A Failure Diagnosis and Impact Assessment Prototype for Space Station *Freedom*

Carolyn G. Baker and Christopher A. Marsh

The MITRE Corporation
1120 NASA Road One
Houston, Texas 77058

Abstract

NASA is investigating the use of advanced automation to enhance crew productivity for Space Station *Freedom* in numerous areas, one being failure management. This paper describes a prototype that diagnoses failure sources and assesses the future impacts of those failures on other *Freedom* entities.

Keywords: Failure Diagnosis, Failure Management, Artificial Intelligence

Introduction

This paper addresses the application of Artificial Intelligence (AI) techniques to failure diagnosis and impact assessment. The Diagnostic Reasoner (DR) is an existing prototype that accepts qualitative status reports that reflect significant changes detected by subordinate systems, determines the likely sources of failures, and projects the impacts of those failures onto other *Freedom* entities. The DR uses AI techniques such as pattern matching, symbolic reasoning, and model-based reasoning to produce a diagnosis to either confirm or correct the subordinate systems' diagnoses.

Background

The National Aeronautics and Space Administration (NASA) is in the process of designing and implementing a permanently manned space station, Space Station *Freedom*, to be put into low earth orbit. In support of Johnson Space Center's (JSC) Automation and Robotics Division, MITRE is developing a proof-of-concept failure management prototype that demonstrates the use of expert system software to assist the onboard crew via decision support in recovering from failures in a dynamically changing, controlled environment. (Baker, 1990)

Space Station *Freedom* has been defined to have a hierarchical, distributed command and control architecture. The highest level, or Tier I, in the architecture is concerned with global, vehicle-wide issues, and includes automated and manual operations, both on the ground and onboard. Onboard Tier I functions include the failure management function. Tier I entities have knowledge of each of the systems and elements, and how these interact. Tier II entities are systems delineated along functional boundaries, such as the Electrical Power System. Each Tier II entity is controlled by a Tier II manager that has knowledge about its own status, but not of the other Tier II entities. With no direct knowledge of other Tier II entities, each Tier II manager produces and reports dynamic data representing the current status and configuration of its components to Tier I for a global assessment of the enterprise as a whole. (NASA, 1989)

The failure management prototype described comprises two components: the DR diagnoses and analyzes the failure; the Recovery Expert (Rx) formulates sets of possible actions to take (Courses of Action) for recovery. The DR and the Rx act at the Tier I level for failure diagnosis and recovery. This article describes the DR; a companion article (Hammen, 1991) focuses on the Rx. The DR and the Rx are not completely automated functions, but are computerized decision aids in which the user is provided interaction capabilities throughout the failure management process.

We adhere to the significant architecture requirements imposed by *Freedom's* environment, as we hope to influence *Freedom's* design with our approach. In particular, the prototypes rely on the existence of hierarchical, distributed management functions. Constraints placed on the prototypes shape the architecture of the software, but do not limit its usefulness to *Freedom* or to space-related domains. This software can be used in other domains without changing the computational portion of the application, instead changing only the data that describe the environment.

The technologies used in the implementation of the failure management prototype are model-based reasoning (the DR) (Davis, 1984) and goal-directed planning (the Rx). The objective of the prototype is to show how we use these technologies, and how the technologies work together synergistically using the same knowledge representation for both the DR and the Rx. The purpose of developing this prototype is to address a new approach to the problem, to apply advanced technologies to it, to examine all aspects of failure management in one integrated application, and to do this in a general way so that the same technology can be used in other domains. The prototype was developed on VAXstation™ 3100 workstations under VMS™, using ART™/Ada and TAE Plus™.

Figure 1 shows an overview of the environment in which the DR and the Rx prototypes operate. The DR receives Tier II reports of significant, qualitative changes. The DR identifies the suspected causes of the failure, and determines the current and possible future impacts these suspected failures are likely to cause on other *Freedom* entities. Once the DR has diagnosed the problem and determined the immediate and downstream effects and impacts of the failure, the Rx uses this information to formulate solutions, or Courses of Action, for recovery. Each Course of Action deals not only with the immediate problem, but also determines what actions to take to mitigate the constraints imposed by the failure. The Procedures Interpreter, an earlier prototype not described in this paper, executes the pre-defined procedures identified in the selected Course of Action, resulting in commands to the Tier II managers. The effects of these commands on the Tier II systems is, in turn, reported to the DR as changes in status or configuration, thereby closing the loop between Tier I and II.

DR Overview

The DR is responsible for determining the likely sources of a failure, and projecting the future impacts of the failure on the rest of the station. The DR uses status reports from the Tier II managers along with structural and behavioral data contained in the Component Model to generate lists of suspects that might be responsible for the failure. These Suspect Lists are updated, merged, and deleted to show the current reasoning of the DR. For each suspect identified, the DR

