SOUTHWEST RESEARCH INSTITUTE
Post Office Drawer 28510, 6220 Culebra Road
San Antonio, Texas 78228-0510

DISTRIBUTED SYSTEMS STATUS AND CONTROL

FINAL REPORT

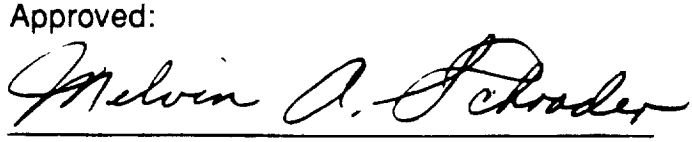
NASA Grant No. NAG 9-387
SwRI Project No. 05-2985

Prepared by:
David Kreidler
David Vickers

Prepared for:
NASA
Johnson Space Center
Houston, Texas

September 27, 1990

Approved:



Melvin A. Schrader, Director
Data Systems Department

TABLE OF CONTENTS

TABLE OF CONTENTS

1.0 INTRODUCTION

The purpose of this document is to present the findings from research performed in the area of Automatic Health Assessment and Control in a Distributed Processing System (NASA Grant Number NAG9-387).

1.1 Statement of Problem

In the future, many computing environments will be characterized by distributed processing in which many different tasks are executed on separate processors (workstations.) The task of monitoring the status of these many different workstations and the local area networks which connect them, detecting and diagnosing problems, and suggesting solutions for the problems is very complex. This status monitoring is necessary, however, in order to effectively maintain the operational integrity of the environment.

Existing centralized environments typically rely upon human experts to identify and suggest solutions to problems. This labor-intensive process becomes dramatically more difficult and error-prone when many different processors (and local area networks) must be analyzed. To accomplish this goal most effectively, automated systems will be required. Such a system will need to gather status information from all workstations and local area networks and assess the health of the environment. These systems will also need to identify and isolate any existing problems and, in some cases, be able to perform remedial actions.

1.2 Direction of Research

The intent of this research has been to investigate concepts for an automated status and control (AS&C) system for a distributed processing environment. System characteristics, data requirements for health assessment, data acquisition methods, system diagnosis methods and control methods have been investigated in an attempt to determine the high-level requirements for a system which can be used to assess the health of a distributed processing system and implement control procedures to maintain an accepted level of health for the system.

A potential concept for automated status and control includes the use of expert system techniques to assess the health of the system, detect and diagnose faults and initiate or recommend actions to correct the faults. Therefore, this research included the investigation of methods by which expert systems have been developed for real-time environments and distributed systems. This investigation focused on the features required by real-time expert systems and the tools available to develop real-time expert systems.

1.3 Basic Results

A distributed processing environment was identified and used as a basis for determining the general characteristics of an AS&C system, the basic functional requirements for which are summarized below:

- Continually collect system data and assess the health of the system;
- Detect system anomalies and diagnose the cause(s); and
- Implement control procedures to recover from detected anomalies.

Besides meeting these basic functional requirements, the AS&C system should be easy to use, flexible and dynamic. The complexity of the AS&C system should not overwhelm the user, yet provide the capability to react to changes in the system structure and components. These requirements can be achieved by providing a graphic user interface that provides the user the following capabilities:

- To quickly identify the area of a problem as well as interactively isolate, diagnose and correct a problem;
- To define the components (objects) of the system and define the system structure in terms of the components; and
- To easily reconfigure the structure of the system that is being monitored.

A final but equally significant requirement of the AS&C is that it must meet the development standards defined by the Hardware Independent Software Development Environment (HISDE) [1].

The remainder of this document describes in detail the environment used as the basis for this research, the characteristics expected of an AS&C system, the basic requirements of data collection, fault detection and control and the areas where further research is required.

2.0 THE ENVIRONMENT

A distributed processing system environment is characterized by independent computers which communicate over one or more networks. The MCC 2.5 Upgrade is an example of such an environment. Unix-based workstations, mainframe computers and other intelligent devices are connected by local area networks (LANs) which carry general purpose and real-time data and information. Various applications at each of the heterogeneous platforms process the data and information from the LANs and/or put data and information on the LANs. The environment can be, and often is, reconfigured from one mission to the next.

An environment that allows for the reconfigurement of numerous heterogeneous devices and applications is both flexible and dynamic, yet, at the same time, is very complex. This makes the task of maintaining the operational integrity of the environment very difficult. For this reason, the MCC 2.5 Upgrade has served as the basis for this research.

3.0 SYSTEM CHARACTERISTICS

For an AS&C system to be implemented in a distributed, real-time environment, it must be flexible and dynamic to adapt to the environment as hardware platforms, configurations and applications change. It must be able to process large quantities of status information and react quickly to changes in the data. It must allow interaction with users responsible for maintaining the integrity of the system. Therefore, expert system, data management, performance and interface issues must be addressed when determining the characteristics of an AS&C system.

3.1 Expert Systems

Real-time computer systems are an expanding marketplace. Applications range from small, simple controllers found in household appliances to large, complex systems for industrial and military purposes [2]. The complexity of real-time systems is also expanding. The amount of data monitored and reported, the rate at which the data must be monitored and the number of factors that must be considered to form a hypothesis or conclusion are increasing, while the time in which this must be done is decreasing. This increasing complexity has led to research in the use of knowledge-based techniques for real-time applications, especially in situations where humans are overwhelmed by information and are unable to interpret the information within the given time constraints. The research has revealed required features of a real-time expert system and tools which can be used to develop them.

3.1.1 Features for Real-time Expert Systems

A system involving expert systems in real-time will be required to have the following features: continuous operation, context focusing, interrupt handling, predictability, temporal reasoning, uncertainty handling, truth maintenance and communication between multiple expert systems [3,4].

