THE ISA EXPERT SYSTEM: A PROTOTYPE SYSTEM FOR FAILURE DIAGNOSIS ON THE SPACE STATION

Christopher A. Marsh
The MITRE Corporation
1120 NASA Road 1
Houston, Texas 77058

Abstract

The Mission Operations Directorate (MOD) at the Johnson Space Center (JSC) is responsible for the safe operation and mission success of manned space flights. The MITRE Corporation, working with MOD, is developing requirements for the Operations Management System (OMS) to automate many aspects of flight control for the Space Station onboard systems. To help develop these requirements, the Integrated Status Assessment (ISA) expert system prototype was built to perform Station-wide failure diagnosis. This paper describes the OMS, how the ISA prototype was built, and how the prototype is being used to help define the OMS.

Currently the Space Shuttle is managed on the ground using a hierarchy of control personnel. A flight director responsible for the overall Shuttle mission, a number of front room flight controllers are each responsible for a different system, and many back room controllers support various front room controllers. This system has been used since the early days of space flight and is very manpower intensive. A similar hierarchy is seen in an automated system for the Space Station. The back room controllers can be augmented with smart components and sensors on the Station and the front room controllers can have their jobs aided with expert systems. At the highest level is the OMS which works at the flight director level getting summary information from the various systems in the Station.

The OMS will include onboard automation, ground based automation, onboard manual operations, and ground based operations. One of the functions of the OMS is to perform a Station-wide status assessment and failure diagnosis of the Station. This Station-wide view is very important because all of the systems interact with each other and a failure in one system will have impacts on the other systems.

It became evident early in discussion with to the MOD flight controllers that prototypes would be very helpful in the process of gathering requirements for the OMS. The prototypes were used to gather additional requirements, to expose people to new ideas and technologies, and study the feasibility of these technologies for systems management.

Because the task of assessing the status of space vehicles is a complex job that requires "expert" knowledge to find heuristic solutions to problems, an expert system approach was chosen to prototype the ISA system. The initial domain of the expert system was the KU band portion of the communications and tracking system (one of the onboard "core" Space Station Systems).

The ISA prototype expert system consists of a knowledge base, an inference engine, and a user *interface. The ISA system was designed as a hybrid expert system using different methodologies: ob-*

ject-oriented programming, rule-based programming using both shallow and deep reasoning, and qualitative modeling.

The ISA prototype has been very successful in its goals. ISA has demonstrated new technology and ideas to the NASA operations community, helped define requirements for the OMS, and shown the feasibility of using hierarchical expert systems to monitor complex systems. The ISA prototype, along with other OMS prototypes, are being integrated in a test bed environment with other space station prototypes to form the Space Station Information System (SSIS) End-to-end Test Capability (ETC) which will have components all over the world. The results of the ETC will influence the Space Station design and be a test environment for software and hardware for years to come.

Introduction

The Space Station systems will be managed by NASA flight controllers in the Mission Operations Directorate (MOD) at the Johnson Space Center in Houston. Unlike most previous programs in which flight durations were approximately one week, the Space Station will be manned for thirty years. New methods, incorporating additional automation, are needed by MOD to manage the Space Station systems efficiently.

The Space Station Program will provide a comprehensive capability to explore and exploit the space environment. Conceptually, it will include a permanently manned central base of operations for research and development in low-earth orbit, co-orbiting platforms and polar platforms. The Space Station and associated elements will be capable of performing scientific, technical, and applications missions for research, commercial, and government customers.

The Space Station Program will evolve in an orderly manner from an initial capability in the mid 1990's to an expanded capability through the next century. The extended life of the program imposes a requirement to exploit changing technology and incorporate it into the Space Station. Thus, the Space Station must be capable of accommodating change to incorporate increased levels of automation.

The Space Station must be highly automated if it is to succeed. A high level of advanced automation will increase productivity, lower operating costs, increase flexibility, improve reliability, increase the feasibility of an autonomous Station, and reduce hazards to humans. (National Aeronautics and Space Administration Advanced Technology Advisory Committee, 1985) These benefits may be achieved by automating portions of tasks that otherwise would be performed by the crew or ground support personnel.

There has been a great deal of progress made in the past few years to bring expert systems into the commercial marketplace (Waterman, 1986). The work which was once done at research organizations is now being applied to real world problems such as planning, scheduling, interpretation, prediction, diagnosis, monitoring, instruction, and control. On the Space Shuttle these functions are being done primarily with highly skilled manpower monitoring all of the spacecraft's systems from the ground. The Space Station now being developed is planned to be operational for thirty years. It is clear to NASA that the Space Station operations can be greatly enhanced by the use of automation and expert system technology.

In 1985 MOD asked the MITRE Corporation to help develop requirements for system management of the Space Station. The MITRE task addressed Space Station systems control and monitoring. It helped develop concepts and requirements for the management of the Space Station onboard systems. The focus was to define the interfaces among the individual systems management functions and the interfaces between integrated systems management and the individual core systems (National Aeronautics and Space Administration, 1986; Spitzer, et al., 1987). This task has lead to the development of requirements for the Operations Management System which performs the integrated systems management function for the Space Station.

MITRE's approach to the task is based on extrapolation from the current systems management approach on NASA's National Space Transportation System (NSTS) Orbiter (Space Shuttle) to

what will be required for the Space Station. This involves comparisons between the way things are done today for Shuttle and the way things should be done for the Space Station.

This effort involved the development of several prototypes of the systems management functions (Harris, et al., 1987; Marsh, 1987). These prototypes include Integrated Status Assessment (ISA), Planning Support Environment (PSE), Procedures Interpreter (PI), On Orbit Maintenance (OOM), and Communications and Tracking System Management (C&T-SMA). These prototypes are being used to demonstrate new concepts, educate the users on available technology, and to further develop requirements for the OMS through comments and feedback from the user community. The prototypes will also be used to assess proposed designs for OMS implementation.

The Space Station Operations Management System

The Operations Management System (OMS) encompasses a set of interrelated functions associated with the coordinated management of the Space Station. To a large degree these functions are based on the experiences from the Space Shuttle and previous programs.

System Management on the Space Shuttle

In developing the systems management concepts the example of Space Shuttle flight control was used. The Space Shuttle is managed on the ground by a flight director responsible for the overall mission, a number of front room flight controllers each responsible for a different system, and many back room controllers who each support a front room controller (Training Division, Systems Training Branch, 1985). This approach has been used since the early days of spaceflight and is very manpower intensive. Figure 1 shows a diagram of Shuttle systems management.