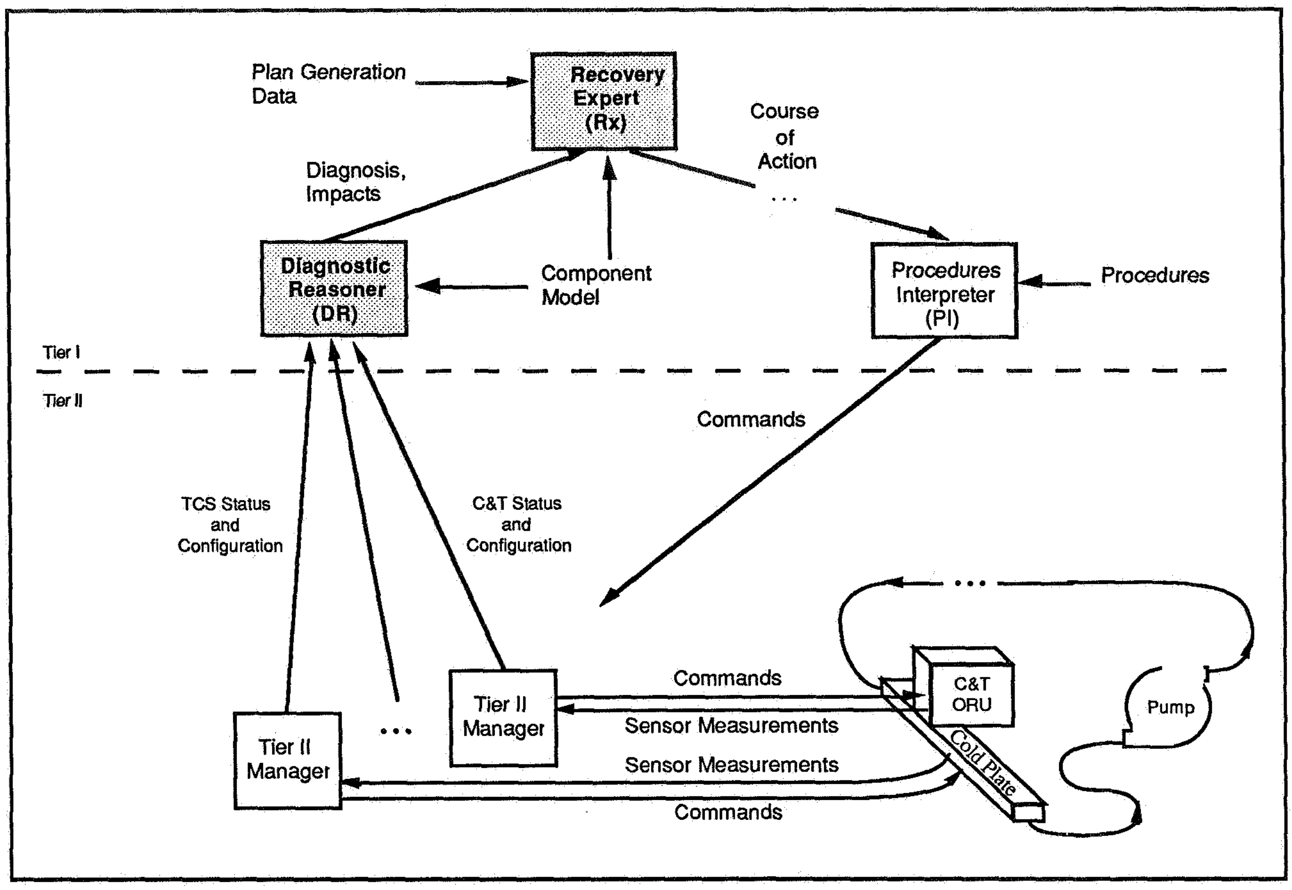


Figure 1. Overview of the DR/Rx Process Flow

uses the Impact Model to assess and project the immediate and future impacts of the failure onto other *Freedom* entities. The suspects and the assessment of their impacts, or Impact Sequences, are passed to the Rx for generation of appropriate Courses of Action for recovery. A high level diagram of the DR processing is shown in Figure 2.

The Tier II managers report significant changes to Tier I. A significant change occurs when the state of some component crosses a qualitative assessment boundary (for example, a component changes from a "nominal" state to a "failed" state, or from a "failed" state to a "nominal" state). In order to determine if there is a significant change, the Tier II manager must assess and evaluate the operational data that it directly monitors. The data that Tier II reports to Tier I represent a qualitative assessment of the system components. The directly-measured quantitative data are also available to Tier I on request.

The DR receives the Tier II reports, and reflects the information in the Component Model, a schematic-like model of the enterprise containing information about both the structure of the enterprise and the expected behavior (and known possible misbehaviors) of the components. If the reports received by the DR indicate a failure, the DR determines possible causes of the failure by searching the Component Model for defined causes of observed misbehaviors. The conclusions drawn by the DR are placed in the Suspect Lists. For each suspect identified, the DR uses the

