SMA Technical MEMO 132
February 1, 1999

Design and Initial Implementation of Diagnostic and Error Reporting System of SMA

Qizhou Zhang

ABSTRACT

We lay out the design of Diagnostic and Error Reporting System of SMA. We describe the implementation of the software and results from test runs on the SMA two-element interferometer.

1. Introduction

SMA is a complex system that involves thousands of electronic and mechanical components and many pieces of controlling software. In such a system, one expects some components fail or do not perform the function they are commanded to. When a component fails or presents a fault status, other components that rely on it may be affected. An extreme example may be the disruption of power to a correlator, which triggers fault in all its components. Should this occur, an array operator would be swamped by error messages were there no error sorting system that performs filtering. In addition, there are pieces of equipment in the array that are delicate. The occurrence of fault status in those components (rising receiver cryostat temperatures, for instance) needs to be brought to the immediate attention of the array operator. And after all, SMA is built to acquire astronomical data. To maximize the use and minimize the corruption of data, one needs to determine the effect of a fault component on the data quality and provide blanking/flagging to the astronomical data steam. All these needs call for an error detecting/sorting system.

The Diagnostic and Error Reporting System (DERS) of the SMA is a piece of computer software that meets these needs. It surveys the status of various test points associated with the hardware and software components in the array. DERS tracks the system fault status and resolves the effect that faults have on data quality. It provides real time blanking and offline flagging information to the astronomical data stream. The software also traces the root cause of the fault and reports error messages to the operator's console and alerts the array operator and engineers via alarm/paging system in case of an emergency.

2. Basic Design

The current design of DERS consists of four major functions:

1. Real time "blanking" information to be passed to the correlator, allowing individual Walsh cycles to be dropped from an integration.
2. Flagging information to be passed to SMADATA after each integration.
3. Error reporting to alert the array operator. A graphic interface to assist troubleshooting.
4. Engineering data to be stored in Sybase for performance analysis.

The underlying design of DERS follows to a large extent the model of the Caltech Millimeter Array (Finch & Scott 1996). The key ingredients in this model are the dependency relationship between nodes and rules that govern fault propagation. An example of the dependency tree is shown in Figure 1. This example was used for the SMA two-element interferometer test during December 1998. Nodes toward the top levels of the tree are the successors of those toward the lower levels. For instance, receiver dewar pressures are logged to reflective memory by the process BALZERS. If BALZERS terminates, the dewar pressure is no longer updated. Therefore, the node Dewar Pressure is a successor of the node BALZERS Timestamp. Conversely, the node BALZERS Timestamp is a predecessor of the node Dewar Pressure. In this example, we have designated the first two working SMA antennas as antenna 2 (node Ant 2) and antenna 3 (node Ant 3) following the nomenclature used in the group.

The idea of a dependency tree remains the central part of the error system. In order for it to work and function effectively, one needs to introduce various concepts associated with nodes and rules governing the fault propagation. We elaborate them in the following:

Node Type: There are three types of nodes in DERS, *sense, diagnostic* and *concatenate.* Both the sense and diagnostic nodes are test points and have their associated values or status. Unlike a sense node, error status in a diagnostic node does not propagate to its successors. This distinction is drawn for a practical need: some nodes that are monitored do not affect the quality of astronomical data. The third type of node is concatenate. They are virtual nodes that have no associated test points. Their status depends solely on the status of their predecessors. An example of such nodes is Ant 2.

Node Status: There are up to three types of fault status associated with each node. Table 1 summarizes the status and corresponding conditions.