Failure Analysis on the Shuttle

Each flight controller spends much of his/her time watching screens full of changing numbers

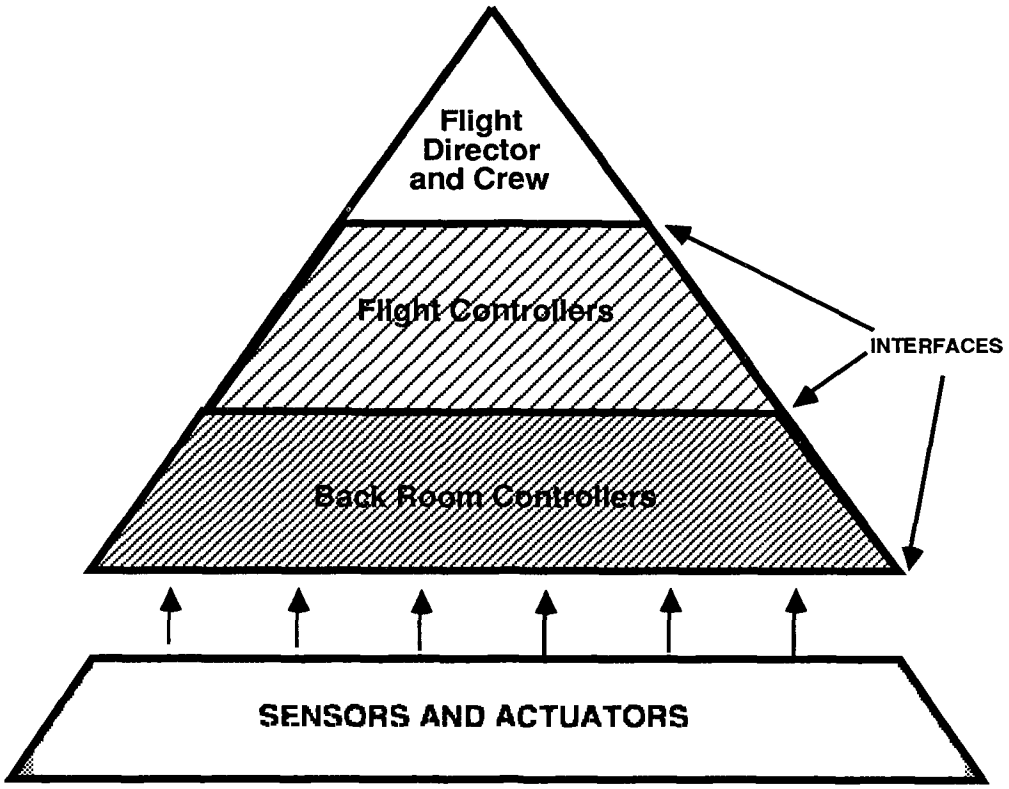

FIGURE 1
SHUTTLE SYSTEMS MANAGEMENT

representing sensor readings onboard the Space Shuttle (see Figure 2 for a typical flight controller screen). When a fault is detected, flight controllers refer to the malfunction procedures and flight rule books to guide them through the isolation of the fault and the reconfiguration of the Shuttle. These books are each hundreds of pages long and sit in book shelves behind each controller. In addition to the books on the ground the astronauts carry hundreds of pounds of material in the Flight Data File (FDF) on each flight to guide them on the operation of the Shuttle. Failure analysis on the Shuttle is very manpower intensive and automation can play a major role in supporting the flight controllers in the future. Much can be learned from the way systems management and failure analysis is done on the Shuttle for the Space Station program.

Space Station Systems Management Hierarchy

Systems management encompasses those functions necessary to safely control and monitor the systems and elements for accomplishment of Station objectives. A hierarchy similar to what is used for Space Shuttle flight control will be used for the Space Station.

This systems management hierarchy for the Space Station will have three levels (see Figure 3). The OMS will perform systems management at the highest level across all systems. This level of management is similar to the management performed by the astronauts on the Shuttle and controllers in the Flight Control Room. Each system will have its own System Management Application (SMA). Each subsystem within a system will have its own System Operations Application (SOA). Management at these lower two levels is similar to what is performed by the controllers in the Multi-Purpose Support Rooms and the other support rooms. Systems management relies upon knowledge of the systems and their operation. As experience with Space Station operations is gained, the applications performing systems management must be refined to incorporate the new knowledge.

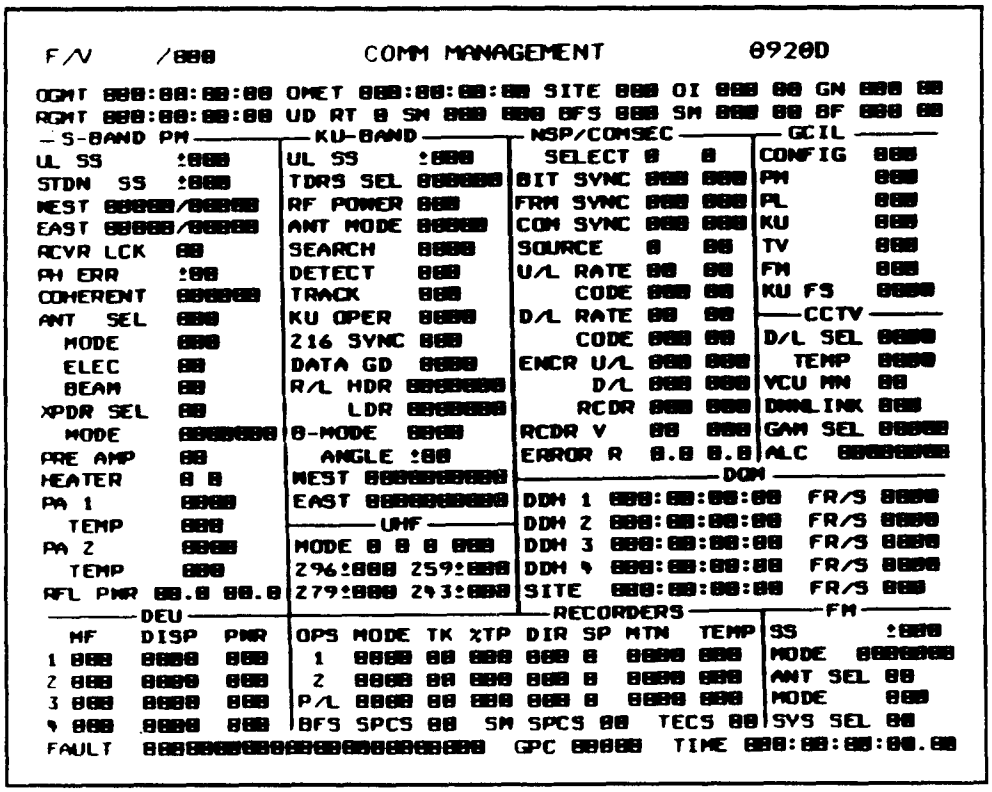

FIGURE 2
FLIGHT CONTROLLER SCREEN

OMS

Because of its position in the systems management hierarchy, OMS will integrate operations across all the Station systems and elements. It will include onboard automation, ground based automation, onboard manual operations, and ground based manual operations. OMS will perform high level functions including station planning, evaluating station status, fault management, system/ payload testing, command management, inventory management, maintenance, and training. The intention of OMS is not to eliminate the work of humans, but to enhance it.

