reports;

4) immediate patching of serious software errors;

5) correction of disk directories and similar data structures after crashes or other software malfunctions;

6) coordination of hardware maintenance services, as required;

7) on-going distribution of documentation;

8) on-going operator and user training;

9) site visits (in the case of remote maintenance) for general hand-holding, and special support for demonstrations and major system changes;

10) check out and distribution of software from other sites;

11) analysis of controlled software for weak spots, inconsistencies, and other problems needing correction; and

12) installation of new and improved software, and its subsequent debugging and maintenance.

These tasks form the core of any maintenance system; a few of them (as noted) are slightly modified when the maintainer is not co-located with the system. Many of these tasks can be handled as easily from remote locations, but the importance of documentation and training (among other things) is increased by the physical separation.

2.3 Things That Facilitate Remote Maintenance

The background of system maintenance was explored in the paper "Toward a Theory of Remote System Maintenance," by B. J. Woznick, R. D. Bressler, and A. G. Nemeth, presented at the ACCAT Principal Investigator's Meeting in October 1978. In this paper, the objectives of system maintenance were explored, and the differences between hardware and software maintenance were outlined. It was observed that the objective of hardware maintenance was, generally speaking, to assure the availability of the system, while much software maintenance was focused on system enhancement. In this paper, it was suggested that the first step in making Remote Maintenance easy was to define carefully what the system was, and reduce the amount of system enhancement as much as possible. Within the framework of a program like the ACCAT, however, there is still a significant requirement for system enhancement.

In general, the greater control that the service organization has over the system, the easier its job is. The following items would contribute to this increased control:

1) if the system has a relatively narrowly defined set of functions;

2) if the system was constructed initially by the service organization;

3) if the system can be changed at will by the maintaining organization (i.e., subject only to internal review);

4) if the system internals are completely hidden from the end-user; and

5) if the system users are either not competent to, or not allowed to change the system.

It is quite clear that the ARPANET maintenance organization has all of these advantages. The current RSM system is based on a general time-sharing system, the UNIX operating system. Its appeal lies partly in the fact that the operating system interface itself is fairly uniform from site to site, and partly in the fact that any reasonably clever user can bend it his own way. Therefore, none of these advantages is available to the RSM system maintainer.

An operational system based on the current RSM system would not suffer as much from these problems, even if the operational system were made by simply copying the code which is running at the ACCAT sites. The code copies would most likely be binary images, absolutely controlled from the central location.

On the other hand, the better prepared the service organization is, the more effective it is. The following factors would contribute to this:

1) really good documentation;

2) trouble-shooting staff members available more or less any time of day that the system user wants help; and

3) site representatives within the service organization.

2.4 Multi-system and Multi-site Maintenance in Research Systems

Answers to the following questions partially define the service provided to the remote sites:

1) Is there a central development site which has configuration control over all other sites, including other development sites?

2) How closely should the remote site software track the software in use at the central site?

3) How do changes developed at one site get incorporated into the systems at the other sites?

4) How great an effort should be made to get all sites to run the same software?

The first question has been answered with a qualified "yes" in this project. The central site (BBN) nominally has configuration control over the software, especially the operating system.

The optimum frequency of distribution is quite situation-dependent. In this project, the time between installations has been between 6 and 12 months. They are first incorporated in the main stream of development, then sent out to the sites. In the RSM project there is an attempt to have the same programs running at all the sites (other than BBN). Since it takes a great deal of time to perform an installation over the network, new tools are needed (and will be developed in follow-on projects) to allow incremental updating to take place over the network. Until these become available, it would seem wiser to do major installations

during site visits.

3 System Issues in Remote Maintenance

The operating system of the RSM has been modified to include monitoring software which collects statistics on system activity. A pseudodevice was added to the UNIX kernel to record console activity, and test points were inserted to account for demand on critical system resources. Initially the plan was to forward this data to a Secure Subnet Monitoring Center (SSNMC) for analysis and for later transfer to BBN. During the course of the RME, a comprehensive data display program was developed to show all of the kernel monitoring data from a single site on the Genisco graphics system, which is part of the RSM. An alternate version of this program displays the data from two sites side by side, allowing qualitative comparisons of the activity.