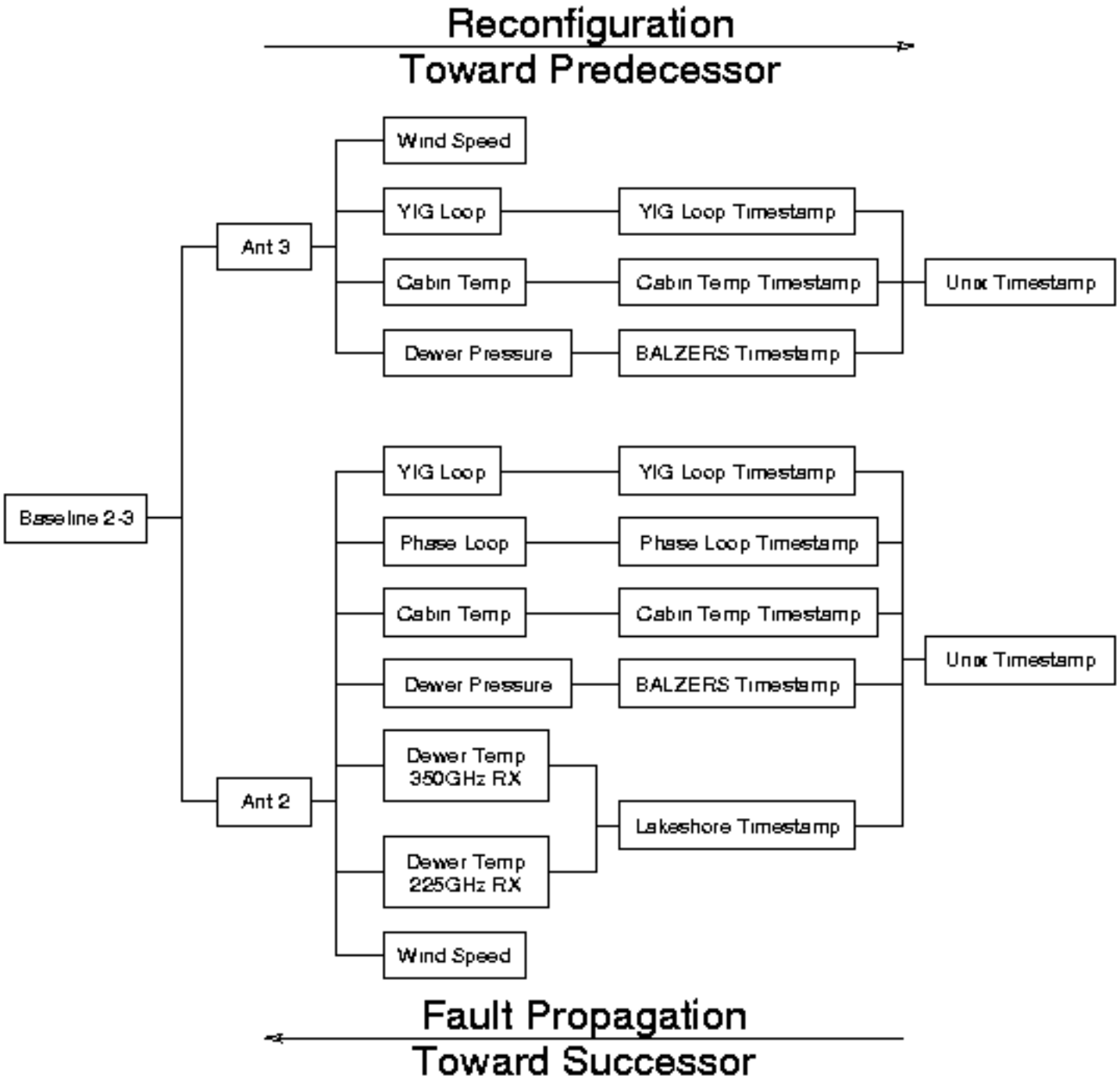

Fig. 1.— DERS logic dependency tree. The nodes on the left are successors to those on the right.