SMA

Each core system SMA will perform systems management functions by integrating the operations of its various subsystems. Each core system performs functions such as fault management, self-safing, subsystem coordination and status reporting to OMS. The several SMAs should be developed in cooperation with the companion Station core systems to facilitate knowledge capture using design and test results.

SOA

Each subsystem SOA will perform basic systems management functions on the system unit level. A system unit is a concentration of components belonging to the same system. A system unit may be a subsystem or a collection of components of one system in the same vicinity. The various SOAs will perform functions such as telemetry range checking, command activation and feedback, calibration of raw data, translation of raw data to engineering units, etc.

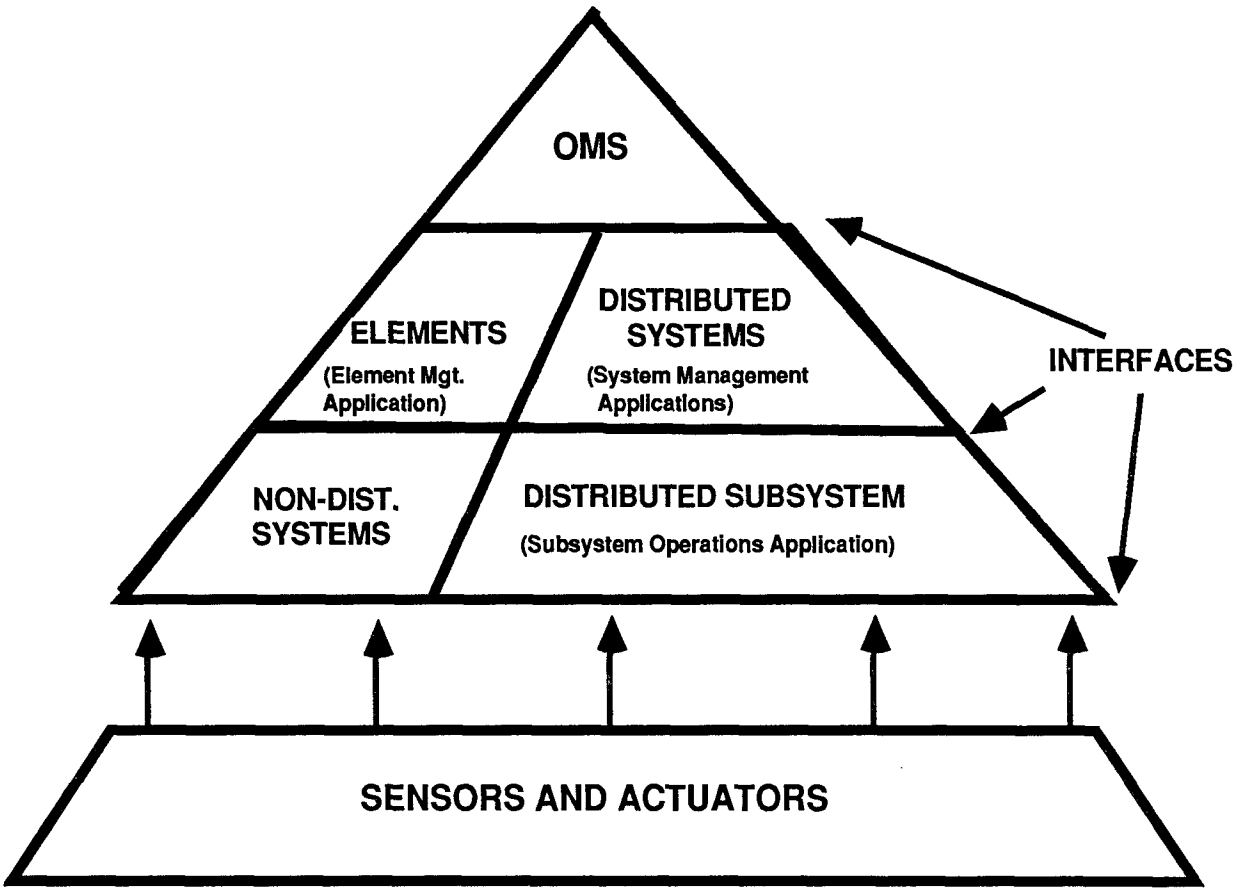

FIGURE 3
SPACE STATION SYSTEM MANAGEMENT

The OMS Functions

Once the need for the highest level of systems management (the OMS) was identified, NASA formed the OMS Working Group. This group is composed of people from MOD, the Engineering Directorate, and the Mission Support Directorate (MSD). The OMS working group defined the OMS and broke it down into eleven major functions in the OMS Definition Document (Operations Management System Working Group, 1986). The OMS functions include:

- Develop, manage and update the short term plan.
- Execute the short term plan.
- Monitor system and payload status.
- Manage inter-system and payload testing.
- Maintain and log global configuration, activity, and state information.
- Detect and manage resource conflicts.
- Manage global caution and warning.
- Perform global fault management and recon figuration.
- Support command and uplink/downlink management (transaction management).
- Provide global inventory and maintenance management system.
- Support onboard training and simulations.

The MITRE prototype work preceded these OMS functional definitions. As a consequence, the prototypes and the definitions do not map on a one-to-one basis. The ISA prototype performs portions of functions 3, 5, 7, and 8 (monitors Station status and configuration, manages caution and warning, and performs global fault management). The PSE prototype performs function 1 (updates the short term plan). The PI prototype performs function 2 (executes procedures referenced in the short term plan). The OOM prototype performs function 10 (inventory and maintenance). Studies were made on function 6 (resource management) (Hammen, 1987), portions of function 8 (Station reconfiguration), and function 11 (training). No prototypes or reports were developed by MITRE on function 9 (transaction management).

