The Space Congress® Proceedings

1987 (24th) Space - The Challenge, The Commitment

Apr 1st, 8:00 AM

Camera Expert System for Space Station Communications and Tracking System Management

Michael Crone
Harris Corporation, P .0. Box 98000 Melbourne, Florida 32901

Paul Julich
Harris Corporation, P .0. Box 98000 Melbourne, Florida 32901

Follow this and additional works at: https://commons.erau.edu/space-congress-proceedings

Scholarly Commons Citation

Crone, Michael and Julich, Paul, "Camera Expert System for Space Station Communications and Tracking System Management" (1987). *The Space Congress® Proceedings*. 3.
https://commons.erau.edu/space-congress-proceedings/proceedings-1987-24th/session-4/3

This Event is brought to you for free and open access by the Conferences at Scholarly Commons. It has been accepted for inclusion in The Space Congress® Proceedings by an authorized administrator of Scholarly Commons. For more information, please contact commons@erau.edu, wolfe309@erau.edu.

EMBRY-RIDDLE Aeronautical University
SCHOLARLY COMMONS

CAMERA Expert System for Space Station Communications and Tracking System Management

by

Michael Crone and Paul Julich
Harris Corporation, P.O. Box 98000
Melbourne, Florida 32901

ABSTRACT

This paper describes Harris research into the use of Expert System technology for the management of the Communications and Tracking System for the Space Station. Harris Corporation has developed the CAMERA (Control and Monitor Equipment Resource Allocation) Expert System to minimize crew workload in managing the communications of the Space Station. The system has been implemented (under NASA contract) for use on a testbed at JSC. The system utilizes a state of the art man-machine interface to allow high level end-to-end service requests.

1. INTRODUCTION

The Communications and Tracking System (CTS) for the Space Station provides a communication network to support command, control, telemetry, payload, audio, and video data flow among the Space Station Program elements and the ground. The CTS provides the flexibility to interconnect communications equipment to support a variety of missions over a thirty year period. The needs range from the transmission of low rate telemetry and command data to 100 Mbps payload data. A typical service might include supporting an EVA operation by providing voice communications, telemetry, and downlink video while monitoring the position of the EVA with remote video. As the EVA exits the Station and moves about in the proximity operations zone, the communications system automatically switches antennas and cameras to maintain communications with the astronaut. Obscuration by the station structure and payloads along with multipath effects must be accommodated. Simultaneously, a variety of other services might be required to support other aspects of the Station mission. With a limited quantity of crew time available, automation is necessary to assure C&T services are available in a timely fashion.

Management of the CTS equipment is the responsibility of the Control and Monitoring Subsystem (CMS) of CTS. Requests for C&T services are received from the Operations Management System. CMS selects the appropriate equipment to fill the service request, computes the parameters associated with the service, and schedules the service by reserving the selected equipment and backup equipment. At the scheduled time of service initiation, CMS provides control messages to the C&T equipment to establish the desired configuration and monitors the initiation of service. Following successful establishment of service, the status and performance indicators and built-in-test signals are monitored to assure that the quality of the service is maintained. In the event of a degradation of service or a trend indicating the imminent loss of service, CMS identifies and locates the problem and initiates the appropriate recovery action.

A C&M architecture envisioned by NASA (Figure 1) is a distributed architecture composed of redundant standard data processors and subsystem controllers internetted by a dedicated 10 Mbps bus. These elements are identical processing elements to those of the Data Management System and other Station systems. The subsystem controllers (e.g., Space to Space, Space to Ground, Audio, Video, and Tracking) contain embedded processors which are responsible for managing the interface to the local subsystem equipment. CTS management software runs in the SDP although the balanced protocol on the C&M bus and the structure of the software allows it to be relocated to the embedded processors.

1.1. THE PROBLEM

Due to the complexity of the communication requirements for Space Station, the automatic allocation of resources and management of such networks in real-time is a goal not attainable via traditional approaches to resource allocation such as linear programming. Associated with the operation of such networks is the problem of monitoring activity for anomalous conditions or trends which suggest imminent problems. The network of the future in space must be automatically reconfigurable to handle such conditions. In addition, it must support diagnostics for repair of faulty equipment.

1.2. AUTOMATION OF CONTROL AND MONITORING SUBSYSTEM

Rapid advancements in Expert Systems have made the application of this technology to the Space Station feasible. Harris Corporation has developed the CAMERA (Control and Monitor Equipment Resource Allocation) Expert System to assess the effectiveness of an expert system in minimizing crew workload in managing the communications for the Space Station. CAMERA provides for automatic management of communications resources, diagnosis of faults, and reconfiguration to restore communications automatically. Prototypes of the system have been implemented (under NASA contract) for evaluation on a testbed at JSC.

1.3. DEVELOPMENT APPROACH

The paradigm of "Expert Systems" has been shown to provide problem-solving computer programs that can reach a level of performance comparable to that of a human expert in specialized problem domains [1,2]. The advantages of a knowledge-based approach to the solution of difficult problems are well known, but the development of such systems differs significantly from that of traditional software [3]. An approach often taken is to use rapid-prototyping to build versions quickly for critique by the expert [3,4]. This approach alone, however, is not appropriate for problems with a great deal of underlying technical structure and real-time requirements, such as the C&M system. In such cases it is necessary to perform a structured analysis to bound the problem and have a basis for the rapid-prototyping. To accomplish this structured analysis, an extension of Data Flow Diagrams(DFD) called Transformation Schemas for the analysis and design of real-time systems are used [5]. The beauty of the Transformation Schemas (which includes extensions to both the DFD symbols as well as the addition of State Tables) is the ease with which they can be mapped to the Automated Reasoning Tool (ART) shell.

In the building of an expert system, Hayes-Roth suggests focusing on the nature of the knowledge and the system architecture required to solve the problem. The distinction to be made being between numeric and symbolic knowledge, factual vs heuristic knowledge, and imperative(how-to-do-it) vs declarative(relationships) knowledge [6]. Our approach to the development of the prototypes was to build the expert system initially on a Symbolics using the ART programming language which provides for "opportunistic reasoning" in the planning, execution and monitoring of activities [7]. This permits focusing on the knowledge base rather than building an inference engine or writing extensive graphics software. As the development proceeds, the architecture begins to take shape, with the migration of some of the code from the ART environment into the Lisp and Flavors languages [8].