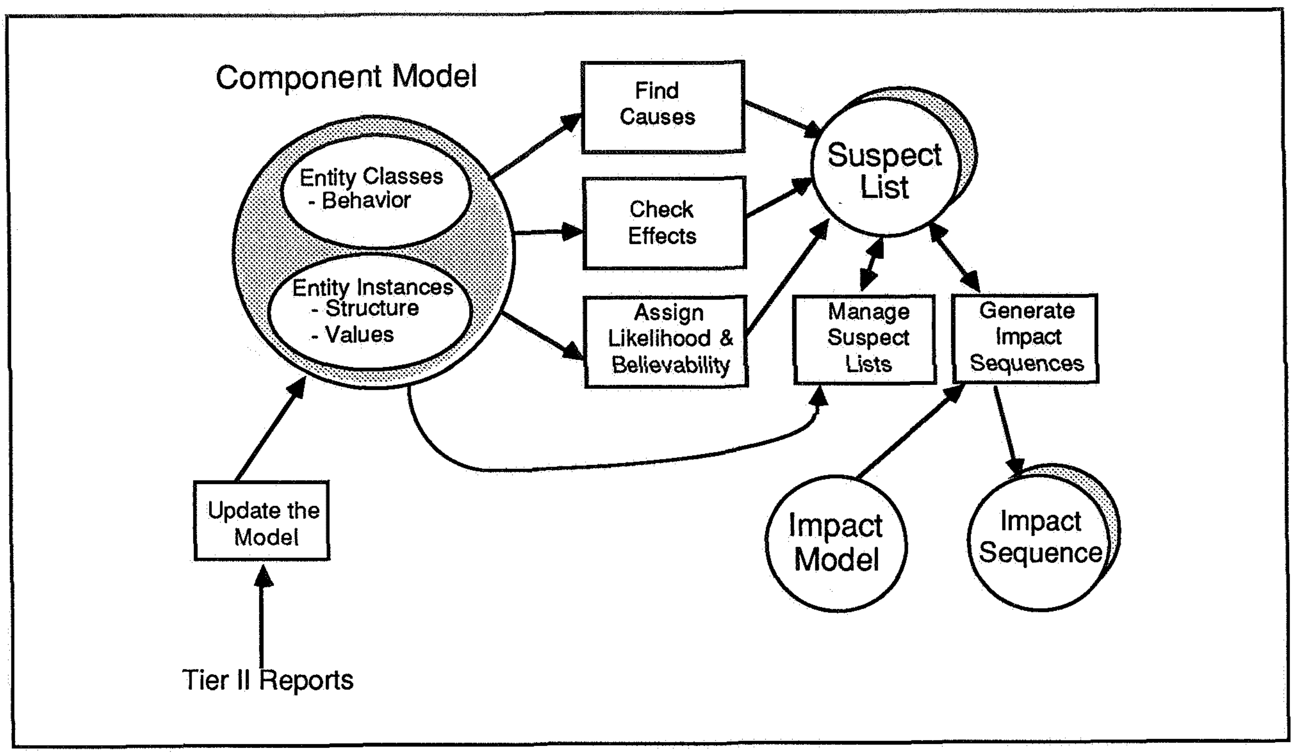


Figure 2. Diagnostic Reasoner Processing

Impact Model to determine the immediate and possible subsequent impacts, and expresses them as a cascade of cause and effect in an Impact Sequence.

Once the DR has diagnosed the problem and determined the immediate and downstream effects, the Rx formulates alternative plans for recovery using the Suspect Lists and Impact Sequences.

Externally Visible Data

The DR either corrects or confirms Tier II diagnoses by updating and using data in the Component Model and the Impact Model. Both these models incorporate an element of time to cope with the dynamic environment. The DR diagnoses, which include both the suspected causes of the failures and the Impact Sequences, are placed in Suspect Lists. We describe the relative time representation and these four data types in this section.

Relative Time Representation

The prototypes are expected to operate in a dynamically-changing environment, where the state of the vehicle changes with time. Timing and temporal relationships among detected events, projected future events, and planned activities are very important to the functioning of the prototypes. The problem is not how to express the specific time at which detected events have occurred (an absolute representation of time), but rather how to express the relative time frame in which future events are expected to occur: this is a matter of timing rather than time.

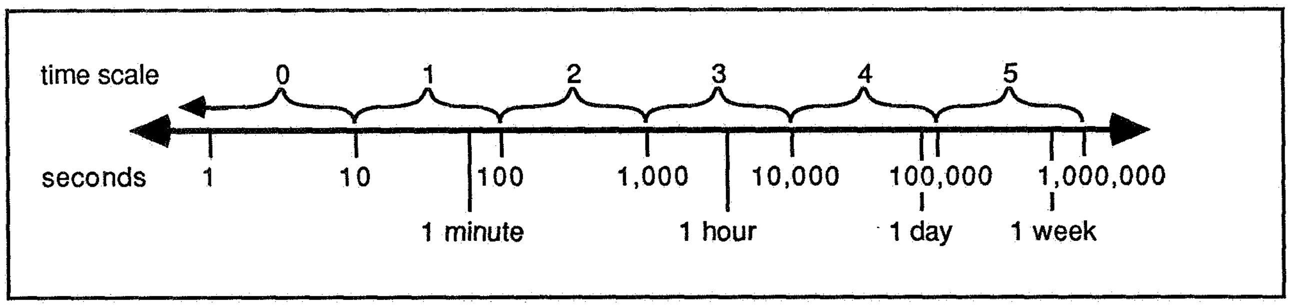


Figure 3. Logarithmic Representation of Relative Timing.

We express relative time on a logarithmic scale of seconds, chunking the time into discrete integral intervals. Figure 3 illustrates this use of time intervals. One example of using this scale is the expression of timing between some event and the expected impact to be felt as a result. If the timing information has a value of 0, the impact is expected to be felt within a maximum of ten seconds. If the timing information has a value of 2, the impact should be felt in 100 to 1000 seconds.

Component Model

The Component Model, a schematic-like representation of the space station and its components, is at the heart of both the DR and the Rx processing. The Component Model incorporates structural, behavioral, and status information. The structural information identifies each individual component's relationships with other components, both physically and functionally. The behavior information identifies, for each type or class of component, the causes and effects of particular conditions with respect to that component type's health and mode of operation. The behavior information describes both internal causal consequences and behaviors across component boundaries. The status information identifies, for each individual component that is operating, the current operational values related to the component's mode of operation, equipment health, and key operational values that are related to behavior.

Figure 4 illustrates three types of inter-component relationships expressed in the Component Model. Locality relationships give information regarding the position in the space station in which components are physically located. Resource Supply and Demand relationships show the functional inter-component dependencies. Note the links between these representations: for example, Node 1 Aft Rack 1 is defined in the Locality relationships and referred to in the Resource Supply and Demand relationships. Resource Supply and Demand relationships show how individual components are functionally aggregated to provide high-level capabilities. Func-tional Connectivity relationships show how the individual components are physically linked together.

Figure 5 illustrates the hierarchical nature of the systems' representation, using the Thermal Control System (TCS) and Communication and Tracking System (C&T) as examples. This representation also encompasses information about characteristics of the components (shown in more detail in Figure 6). Connections between the Levels (0, 1, 2 and 3) show the inheritance pathway in the Component Model. At the lowest level in the figure, the instantiation of specific components is shown, where each component class is likely to have many individual instantiations.