The OMS and Failure Diagnosis

When a failure occurs on the Space Station, OMS functions 3, 5, 7, and 8 will be performed. OMS functions 3 and 5 monitor the Station's systems at a high level and log the failure. Function 7 would annunciate failure through the station caution and warning system. Function 8 would isolate the failure and recommend any necessary reconfiguration to maximize the objectives of the Station. Today on Shuttle, all these functions are performed by the astronauts, the flight director, and the front room flight controllers. Through the use of expert system technology and automation, these functions can be made more efficient and the work of the ground and flight crew can be greatly enhanced. The ISA prototype illustrates this type of automation by performing all of these functions except recommending a new Station configuration.

Approach

Early in the process of interviewing the MOD flight controllers to gather system management requirements, it became evident that a prototype would be very useful. One of the goals of the prototype was to gather additional requirements by letting users see ideas and give feedback. This included such things as defining the boundary between the OMS and the systems. The prototype was also used to expose the operations community to new technologies and study the feasibility of these technologies for system management.

Because the task of assessing the status of space vehicles is a complex job that requires "expert" knowledge to find heuristic solutions to problems, an expert system approach was chosen to prototype the ISA system. This section will describe expert systems and knowledge engineering in general, and then the initial domain of knowledge chosen for the ISA prototype.

Expert Systems

An expert system is a software system in which specialized problem solving expertise from a human "expert" has been coded in the form of knowledge. For the ISA expert system NASA flight controllers were the "experts".

In a rule based expert system the knowledge is coded in the form of IF-THEN rules and facts. The overall control of a rule-based system is governed by a software component called the inference engine which does the matching and execution of the rules. For the ISA expert system, flight controllers were questioned on how they performed failure analysis on the Space Shuttle, and their responses were used to construct the prototype.

Knowledge Engineering Process

The process of building an expert system is known as knowledge engineering. Knowledge engineering involves the acquisition of knowledge and putting it into a useful system. This section will describe that process in general and then specifically focus on the ISA expert system.

Knowledge Acquisition

There are five stages to knowledge acquisition (Hayes-Roth, Lenat, Waterman, 1983). Identification is the first step and involves identifying problem characteristics to define preliminary requirements which are used for the next step, conceptualization. Here concepts are developed to represent the knowledge. These concepts lead to the next step which is formalization. Formalization involves designing a structure to organize the knowledge. This structure is then used for the implementation. In a rule based expert system the implementation will involve the formulation of rules to embody some of the knowledge. The last step is testing. In testing the system the "expert" looks at what the knowledge engineer has done and critiques the system. Reformulations, redesigns, and refinements are often needed in the system and the above steps are repeated until the desired system is completed.

ISA Knowledge Engineering

The knowledge engineering process took several months for the ISA prototype with the expert spending about a half day per week suggesting changes to the system. After most of the knowledge engineering process was completed an elaborate user interface was added to the system. Next, many other operations people were shown the prototype and their ideas and knowledge were used to further refine the system. Requirements learned from this process were input to the OMS working group and later turned into Space Station requirements.

DOMAIN

The initial domain for the ISA prototype was the communications and tracking KU band system and that system's interfaces. This included the power busses, cooling loops, Tracking and Data Relay Satellite System (TDRSS), and the interface to the DMS. This area was chosen for several reasons:

First, while the design had not been chosen, it would have to resemble the Shuttle system because both systems would use the TDRSS. Therefore it was possible to derive a reasonable schematic of the system.

Second, it contained the intersections of several systems; a fault in one system could cause other systems to malfunction.

Third, a flight controller with expertise in this area was eager to work with us.

KU Band

The KU band subsystem consists of two redundant strings of components (String A and String B) for both uplink from the ground and downlink to the ground (see Figure 4). These components can be cross-strapped when a failure occurs. The components from one string are tied to one power bus while the other string is tied to another power bus.