There is a need for further development of the monitoring and analysis system. Crash anticipation software could use information about soft hardware errors (for example, an increase in the frequency of recoverable disk errors) to point to an impending system collapse. More detailed performance analysis reports could also identify system bottlenecks. These reports would include the names of processes or programs which were consuming unnatural amounts of system resources, and, when possible, would suggest changes in the hardware or software to provide better response time.

4 Maintainer Issues in Remote Maintenance

4.1 System Control

4.1.1 An Outline of System Control

The software associated with the RSM consists of several hundred components. Any of these may be undergoing modification, and varying versions may be installed at different locations. The software configuration control system for RSM consists of the following major elements:

- a series of libraries associated with the various releases of the software;
- programs to create, manage, and transfer these libraries;
- programs to install the new releases in such a way that the individual components can be identified as part of that release; and
- programs to verify the configuration of the software installed at a location.

A number of systems have been implemented to deal with system control. Probably the best known is the Source Code Control System of the Programmer's Workbench, which is discussed in "Toward a Theory of Remote System Maintenance." In brief, the SCCS is best adapted to the control of single large modules, while the UNIX system and its major utilities and applications consist of hundreds of related modules and documentation files.

In order to examine this problem in more detail, a pre-prototype system for source and object control has been implemented. It consists of a preliminary data base describing the system and a few commands which manipulate this data base. The commands which construct and manipulate the data base were built from available UNIX commands, notably the stream editor 'sed', the report generator 'awk', and the Version 7 shell (command processor). This pre-prototype system now consists of four main commands (all implemented as shell files):

- o mkcont, which analyzes a complete directory tree and constructs a sorted list of lines consisting of element names and complete pathnames. The list is augmented with the names of corresponding Makefiles and Build.info files;
- o fileloc, which accesses the directory and produces a list of elements matching the input line specification;
- o getfile, which extracts copies of the elements that correspond to the input line specification, generates a control file for the return operations, and places a journal entry in the central control file; and
- o storefile, which uses the local control file to return the elements, making backup copies if they have been modified.

The original pre-prototype data base directory has been mechanically constructed. In some cases this will be inadequate and individual modifications will be needed. The required information will be captured as part of the storefile operation.

A more complete list of the functions which allow programs, subroutines, and documents to be entered, extracted, and identified may be found in Quarterly Technical Report No. 19

(November 1980).

4.1.2 Software Distribution

Continuous distribution of updates is awkward; on the other hand, users wish to have access to the latest, and presumably best, software. In the course of the RME, BBN batched the installations, and, with the single exception of a complete over-the-net installation in December 1979, performed the installations during site visits. Recently, the problem of scheduling routine distributions to the Remote Site Modules over the network has been examined in some detail. There is a detailed discussion in Quarterly Technical Report No. 22 (August 1981).

4.2 Diagnostics

4.2.1 Cross-Network System Debugging

The program debugger on BBN-UNIX has been enhanced so it can communicate with remote processes, possibly residing in a different network, using a protocol that is based on that specified in RFC #643. The cross-network debugging protocol is designed to be robust in the face of message duplication and message loss. A test version of the UNIX Operating System contains code that allows it to be debugged by BBN-UNIX's

enhanced debugger in a controlled crash situation which invokes the "panic" routine of the kernel.

The enhanced debugger consists of three logical modules. These are the command interpreter, the local process debugging module, and the remote process debugging module. The command interpreter provides the interface to the user. The remote process debugging module translates debugging requests into network messages and vice versa. Communication with the net uses the BBN-UNIX Raw Message Interface. Besides adherence to the cross-network debugging protocol, transmitted network messages are prefixed with internet leaders. The local debugging module is the standard UNIX program debugger. A further discussion of the implemenation of this cross-net debugger may be found in Quarterly Technical Report No. 14 (August 1979).

The initial implementation of the cross-net debugger was not installed in the RSM system for two reasons. First, it required more kernel space than was conveniently available in the RSMs; second, the RSM user interface is based on an available UNIX system debugger. The best UNIX debugger ('adb') was not part of the Version 6 system available at that time, but is included in the Version 7 distribution. The usefulness of an XNET based on this debugger is greater than one based on any other available debugger.