Node State and Reconfiguration: Array operations sometimes require taking a component in the dependency tree offline or disable the entire branch of the dependency tree. For example, a malfunctioning sensor generates false alarms and the status reported from the sensor needs to be ignored. This capability, which is part of reconfiguring dependency trees,

Table 1: Node Status

Node Type	Status	Conditions
Sense/ Diagnostic	GOOD	if within tolerance and all predecessors GOOD
	BAD	if out of tolerance and all predecessors GOOD
	AFFECTED	if one of predecessors BAD or AFFECTED
Concatenate	GOOD	if all predecessors GOOD
	AFFECTED	if one of predecessors BAD or AFFECTED

is accomplished by introducing node state. A node state can either be ONLINE/OFFLINE, or ENABLED/DISABLED. If a node is taken offline, its state is changed to OFFLINE. The reconfiguration of the dependency tree obeys the following rules:

1. The reconfiguration proceeds toward predecessors starting from the node being taken offline;
2. Propagation stops when there are no more predecessors;
3. Propagation stops when an offline node is encountered;
4. The node state changes to DISABLED if all the successors are in OFFLINE or DISABLED state.

Fault Propagation: We design fault propagation rules following the error detection system of the Caltech Millimeter Array. A node status propagates only to its successors along the dependency tree. Both the sense and diagnostic nodes are sampled for status update. However, diagnostic nodes are sampled only for monitoring purpose and their status does not propagate along the dependency tree. Only the sense node or concatenate node status propagates through the tree. The propagation obeys the following rules:

1. The node status propagates only to successors;
2. Propagation stops when the node status are the same as its successor's;
3. Propagation stops when a diagnostic node is encountered;
4. The node status changes to AFFECTED if one of the predecessors are in BAD or AFFECTED status;

5. The node status can only change to GOOD if the node value is within tolerance (for sense/diagnostic nodes) and all the predecessors are in GOOD status.

Error Reporting: As mentioned earlier, the failure of a key component in the array may trigger fault status in large number of nodes. For example, a power disruption to the correlator (we pray that this never happens!) will generate massive amount of error status. To minimize the flux of error messages and more importantly to help locating the trouble spots, it is desirable to have only the root cause of the fault reported. To accomplish this, we introduce the following rules:

1. Messages are sent only for sense or diagnostic nodes;
2. A message is sent only when a status change is detected and the change is not caused by the change in its predecessors.

The second rule ensures that the error is reported only for the root node that initiated the fault.

3. Implementation

A prototype DERS was implemented and tested later during December 1998 on the SMA two-element interferometer. Figure 2 shows the flow chart of DERS. At run time, DERS reads ASCII files that define the dependency relationship among nodes. DERS accesses both the shared memory (SM) and the reflective memory (RM) area to update values for the sense/diagnostic nodes. These values are compared with the associated limits, which derives the updated status. The status then propagates through the tree according to the fault propagation rules. Error messages are logged to SMA1 via a RPC (Remote Procedure Calls). A shell script is defined for SMA staff to view the error messages. For critical nodes such as dewar temperature and dewar pressure, we also implemented e-mailing mechanism when values are out of tolerance.

Figure 3 shows an on-screen print-out of the DERS dependency tree for the two-element interferometer test. Compared to Figure 1, node names are prefixed and suffixed to identify the antenna and the data type of each node. To ensure that the diagnostic data that DERS is reading are being updated, we introduced timestamps for processes running on computers. Those processes update values for relevant nodes to the reflective memory. If any of the processes die, errors in its succeeding nodes are ignored until the concerned process