Using this development environment four prototypes were completed over the course of a year. Each prototype was demonstrated to the NASA experts for critique and comments and suggestions incorporated into the succeeding prototype. The current prototype consists of over 550 ART rules, over 7K lines of ART schemata, and several thousand lines of Lisp and Flavors.

2. RESOURCE MANAGER EXPERT SYSTEM

The CAMERA expert system can really be considered to be two closely coupled expert systems - one for resource management, and one for automatic monitoring. The expert system for resource management includes planning and scheduling of communications resources, and the control of those resources through the duration of the requested service. The expert system for automatic monitoring accomplishes both predictive as well as adaptive reconfiguration of resources based on measurements of equipment states.

The basic architecture of the CAMERA system is shown in Figure 2. The MMI, Simulator Control, Resource Manager(RM), and Automatic Monitor(AM), shown in the upper half represent the Expert System software written primarily in ART and residing on the Symbolics. The Simulator and Status Monitor (SIMON) shown in the lower half are currently written in Lisp and Flavors on the Symbolics, but will be migrated to a Unix environment and rewritten in an Object-Oriented C language developed by Harris called CFLAVORS. All communications above the center line in the figure occur through the ART data base. All rules match on information asserted into this data base and fire in a data-driven, asynchronous fashion. Communications from CAMERA to SIMON occur via Lisp interface code, and information from SIMON to CAMERA is done via assertions into the ART database.

Service requests to the C&M system come from two sources: (1) the crew , and (2) Operations Management System(OMS). Although the CAMERA software makes no attempt to simulate the activities of OMS, the design of the RM software has been done to make such an interface straightforward. This has been accomplished by converting all communications with the RM into a message-level protocol. The command language of CAMERA is defined at multiple levels to support high-level script-type concepts down to low-level commands which actually activate and control the simulator. CAMERA commands are transformed into a tree of schemata upon which rules such as the scheduler portion of the RM can match and act. Their actions are limited to modifying information in the ART data base and/or asserting or retracting information in the data base. The MMI monitors this data base and modifies the Symbolics display accordingly. The AM also monitors the data base for changes to equipment allocations and equipment measurements.

The CAMERA system uses the SYMBOLICS display for all MMI purposes including graphic routines to display trend information on the SYMBOLICS terminal. The display on the Symbolics consists of several windows - the arrangement and presence of which depends on the state of the system and user requests. The primary input device is the mouse, augmented by the keyboard for some parameter entering. A spaceborne implementation might use a trackball or joystick. All menus items are activated via the mouse, or in Symbolics' terms are "mouseable". The user may also move, reshape, or bury windows at his convenience. Several of the windows serve as "real-time" status displays where information such as alerts, anomalies, trend data as well as a time view of service requests in the context of the operations schedule, are updated asynchronously. Shown in Figure 3 is a copy of a display of the MMI in the latest CAMERA prototype. The windows labeled: current time, operations schedule, and system messages are updated asynchronously. The command menu consists of pull-down menu categories such as SERVICE-REQUEST. All windows are scrollable if the display exceeds the window size. In Figure 3 the Service Request window shows a boilerplate which the operator has filled in to define a service request. After submission it it will be entered in the Operations Schedule window in the pending state while a determination is made as to whether the service can be satisfied. The scheduler will determine whether the required equipment and channel capacity exists during the required time frame by applying temporal constraints [9].. If it can it will proceed to "entered" state until being activated at the appropriate time. If all proceeds normally it will pass to the "completed" state at the end of the programmed time. If during the process of equipment allocation equipment is unavailable for the requested time frame but could be made available by preempting a previously scheduled request, the operator is asked whether or not he wants to preempt the lower priority request.

Information within a window is often an abstraction of more specific information, and can, in many cases, be expanded hierarchically for more details. One such example is that of the OPERATIONS SCHEDULE shown in Figure 3, in which service requests are shown scheduled over some time window. These are displayed in their most abstract form as rectangles labeled with service request IDs, type of request, priority, duration, and state of request. If the user desires to look at the next level of detail for a service request he can mouse on a rectangle and information about the request will be displayed. If he desires to go another level down, he can mouse on an equipment item listed below in the scrollable window. At this point information is displayed showing the state of the equipment such as the TDRSS-KU-BAND-R-T-1 shown in Figure 3. The same approach is taken for other system information, such as trends, as appropriate. The advantage to this approach is that a user requires little operational experience to logically traverse the system.

The primary input to the RM function is a service request. A set of primitive service requests such as "EVA to Space Station Audio" which represent end-to-end communications, have been defined and exist in a menu for scheduling. In addition, a service request may contain embedded service requests which may be satisfied in parallel, or which must occur in some sequence. To support the definition of these more complex service requests, a graphic editor has been developed which allows the user to define logical communication links along with the types of communications. The system will build the service request from the primitives. Figure 4 shows the building of such a request to support an EVA. In this example, where the service-type-name has been given as "SPACE-1" by the user, An EVA will communicate to both the station and the ground using both audio and video (the default), and the visual activity will be monitored on the station. Note that the user specifies a logical link from the EVA to ground. The system knows that such a path must physically pass through the station, so appropriate equipment is allocated. In the current prototype, the user can select one of the four quadrant areas outside the station where the EVA will take place. He may also select the area which will monitor the video. Both of these are shown in dark in Figure 5. Finally he can mouse on the hab module and select the monitor as shown in Figure 6. When he is satisfied with this service request it will be added to the service types menu and can be scheduled as any other request. He can also re-edit the service request at a later time, and ask for a high-level connectivity diagram showing the subsystems involved in satisfying the service request.

We have also identified the need to define a capability for repetition (e.g. do this request every day at 0900). Such an approach allows for procedural definition (to include error handling capability) of service requests while maintaining the parallelism and "opportunistic reasoning" capabilities of a tool such as ART. By "opportunistic reasoning" it is meant that a search for a solution can proceed in both backward and forward directions as dictated by the data. By allowing the definition of embedded service requests, the system can support the concept of an entire mission as an abstract service request where each embedded request may be either a literal service request or contain other embedded service requests.