4.2.2 A System Dump Analysis Tool

Often a system problem must be diagnosed from no more than a system dump and a vague description of the behavior of the system just before the crash. The UNIX debugger, 'adb', is sufficient for many user-level programs, but is inadequate for crash dump analysis, because:

1) the format of crash dumps differs from that of ordinary UNIX core dumps,

2) the format of the executable file from which UNIX is booted differs from that of ordinary UNIX executable files, and

3) 'adb' is unable to handle C structures symbolically.

Structures are used very heavily in the kernel, and calculating the offset of each member of every structure is a tedious, error-prone process, as is entering such offsets to explore the contents of a structure.

Several modifications and extensions were made to 'adb' to solve these problems. First, options were added to adb which would automatically set up the internal storage map for kernel files and system dumps. Second, predefined formats were devised for all kernel structures; each member of the structure is displayed in the appropriate format (octal, decimal, etc.) and labeled. The formats are defined as macros suitable for use by the general UNIX macro pre-processor ('m4'). The new command, 'udb', invokes 'adb' but pipes the user's input through this

pre-processor. This combination gives the user the full power of 'adb' (since the pre-processor transparently copies its input to its output when it does not find a macro), plus the ability to call on predefined macros in order to print out the contents of a kernel structure.

4.2.3 Graphics Diagnostics

A complete set of diagnostics was developed for the Genisco processors and the associated Conrac displays. These include processor test, memory test, color convergence tests and other display-oriented tests. These have been designed to be useful in network debugging situations to the greatest degree practical, and the results of diagnoses are returned to the terminal or network virtual terminal requesting the tests. Clearly, certain kinds of examination require a human to interpret the results.

4.3 Software and Data System Repair

Many system crashes require only the most mechanical of operations in order to bring the system back up. Such an "auto-reboot" requires a new program for filesystem repair which can be safely used by a naive computer program. It should cure simple problems, but stop when it encounters dangerous situations that necessitate human intervention.

The auto-reboot facility would be engaged automatically when the system performs a controlled crash (a "panic"). It would bring up the system again using a known-good filesystem, fix the other filesystems, and if no serious problems develop, it would enable users to access the system. If the auto-reboot facility detected serious file sytem errors, Remote Maintenance would be requested.

An extended discussion of the characteristics of file systems that allow one to construct the file repair program, along with the methods used by this program, can be found in Quarterly Technical Report No. 18 (August 1980).

5 User Issues in Remote Maintenance

5.1 General

One would think that the "remote" part of remote maintenance makes little difference to the user. This is not always the case: 1) the user may feel that the support is less good; 2) the user cannot appeal directly to the maintainer for help at all times; and 3) the bookkeeping necessary to make things work is significantly more complex.

When a system is being maintained by a local expert, the user develops confidence in both the system and maintainer by observation. When the user is remote from the system, or if both user and maintainer are remote from it, substantial concerns (even if often groundless) arise about the system. If the maintainer's contact is intermittent, these concerns are magnified.

5.2 Problem Reporting

When one says "the UNIX system," one means the operating system itself, the ordinary utilities (over 200 programs), and specialized applications programs which are maintained at NOSC or at other sites. Whenever anything goes wrong, the average user feels that the system has failed. Furthermore, the user often does draw a line between the software and hardware. To provide

adequate service, it is necessary to receive, analyze, and respond to comments and complaints about any of these system components.

An automated problem-reporting system cannot depend simply on the message system and a large number of mailboxes. It should receive comments, complaints, and bugs on any of the programs, categorize the message to the greatest degree possible, permit simple forwarding, produce ticklers to assure that the problem is attended to, and manage the distribution of corrections to all of the sites on the network.

The current system, however, continues to depend on individuals to receive, examine, and diagnose the problem. The specifications for an effective problem-reporting system, and its integration into a user-friendly environment, need substantial additional research.

5.3 Documentation: On-line and Paper

In the past, the computer community has not been known for prompt preparation of documentation, and useful documentation has been rare. The present situation is better in some ways and worse in others. The UNIX system, for example, has fairly good user documentation. Every command has some sort of entry in the user manual, and the more complicated ones have reference manuals

and some sort of tutorials.

In the course of the RSM effort, improvements have been made in the documentation system. Even so, there is a feeling that the on-line documentation system is not adequate. Some of the deficiencies are discussed in Quarterly Technical Report No. 19 (November 1980). Many of the issues raised there warrant further research, since the basic method for learning how to use a complex system is by word of mouth, which appears to be quite inefficient.