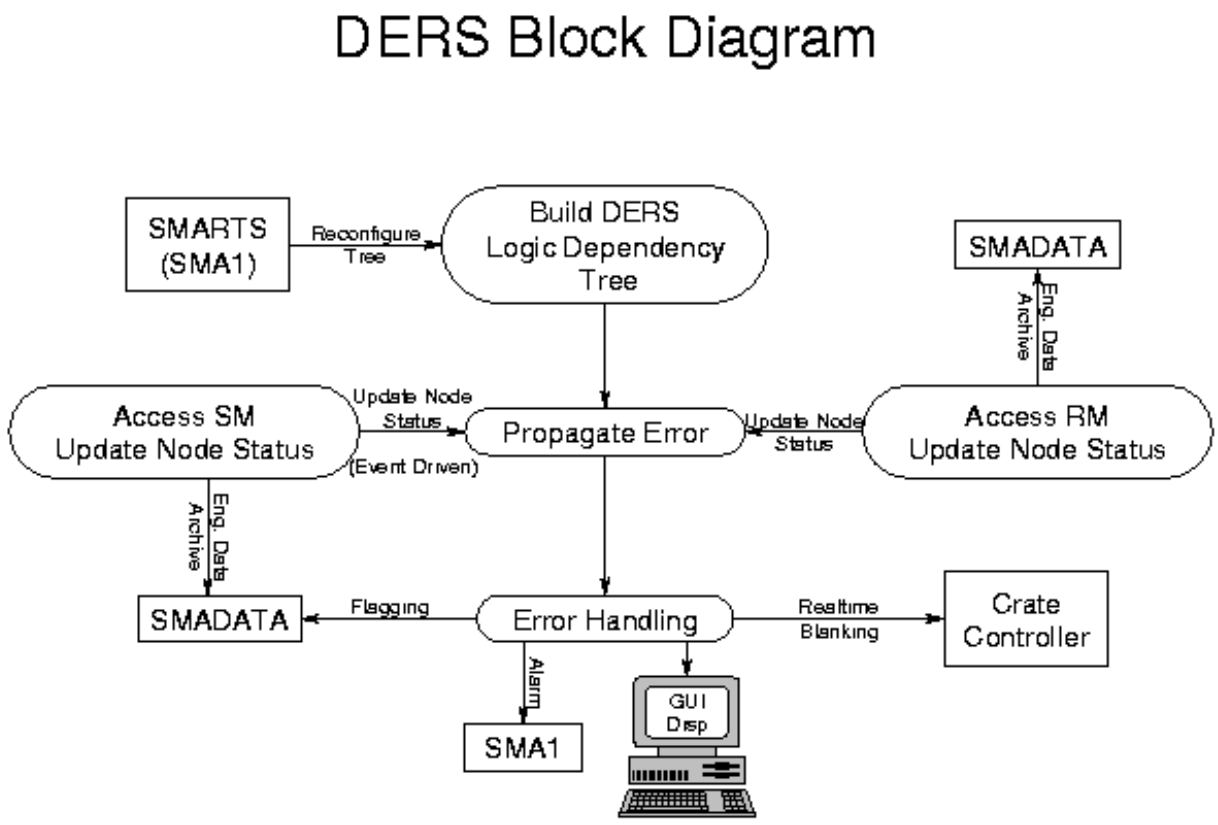

Fig. 2.— Block diagram for DERS. SMA1 and SMADATA are two Sun workstations.

is started again. For example, the process BALZERS updates receiver dewar pressures to reflective memory. If BALZERS is stalled, the pressure is no longer updated. In this case, a fault related to the process BALZERS will be reported. However, any errors related to the node DEWAR_PRESSURE (if exists) will not be reported unless BALZERS is restored. The test points in Figure 3 represent essentially all the relevant nodes in the reflective memory available for monotoring. During DERS test runs, data blanking/flagging was not performed.