Just as the user can define service requests in the direction of abstractions, the design supports the introduction of commands in the direction of specialization. In other words, the knowledgeable user can enter commands at any level in the hierarchy down to the commands which actually control equipment. He can also specify particular instances of equipment when scheduling a service as opposed to letting the allocator choose for him. Although this approach can lead to the termination of either a pending or active service request, it is imperative to provide the crew with a manual override capability. It is not necessary for the user to know the exact command syntax for the override commands, because the MMI will allow him to issue logical commands such as "turn off transmitter", or allow the more experienced user to bring up an equipment description (schema) and modify any of the slots directly.

After entering a service request the user receives back an indication of either a successful scheduling of the required equipment in the specified time frame or an unsuccessful scheduling attempt. Currently the successful service request is entered automatically. In future versions the user will be able to specify in advance whether he wishes to approve the hypothesized schedule or let it be entered automatically. This will allow the user to accomplish "what if?" planning without effecting the system configuration. In the current system when a user issues a service request of a higher priority than a scheduled or active service request he is asked to ok the pre-emption of the effected service request. Each service request is associated with user (crew) names. As such, the system can impose limitations as to pre-emption levels available to each individual.

From the user's perspective the status of the system is the other main concern. If things are progressing as scheduled, then the the operations schedule window will be modified appropriately. If the service request has associated trend information or critical parameters that the system is monitoring, the user will be able to specify a window into the system for this purpose. This window may display plots of trend data in quasi real-time. An example of such a plot is shown in Figure 7. The output information currently displayed in an asynchronous manner are any alerts or failures.

3. AUTOMATIC MONITOR EXPERT SYSTEM

The purpose of the Automatic Monitor(AM) Expert System is to detect faults in the communications system, isolate those faults to orbital replaceable units (ORUs), notify the communications specialist, and report the failure to the Resource Management Expert System. The current version of the AM actually isolates faults to equipment within the ORU, such as a low-noise amplifier. In future versions it will also support the diagnostic activities on Space Station.

Fault isolation has progressed in recent years from using shallow reasoning approaches which were strictly rule-based, to model-based reasoning based on function and structure[10,11]. Other approaches have emphasized more general topological knowledge (connectivity plus directionality) which must suspect all modules upstream when a failure is detected. Our current prototype for AM deals mainly with the structure, and is represented by a fault network residing as schemata in the ART database.

3.1. STATUS MONITORING

The purpose of the Status Monitor is to report measurements to the AM whenever so directed, or whenever a measurement is not within expectations. The default behavior of the STATMON is to test new measurements against the expected value of the measurements. When a measurement begins to go awry and enters the guard band, a warning message is sent to the AM, and the system will begin to automatically collect trend data. Statmon will automatically collect values until the measurement returns to a normal status and stays there for some pre-defined period (useful for detecting oscillation). Statmon can be considered a programmable filter for screening input measurements. The AM has total control over the filters that Statmon uses, what data is collected for trend analysis, and what information is reported back.

Statmon receives input from the RM, the Simulator, and the AM. RM sends Statmon the same commands that it sends to the simulator (or to the actual equipment if present). The simulator sends new measurements to the Statmon, and the AM sends Statmon messages which control which data the Statmon evaluates and sends to AM. The Statmon uses a message queue and command table like the simulator and uses these to compare against its own expected measurement data base as shown in Figure 8. It functions much like an embedded simulator and creates expectation measurements which track simulated measurements by applying the same functions as the simulator. This allows the guard band and red lines to follow complex signals as shown in this figure.

3.2. FAULT DETECTION, ISOLATION, AND RECOVERY

As measurements are asserted into the ART DB by the Statmon, AM rules are fired to instantiate portions of the fault net for reasoning. In particular one or more "symptom specimens" corresponding to real symptoms are instantiated. Other AM rules will then instantiate problem specimens linked to the symptoms via "local-causes" relations. The AM rules will then try and associate the problems via "causes" relation from the fault network. The highest problem in the "causes" chain (usually the one nearest the signal source) is marked as the real problem and the others are retracted. The AM will send this information to the RM where a backup device will be switched in if one exists, and also notify the operator of the failure.

4. EVOLUTIONARY APPROACH TO DEPLOYMENT

The inclusion of Expert Systems technology for the Space Station should be evolutionary in nature. An object-oriented C&T design can accomplish this evolution in a straight-forward manner. The key to this task is in the degree of coupling between the software Computer Programs (CP - a grouping of software which accomplishes a major function such as control and monitoring of C&T equipment) and the units within each program. By taking an object-oriented approach to the system, the emphasis can be placed on the message passing between objects, and the encapsulation of methods and data within these objects. The actual location where the objects reside, and the details as to their internal implementation then becomes more of a networking and therefore communications protocol and bandwidth problem. Whether the objects accomplish their tasks with a minimal intelligence as with the initial traditional architectural design or with "expert level" intelligence does not effect the design of the system. It only effects the robustness of the decision. This will, in turn, effect the amount of total system utilization but will not require redesign of other objects. The actual evolution of the objects can occur in several ways. The main point is that encarnations of the various objects may exist on the ground, in space, or both, depending on the phase in the evolution.

Before discussing the approach, consider a possible evolution. First, at IOC, a non-Expert System software architecture based on ADA and object-oriented design exists on the Station. We might refer to this as Control and Monitoring Local (CAML) system. On the ground at IOC would be Control and Monitoring Software capable of creating complex schedules to be relayed to the CAML system for implementation. We might call this the Control and Monitoring Plenary (CAMP) system. The functions of CAMP will be explored in the next paragraph. The CAML system will serve initially in a support role to the predetermined plans submitted by the CAMP system. It will, however, provide the onboard crew with the capability to make modifications to the CAMP plan and to restore space-to-ground communications if necessary. The CAML software will be designed as hierarchical objects with levels of responsibility. By allocating the functions of resource management and redundancy management at both the system control level (C&M) and with each subsystem (Audio, etc.) an object-oriented approach can be maximized, and Expert System evolution is natural. For instance, the C&M receives service requests along with constraints from OMS. It maps these into lower level service requests and constraints which are passed to the subsystems. It is the subsystem's responsibility to determine the instances of equipment and command sequencing to accomplish the service (with the exception being when the actual instance of equipment is passed as a constraint from the higher level). This will initially be an algorithmic or table look-up function, but can made more intelligent by adding Expert Level knowledge to the process. To reemphasize an earlier point, the results of the service request will be the same - only the quality will change. By taking the message passing paradigm, the interface between CP's need not change - only additional methods will be required of the receptor CP. Using this same approach, planning can be eventually added to the system, by changing the mode of the CP in response to a service request. Another point to be made about this design approach is that by treating the management functions as merely levels of abstraction in solving the allocation problem, the code which exists at each level can be common with its "tables" being populated with level specific data. This approach also supports an incremental growth in the SDP's and EDP's by adding additional memory and processor boards within the SDP and EDP enclosures.