3.1.1.1 Continuous Operation

A real-time expert system must be capable of continually monitoring and analyzing data, even when a problem has been detected. It cannot stop to diagnose the problem but must perform the diagnosis in conjunction with continued monitoring and analysis.

3.1.1.2 Context Focusing

Depending upon the current conditions of the system, a real-time expert system should be capable of defining a context in which only specific rules apply. This will limit the number of rules applied at any given time. Also, the real-time expert system should be capable of changing the type or rate of data input based on the current context. For example, the rate of data may have to be increased when a problem is detected, additional or different data may be required during diagnosis, or data may be required less frequently during monitoring due to past data trends.

3.1.1.3 Interrupt Handling

A real-time expert system should be capable of being interrupted to handle a high-priority message, such as a critical condition alarm. The alarm could cause an operator to be alerted, the initiation of recovery procedures, or the modification of the planning goals of the inference engine. This is one way in which context focusing could be achieved.

3.1.1.4 Predictability

Within expert systems, a situation can arise that is so complex that it is very difficult to determine all the factors that led to the situation. For this reason it is difficult to predict the amount of processing required to arrive at every conclusion. A real-time expert system requires that the amount of processing done within a given time constraint be predictable. Therefore, additional heuristics may be required to determine that an acceptable conclusion, not necessarily the optimal, can be found in a reasonable time.

3.1.1.5 Temporal Reasoning

A real-time expert system should be capable of reasoning with or about time. That is, it should allow and handle temporal relationships. To have temporal relationships, there must be a distinction between past and new data. Therefore, the real-time expert system should be capable of associating time with acquired data, and using the time to archive, retrieve and evaluate historical data. The real-time expert system should also be capable of recognizing, based on the times associated with acquired data, the loss or lack of data (which could occur due to a communication or data acquisition problem.)

3.1.1.6 Uncertainty Handling

Uncertainty can originate in a real-time expert system due to untimely data, noisy data, incomplete data and inconsistent data. The real-time expert system should incorporate mechanisms for data validation and reasoning with uncertainty.

3.1.1.7 Truth Maintenance

In a real-time environment, the validity of data decays over time. Therefore, the real-time expert system should have the capability to determine when the data is invalid and retract any assertions based on the data. Also, the real-time expert system should be capable of determining when to replace assertions with new assertions based on changes in data.

3.1.1.8 Communication Between Expert Systems

When an expert system is distributed and there are separate agents cooperating to perform a given task, there should be a mechanism by which the expert systems can communicate. Also, the expert system should be capable of, either by design or dynamically, efficiently using the communication and computing resources. There is a trade-off between communication and computation. When an agent makes a decision, there is either a cost associated with the communication of data to the agent or a cost associated with the computation done by the agent to make the decision without the data. The cost depends upon the limits of the communications bandwidth and the limits of processing time.

3.1.2 Tools

Over a hundred expert system building tools are commercially available, of which only a few are applicable to real-time expert systems. The reasons for this are many and are related to the features expected for real-time expert systems: the tools are difficult to integrate with conventional software; the tools have few or no capabilities for context focusing; the tools do not allow asynchronous input or interrupts; the tools are not fast enough and cannot guarantee response times; and the tools have few or no capabilities for temporal reasoning and cannot be integrated with a real-time clock [3]. There is, however, a commercial tool designed specifically for real-time monitoring and control, G2.

G2 [4] is implemented in Common Lisp and is available on a wide range of general purpose computer systems. G2 has the following features:

- Context focusing, using metaknowledge;
- Truth maintenance and temporal reasoning, by attaching expiration times, used by the inference engine, to each data value; and
- Communication between expert systems via interprocess communication.

Besides the above features, G2 also has a high-level interface and development support. The interface provides an object-oriented graphic representation for system elements and structured natural language for knowledge, models and other information. The development support provides capabilities for rapid prototyping, interfaces with other systems, and simulation for testing the knowledge base.

G2 has many of the features expected for real-time expert system development plus development support. It is currently available by the Gensym Corporation in Cambridge, Massachusetts. It would be a good tool for developing a prototype for an AS&C system in a distributed real-time environment.

3.1.3 Examples of Real-time Expert Systems

Numerous examples of real-time expert systems exist in areas such as aerospace, communications, finance, medicine, robotics and process control. A survey of the area was done by Thomas J. Laffey et. al. [3]. Another survey applicable to this research, done by Daniel L. Dvorak [5] describes expert systems being developed for monitoring and control applications.

One relatively new example is worthy of note because of its similarity to this research. It is the Real-Time Failure Management System (RTFMS) being developed by G.E. Aerospace [6] for complex ground systems. The RTFMS meets the basic functional requirements of an AS&C system as presented in this report. It:

- Collects state-of-health status information from spacecraft equipment, ground equipment and software components;
- Evaluates the status information to detect anomalies; and
- Initiates corrective actions for detected anomalies.

The RTFMS is database driven; databases are used to define the system configuration, command formats, telemetry status data blocks and system status constraints. This design allows the system to be expandable, modifying only the databases in order to modify information concerning the processes or equipment being used. The difference between the RTFMS design and the AS&C concept is that the RTFMS status monitoring is based on the commands and telemetry status data, which are application specific, and the AS&C concept is based on data collected at each workstation, which is not tied directly to any application.

3.2 Data Management

An AS&C system must gather status data elements to assess the health of the system which it is monitoring and to detect and diagnose faults. Optimally, the more data collected and analyzed, the more informed the system is and the better the fault detection and diagnosis. Yet more data requires a longer processing time and, in a distributed system, a larger communication bandwidth. A balance must be achieved between the amount of data collected and the certainty of the fault detection. One approach is to distribute the decision making concerning the value of a data item. For example, a device on a LAN can decide whether or not to send data to a higher level system (i.e. expert system agent) based on whether anything "interesting" or "unusual" is happening. By the same token, the higher level system can choose to poll the device to verify that the device is still present and capable of providing data.