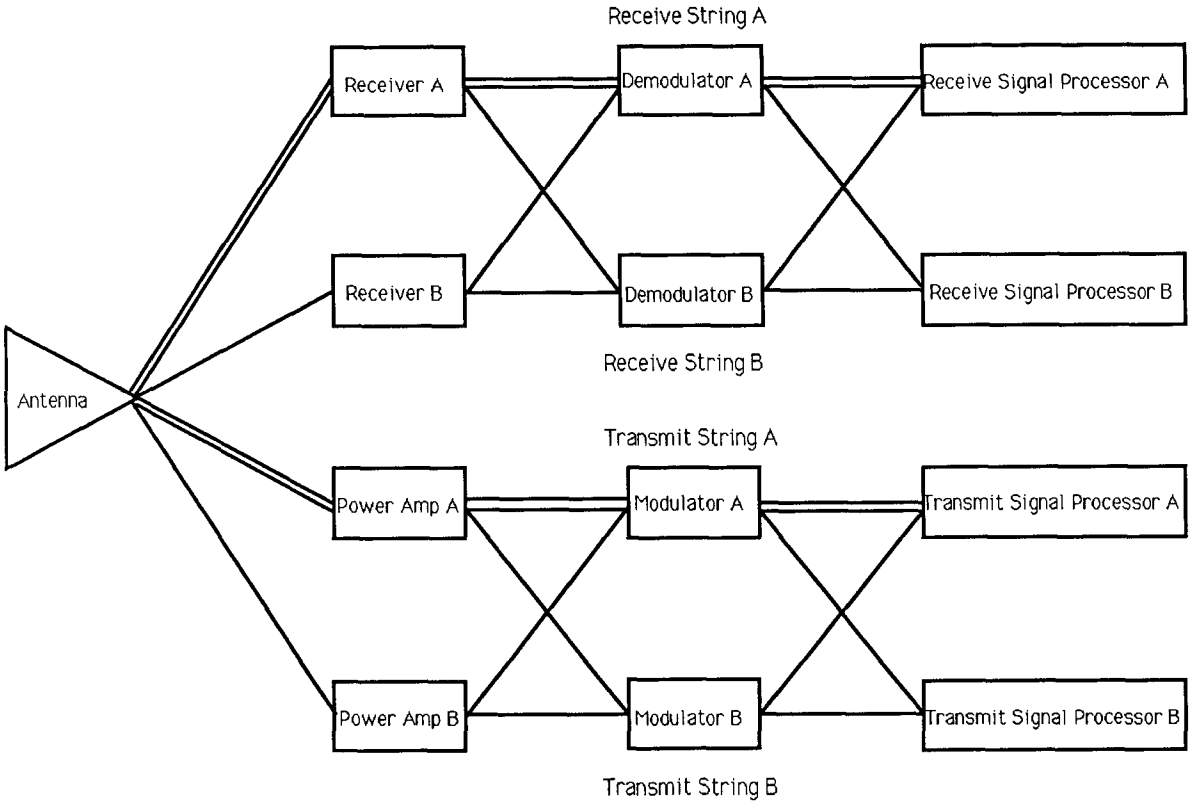

FIGURE 4
KU BAND SUBSYSTEM

Communications System

The communications system will function like an assembly line with signals being passed from one component to the next. For the receive string, a signal starting from the ground which contains multiple "messages" for applications on the Station will be sent up to the TDRSS. From the TDRSS it would be relayed to the Station. At the Station it would be picked up on the antenna and sent to one of the two receivers. The receiver would tune in the signal and send it on to one of the two demodulators. The demodulator would make the signal a simple baseband signal and then send it to one of the two receive signal processors. The receive signal processor finally splits up the signal into its component "messages" that could then be sent to the proper application on the Station. If any one of the components in the string failed, the "messages" would not be received and a failure would be detected. There has to be enough sensor information on the components to detect where the signal stopped being processed. The receiver, demodulator, and signal processor all would have redundant units that could be "switched in" if a unit failed. The transmit string will work in the same manner as the receive string except that the signals go from the Station to the ground.

OMS - System Interaction

One of the goals of the prototype was to better define the hierarchy of responsibility between the OMS and the Space Station systems. The systems would be responsible for gathering the data from the sensors and some level of data processing. The OMS would be responsible for the high level decision making based on information from all the systems.

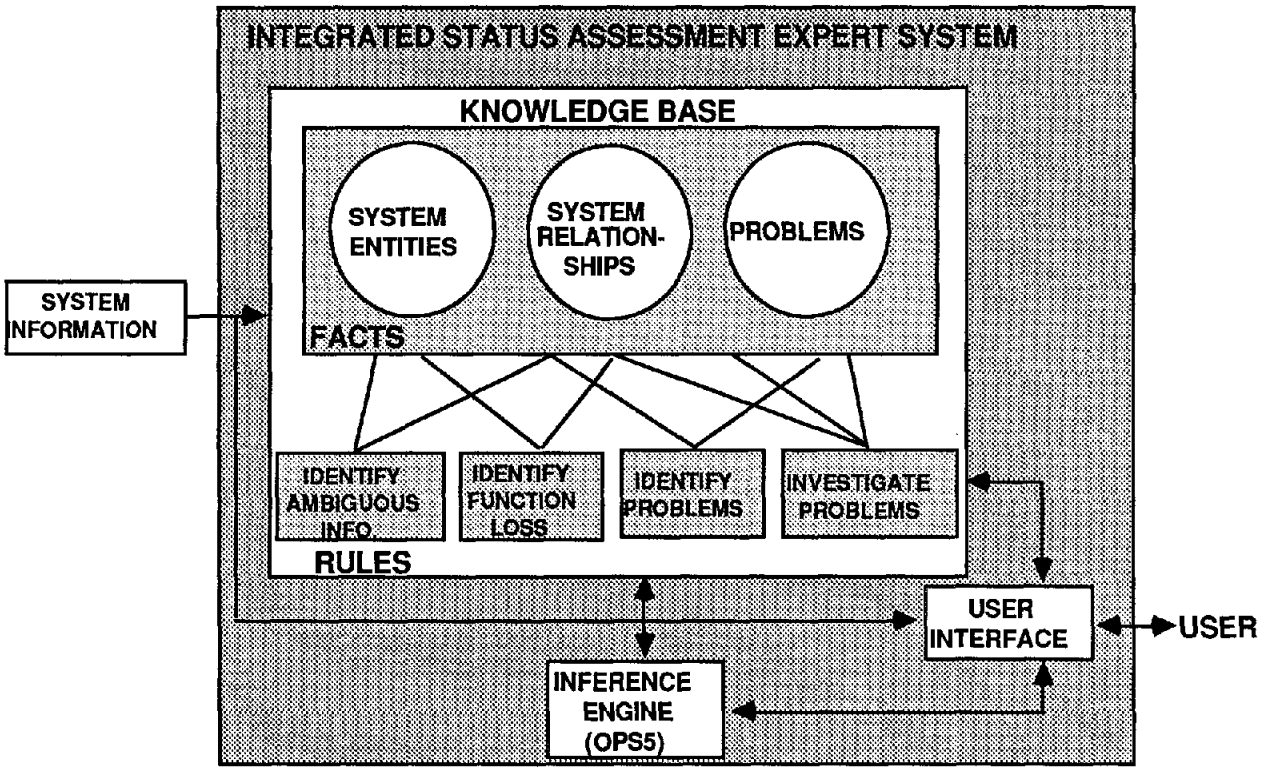

FIGURE 5
ISA PROTOTYPE

Design

The ISA prototype expert system consists of a knowledge base, an inference engine, and a user interface. The knowledge base contains the facts about the Space Station and rules to perform fault isolation. The inference engine controls how the knowledge base interacts with itself. The user interface gives the user overall control and allows the user to examine and modify the knowledge base. Figure 5 contains a diagram of the system.

The ISA system was designed as a hybrid expert system using different methodologies: object-oriented programing was used to describe the different parts of the Space Station; rule-based reasoning was used to encode the knowledge on how to perform the failure diagnosis; and qualitative modeling was used to describe the state of the various components. These methodologies used together make a flexible model-based expert system which can perform diagnosis over different domains. The final package was tied together with a user friendly interface. The ISA prototype was hosted on a symbolics 3600 series computer and written in ZetaLisp and OPS5.

Object-Oriented Design

Object-oriented programming is a technique popular in building expert and knowledge-based systems in which the common elements of similar objects are grouped together using frame hierarchies for inheritance and procedural attachment (Cox, 1986). Common elements do not need to be repetitively associated with each individual object; instead, the common elements of an object are inherited from a more abstract generic object. Only the association of the individual object with the abstract generic object needs to be noted. The similar individual objects are often called instances of the generic object. Generic objects may themselves be instances of a more generalized generic

object. When a system has several similar but not identical objects, the use of inheritance to model these objects provides a significant reduction in both the size and complexity of the system.