Figure 4 shows information DERS reported to the screen at run time. It is a snapshot and is updated each time after all nodes are sampled. In this example, the phase lock loop in the 225 GHz receiver in antenna 2 was out of locked and presented a BAD status. This fault propagates to its successors ANT2 and BSLN_2_3. Both nodes are shown to be AFFECTED. A message was sent to SMA1 via RPC, which is shown in Figure 5a. We also have a few nodes that are deemed critical (receiver dewar pressure, for instance). When their values go out of tolerance, electronic mails are sent to related SMA staff, in addition to regular

```
0(BSLN_2_3)
    1(ANT2)
        2(WEATHER_WINDSPEED_F)
        3(ANT2_DEWAR_TEMP1_S)
            11(ANT2_LAKESHORE_TSTAMP_L)
                14(UNIX_TIME_L)
        4(ANT2_DEWAR_TEMP3_S)
            11(ANT2_LAKESHORE_TSTAMP_L)
                14(UNIX_TIME_L)
        5(ANT2_DEWAR_PRESSURE_F)
            10(ANT2_BALZERS_TSTAMP_L)
                14(UNIX_TIME_L)
        7(ANT2_CABIN_TEMP_F)
            6(ANT2_CABIN_TEMP_TSTAMP_L)
                14(UNIX_TIME_L)
        8(ANT2_PHASE_LOCK_S)
            12(ANT2_PHASE_LOCK_TSTAMP_L)
                14(UNIX_TIME_L)
        9(ANT2_YIG1_LOCKED_S)
            13(ANT2_YIG_SVC_TSTAMP_L)
                14(UNIX_TIME_L)
    15(ANT3)
        2(WEATHER_WINDSPEED_F)
        16(ANT3_DEWAR_PRESSURE_F)
            18(ANT3_BALZERS_TSTAMP_L)
                14(UNIX_TIME_L)
        17(ANT3_YIG1_LOCKED_S)
            19(ANT3_YIG_SVC_TSTAMP_L)
                14(UNIX_TIME_L)
```

Fig. 3.— On-screen print out of the DERS dependency tree.

warnings sent to SMA1 through RPC. An example of such a warning is shown in Figure 5b.

4. Results from Test Runs

DERS has been running on HAL at the SMA Haystack site since December 1998, with brief disruptions for revision. During the run, DERS has proven to be very effective in tracing the root cause of errors in the system. In a number of occasions, the process UNIX_TSTAMP stopped running, which affected the entire dependency tree. However, only the error on UNIX_TSTAMP was reported.

At the time of this writing, antenna 2 has been taken apart and began to be shipped to

the SMA site in Mauna Kea. The node ANT2 was taken offline.

5. Future Work

DERS will expand as new pieces of hardware are integrated into the array and more test points become available. In the immediate future, we will implement the engineering data archive, DERS-SMARTS communication capabilities and the graphic status display.