Expert System technology in the area of resource management could exist at IOC, but it would be most appropriate to locate this software on special purpose hardware on the ground. This software (dubbed CAMP earlier in the document) would have access to the same schedule and status data base as exists on the station. Its job would be similar to the Harris CAMERA Expert system resource manager which uses a knowledge-based approach to assign specific pieces of equipment to service requests. The results of this assignment along with other constraints would be transmitted via TDRSS to the station as a service request with specific instances of equipment being included as constraints to the C&M. The C&M software would pass the constraints along with the service requests to the appropriate CP's for implementation. Of note is the fact that the C&M software could have been given the original service request and used its table lookup scheme rather than receiving a more elaborated one from the ground. The only difference in the result is that a better system utilization will occur from the knowledge-based approach. The same message passing scheme with service requests and constraints is still used.

The next stage in the evolution is to move some of the knowledge-based capability to the C&M and then to selected CP's and finally to all CP's. The architecture need not change from an inter-CP perspective. The final stage would be to add planning support to the system on the ground and migrate it eventually to the Station. Once again the architecture is left intact, but additional information flow will take place logically from the lower levels to the upper levels. This final task may not actually take place via an object-oriented approach, but may instead take advantage of Relational Database products. These tools are adding inferencing capabilities which will allow a very tight coupling between Expert Systems and Databases. This may make the transition to Expert Systems even smoother than currently envisioned.

As a final note, a similar approach can be taken for the redundancy management software. This is especially true in the area of fault isolation and diagnostics. This area will require more transmission bandwidth to get trend data and BIT from the station, but will only be used on the ground to get ORU's back in operation. The redundancy switching will occur on the station in real-time.

5. SUMMARY

The development of the CAMERA Expert System has shown the feasibility of Expert Systems for the Space Station. Work is continuing at Harris to integrate mission planning with the CAMERA system and to enhance the diagnostic capabilities. Investigation into porting the CAMERA system onto a traditional computer architecture in ADA is also being pursued, along with retargeting to a parallel processing environment.

Acknowledgements

Contributions to this work were made by AI researchers at Harris including: Lynn Carley, Ellen Dash, John Dixon, Jim Jacobs, Paul MacNeil, Laura Thomae, Bill Wavering, and Tom Wells.

6. References

(1) *The Handbook of Artificial Intelligence*, ed. by A. Barr and E. Feignbaum, HeuristicTech Press, Stanford, Cal. 1982

(2) "Expert Systems: Limited but Powerful", W.B. Gevarter, IEEE Spectrum, August 1983, pp 39-45

(3) "Comments on the Procurement and Development of Expert Systems", M. Crone, D. Hall, Proc. Expert Systems in Government Symposium, IEEE Computer Society, 1985

(4) *Building Expert Systems*, Edited by F Hayes-Roth, D. Waterman, D. Lenat, Addison-Wesley, 1983

(5) *Structured Development for Real-Time Systems* , P. Ward & S. Mellor, Yourdon Press, New York, 1985

(6) S.1 Systems User's Guide, Teknowledge Inc.

(7) ART Reference Manual, Inference Corporation, Los Angeles, Cal., 1985

(8) Lisp Machine Manual, Weinreb, D. & D Moon, Symbolics Inc. 1985

(9) "Constraint-Directed Search: Case Study of Job-Shop Scheduling", M. Fox, Ph.D Thesis, *Computer Science Department, Carnegie-Mellon University, 1983.*

(10) "The FIS Electronics Troubleshooting System", F. Pipitone, *IEEE Computer*, 1986

(11) "Diagnosis Based on Structure and Function", R. Davis *Proceedings AAAI Conference*, 1983

REGRETFULLY, A QUALITY REPRODUCTION OF THIS ARTICLE WAS NOT POSSIBLE, DUE TO QUALITY OF FURNISHED ORIGINALS.