To perform temporal reasoning, the collected data elements must be tagged with a time stamp in some manner when acquired. This time association is required for trend analysis, consistency checking and diagnosis.

The data gathering process should be flexible and dynamic to support context focusing. That is, the data elements collected and the rate at which the data elements are collected should be programmable.

3.3 Performance

A major problem found in implementing an AS&C system in a real-time environment is performance. Basically, the AS&C system must perform in real time; it must be able to collect status data and detect and diagnose faults fast enough to keep up with the data rates and the occurrences of faults. The AS&C must also do this while staying within the resource limits imposed by the computer system. For example, if the AS&C shares resources with the same processes and devices that it is monitoring, it must use these resources in a manner that has a limited or predictable affect on the data.

3.4 Graphical Interface

To facilitate the identification of system anomalies, the AS&C system should have a graphically-oriented interface that provides a comprehensive graphic surveillance of the entire system structure that can be used to quickly ascertain the state-of-health of the system. A graphic depiction of the system structure gives the user an understanding of what is being monitored and allows the user to associate the system structure with the data that is being collected.

The interface should also provide the user the capability to select fault areas of the system, interactively diagnose and isolate the faults and initiate fault resolution and/or recovery procedures. For maintenance purposes, the AS&C should allow the user to modify the structure of the system being monitored, both graphically and physically.

The graphic interface must be supplied at a minimum CPU cost. That is, since graphic interfaces are generally CPU intensive, the interface should be designed in a manner that minimizes its complexity and thus the CPU requirements.

4.0 TYPES OF DATA AVAILABLE

In the MCC 2.5 Upgrade environment, there are numerous sources and, thus, types of data that can be useful in discerning the state-of-health of the system. A great deal of useful data is maintained by the UNIX operating system; other data is maintained by applications. Types of data include workstation information, file system statistics, memory statistics, general I/O statistics, communications statistics, process statistics, user information and exceptions.

4.1 Workstation Information

Workstation information includes general information about the hardware platform: the manufacturer, processor, RAM and disks attached. The data available is summarized below.

Workstation:

- Workstation manufacturer
- Operating system version
- Workstation Executive (WEX) version
- Hardware version
- Hardware identifier
- Mode of the system (development, certification, or operational)
- Workstation fault tolerance status (active, or backup)
- Application software version
- Mission/flight identifier

Processor(s):

- Workstation processor type
- Number of processors on workstation
- Speed of processor
- Type of math co-processors
- Number of math co-processors
- Speed of math co-processor
- Total percent of CPU utilization
- Number of context switches

RAM:

- Total amount of RAM on each CPU
- Speed of RAM (and amount for each type if more than one type)
- Amount of cache RAM by processor
- Speed of cache RAM by processor
- Percent of RAM used
- Cache hits that can be used
- Cache misses
- Number of enters done
- Number of enters tried when already cached
- Long names tried to enter
- Long names tried to look up
- Number of purges of cache
- Maximum memory per process

Disks:

- Number of disks
- Number of inodes per disk
- Device name for each disk
- Number of bytes/sector for each disk
- Interleave for each disk
- Manufacturer for each disk
- Disk identifier for each disk
- Speed of each disk
- Amount of disk in use for each disk
- Largest free space for each disk
- Number of inodes available per disk
- Number of buffers sent to disk by disk
- Number of buffers received from disk by disk
- Device of the root
- List of defective sectors for each disk

4.2 File System Statistics

Data available concerning each file system is summarized below.

- List of currently open files
- Status of files currently open (read, write, read/write, locked, etc.)
- Fundamental file system block size
- Total blocks in file system
- File system id

Other information related to the file system, that concern file I/O, include data pertaining to buffers, streams and maps. This data is defined under general I/O statistics.

4.3 Memory Statistics

There is data available concerning both shared memory and virtual memory. Also, there is data available concerning the address cache associated with virtual memory. This data is summarized below.

Shared Memory:

- Maximum shared memory
- List of shared memory allocations with assigned processes
- Last attach time for shared memory
- Last detach time for shared memory

Virtual Memory:

- Maximum amount of virtual memory
- Virtual memory page size
- Number of pages swapped in
- Number of pages swapped out
- Number of program pages loaded
- Average amount of virtual memory used
- Maximum amount of virtual memory used

- Number of pages paged into memory
- Number of pages paged out of memory
- Total number of page faults
- Remaining blocks of free memory
- 5 second moving average of remaining free blocks
- 30 second moving average of remaining free blocks
- Maximum paging I/O per second before starting swapping
- Maximum sleep time before a process is very swappable
- Maximum number of pages free before clock freezes
- Minimum number of free pages before swapping begins
- Number of pages to try to keep free via a daemon
- Number of pages not to steal

Virtual Address Cache Flush:

- Number of context flushes
- Number of segment flushes
- Number of complete page flushes
- Number of partial page flushes
- Number of nonsupervisor flushes
- Number of region flushes

4.4 General I/O Statistics

Information available concerning general I/O includes buffer, stream, semaphore and map data. This data is summarized below.