The two main types of objects used in the prototype are entities and relationships. Entities represent major system components and relationships represent the connections among these components. All entities and relationships can be represented at different levels of abstraction. This way a user can look at the Station's components at a high level for normal operations and focus down into a specific area for off-nominal situations.

Entity-Relationship Model

The component status and configuration information was represented in an entity relationship model. Entity-relationship modeling is currently being researched and used in state-of-the-art data base design (Ross, 1987). In an entity-relationship model the subject area of interest is represented as entities and relationships among the entities. The facts portion of a knowledge base is essentially a data base and therefore lends itself nicely to the new entity relationship techniques. By using the entity-relationship knowledge representation, the structure and behavior knowledge is segregated from the trouble shooting heuristics. The ISA prototype has two basic object types; entities for components, and relationships for connections between components.

The data structure for a entity (component) is:

NAME (The component's name)
STATE (The state of the component (nominal, suspect, or failed))
MODE (The operational mode (on or off))
PERFORMANCE (The component's performance)

The values of a component's name, state, and mode attributes are used in locating faults.

The data structure for a relationship between components (entities) is:

FROM (The "from" component's name)
TO (The "to" component's name)
TYPE (The type of relationship)
STATE (The state of the relationship)
DELAY (The time delay for the relationship)

The values of a relationship's from, to, type, state, and delay attributes are used in locating faults.

Multiple Levels of Abstraction

The ISA prototype supports multiple levels of abstraction. The system groups entities (components) together by using their class value. The value of an entity class is itself a name of a higher level entity (system). The higher level entity is made up of all the entities which have the higher level entity as their class value. For example, the communications system entity is made up of smaller entities such as receivers, and signal processors. A receiver will have as its class value "communications system". There is no limit to the number of levels the prototype can handle. Whenever an entity goes to a suspect or failed state at one level, all higher level entities go to a suspect or failed state. This allows the user to look at a system at any level of detail provided the model is there to support it.

Rule Based Reasoning

One technique used in building expert systems is rule based programming. Rule based programming involves coding the knowledge into IF-THEN rules and facts. The rules are the knowledge "rules of thumb" that an expert uses in solving a problem. The left-hand side of a rule is the condition part or situation. The right-hand side of a rule is the action part of a rule or consequent.

The left-hand side of a rule is usually a Boolean combination of clauses. The types of Boolean operators vary with the system used. Many languages allow only the operators AND and NOT.

The right-hand side of a rule, or action part, is generally a list of modifications to be made to

data memory when the rule is executed. These actions usually add, modify, or delete facts and they may also perform operations such as reading or writing to an input/output device.

The overall control of a rule-based system is governed by the inference engine. The inference engine executes the rules by matching the condition part of a rule with the current facts, and then performing the action part of the rule. If multiple rules can be executed at the same time, then the inference engine does a selection using a control strategy or conflict resolution.

Shallow Reasoning

Traditionally, diagnostic expert systems have used "shallow reasoning" to perform failure analysis. This means that the rules about the system's behavior are not deter-mined from the structure of the system. Diagnostic expert systems in the medical area typically do not know how a human is built. The Shuttle malfunction procedures are written in the form of a shallow reasoning expert system. In building the ISA prototype, the rules were initially coded from the malfunction procedures used in Shuttle. This design approach was quickly dropped because the number of rules grew for each component that was added to the system. Instead, a hybrid design was used that has deep reasoning rules to narrow down the problem to a specific component and shallow reasoning rules to reason about that component. The shallow reasoning rules can handle many situations that evade deep reasoning rules which depend upon specifics about a component's structure.

Deep Reasoning

Today, there are expert systems being built that use "deep reasoning" to perform diagnostic functions (Bobrow, 1985). These expert systems reason from the structure of the system being diagnosed. Each component has some associated

The ISA prototype has the Space Station design knowledge built into the facts portion of the knowledge base. This allows the use of deep reasoning for general purpose troubleshooting. The troubleshooting rules are general purpose and are not component specific. This greatly reduces the size of the rule portion of the knowledge base.

Given enough state information on the system components, the troubleshooting rules can find the source of faults in a system. They can also detect ambiguous information that would be the result of incorrect state information on a component. This is a common occurrence on the Space Shuttle when sensors fail. Once the source component of a failure is found the more traditional shallow reasoning rules can be invoked.

Trace and Explanation Facility

The inference engine used for the prototype has three levels of trace which can be used to follow the firing of rules. The highest level shows the name of the rule that fires and all facts that change in memory. The second level of trace only shows the name of the rules that fire. The lowest level of trace gives no information on the rules or the facts being used in the knowledge base.

The ISA prototype also has special LISP functions that control the inference engine and print out English explanations of the reasoning.

Both the trace information on the rules and facts and the English explanations can be printed to a file for later examination by the user.

Qualitative Model

The ISA prototype uses a qualitative model to describe a system's behavior. A qualitative model is one which uses terms such as "good" or "bad" to describe components instead of numerical information such as "3.54 MHz" or "34.5 degrees C" that would be used in a quantitative model. The SOAs are responsible for gathering the sensor data, converting it into engineering units, and doing limit sense operations. The SMAs perform reasoning, logging