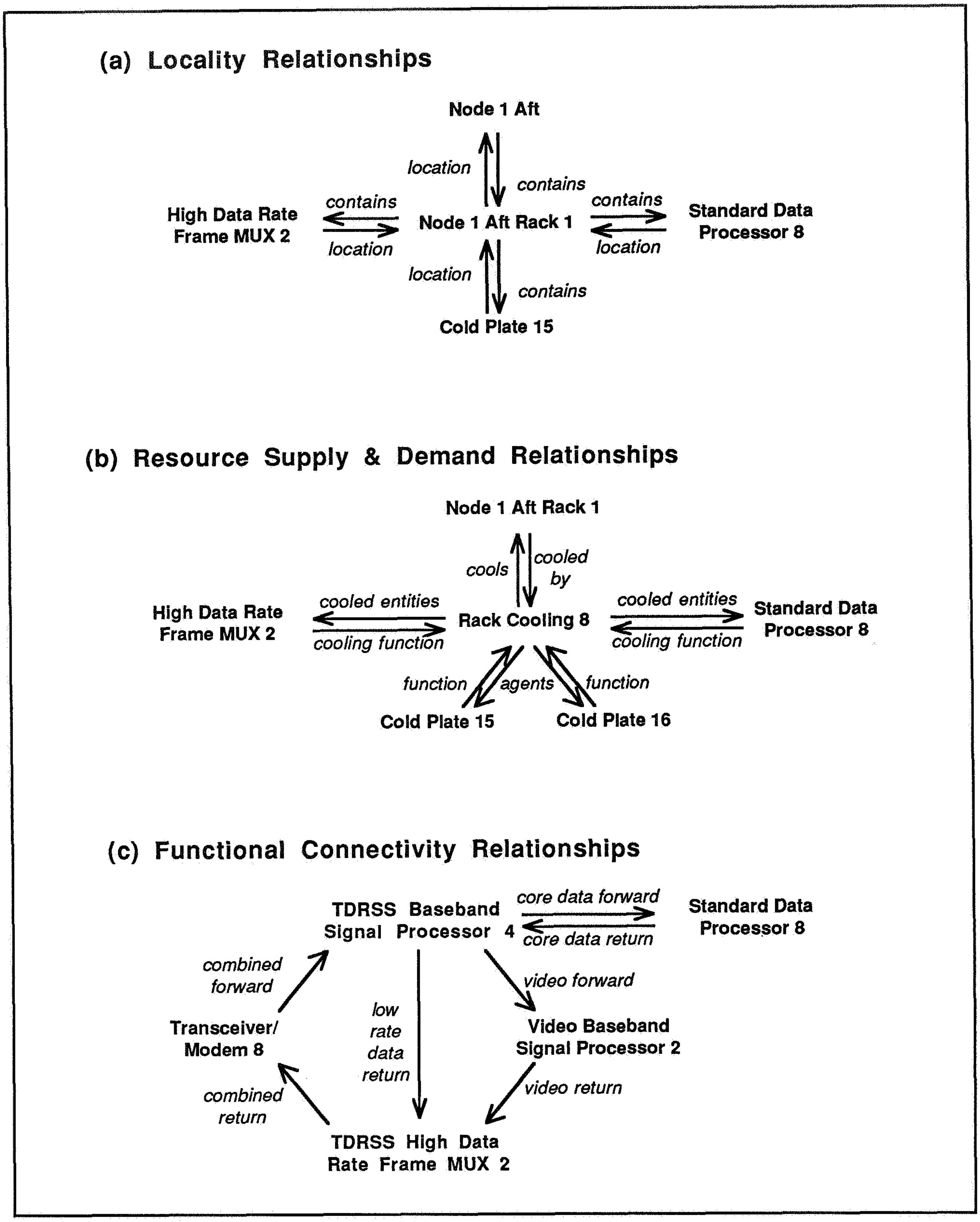


Figure 4. Relationship Types Within the Component Model

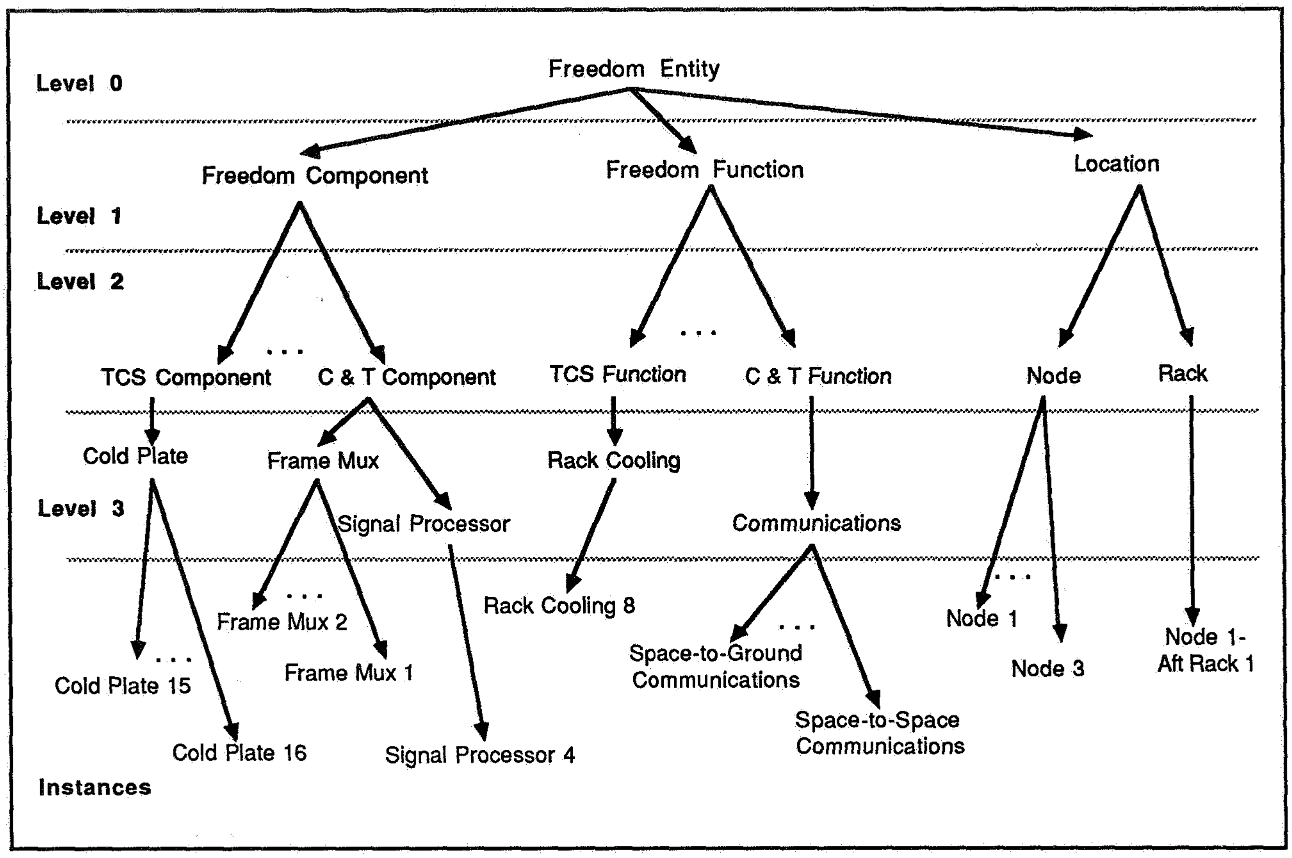


Figure 5. Component Model Hierarchy and Inheritance Structure.