Buffers:

- Total buffer read requests
- Times an aged buf was allocated
- Times an "lru buf" was allocated
- Times a process had to sleep for a buffer

Streams:

- Estimate of number of bytes on queue
- Queue state
- Minimum packet size accepted by this module
- Maximum packet size accepted by this module
- Queue high water mark
- Queue low water mark
- State/flags
- Count of procs waiting to do ioctl
- Timeout for sd_vmin
- Amount needed to reach sd_vmin
- Number of pushes done on stream
- Logical OR of all siglist events
- Logical OR of all pollist events
- Data block allocation count
- Data block allocation low
- Data block allocation medium
- Maximum stream message size
- Initial number of stream event cells
- Flag: are there queues to run

- Current item usage count
- Total item usage count
- Maximum item usage count
- Count of allocation failures
- Count of calls to "put" process
- Count of calls to "service" process
- Count of calls to "open" process
- Count of calls to "close" process
- Count of calls to "admin" process

Semaphores:

- Number of entries in semaphore map
- Number of semaphore identifiers
- Number of semaphores
- Status of each semaphore
- Time of completion of last operation on each semaphore
- Number of undo structures in system
- Maximum number of semaphores per id
- Maximum number of operations per "semop" call
- Maximum number of undo entries per process
- Size in bytes of undo structure
- Adjust on exit maximum value

Maps:

- Number of free slots in map
- Number of processes sleeping on map

Other I/O:

- List of tty's on system
- Settings for each tty
- Map of users to tty's
- Number of video outputs
- Types of video outputs
- Status of each device

4.5 Communications Statistics

Information available concerning communication includes network, packet, message and socket data. This data is summarized below.

Communications Information:

- Server node(s) (file server, directory server, etc.)
- List of networks connected to (registered/logged on)
- Number of packets/buffers sent at each level
- Number of packets/buffers received at each level
- Number of communications retries (or collisions) at each level
- Number of nodes on each network
- Maximum total throughput
- Current total throughput
- Throughput by current node and process
- Count of packets with CRC errors
- Count of packets with alignment errors

- Count of discarded packets
- Count of overrun packets
- Number of missed packets
- Number of output packets requeued

Message Queues:

- Maximum number of message queues
- Maximum length of message queues
- Number of messages in message queues
- Length of each message in the queue
- Last process to send a message to each queue
- Last process to receive a message from each queue
- Time last message was sent to each queue
- Time last message was received from queue
- Average queue length for each queue
- Average time occupied for each queue

Socket Information:

- Number of connections for each socket queue
- Maximum number of queued connections
- Connection timed out
- Errors affecting connection

4.6 Process Statistics

Data available concerning the processes and applications running on a system is summarized below.

- List of active programs
- Status of each active program
- Location in memory for each program segment
- Location on disk for each active program segment swapped out
- Total accumulated user time for each active process
- Total accumulated system time for each active process
- Total accumulated user time for children of each active process
- Total accumulated system time for children of each active process
- Total accumulated CPU time for each active process
- Parent ID for each process
- Priority by process
- Value of "nice" for each process
- Start time for each process
- TTY associated with initiation of each process
- Initiating command for each process
- Percent of CPU utilization per active process
- Seconds resident (for scheduling)
- Seconds since last block (sleep)
- Signals pending to this process
- Current signal mask
- Signals being ignored
- Signals being caught by user
- User id
- Unique process id

- Exit status for wait
- Process credentials
- Size of text
- Size of data space
- Copy of stack size
- Current resident set size
- Resident set size before last swap
- Decayed percent of CPU time for this process
- Number of swap "vnode" locks held
- Bit mask summarizing nonempty queues

4.7 User Information

Data available concerning the users of a system is summarized below.

- Users logged onto system
- Current environment by user
- Logon times for users
- Absolute limit on disk blocks allocated
- Preferred limit on disk blocks allocated
- Current number of blocks allocated
- Maximum number of allocated files
- Preferred limit on number of files
- Current number of allocated files
- Time limit for excessive disk use
- Time limit for excessive files

4.8 Exceptions

Data available concerning system exceptions is summarized below.

- Memory errors (soft or hard)
- Location of Memory errors
- Number of failures for memory location (for soft errors)
- Disk errors (soft or hard)
- Location of disk failures
- Number of failures for each disk location (for soft errors)
- Self test errors
- Application errors

5.0 DATA COLLECTION METHODS

An AS&C system should be capable of collecting data from the several sources listed above. It should take advantage of any data which is already collected and which would be useful in determining the status of the system. It should also be capable of collecting varying amounts of information from the systems being scrutinized depending on the immediate requirements. It should use existing functions to facilitate the collection of data, as well as providing any additional data collection capabilities needed.

5.1 Data Already Collected

The UNIX operating system, in current standard implementations, collects or maintains many forms of data which would be very useful to the AS&C system. Some of the data collected by the UNIX systems is available from standard UNIX commands and can be easily retrieved. Other data collected by the UNIX system is available from the remote health and status program currently being implemented. However, some of the data which the UNIX operating system collects or maintains could be collected only through specific functions written for the AS&C system and other data may need to be collected in a similar manner for performance purposes.

5.1.1 Data Available Through Existing UNIX Commands

Much of the data useful to the AS&C system collected by the UNIX operating system is available directly through user commands or through programmatic function calls.

The "ps" command provides a wide variety of information. The status of processes currently scheduled is available using the "ps" command. It also provides considerable information concerning the status of memory use and the status of swap space. It indicates the total amount of time each process has executed. It can be used to determine the pedigree of any of the scheduled processes. It can provide the environment in which each process is running, as well as providing indications of the use of other resources by each process.

Other existing UNIX commands can provide data useful to the AS&C system. The disk space available can be determined using the "df" command and the usage and limits for each user are available via the "quota" command. Many of the environmental parameters active in a system are available using the "env" command. Information concerning users may be available using the "finger" command. The "ipcs" commands provides information related to the active message queues, the shared memory segments and the active semaphores. The "lpq" command provides information about print spooling jobs. The "users" and "who" commands indicate the users currently on the system and the "w" command indicates who is on and what they are doing. The "dkinfo" command can be used to collect information about specific disks. Input/Output statistics are available via the "iostat" command. If the profiling option has been generated into the operating system, then the

operating system profiles may be obtained using the "kgmon" command. Some network status information is available using the "netstat" command. If the file system is based on the network file system, then the "nfsstat" command can be used to retrieve some information about the status of the file system. Some process table information can be obtained using the "pstat" command and some virtual memory statistics can be retrieved using the "vmstat" command.