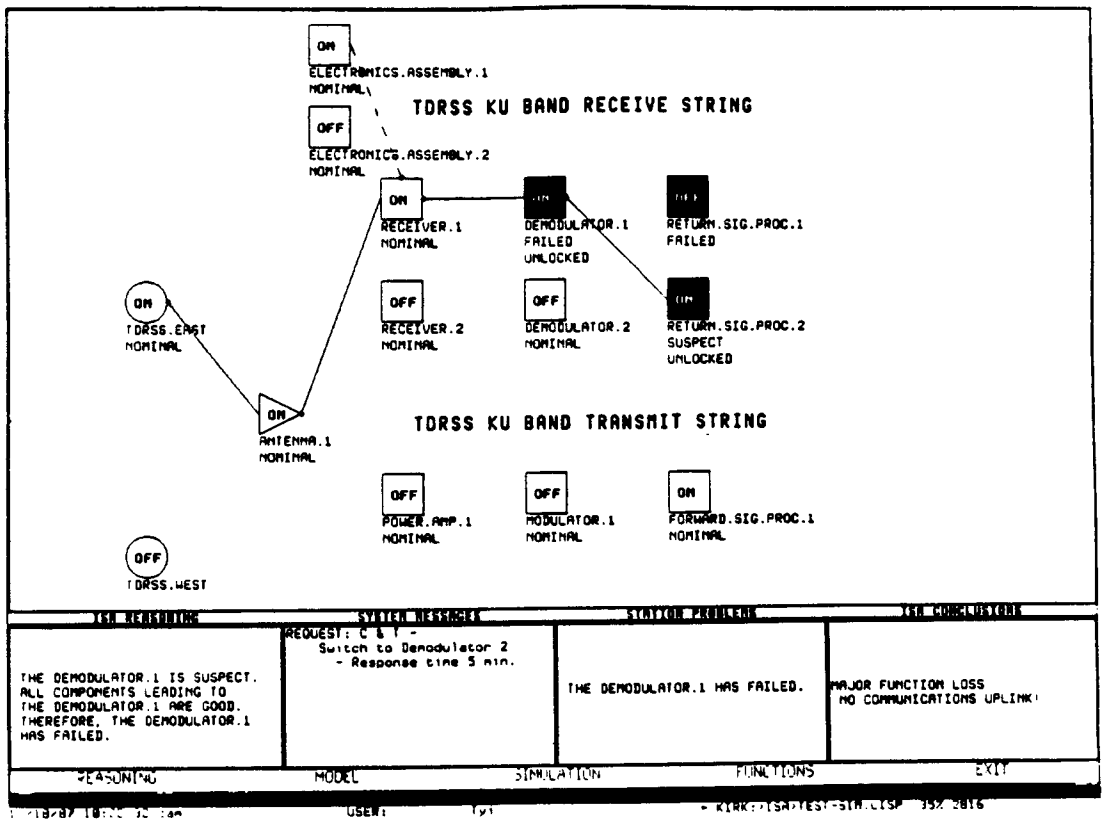

FIGURE 6
ISA PROTOTYPE SCREEN

the data, and finally determining the qualitative terms to best describe each part of the system and giving that information to the OMS. The ISA prototype takes that information to determine the source of failures.

The qualitative model used in the ISA prototype uses the terms "nominal", "suspect", and "failed" to describe the state of a system component. Nominal means that the component is operating normal and everything is fine. Suspect means that the component is not operating properly. A suspect component may be failed or it may be not operating properly because another component connected to the suspect component is not operating properly. A failed component is one which has been found to be the source of a failure.

By following through the suspect components using their connections to each other, the ISA prototype can trace back to the failed component.

User Interface

The ISA prototype has an elaborate user interface to build system models, observe system operations, and to control and observe the expert system operations. A keyboard, mouse pointing device, and a high resolution display area with windows and mouse sensitive areas are used as the interface to the user. Refer to Figure 6 for a view of the user interface.

A large area in the center of the display is the system diagram window and is used to show system schematics at different levels of detail. This window is actually a viewport onto an infinitely larger window. The user can pan the displayed area around to look at different areas in the system. There are multiple numbers of system diagram windows that the user can look at, each of which has a different system displayed. Only one system diagram window can be viewed at a time. Each component on the system diagram window is mouse sensitive. The user can display additional information on the component, the component's

relationships to other components, and move up or down levels of abstraction by "clicking" the mouse on a component. The user can also change system information including its graphical representation by using the mouse.

The four smaller windows below the main center display are used to display information on the expert system reasoning, display messages from the systems, show identified failures and show final conclusions.

At the bottom of the display are five command items which lead to other menus to operate the prototype. The reasoning command brings up a menu that explores the knowledge base and runs the expert system. The model command brings up menus to create, delete, and modify system components. The simulation command brings up a menu which allows the user to run the prototype in a simulation mode where it will update information on the screen based on data found in a file on the network and then make assessments based on this new information. The functions command brings up a menu of commands for moving around the main display screen to different portions of the knowledge base. The exit command lets the user leave the prototype to a main Space Station display screen.

Building System Models

The ISA prototype required an easy method of building and modifying system models. There are a large number of system components that will exist in the Space Station and the concepts for the design of the Space Station are rapidly changing. By using the menus and the mouse pointing device, models can be easily created, modified, saved, and reloaded into the prototype. A library of graphic symbols exists to represent components on the schematics and the user has a number of options for displaying the connections between components.

Operation

The ISA prototype has several modes of operation. In the simplest mode faults can be manually inserted into a system and a diagnosis run to determine the source of the failure. In the more complex modes, simulated or real system data can be read into the system through the file system and the model will be updated to reflect the new information. After the model is updated, the expert system will be activated to perform a diagnosis. The user has full control over the cycle time to read data and make a system diagnosis including the option to single step through the process.

Conclusions

The ISA prototype has been very successful in its goals, demonstrating new technology and ideas to MOD, defining requirements for the OMS, and showing the feasibility of using hierarchical expert systems to monitor complex systems. The ISA prototype, along with other OMS prototypes, are being integrated with other system prototypes to form the Space Station Information System (SSIS) end-to-end test capability which will have components all over the country.

Results

The ISA prototype was built with modeled portions of the Communications and Tracking System, the Electrical Power System, the Data Management System, and the Environmental Control and Life Support System. It has been demonstrated to many groups at NASA including Astronauts, flight controllers, engineers, and many different potential Space Station contractors. At each demonstration new ideas to improve the prototype and drive new requirements were solicited. These ideas and requirements were incorporated into NASA documentation (National Aeronautics and Space Administration, 1986) and the OMS definition (Operations Management System Working Group, 1986).

The OMS definition document was incorporated into the Space Station Architecture Control Documents (ACDs).

In early 1987 the ISA prototype was tied to a Communications and Tracking System Management prototype. This demonstrated the systems management hierarchy concept, the use of qualitative models, and the feasibility of cooperating expert systems across a network.

To demonstrate the prototype's use of entity-relationship modeling, qualitative modeling, and deep reasoning across different domains, a model of a nuclear plant was built with data to drive the simulation.

Because of the success of the ISA prototype, MOD has funded work to install automation and expert systems into the existing Space Shuttle control room. This work, the INCO expert system, will involve providing automation aides to the task performed by the INCO position in the MCC.

Currently the ISA prototype together with the Procedures Interpreter (PI) prototype are communicating on the DMS Testbed to the Guidance Navigation and Control (GNC) emulator Testbed. Figure 7 shows this integrated configuration. Knowledge about GNC configuration and fault management have been put into ISA system. This information along with GNC jet operations data and status and recommend action during a reboost scenario. Integrated tests and demonstrations between the OMS prototypes and GNC systems are now taking place.