In addition, there are several problems with the on-line manual system. For example, the following questions are not easily answered through the ordinary manual system.

- o Where do I find the tutorial on the editor?
- o What is the configuration of this system?
- o What is the assignment of terminal lines?

Furthermore, there is no convenient place to store reference cards for the more important commands, logs of system updates (such as changes in the on-line manual), policy and procedure statements, or general advice. An experimental facility (called 'info') has been developed to deal with these problems. A description of this command and the facilities included may be found in Quarterly Technical Reports No. 21 (May 1981) and No. 22 (August 1981). The entire 'info' command is implemented as a

shell file. This permits rapid development and extensive experimentation with features. The disadvantage is that the resulting command is slower than one would wish.

6 Remote Site Module Acquisition

6.1 Equipment Acquistion and Installation

During the course of the current project, BBN acquired, tested, and installed the complete RSMs for NPS and CINCPACFLT, added Genisco display equipment to the PDP-11/70 at BBN, and obtained additional Genisco equipment for the NOSC RSM. There are summaries of these activities in Quarterly Technical Reports No. 11 (November 1978) and No. 14 (August 1979).

An acceptance test for RSMs was jointly developed by the various sites and BBN, and the tests were performed at NPS during December 1979 and at CINCPACFLT during January 1980.

A brief training course in the UNIX system and UNIX operations was conducted at NPS in connection with the installation, and at CINCPACFLT at the time of the acceptance test.

7 System Operations

7.1 Network Operations

7.1.1 Initial Network Operations

The initial plan for network operations assumed that the BBN Remote Site Module would join the ACCAT secure subnet whenever remote maintenance was needed. This required preparation of a facility which could be used for this purpose at BBN, and the approval of that facility by the cognizant authority.

In the early stages of this project, the NPS RSM also operated in a non-classified mode, and a number of exercises were carried out between the BBN RSM and the NPS RSM. The largest remote maintenance operation of that series was performed in December 1979. The entire operating system and all the controlled utilities were transmitted in source form over the ARPANET and everything was installed from these sources. This exercise, which took approximately one week, provided valuable experience in over-the-net installations. It demonstrated that, while such an operation could be carried out successfully, substantial improvements were needed in the methods for transferring source code and in building the objects. This has been a theme in the development of the RME from the beginning; the methods used have been described in the various Quarterly Technical Reports and have been summarized in this report.

BBN received permission to operate on the ACCAT classified sub-net in late January 1980. The first contact was made over the network on February 15, 1980. When this project was first planned, it was assumed that the BBN RSM would use a different encryption key from the one which was in routine use for the other sites. This meant that the operations were disrupted at the other sites whenever BBN joined the network, and it was hard to agree on the times at which this work could be done. After some experimentation, and a lot of consultation, it was agreed that the other sites could use the "BBN" keys almost all the time. As a result, it became quite easy to join the net and perform whatever operations were necessary. At first, RSM exercises were performed mid-day Eastern time, but later they were transferred to evening hours Eastern time.

As one might expect in a research project, a number of the original plans have changed greatly. Only rarely, for example, has it been necessary for BBN to join the net for an emergency RSM. Routine monitoring of the various systems has not played a large role as yet, although it is clear that host-level monitoring will be part of a fully developed remote maintenance system.

7.1.2 CINCPACFLT Exercise March 1981

During the first two weeks of March 1981, CINCPACFLT conducted an extensive wargaming exercise using the ACCAT system from the Oahu Remote Site Module. This exercise was an unusual test of the entire system, since it ran twenty-four hours per day for about ten days. Although the ACCAT Central site is up around the clock, operational staff and service are normally supplied during the business day only. Special procedures were adopted to assure that problems at the RSMs could be handled promptly. The overall performance of the system was judged adequate, although the stresses placed on it were far beyond those anticipated in the original plan. A further description of this important exercise may be found in Quarterly Technical Report No. 21 (May 1981).

7.2 On-site Operations

7.2.1 System Staff

From 1978 to 1981 BBN had a staff member stationed at NOSC while the remote maintenance system was being developed and proven. Part of the time this position was filled by rotating various members of the BBN technical staff through NOSC.