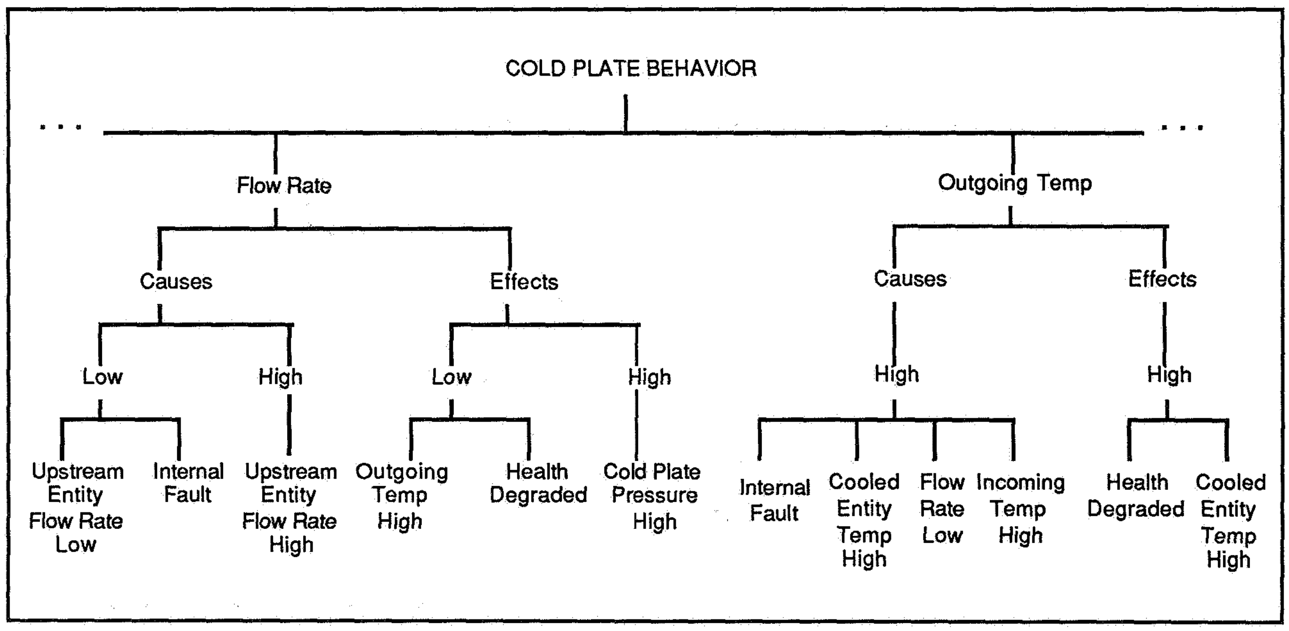


Figure 6. Behavior Measures for a Cold Plate Component Type.

The description of a specific component's behavior and characteristics is based on the generic description of a class of related components; the class description, in turn, is defined as a hierarchy of descriptions. The description of a class of components includes descriptions of types of relationships with other components, component behavior causes and effects, and attribute definitions for status information. The values for the status information includes the specific behavior measures and operating conditions of each individual component.

Figure 6 elaborates on a subset of the behavior information for the specific component class Cold Plate. This information is expressed at Level 3 of the hierarchy as shown in Figure 5. Behavior information for a component type is expressed as a cause-and-effect model for possible observed behaviors. In Figure 6, for example, two possible causes for a cold plate to exhibit a low flow rate are "Upstream Entity Flow Rate Low", or "Internal Failure". Upstream Entity is an abstraction that is instantiated for the specific component instances using information extracted from Functional Connectivity relationships (Figure 4(c)). Figure 7 illustrates the dynamic operational values related to measurements of behavior for a component instance. This type of representation is repeated for each component in the Component Model.