REFERENCES

Finch, R. P., & Scott, S. T. 1996, PASP, 108, 714

```
Node Id = 0: (BSLN_2_3)       baseline 2-3
        parents = NONE
         children = 1 15
        type = CONCAT   state = ONLINE,ENABLED   status = AFFECTED
Node Id = 1: (ANT2)       Antenna 2
        parents = 0
         children = 2 3 4 5 7 8 9
        type = CONCAT   state = ONLINE,ENABLED   status = AFFECTED
Node Id = 2: (WEATHER_WINDSPEED_F)  wind speed
        parents = 1 15
         children = NONE
        type = DIAG    state = ONLINE,ENABLED    status = GOOD
Node Id = 3: (ANT2_DEWAR_TEMP1_S)   Ant 2 225 GHz receiver dewar temp
        parents = 1
         children = 11
        type = SENSE   state = ONLINE,ENABLED    status = GOOD
Node Id = 4: (ANT2_DEWAR_TEMP3_S)   Ant 2 350 GHz receiver dewar temp
        parents = 1
         children = 11
        type = SENSE   state = ONLINE,ENABLED    status = GOOD
Node Id = 5: (ANT2_DEWAR_PRESSURE_F) Ant 2 dewar pressure
        parents = 1
         children = 10
        type = SENSE   state = ONLINE,ENABLED    status = GOOD
Node Id = 6: (ANT2_CABIN_TEMP_TSTAMP_L)    Ant 2 cabin temp time stamp
        parents = 7
         children = 14
        type = SENSE   state = ONLINE,ENABLED    status = GOOD
Node Id = 7: (ANT2_CABIN_TEMP_F)     Ant 2 cabin temperature
        parents = 1
         children = 6
        type = SENSE   state = ONLINE,ENABLED    status = GOOD
Node Id = 8: (ANT2_PHASE_LOCK_S)     Ant 2 phase lock
        parents = 1
         children = 12
        type = SENSE   state = ONLINE,ENABLED    status = BAD
Node Id = 9: (ANT2_YIG1_LOCKED_S)    Ant 2, RX 1 YIG lock status
        parents = 1
         children = 13
        type = SENSE   state = ONLINE,ENABLED    status = GOOD
Node Id = 10: (ANT2_BALZERS_TSTAMP_L)   Ant 2 monotorBalzers's time stamp
        parents = 5
         children = 14
        type = SENSE   state = ONLINE,ENABLED    status = GOOD
Node Id = 11: (ANT2_LAKESHORE_TSTAMP_L) Ant 2  monotorLakeshore's time stamp
        parents = 3 4
         children = 14
        type = SENSE   state = ONLINE,ENABLED    status = GOOD
Node Id = 12: (ANT2_PHASE_LOCK_TSTAMP_L)Ant 2  phase lock's time stamp
        parents = 8
         children = 14
        type = SENSE   state = ONLINE,ENABLED    status = GOOD
Node Id = 13: (ANT2_YIG_SVC_TSTAMP_L)   Ant 2  YIG lock's time stamp
        parents = 9
         children = 14
        type = SENSE   state = ONLINE,ENABLED    status = GOOD
```

```
Node Id = 14: (UNIX_TIME_L)     Unix time stamp on HAL
        parents = 6 10 11 12 13 18 19
         children = NONE
        type = SENSE   state = ONLINE,ENABLED    status = GOOD
Node Id = 15: (ANT3)       Antenna 3
        parents = 0
         children = 2 16 17
        type = CONCAT   state = ONLINE,ENABLED   status = GOOD
Node Id = 16: (ANT3_DEWAR_PRESSURE_F)   Ant 3 dewar pressure
        parents = 15
         children = 18
        type = SENSE   state = ONLINE,ENABLED    status = GOOD
Node Id = 17: (ANT3_YIG1_LOCKED_S)    Ant 3, RX 1 YIG lock status
        parents = 15
         children = 19
        type = SENSE   state = ONLINE,ENABLED    status = GOOD
Node Id = 18: (ANT3_BALZERS_TSTAMP_L)   Ant 3  monotorBalzers's time stamp
        parents = 16
         children = 14
        type = SENSE   state = ONLINE,ENABLED    status = GOOD
Node Id = 19: (ANT3_YIG_SVC_TSTAMP_L)   Ant 3  YIG lock's time stamp
        parents = 17
         children = 14
        type = SENSE   state = ONLINE,ENABLED    status = GOOD
```

Fig. 4.— On-screen print out of the state and the status for all the nodes shown in Figure 3. Note that the phase lock loop on antenna 2 (see Node ID = 8) was out of locked and displayed a BAD status. This affected its succeeding nodes ANT2 and BSLN_2_3.

```
1998 12 23 22 12 45 ANT2_PHASE_LOCK_S=0.0  Ant 2 phase loop not locked. ^G
1998 12 23 39 34 27 ANT2_PHASE_LOCK_S=1.0  Back to normal ^G
```

Fig. 5a.— Messages sent to SMA1 via RPC regarding the phase lock loop. The first six fields in the message are year, month, day in the month, hours, minutes and seconds in UT, followed by the node name, node value and a brief message. When the phase loop is locked, an all clear message was sent.

```
From hal@sma1.haystack.edu Wed Jan 06:19:32 1999
Subject: Warning from DERS
Content-Length: 105
X-Lines: 3

ANT3_DEWAR_PRESSURE_F=0.098000
Ant 3 dewar pressure high.
```

```
From hal@sma1.haystack.edu Wed Jan 06:19:34 1999
Subject: Warning from DERS
Content-Length: 105
X-Lines: 3

ANT3_DEWAR_PRESSURE_F=0.000019
Back to normal.
```

Fig. 5b.— E-mail messages sent to the SMA staff regarding the rising and restoration of the receiver dewar pressure on antenna 3. Each e-mail contains the node name, node value and a brief description of the situation. A relief e-mail is sent after the node value is restored to within the limit.