Although the ACCAT central site at NOSC has a large staff, the RSM operator is relatively isolated, since he is the only

person who has immediate knowledge of the UNIX system. This person could play at least three different roles: operator, remote maintainer (for the other sites), and system consultant. The traditional way to deal with system staffing would be to provide an operator and a system consultant at each site. Part of the RME was to demonstrate the practicality of simplifying the staff at the remote sites. The job description of the COS operator evolved throughout the period in question, but the description below is aimed primarily at the current view of the position.

7.2.2 Hardware Maintenance

The hardware maintenance for the PDP-11/70s and the display hardware (except for the Geniscos at NOSC) is also part of this project. One of the elements in the remote maintenance experiment is the development of procedures for detecting hardware failures and obtaining appropriate service. In particular, extensive diagnostics for the graphics subsystem have been developed for over-the-net testing.

8 Other System Components

8.1 Kernel Modifications

There are major differences between the BBN version of the UNIX kernel and the Version 6 kernel distributed by Western Electric. The BBN version includes:

- o support of the ARPANET;
- o improved interprocess communication;
- o additional device drivers; and
- o support of certain Version 7 facilities, including environment passing and improved accounting methods.

A number of other changes have been made and these are mentioned briefly below.

The Bell version of the software does not support either network protocols or the network interface device. The BBN version supports the ARPANET Network Control Program (NCP) protocols. It also supports, through the Raw Message Interface, a preliminary version of the Transmission Control Protocol which was developed for research rather than production use.

The BBN version of the kernel contains significant improvements in interprocess communication (see BBN Report No. 3949, "A Standard for UNIX Interprocess Communication," J. Haverty, J. Davidson, and R. Rettberg, October 1978). These improvements include:

- o an enhanced version of the standard pipes and the RAND

port mechanism (this offers significant improvements in performance over other versions);

- o the RAND extension system calls: 'gproc' which gets the process table, and 'sfork' which forks a new process with all signals off;
- o the BBN extension system calls: 'await' which allows a process to await activity on a file and 'capac' which gives the capacity of a pipe or other file to receive or distribute characters without blocking; and
- o the required modifications in all character device drivers to support these mechanisms efficiently.

The following device drivers have been modified or added in this version:

- o character devices: DZ, DH;
- o RP06 disk; and
- o TU16 tape.

In addition, the TTY handler has been extensively modified to support flow control (XON/XOFF) and to provide other conveniences which users desire.

A scattering of corrections and enhancements have been made elsewhere in the kernel; the more significant ones are listed below:

- o new timer system calls which provide support for a clock in the user's program;
- o support of non-kernel space buffers;
- o text segment overlay (405 file) support; and

- o buffer aging in the buffered I/O subsystem.

The modifications in support of Version 7 facilities are outlined briefly below ("Version 7 Conversions") and may be found in Quarterly Technical Report No. 20 (February 1980).

8.2 Graphics Software

The graphics programs are of great importance to the RSMs, which can be thought of as intelligent terminal handlers during the war-gaming activities. Substantial work has been performed on both the programs and documentation for the Genisco Programmable Graphics Processor subsystem. In particular, the following tasks were performed:

- o all of the available software has been adapted to the UNIX environment;
- o a comprehensive set of diagnostic and test programs has been prepared;
- o two libraries of subroutines for use with the Genisco system have been developed; and
- o documentation of hardware and software has been written in the form of UNIX User's Manual pages.

The collection, modification, and installation of the software components is described in Quarterly Technical Reports No. 11 (November 1978), No. 14 (August 1979), and No. 17 (May 1980). The relevant pages of greatest interest in the UNIX User's Manual

are: gld3(1), pgp3(1), and gasm3(1), which describe the commands; libsg(3) and libgn(3), which describe the libraries; gnn3(4), which describes the Genisco driver in the operating system; and gensico_h(4) which describes the hardware itself.

8.3 Virtual Terminal Software

The UNIX version running at NOSC at the beginning of the RSM project supported certain virtual terminal operations on the Genisco displays. It was possible to have several jobs running in different terminal windows when this system was used. There was considerable interest in maintaining the facilities provided, and extending them to simpler terminals. On the other hand, there were a number of features found in the BBN version of network UNIX which would not fit into the relatively cramped kernel space of the PDP-11/70 if the virtual terminal functions were implemented as part of the operating system itself. This difficulty was resolved by providing a user-level program, called 'screen', which manages the activities on a terminal screen.