The ISA prototype has demonstrated new technology to the NASA operations community, helped define requirements for the OMS, and shown the feasibility of using a hierarchy of expert systems to monitor complex systems.

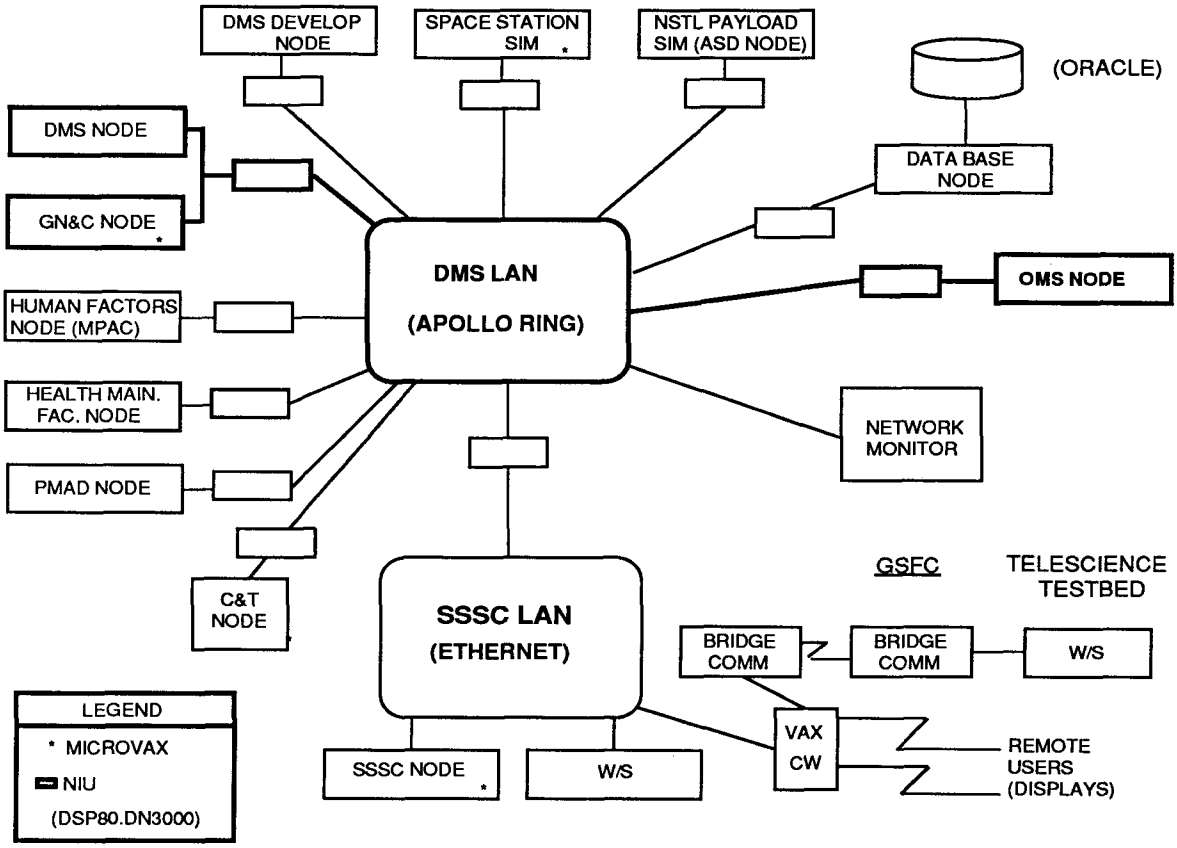

FIGURE 7
OMS TO GNC CONFIGURATION

Future Plans

The ISA prototype will tie together with other Space Station system prototypes through the DMS Testbed. The Thermal system, Communications and Tracking Testbed, Power Management and Distribution models, Payloads, and the European Columbus module all have plans to tie into the OMS prototypes. All these system ties together will form a Space Station Information System (SSIS) End-to End Test Capability (ETC). The ETC will become an environment to try out system management concepts as well as communications protocols. The first Phase of this work is well under way with the GNC Emulation Lab. The next phase will start later this year and will involve the investigation of the use of the Ada as a host language for the OMS.

The results of the SSIS end to end testbed project will influence the Space Station design and be a test environment for hardware and ideas for years to come. From what is learned in the test bed effort, additional requirements will be generated for the phase C/D Space Station contractors. The prototypes in the test bed will also act as gauges to evaluate potential OMS designs from the final contractors. Finally the lessons learned from this entire effort will be used in monitoring the actual implementation of the OMS in the Space Station and on the ground.

List of References

Bobrow, Daniel G., (1985) Qualitative Reasoning About Physical Systems, The MIT Press.

Cox, Brad J., (1986) Object Oriented Programming, An Evolutionary Approach, Reading, MA: Addison Wesley.

Hammen, David G., (1987) Space Station Resource Management, MITRE Working Paper, Houston, TX: The MITRE Corporation.

Harris, R. A., Marsh, C. A., Reynolds, S. J., Muratore, J., (1987) "Hierarchical Expert Systems for Space Station Systems Management", Poster Session Abstracts, The IEEE Computer Society.

Marsh, C. A., (March 12, 1987) "ISAES - An Expert System for Evaluating the Condition of the Space Station", Proceedings of JAIPCC '87 Mini-Symposium, Instrument Society of America, Clear Lake - Galveston Section and The Institute of Electrical and Electronics Engineers, Inc., Galveston Bay Section.

National Aeronautics and Space Administration Advanced Technology Advisory Committee, (1985) Advancing Automation and Robotics Technology for the Space Station and for the U.S. Economy, NASA TM-87566, Houston, TX: NASA Johnson Space Center.

National Aeronautics and Space Administration, (1986) Space Station Systems Operations Concepts for Systems Management", JSC-20792, Houston, TX: NASA Johnson Space Center.

Operations Management System Working Group, (1986) Operations Management System Definition Document, Houston, TX: NASA Johnson Space Center.

Ross, Ronald G., (1987) Entity Modeling: Concepts and Application, Boston, MA: Database Research Group, Inc.

Spitzer, J. F., et al. (January, 1987) Overview of AI Applications for Space Station Systems Managementt, AIAA 25th Aerospace Sciences Meeting, Reno, Nevada.

Training Division, Systems Training Branch, Mission Operations Directorate - Orientation Manual, JSC 20349, Houston, TX: NASA Johnson Space Center.

Waterman, D. A., (1986) A Guide to Expert Systems, Reading, MA: Addison Wesley, 1986.