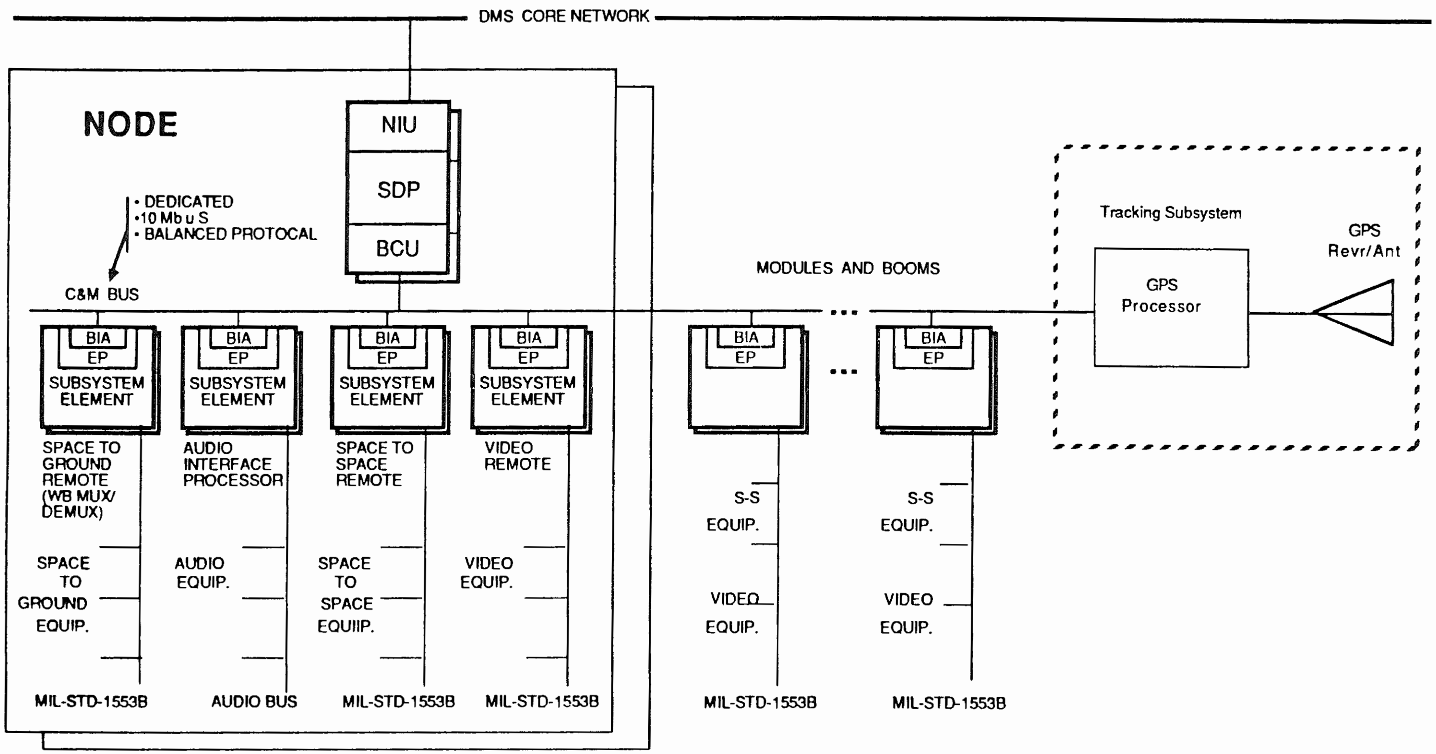

Figure 1: C&MS HARDWARE ARCHITECTURE

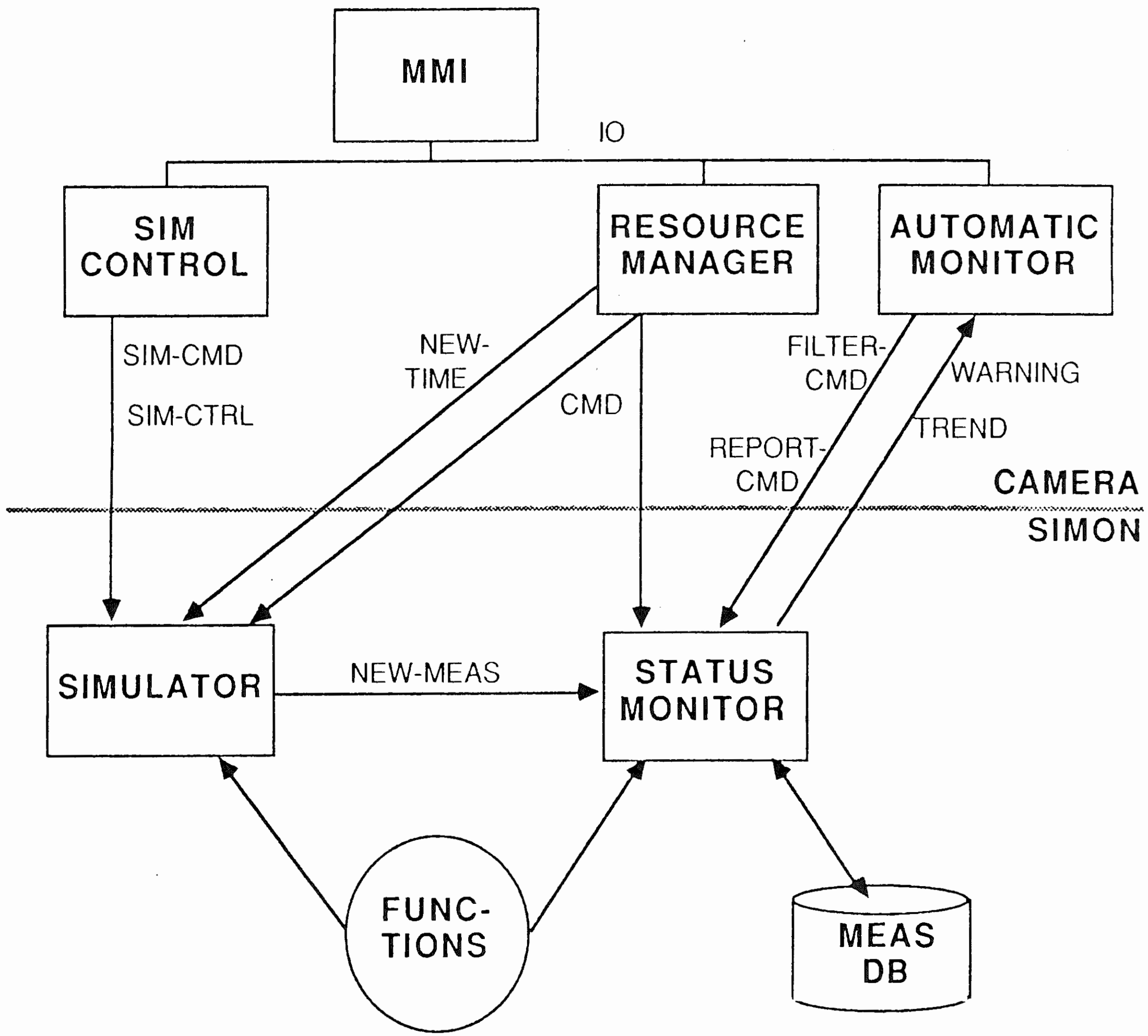

Figure 2. CAMERA Architecture

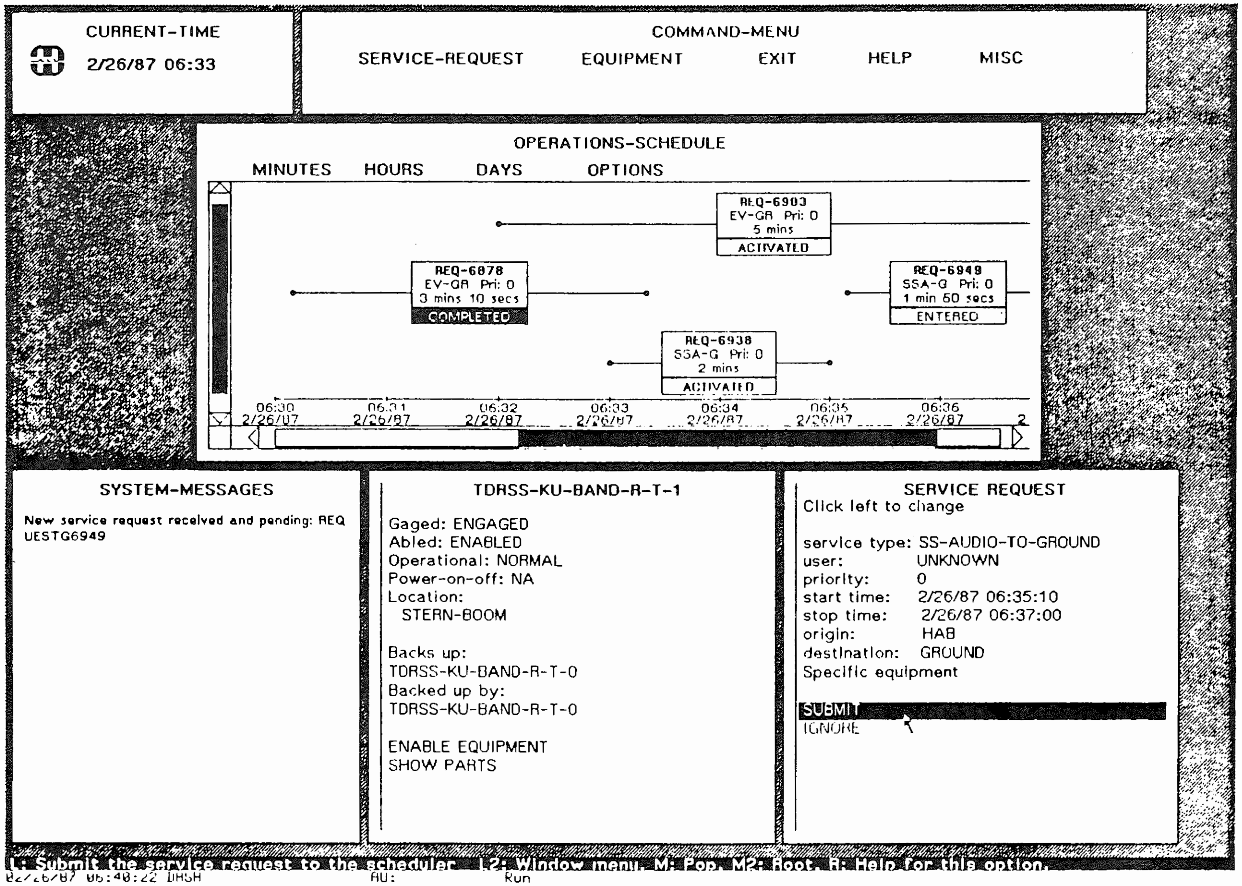

Figure 3. Operational Man-Machine-Interface