The screen handler works on terminals like the Ann Arbor (or the DEC VT52 or VT100). It can run under any UNIX system which supports pseudotty devices and the 'await' and 'capac' system calls that are found in the BBN network software. 'Screen' allows the user to divide his screen into several rectangular regions, treat each of them as an independent output device, and

move the keyboard logically to any of the windows on the screen.

Details of the operation of the screen handler may be found in the UNIX User's Manual on pages mkscrinit(1) and screen(1).

8.4 Network Bandwidth

While there are a number of programs under development as a part of the ACCAT, the major initial use of the network will be for war games using the Warfare Environment Simulator (WES) program developed under the direction of the Naval Ocean Systems Center (NOSC) for the ACCAT system. The WES program is a complete war-gaming facility; that is, it allows several players to formulate hypotheses about the enemy forces within the game area, issue commands which cause actions to take place, and observe the results of these actions. The player's station normally consists of a Genisco color display showing the game situation on a map, and two Ann Arbor alphanumeric displays, one used as a status board and the other used to enter commands. The contents of the color graphic display (except for static map information) and the status boards are controlled from the ACCAT site. The graphic display is transferred over the network using a private protocol.

The performance of the RSM during WES exercises depends critically upon the performance of the network between the ACCAT

site at NOSC and the RSM. Since the potential bandwidth between ACCAT and CINCPACFLT is limited, it was quite important to analyze the demands which WES put on the network, and to find out if there was any simple way to reduce these demands. A study was undertaken during the summer of 1979 to examine these issues and the results were reported in "Network Bandwidth Study," Woznick et al, at the ACCAT PI meeting at NPS, October 1979.

8.5 Version 7 Conversions

The RSM uses the Version 6 UNIX operating system kernel, modified to support the ARPANET protocols, additional devices, and the monitoring software. The user-level commands available include the standard UNIX commands, specialized commands developed for ACCAT in general or for the remote software maintenance experiment in particular, and other programs developed in the UNIX community. The release of the Version 7 UNIX system in June 1979 made it possible to acquire certain additional useful programs from Western Electric. The two versions of UNIX are quite similar, but there are a number of important differences between the two operating system kernels. These differences required the formulation of a strategy that allows the Version 7 programs to be run with a modified Version 6 kernel.

Because the system is not a Version 7 kernel, certain differences from Version 7 are unavoidable. Most of these, however, are not visible to the programmer using the special library automatically provided with the compiler. The details of the differences, and the general strategy used to convert from the Version 6 to Version 7 environment are described in Quarterly Technical Report No. 20 (February 1981).

8.6 Archival Storage

There is a need within any dynamically modified system to store copies of text in such a way that one can call them back as needed. Such an archival storage system may also be used to handle user data which may be needed again, but which should be removed from system storage until that time. The term "archival storage system" is used here to refer to a system in which the user explicitly identifies files which he wishes to preserve for an extended period of time. This is in contrast to the automatic incremental backup system which UNIX already provides. detailed description of the functions of this system may be found in Quarterly Technical Report No. 22 (August 1981).

Bibliography

BBN Report No. 3984, Combined Quarterly Technical Report No. 11, November 1978.

BBN Report No. 4068, Combined Quarterly Technical Report No. 12, February 1979.

BBN Report No. 4133, Combined Quarterly Technical Report No. 13, May 1979.

BBN Report No. 4184, Combined Quarterly Technical Report No. 14, August 1979.

BBN Report No. 4270, Combined Quarterly Technical Report No. 15, November 1979.

BBN Report No. 4342, Combined Quarterly Technical Report No. 16, February 1980.

BBN Report No. 4399, Combined Quarterly Technical Report No. 17, May 1980.

BBN Report No. 4474, Combined Quarterly Technical Report No. 18, August 1980.

BBN Report No. 4526, Combined Quarterly Technical Report No. 19, November 1980.

BBN Report No. 4609, Combined Quarterly Technical Report No. 20, February 1981.