While much of the data collected by the UNIX operating system is available to the user via commands, the AS&C system may need to directly access that data in order to meet the necessary performance requirements.

5.1.2 Data Collected by Remote Health and Status

The remote health and status program is currently being implemented for the NASA workstation environment. It allows much of the data collected by the unix operating system to be transmitted over a Local Area Network (LAN) to a set of data collection sites. It also specifically collects other data that will be useful to the AS&C system. The data collected by the remote health and status program is defined in the Workstation Health and Status Level B/C Requirements Document [7], prepared by Ford Aerospace Corporation for NASA-JSC.

While much of the data collected for the remote health and status program is the same data as that needed for the AS&C system, there may need to be separate systems to provide the data in order for the AS&C system to be as flexible as needed and in order for it to meet the necessary performance requirements. Some of the same functions may be used to retrieve the data, but the selection of when to retrieve the data and exactly what data is retrieved may need to be controlled by the AS&C system.

5.1.3 Other Data Available from UNIX

The UNIX operating system collects data which would be useful to the AS&C system but which is not currently accessible through either user commands or through the remote health and status program. Much of that data is configuration dependent or hardware dependent. Some of the otherwise unaccessible data may need to be collected, especially during the diagnostic phases of the AS&C systems functions. In some situations, the AS&C system will, therefore, need to be capable of determining what specific hardware is involved and capable of using specific diagnostics, data access routines, and control responses based on the specific hardware involved.

5.2 Data Available from Existing Tests

There are several types of existing tests which can be used by the AS&C system. These tests include on-line self tests, off-line self tests and boot-up tests.

5.2.1 On-line Self Tests

The most useful of the existing diagnostic tests for the purposes of the AS&C system are the on-line self tests. Their major advantage is that the AS&C system can initiate the tests without having to bring down the system in question.

In general, however, because these tests are run with the system still executing, they are generally not as helpful in the diagnostic process as the ones run at boot-up or off-line. There are several areas where tests either exist or can be developed where on-line tests can be very useful and effective. Some of those areas include math co-processor tests, memory tests (while ensuring nothing else is in the areas tested while the tests are on-going), and disks tests (if the areas being tested are locked and saved). If used on a periodic basis, these on-line tests can, with little degradation of the systems performance, provide rapid detection of significant failures and indications of gradual failures.

5.2.2 Boot-up Tests

When a system is initially booted up, a very thorough test is usually done of the hardware. If the boot-up test finds some faults but does manage to boot up, then some indication of the discovered fault may still be accessible to the AS&C system when it is executed. However, for many boot-up tests, either the system will not get far enough to run the AS&C system, or there will be no record of the faults left.

5.2.3 Off-line Test

The off-line tests have many of the same problems as the boot-up tests. In fact, the off-line tests may be a little worse, since the boot-up tests at least intend to end up with a normal running system. The off-line tests are often run outside of the normal UNIX environment and very well may not be able to save their results nor to communicate with other processors. The major advantage of the off-line tests is that they can often be much more thorough than the on-line tests, since they do not have to contend with the system running at the same time. Overall, it is possible some useful data could be collected from off-line tests, but it is not likely.

5.3 Other Data Collection Methods

There are several other methods which may be used in the collection of data about the status of nodes on a network. Data can be collected by the monitored machine and those machines polled for the data periodically. The monitored machine could periodically transmit messages to demonstrate that it is still functioning correctly. Finally, the monitoring machine could monitor all of the messages on a LAN to determine the status of all the machines using the LAN.

5.3.1 Polled Data Collection

The program currently being developed to check the health and status of workstations has the capability to operate in a polled mode. The AS&C system will also need to be able to operate in a polled mode. In a polled mode the system will collect data during normal operation of the UNIX operating system and have that data available when the monitoring system sends a message requesting the data.

There are several capabilities which a polled system should have. In order to minimize the resources consumed by the AS&C system, while still providing the capability to detect problems and diagnose them, the system should contain considerable flexibility. The AS&C system should be capable of collecting only data specifically selected. It should be capable of collecting that data at multiple data collection rates. It should be capable of varying the data collected and the data collection rate as necessary.

5.3.2 Event Data Collection

The health and status program also can operate in an event mode. In an event mode, the workstation being monitored also analyzes some of the data collected. If certain events, thresholds, or combination of events occurs, then the monitored station sends a message to the monitoring station indicating what has occurred.

The event reporting mode has the advantage of much more rapid response to the problems indicated by the events reported. The major drawbacks of the event reporting mode is the additional local machine resources used in the analysis and the possibility that a system with a problem will not be as able to detect its problems as a separate machine could.

For some specific events, the AS&C system will need to be able to act in an event mode. However, those events should consist primarily of catastrophic or fatal occurrences. The use of local resources to analyze the data being collected on the monitored machines should be kept to a minimum.

5.3.3 Keep Alive Signals

One mechanism which may be used between or instead of polling the monitored machines is to have them send out messages periodically so that the monitoring system can be sure that the machines being monitored are alive and operating. While this mode is very similar to the polled mode, it requires different capabilities to be working in the monitoring system and in the system being monitored.

5.3.4 LAN Monitoring

A mechanism which can be used in addition to or instead of the other data collection mechanisms is to have the monitoring system monitor all of the messages on the LAN. If the AS&C system monitors all of the LAN messages, then it should have good knowledge of the status of the system as a whole.

It should be possible, given a good knowledge of the system as a whole, to determine when one of the machines on the network is not responding correctly.

There are several levels at which a monitoring system could operate. It could just check to ensure that the contents of the messages sent were reasonable. It could monitor certain basic system statuses and ensure that all of the messages reflect the status of the system. Finally, the monitoring system could collect data regarding the normal operations of the system and attempt to determine if the system is operating within normal ranges.