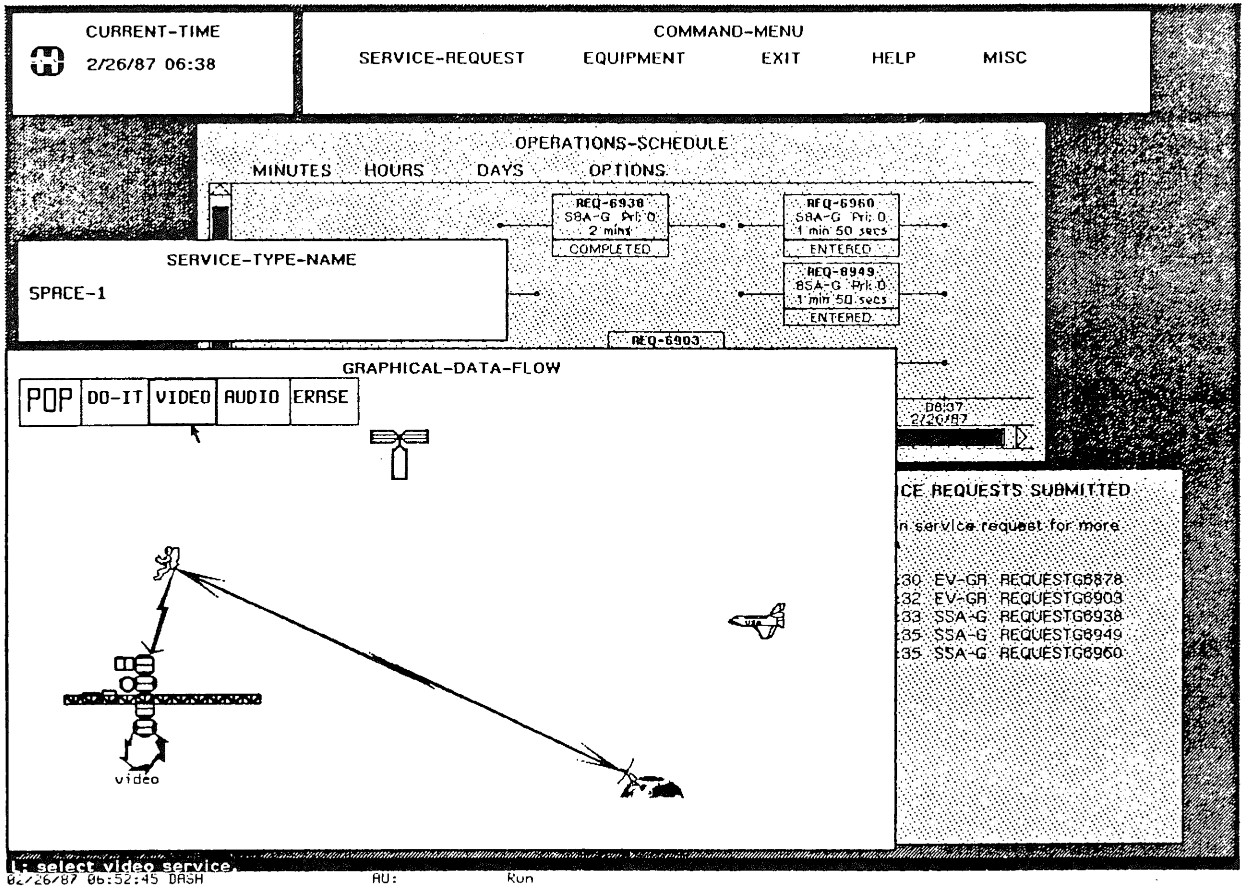

Figure 4. Graphical Data-Flow Editor

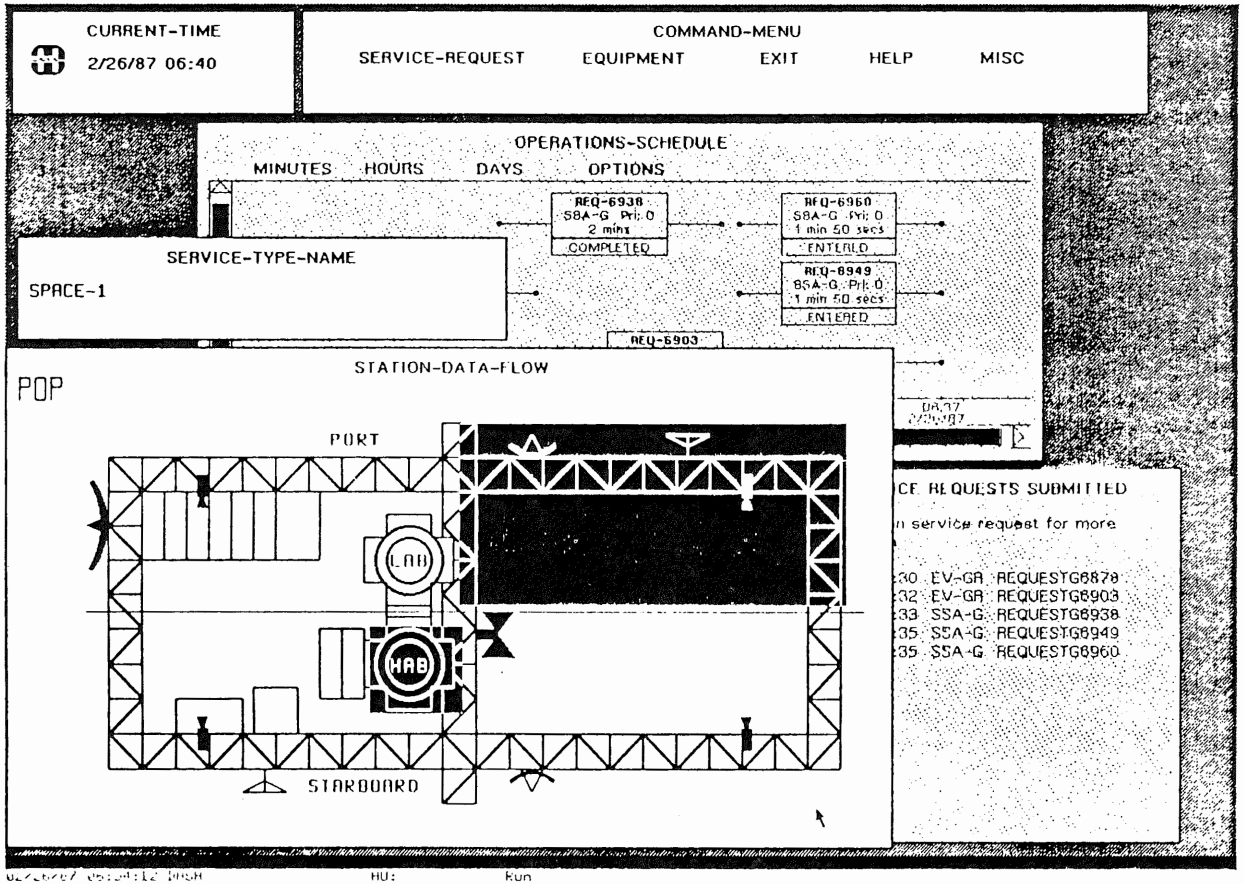

Figure 5. Station-level Data-Flow

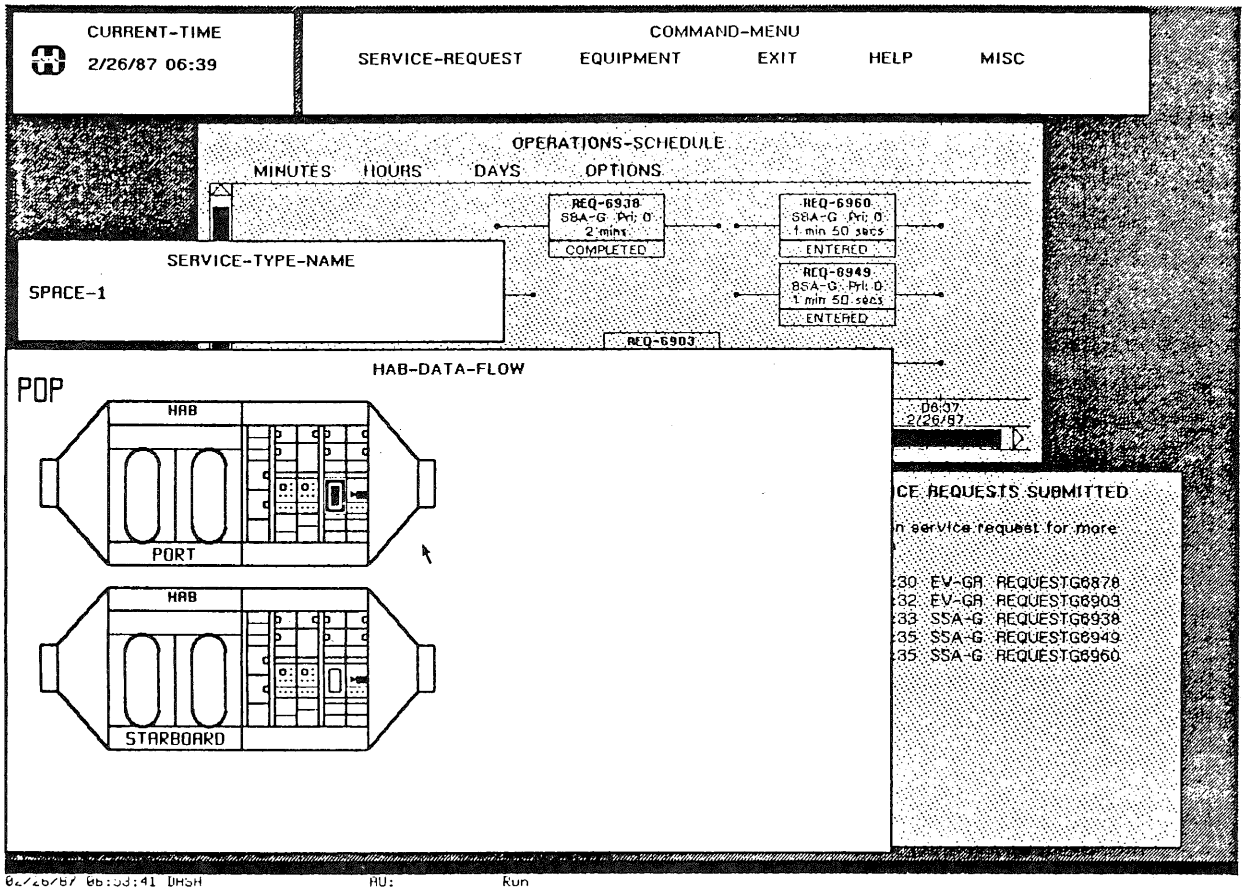

Figure 6. Hab Data-Flow