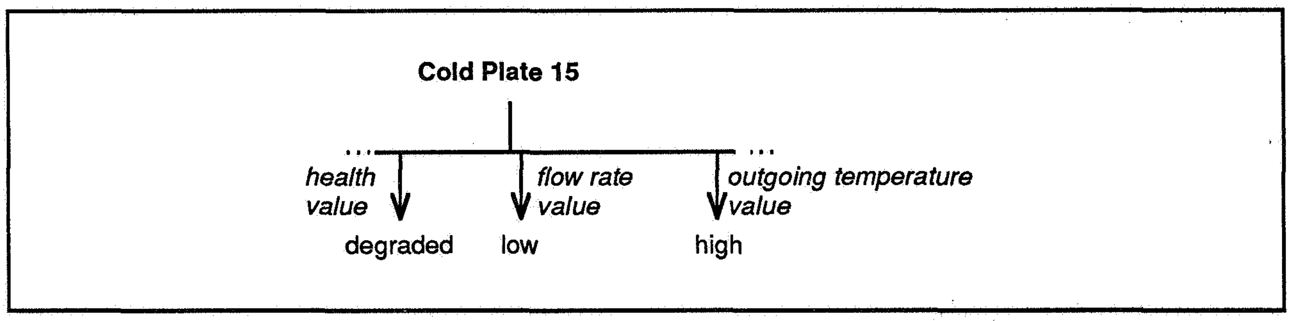


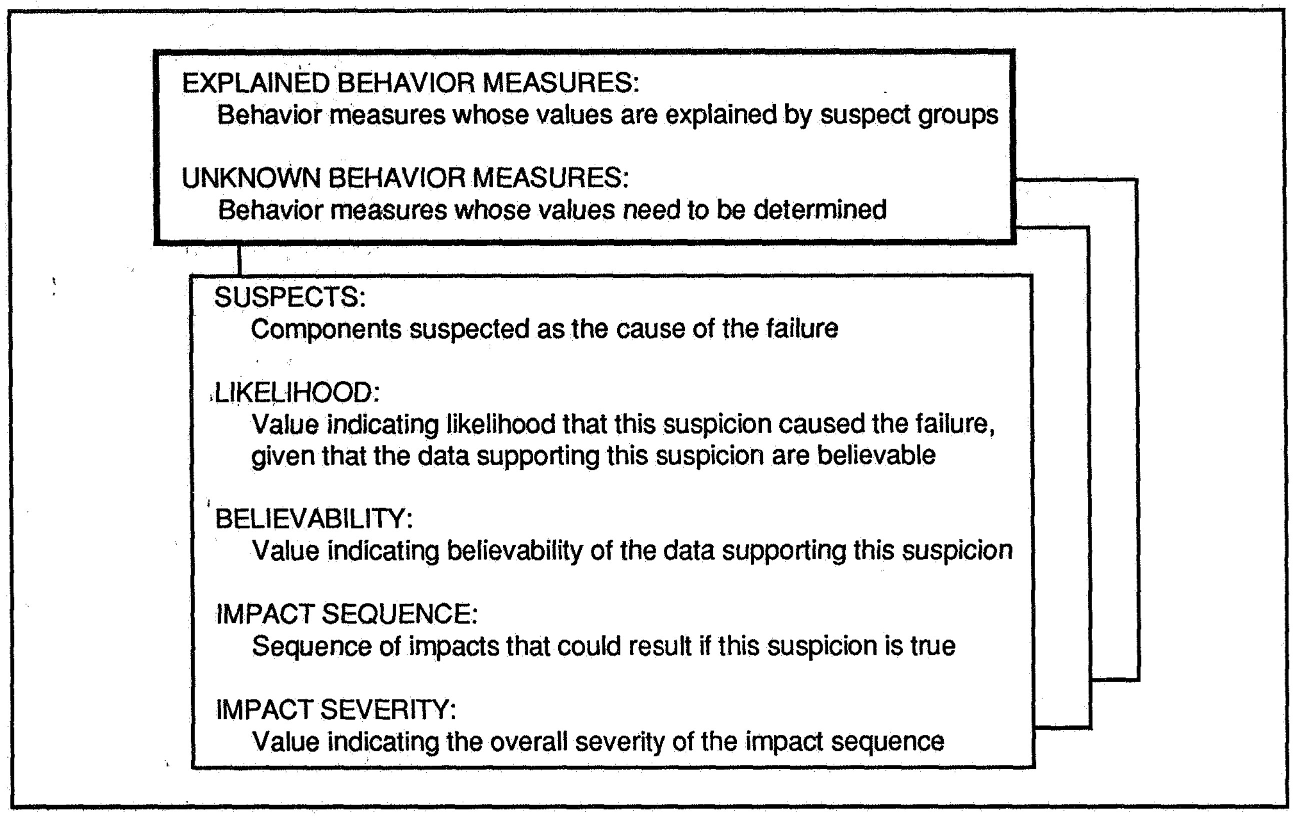
Figure 7. Cold Plate Instantiation.

Suspect List

A Suspect List identifies a set of related observed (off-nominal) behavior attribute values and those components whose off-nominal performance could result in those observations. Figure 8 shows the structure of a Suspect List.

A Suspect List is associated with a specific failure. When diagnosing a failure, the same failure situation might be explained by more than one possible cause; that is, one failure situation can yield more than one possible diagnosis. In this case, the DR generates a single Suspect List containing several Suspect Groups, each group providing a different explanation for the same failure. In some cases, a single component may not be the cause of failure. A set of observable behaviors resulting from multiple component failures may be jointly responsible for the failure. In this event, the DR lists all of the components as a possible cause in a single Suspect Group.

A Suspect List also can identify key unknown behavioral measures: operational behavior measures whose value is not available directly. The assessment of some behavioral measures will require special resources (e.g., asking an astronaut to execute a procedure to collect data) or will induce inter-system interactions (e.g., taking a component offline to perform diagnostics). When the DR encounters one of these unknown measures along one of its diagnostic pathways, it posts the measure in the Suspect List as an unknown whose assessment should help refine the diagnosis.



EXPLAINED BEHAVIOR MEASURES:
Behavior measures whose values are explained by suspect groups

UNKNOWN BEHAVIOR MEASURES:
Behavior measures whose values need to be determined

SUSPECTS:
Components suspected as the cause of the failure

LIKELIHOOD:
Value indicating likelihood that this suspicion caused the failure, given that the data supporting this suspicion are believable

BELIEVABILITY:
Value indicating believability of the data supporting this suspicion

IMPACT SEQUENCE:
Sequence of impacts that could result if this suspicion is true

IMPACT SEVERITY:
Value indicating the overall severity of the impact sequence

Figure 8. Suspect List Structure

For each Suspect Group in the Suspect List, the DR adds likelihood and believability values, as well as impact severity values and an Impact Sequence. Likelihood indicates how likely it is that the Suspect Group caused the failure: for example, a component that fails frequently is more likely to be the cause than one that fails rarely. Believability indicates how reliable the data are that are used to determine the Suspect Group as the cause of failure: for example, some sensors are known to frequently fail or be unreliable. The impact severity value is an indication of the severity of the failure based on the severity of projected consequences. This value is used by the Rx to determine whether a recovery Course of Action is generated for that suspect for safety's sake, regardless of the likelihood that it really is the source of the failure.

Impact Model

The Impact Model is used to predict the cascade of effects that are expected to result from a specific failure type, or Impact Sequence. The Impact Model contains information much like that needed by a person to formulate Failure Mode, Effects, and Criticality Analysis (FMECA) trees. This includes, for each system in the enterprise, and for each component in each system, the relationships between each component's failure modes and the effects of those failures with respect to each system operating mode as well as the operational requirements of those modes. These data provide the basis for determining the effect of the failure on the entire enterprise with respect to the time of the failure's occurrence, the component's as well as the system's operating mode at that time, and the particular set of operational requirements against which the effect of the failure must be assessed. (Ireson, 1988)

The Impact Model focuses on the immediate and near-term future operational causes and effects. The data contained in this model is partitioned into sections, where one section expresses one type of cause-and-effect sequence. Each cause-and-effect sequence contains information such as the failure type, the component or function class affected by or involved in the failure, the failure type expected to appear as a result of the failure, the component(s) or function(s) to which the failure type is expected to propagate, a heuristically assigned severity value for the failure type, and the temporal aspects of the failure. For a specific failure, these sequences are instantiated and linked together, based on the current configuration of the enterprise.