BBN Report No. 4679, Combined Quarterly Technical Report No. 21, May 1981.

BBN Report No. 4761, Combined Quarterly Technical Report No. 22, August 1981.

"Toward a Definition of Remote System Maintenance," B. J. Woznick, R. D. Bressler, and A. G. Nemeth, ACCAT PI Conference, CINCPACFLT, October 1978.

"A Standard for UNIX Interprocess Communication," J. Haverty, J. Davidson, and R. Rettberg, October 1978.

"Network Bandwidth Study," B. Woznick, et al, ACCAT PI Conference, NPS, October 1979.

"BBN UNIX Mail System Tutorial," E. R. Shienbrood, BBN Report No. 4643, April 1981.

"Some Notes on the BBN-UNIX Network Software," D. Franklin, BBN Report No. 4684, June 1981.

RFC #643, Network Debugging Protocol, E. Mader, July 1974.

Appendix A Remote Site Module Configurations

Minicomputer:

CM70-CVA-LK Computer consisting of:

- 11/70 central processor with memory management,
- 2K bipolar cache memory,
- 128K-byte parity core memory,
- Bootstrap/Diagnostic Loader,
- Line Frequency clock,
- DECwriter II console terminal,
- Two cabinets (one for the CPU, one for core memory),
- RWP06-AA Single access 176-MByte disk control (with 1 drive),
- TWE16-EA 1600/800 bpi magtape drive with control.

RP06-AA additional disk drive (control above used for this drive)

Disk pack, one for each drive

TE16-EE additional tape drive

PF11-C Floating Point Unit

H960-DH Cabinet with nine SU expander boxes

DH11-AE Programmable asynchronous multiplexer for EIA/CCITT terminals or lines

DD11-DK Backpanel

IMP 11-A Interface to ARPANET IMP

Serial Line Printer: Tally 1612 RO

Alphanumeric Video Terminals:

16 Ann Arbor Terminals, modified to support the editor NED

Display Subsystem:

3 GCT-3011 Programmable Graphical Processor

10 GCT-3026-08 Memory Units

3 GCT-3041 Chassis and Power Supply

3 GCT-3031 Video Control

5 GCT-3032-3 Monitor Controls

3 GCT-3038-X Character/Vector Generator

3 GCT-3052 PDP-11 Interfaces

5 GCT-3084 25" Monitors

3 GCT-3071 Keyboards

3 GCT-3073 Joy Sticks

15 GCT-3094 BNC cables

6 GCT-3095 RS232 Cables

3 GCT-3052A Unibus Cable

Graphics Hardcopy Unit:

Versatec 1640 with 8 channel video multiplexer and dma adapter

Joystick Control Subsystem:

PDP11-03-KA computer with 16K of mos memory

DLV11-J 4 serial line interfaces

KEV11 Extended instruction set

Appendix B Controlled Software

Tables of Controlled Software (9/81)

The current lists of controlled software are found in the following tables. The first two tables ("Commands Normally Stored in the Directory /bin" and "Commands Normally Stored in the Directory /usr/bin") reflect the distribution of programs on the Cambridge RSM; in most cases, a command may be moved from the /bin to the /usr/bin directory without affecting the behavior of the system. In addition, it should be observed that this list is dynamic. In particular, the Version 6 shell and its associated programs are considered obsolescent, and will eventually be removed from the list. Such programs are marked with an asterisk (*). Unsupported programs are not generally removed from the controlled directories, and programs which have not yet been accepted as part of the Controlled Software list may be installed in some sites.

Table B-1

Commands Normally Stored in the Directory /bin

[	disable	logout	retrieve
adb	diskcheck	lpr	rm
ar	dpy	ls	rmdir
as	du	make	roll
asgrp	dump	mkdir	send
backup	e	modtty	sh
basename	echo	mount	sh6*
cat	ed	murder	sh7
cc	enable	mv	shvec*
cc7	exit*	ncheck	size
chgrp	false	net	sort
chmod	ftp	net_test	strip
chown	goto*	netproc	stty
chroot	grep	netstat	su
clri	hold	newgrp	sync
cmp	icheck	nm	time
copy	if*	od	tp
cp	install	passwd	true
daemon	iter	path	tsort
date	keepopen	pcc*	umount
dcheck	kill	printf	wall
dd	ld	pwd	who
df	ln	rawstat	write
di	login	reboot	xsum
dir	logo	restor	