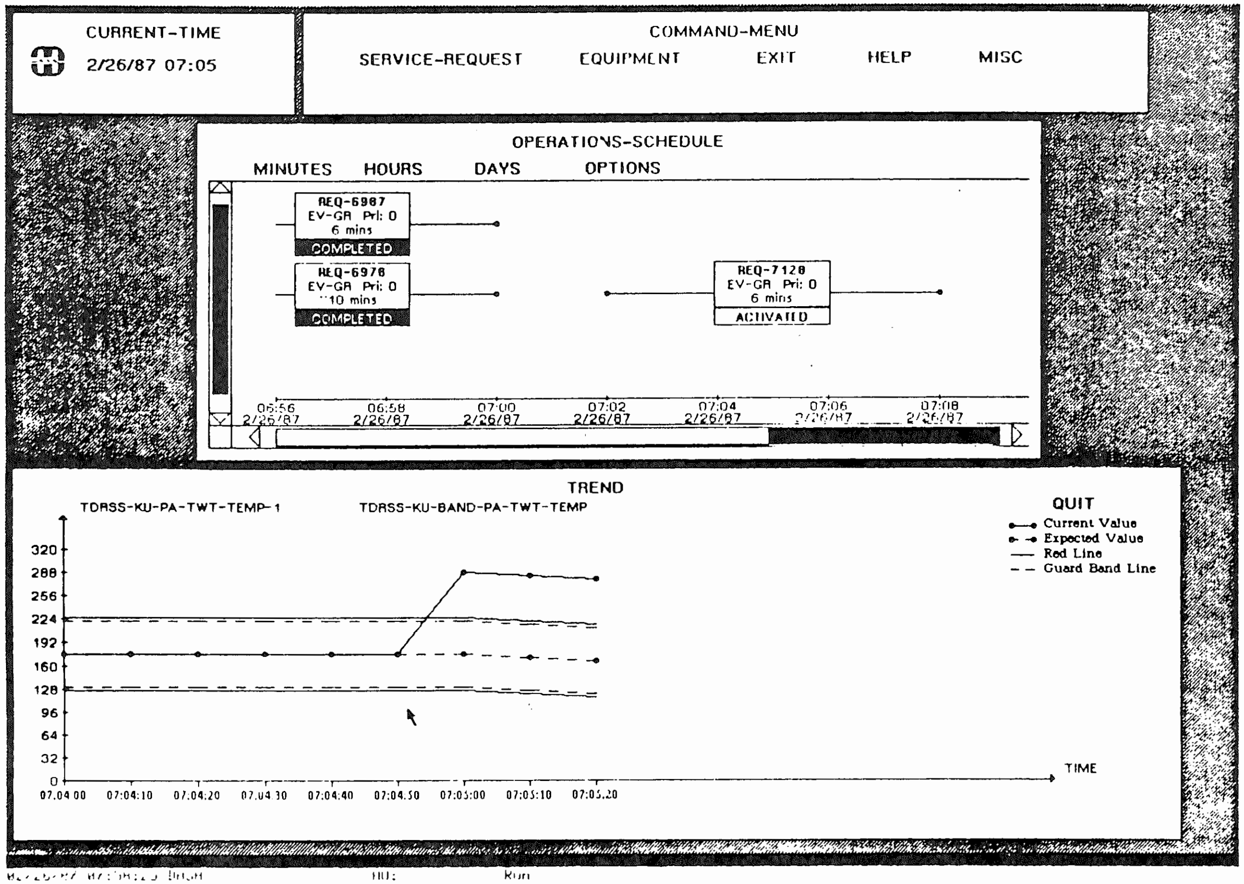

Figure 7. Trend Display for TWT Temperature

STATUS MONITOR DATA STRUCTURES

MESSAGE QUEUE (like simulator)
COMMAND TABLE (like simulator)

MEAS-DB

CURRENT-VALUE	7
EXPECTED-VALUE	8
ACCEPTABLE-VARIANCE	2