Figure 9 provides an illustration of a portion of the Impact Model: the Impaired Operations of the Active Thermal Control System (ATCS) failure type. One cause-and-effect sequence is associated with this failure type (i.e., a Reduced Resource Supply in the ATCS cooling); however, the model structure can support multiple effects. Again, aspects of the model are represented symbolically (e.g., ATCS Cooling-Location), and must be instantiated for specific failures.

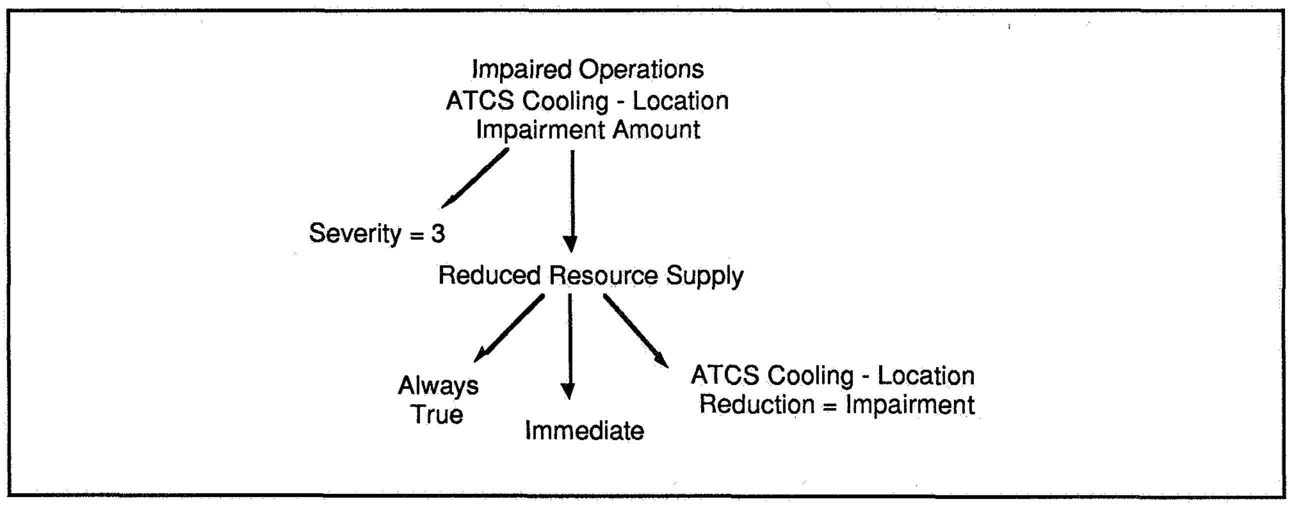


Figure 9. Impact Model Example

Impact Sequence

The Impact Sequence expresses the cascade of immediate and subsequent effects seen as the result of a particular failure. The concept is similar to that of the FMECA, expressed in the following example from the seventeen-century of a component failure that resulted in catastrophic failure: "For want of a nail the shoe was lost, for want of a shoe the horse was lost, for want of a horse the rider was lost ..." and so goes the cascade of subsequent effects of a minor failure in the communication system until, at last, the kingdom was lost. (Ireson, 1988)