Table B-2

Commands Normally Stored in the Directory /usr/bin

a	gasm3	neqn	sleep
aa	genstats	newer	snap
ac	getman	news	sndmsg
add_user	gld3	nice	spell
append	head	nohup	split
ask	help	nroff	splmsg
b2mconv*	host	nroffman	spooler
backupname	ie	p	storepic
banner	indent	pack	stp
bc	info	pagetype	sum
blkcnt	join	pcat	syms
build*	just	pen	tab
c6to7	lastboot	pfe	tail
cbu	lex	pgp3	tally
ce	lint	pload	tar
cid	lnall	pr	tbl
cindex	logdump	printer	tcp_telnet
cleanup	look	prkeys	tee
col	lorder	prman	telnet
comm	m4	prof	tk
compare	makebuild*	ptx	tr
con	makeman	qr	trita
cpall	man	qrita	tty
crash	manix	query	udb
cref	mc	quiz	umountdoc
cref11	mem100	rc	uniq
crpost	mem4023	remind	unmountdoc
dc	mema	restorepic	unmountman
deleteman	memd	rev	unpack
delrem	memmap	rew	unroll
diff	memp	rita	untab
dm	mesg	rmline	upost
edtypo	mkinit	roff	usort
ee	mkkeys	rpl	ux
egrep	mkscrinit	rr	ve
expr	mountdoc	rrn	vroff
f77	mountman	sa	wc
fgrep	msg	sb	wh
file	mvall	sbu	whois
fill	ned	screen	yacc
find	ned-filter	sed	yacc7

Table B-3

Specialized Commands and Databases

THISHOST	getty	mknod	talksh
TIMEZONE	glob	mount	terminals
accton	host_status	mtab	ttys
badblock	init	patchfs	types
cron	logger	prune	umount
diskcheck	meter	rc	update
dtab	mkfs	reboot	wall
/etc/genisco:			
buffdump	grcxtalkc	grmemtstc	grytalkc
dumppgp	grdlytstc	grxtalkc	
/etc/net:			
allhosts	hosts.txt	rmi_inputd	tcpecho
date_hosts	largedaemon	rmi_killd	tcpoff
datedaemon	mailer	rmioff	tcpon
fcpserver	ncpdown	rmion	tcpspec
ftpmain	netoff	sghost	tcptable
ftpsrv	neton	shorttext	tcptable.accat
ftptty	netser	smalldaemon	telserv
host_map	nftpserv	specserver	
host_map.bin	nsend	srvrftp	
hosts.ai	reset	tcp	

Table B-4

Libraries and Library Files

/lib:

as2	liba.a	libpc.a	nc0
crt0.o	libc.a	libs.a	nc1
crt07.o	libgc.a	libsg.a	nc2
error_table	libgn.a	libstr.a	nfc0
fcrt07.o	libl.a	libtcp.a	nfc1
lib7.a	libn.a	liby.a	pc0
libS.a	libp.a	mcrt0.o	pc1
libY.a	libpa.a	mcrt07.o	pc2

/lib/genisco:

READ-ME	floridag	grcxtalkg	restorepicg
clearset	gann	grdlytstg	siggers
clrbuf	geng	grmemtstg	storepicg
colscan	gloadcode	grxtalkg	tgann4
crosshatch	gng	grytalkg	vltshow
dumpbuf	grayscale	mgng	

/lib/rita:

READ-ME	ritamon	tritamon	
errors	ritap	tritap	
aign	eign	libm.a	ratfor
atab	etab	libmp.a	salt
atrun	f77pass1	lint1	suftab
cign	lib.b	lint2	udb_macros
crontab	libF77.a	llib-lc	w2006
ctab	libI77.a	llib-lm	yaccpar
diffh			

/usr/lib/font:

HB	HI	HR

/usr/lib/lex:

ncform

/usr/lib/term:

300	300s-12	600	e8
300-12	37	600-12	qume
300s	450	e6	tn300

/usr/lib/tmac:

tmac.a	tmac.d	tmac.o	tmac.t
tmac.an	tmac.e	tmac.r	
tmac.c	tmac.m	tmac.s	