6.0 PROBLEM DETECTION

There are several ways that potential problems can be recognized. They range from relatively simple methods, to determine and implement, to complex combinations and series of events. Some of the symptoms may be easily directly related to the cause of the problem. Other symptoms may be only secondary indications of problems which cannot be viewed directly. The AS&C system will need to be able to easily include the information necessary to detect the simple problems while, providing the capability to collect the complex information necessary to detect selected difficult problems.

There are three basic categories of information to be collected. The information may be in the form of the occurrence of a simple event. The information may consist of some sort of threshold violation. Finally, the information may be in the form of some complex set or series of the simple events.

6.1 Node Status

The basis of all of the problem detection methods must be a knowledge of the status of the node or LAN being monitored. All decisions concerning problem detection for a specific node must be based on the requirements and characteristics of that particular node. A knowledge of which processes could be executing in a particular node is necessary. A general knowledge of the resource requirements for each of the processes will facilitate the analysis of nodes on which they are executing.

There are also many static parameters defining nodes which will be required in order to determine whether or not the variable parameters are in a problem range. For example, the AS&C system must have the memory, swap, and disk sizes available for each node in order to determine if the current level of use is a problem or not.

All of the problem detection capabilities listed below must be capable of being modified by the status of the nodes being monitored and by other status information as it is determined.

6.2 Simple Events

There are many simple events which should be recognizable by the AS&C system as an indication of a problem. For example, almost every error report which causes a reboot of the system on which it occurs should cause an alarm to be raised at the very least. Any error report which indicated that important user applications programs were being abnormally aborted should also cause an alarm. Errors which occur occasionally and leave a system in an unusable state should cause that system to be reset.

Simple events may be detected either at the system being monitored or at the monitoring system. However, the major characteristic of a simple event is that it consist of a single occurrence of an isolated nature which is

recognized as implying that some fault has occurred or that some problem exists.

6.3 Thresholds

There are several different kinds of thresholds which may be recognized by the AS&C system as potential problems. They vary from a parameter going above or below a particular value, through a parameter changing by more or less than a given value, to the value of a parameter being above or below the value of some other parameter. Overall, there are many ways that thresholds can be used to detect potential problems in a workstation or in a network.

6.3.1 Simple Thresholds

There are several examples of potential problems which could be detected using simple thresholds. For example, if the amount of free space available to nonsuper-users on the disk of a file server falls below a given level (e.g. the amount allocated for temporary files), then the system could indicate that a potential problem exists. The problem possibly could be corrected by removing any files which are no longer needed from the disk. Such a potential problem then could be corrected even before it had an adverse impact on the operations overall. Similarly, if the amount of memory available or the amount of swap space available became similarly restricted, the problem potentially could be detected in time for real impacts to be avoided.

6.3.2 Variations on Thresholds

There are also many other types of thresholds that may be used to detect potential problems. Some of the other types include: thresholds on changes in values, thresholds on variation from decaying averages, as well as variations from other parameters or from functions of other parameters. As more complexity is added to the threshold calculation, more time and other resources are used for the check, but the more discriminating the calculation is as a test for potential problems. As the discrimination of the test is increased, the proportion of potential problems which are actually problems should increase and, therefore, the usefulness of the AS&C system should increase.

An example of a more complex threshold would be one based on a weighted average of the rate of allocation of a shared resource versus the rate of deallocation of the resource, modified by the percentage of the resource which is unallocated. A threshold such as this one might give a much better picture of whether a problem is really imminent or whether the situation is, even if abnormal, not a problem.

Other potential problems could, perhaps, be best detected by any one of a number of thresholds. For example, a particular disk drive might be singled out for additional testing if the number of disk read errors exceeded some threshold, the number of errors per read exceeded a threshold, the number of errors per read increased by some threshold, or if

the number of read errors for a particular platter of the drive exceeded the number of errors for all other platters by some threshold. Any of these tests could provide an indication that a drive was experiencing a problem.

One of the areas requiring further research is determining a set of thresholds which will allow the AS&C system to distinguish between a potential problem and a normal situation. The health and status program currently being implemented could be used as a tool to gather information to be used to determine reasonable thresholds.

6.4 Complex Mechanisms

In addition to any particular event, whether it is a simple event or a threshold event, being the basis for detecting potential problems, events in various combinations could also be used. Some of the ways that combinations of events can be used include sets of events, series of events, and time dependent events. Of course, combinations of sets, series and time dependencies are also possible.

6.4.1 Sets of Events

The use of sets of events will allow the AS&C system to further discriminate between situations which are potential problems and those which are not. Through sets of events, the node and LAN status information referenced above can be coordinated with the simple events and the threshold violations. In many ways, the more complex thresholds mentioned above also could be considered as sets of thresholds. All of the different types of information can be coordinated to refine the definitions of potential problems in order to increase the effectiveness of the AS&C system.

As mentioned above, the complex threshold examples could be considered as sets of thresholds. Other examples of sets of events might include: a specific error message from a particular process on a particular node; a specific set of threshold violations while running a particular process on any node; or the failure of a node to communicate at the appropriate time while in a specific mode and executing one of several processes. Many other examples fit into the same basic mold of sets of statuses, events, or conditions.

6.4.2 Series of Sets

An additional complexity that may need to be considered in the AS&C system is that not all the evidence of a problem will necessarily occur at the same time. It will be necessary to be able to detect the occurrence of a series of sets of events instead of just a single set of events.

Once again, some of the status or specific event information referred to previously can be included in this category. There is not much difference between saying that something happens while a process is running and saying that it happened after the process was started and before the process

terminated. While the AS&C system could be set up to keep status information to indicate these situations, in many cases the definition of the potential problem situation is easier in terms of a sequence of conditions than in a set of conditions which include status information. In particular, if the series includes multiple events occurring at different statuses, the situation will be much easier to describe as a series of conditions.