RED-LINE-OFFSET	3
GUARD-BAND-OFFSET	2
FILTER	GUARD-BAND
TREND-VALUES	((time curr-value exp-value) ...)
COUNT-OF-OK-MSMTS	value
REPORT-ON	trends, warnings, values

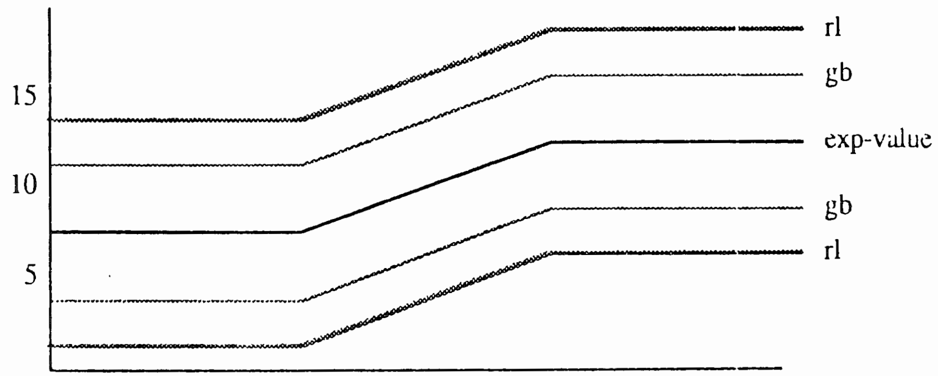

Figure 8. Status Monitor Data Structures