6.4.3 Time Dependencies

One final additional consideration in determining whether a situation indicates a potential problem or not is a time dependent relationship between the events. Even in a real-time environment, these time dependencies will usually take the form of some condition needing to occur within a certain period of time after some other condition, or some set of events not occurring within some time of each other. However, in a real-time oriented system, an AS&C system should be able to discriminate problem conditions based on particular time dependencies between the conditions or events of interest.

7.0 CONTROL STRATEGIES

Once a potential problem has been recognized, there are several different strategies that may be used to respond to the situation. The responses to particular situations may also change as experience is gathered as to the relationship between symptoms and the actual problems. Some potential problem situations will have normal explanations. Other problems should cause alarms to be generated. As the AS&C system gains capability, it may recommend what actions to take in particular situations or even initiate a sequence of actions based on the situation.

7.1 Generate Alarms

The simplest action the AS&C system may need to execute is to generate an alarm. Alarms should be generated only when there is a significant chance that a real problem situation exists. If too many false alarms are generated, then no one will pay any attention to the real ones. Depending on the nature of the alarm condition, the alarm may need to be displayed in a variety of locations. Alarms with no other action or suggested action should occur only when the symptoms could have many causes or when they have no known causes.

7.2 Recommend Actions

For those sets of symptoms which are recognizable as probably springing from a limited set of possible causes, the AS&C system should be capable of at least suggesting a course of action. The suggested actions could lead to a solution to the problem, or they could narrow down the cause of the problem so that effective action can be taken.

If actions are only recommended, then the operator would be required to initiate any actions necessary, whether they were the recommended actions or some other actions.

7.3 Initiate Further Diagnostics

The next level of activity which could be taken by the AS&C system would be to actually initiate actions on its own. The type of actions which initially would be most likely to be executed by the AS&C system would be further diagnostics. This would be especially true in situations where the diagnostics could be run without a major impact on the system operations.

One of the ways the AS&C system could easily initiate further diagnostics is to let it retrieve more of the data being collected by the stations being monitored. Another step the AS&C system could initiate is to have the monitored systems collect even more data than they normally collect. Another possibility is for the AS&C system to cause the recently collected

data to be archived where it will not be destroyed and can be used for analysis during the problem or later. The post problem analysis will be necessary to ensure that the AS&C system continues to improve in its forecasting and diagnostic abilities.

After an initial period of analysis and learning, the AS&C system operators may also develop specific sets of diagnostic actions which the AS&C system should take in certain given situations. These actions would be designed to provide the operators the information needed to diagnose and cure the problem.

7.4 Operator Confirmed Sequences

Once the AS&C system has gotten to the point that it can recognize particular problem situations with a fairly high degree of accuracy then it may begin to execute some of the actions required to solve the problems found. Initially, however, there is likely to be a confirmation required by a human before the system is allowed to actually initiate a series of actions. Indeed, there may be requirements for the possibility of humans to specifically confirm several of the steps in such a sequence.

These types of sequences should be put in place for problems that occur relatively frequently, and are almost always solved by using a very similar set of actions. For example, if a disk occasionally gets too full and there is a class of files which may be deleted if needed, the system may indicate to the operators that the disk is too full and ask if it is all right to remove the set of files. If the situation is normal then the operator could tell the system to delete the files. If, however for some reason one or more of the files should not be deleted, then the operator could keep the system from deleting them.

7.5 Automatic Sequence Initiation

Once the AS&C system has been in operation for a while and sufficient confidence has been gained in the systems ability to recognize a subset of the problems accurately and reliably, the AS&C system may be allowed to initiate sequences of actions without the confirmation of a human. This step will only occur for problems which can not only be recognized, but also can reliably be solved with a particular set of actions and usually only for those problems which require a solution in a short enough time span that getting human confirmation of the sequence could hinder system performance.

Other situations where the AS&C system may take actions without human intervention might include failover to backup units or periodic tasks in the system maintenance or system diagnostic areas.

8.0 DIRECTIONS FOR FURTHER RESEARCH

Sources of data, methods for collecting data, methods for determining problems and strategies for control have been identified, but the knowledge base necessary to define the relationships among the data collected, the anomalies that occur and the control procedures necessary to recover from the anomalies, remain to be developed. Also, diagnostics that can be used to isolate faults should be determined, defined and combined with the knowledge of fault detection and recovery, to develop a prototype of an AS&C system.

8.1 Knowledge Base Development

There are several ways that information can be collected for the knowledge base needed for the AS&C system. Some knowledge will be available from vendors and site maintenance offices. Additional knowledge may become available in technical or trade journals. Some data can be obtained by instrumenting a network and inserting faults into nodes on the network. Finally, knowledge can be collected by instrumenting real active nodes and networks and collecting data as actual problems occur.

8.1.1 Existing Systems Knowledge

While there is not a long basis of experience with large networks of UNIX based workstations, there is some experience. The major problem with trying to use the existing experience is that almost none of it relates to the automatic recognition of problems.

Most of the current experience in the vendor areas consist of technicians or maintenance engineers who are called in when a system is not working and are tasked to fix the system. The method used to fix the machines most often consist primarily of replacing boards or other parts until the system is fixed. There is relatively little diagnostic type knowledge available and almost no problem forecasting type knowledge available.

The system administrators and operators are another source of existing knowledge needed for the AS&C system. However, there is relatively little incentive for them to pay attention to preliminary symptoms and very little incentive for them to note the symptoms and resultant problems or to organize the experience base in any way.

Overall, although there is some experience with faults on UNIX based networks, there is very little existing knowledge that can be used to build the knowledge base needed for the AS&C system.

8.1.2 Fault Insertion

One way to increase the pace at which knowledge needed for the AS&C system can be gathered is to insert faults into systems and document the results. There are several kinds of faults which might be inserted into systems to provide additional data. It would, however, be necessary to instrument the

network and the node into which the faults were to be inserted in order to collect the necessary data.

The health and status program currently being developed could be used to instrument a network for fault insertion. It could be specifically directed to collect data which might have a bearing on the fault to be inserted. It also could be instructed to archive the data for those faults which are fatal. Because the health and status program operates in a network environment, it can also be used to record the affects on the LAN and the other nodes on the network of the faults inserted.

There are several drawbacks to using fault insertion to collect the needed knowledge. One of the problems is that at least one node, and potentially a whole network, must be dedicated to the task. When inserted faults cause a node or network to fail, no other work can be going on in those facilities. Another problem is that the fault insertion method of gathering knowledge will not provide real life experience in the detection, diagnosing and solving of problems. Fault insertion, however, does have some advantages including being able to collect data on many more problems in a much shorter time, and it can provide some insight to a selected and limited set of problems.

8.1.3 Network Monitoring

One of the most promising areas of research is the instrumentation of networks which are actually being used so that real-life data can be collected. The data collection system must run continuously, and whenever problems occur, additional data must be collected so that the collected data can be correlated with the problems and the solutions.

Once again the health and status program is a candidate as a monitoring program. It has the advantage of already being under development, of being UNIX network based, and of being relatively flexible. The amount of data collected and archived will need to be carefully defined so that the data needed to generate the knowledge base is collected and saved with a minimum of interference with the normal operations of the nodes and the network.

In addition to the data collected by the health and status program, procedures will need to be instituted to save the collected data whenever a potential problem situation exists. The process used to determine whether there really is a problem and what the problem was will also need to be documented. As the knowledge base grows, the data collected and the procedures associated with the knowledge acquisition can be refined.

There are at least two networks that are good candidates for being instrumented. The UNIX based network at SwRI consisting of SUNs and Masscomps would be a suitable place to begin the data collection. The initial iterations of deciding what data to collect, how often to collect the data, how often to archive the data, as well as, what to do in potential problem situations and how to even recognize those situations could be done on the SwRI network. The second network which is a good candidate for additional data collection is the TFCR network. Once the

initial data collection decisions are made and the procedures are defined, additional data can be collected by setting up the procedures and data collection options on the nodes in the TFCR network. The additional network and nodes should significantly increase the rate of knowledge acquisition.

8.2 Diagnostics

It is not possible, without overloading communication and CPU resources, to collect all the data all the time. Therefore, the minimum data required to detect an anomaly should be acquired. This will allow an anomaly to be detected, but not necessarily identified, as the anomaly could be due to many possible reasons, any of which could be suggested by the data acquired. Therefore, diagnostics will be required to isolate the source of the anomaly. The diagnostic may do nothing more than acquire additional data, beyond the minimum requirements, that is necessary to isolate the anomaly. Or, the diagnostic may execute a test or series of tests which confirm or deny a possible source of the anomaly. The diagnostics can be identified during the development of the knowledge base.

8.3 Prototype

During the development of the knowledge base, a prototype can be developed to include a user interface, data acquisition and anomaly detection for an initial set of anomalies. The prototype can be used to show proof of concept, test the user interface, test the structure of the knowledge base, test the information in the knowledge base and serve as a basis for further development of the knowledge base, diagnostics and the AS&C.

9.0 SUMMARY

An AS&C system has the basic requirements of monitoring the health of the target system, detecting and diagnosing anomalies and recovering from the anomalies. To realize these requirements in a real-time environment, the AS&C will have to be flexible, dynamic and adaptive. Issues such as data management, performance and user interface will have to be addressed. Expert system techniques may be utilized, but the techniques will have to be extended with features necessary for a real-time environment.

In the Unix environment, there are many sources of data available which can be used to detect anomalies. Methods for collecting data, techniques for detecting anomalies and strategies for recovering from anomalies exist, but the relationship between acquired data and a system anomaly does not exist. The next step in this research is the determination of the relationships between the data collected, anomalies that occur and control procedures necessary to recover from the anomalies. This knowledge will have to be obtained from vendor, maintenance personnel and operator experience as well as from data collected from fault induced testing and data collected continuously from existing networked, workstation environments. Along with this task, diagnostics that can be used to isolate faults should be determined and defined. These diagnostics, along with the knowledge of fault detection and recovery, can be used to develop a prototype of an AS&C system.

REFERENCES

1. Steven W. Dellenback PhD., Mark D. Collier, Andrew K. Knipp, Hardware Independent Software Environment Standards Evaluation, Final Report, Project No. 05-2217, Southwest Research Institute, December 1988.

2. M. Lattimer Wright, Milton W. Green, Gudrun Fiegl and Perry F. Cross, "An Expert System for Real-Time Control," IEEE Software 3 No. 2 (March 1986), pp. 16-24.

3. Thomas J. Laffey, Preston A. Cox, James L. Schmidt, Simon M. Kao, and Jackson Y. Read, "Real-Time Knowledge-Based Systems," AI Magazine 9, No. 1 (Spring 1988), pp. 27-45.

4. Robert Moore, Gregory Stanley and Rick Smith, "The G2 Real-Time Expert System," AIAA Computers In Aerospace VII Conference, Monterey, California, 1989, pp. 1123-1129.

5. Daniel L. Dvorak, "Expert Systems For Monitoring And Control," Report AI 87-55, Artificial Intelligence Laboratory, The University of Texas at Austin, 1987.

6. Michael E. Cortese, "Real-Time Failure Management System," AIAA Computers In Aerospace VII Conference, Monterey, California, 1989, pp. 767-775.

7. W.L. Feindel, Workstation Health and Status Level B/C Requirements Document, Contract NAS 9-15014, Ford Aerospace Corporation, August 1989.