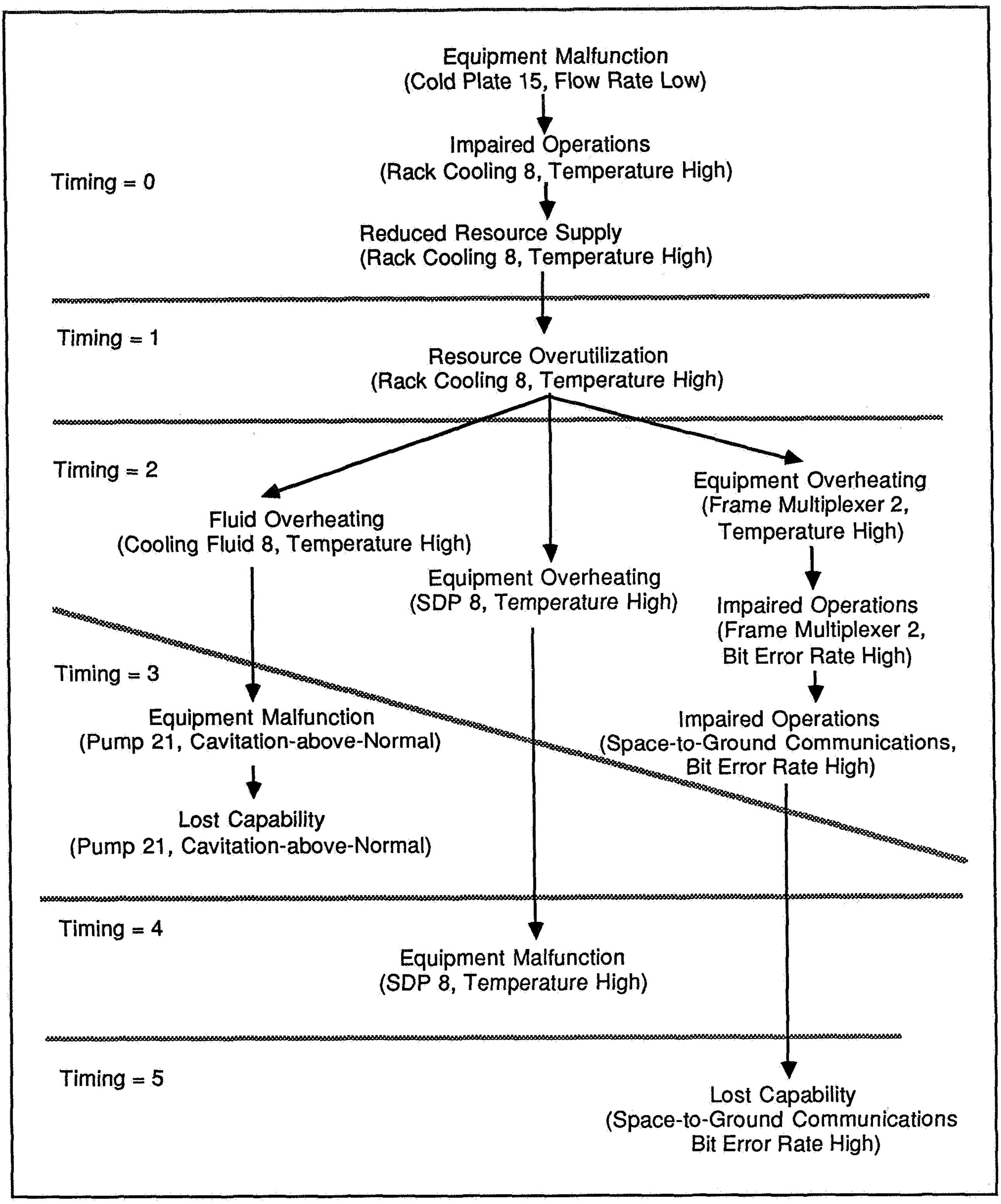


Figure 10. Impact Sequence

The Impact Sequence is a tree-like structure constructed by starting with a root cause (e.g., Equipment Malfunction), examining it in the given situation, and linking together the causes and effects as they apply to situation. As with FMECA work, it is not feasible to identify and control every failure in every system with the potential for catastrophic effects; rather, the objective of the Impact Sequence is to provide a sound basis (to the RX) for guiding the allocation of resources such that the probability catastrophic failure is reduced as much as possible. (Ireson, 1988) Figure 10 illustrates an Impact Sequence.

The DR Processing

The following sections describe the DR processing, as illustrated in Figure 2, using the preceding data descriptions.

Updating the Component Model

The Tier II managers notify the DR of changes in the both the status and configuration of system components through system reports. Only significant qualitative changes (for example, a cold plate's temperature changing from "nominal" to "hot") are reported to the DR; minor quantitative deviations are not. The DR uses the information in the system reports to update the Component Model.

Identifying Suspects

When the system reports indicate a problem, the DR searches for suspected causes using the reported status and the modeled behavioral cause-and-effect. Once a suspect is identified, it is placed as a Suspect Group within a Suspect List. If the DR identifies a multiple source failure, the Suspect Group will contain more than one suspect. After a suspect is identified, the DR verifies that the expected behavioral effects have occurred with respect to the suspect(s) and their related components in the expected time frame. This process is repeated until no new suspects can be identified.

For each Suspect Group, a likelihood value, which incorporates two factors, is assigned. The first factor considers how likely the Suspect Group is to be the cause of the failure, taking into account the component's failure rate. The second considers the likelihood that a multi-component Suspect Group is the cause of the failure. A believability value, also assigned to each Suspect Group, is an assessment of how much belief is given to the information from the systems. Suspect Groups containing component known to have sensors that produce unreliable values listed as the cause will have a low believability. The inclusion of a believability value for a Suspect Group was driven by the fact that false sensor readings have been a problem in previous space programs.

The DR uses modeled component misbehaviors to identify suspects. The drawback of this technique is that diagnoses are limited to predefined known failures in the component model. If a component fails in a mode that is not in its modeled misbehaviors, the DR cannot explain the failure properly. These unanticipated failures will be diagnosed either as multiple failures of related components or as conflicts in expected component behaviors. Conflicts between the observed component behavior and the expected modeled behavior will alert the DR to notify the user that some form of unmodeled misbehavior is taking place.

Other model-base diagnosis techniques that use nominal component behavior are not as limited in their isolation of failures because they look for conflicts between a model of the nominal system and the observed behaviors (Davis, 1988; Holtzblatt, 1989). These techniques are not currently used by the DR because they require detailed knowledge for each component, with a concomitant increase in the amount of processing power required to perform a diagnosis on a complex system. This is especially a problem when diagnosing multiple failures. It is also very difficult to build a detailed model to use these techniques for something as complex as an entire space station. The use of nominal behavior in high-level models to diagnosis failures will be an area of future research for the DR.

Using the Cold Plate example, if the Thermal Control System reports that a specific Cold Plate's outgoing temperature is high, the DR looks for possible causes to explain this behavior in a cold plate. The modeled behavior shows that a Cold Plate's high outgoing temperature could be caused by an internal failure, a low flow rate, the upstream entity having a high outgoing temperature, or a cooled component's temperature being high. As an example, if the DR sees that the observed flow rate for this specific cold plate is low, it searches the behavioral information in the Component Model for an observed behavior that causes a low flow rate. Once a root cause (suspect) is identified for the high outgoing temperature, the DR assigns this suspect to a Suspect Group within a Suspect List. The DR then checks the operational data to see if the other effects from that root cause have occurred in the predicted time frame. If they have not been observed and sufficient time has elapsed, the DR places a lower believability on that Suspect Group. Next, the DR assigns a likelihood to the Suspect Group, based on what operational reliability parameters are available for the suspect. This same process is repeated until all the components whose failure mode might cause Cold Plate 15's outgoing temperature to be high are identified and placed as Suspect Groups in the Suspect List.

Managing the Suspect List

The Tier II managers do not observe all of the effects of a failure at the same time. Consequently, the DR will generate Suspect Lists without complete knowledge of the problem. When the DR receives the first system report, the DR responds given the available information and generates an initial Suspect List. As additional information becomes available, rather than building a completely new Suspect List, the DR first determines if the new information is related to an already-existing Suspect List. The DR relies on the expected behavioral causes and effects of identified suspects that are described in the Component Model to update, merge, or delete existing Suspect Lists generated as the result of the same failure. This yields new Suspect Lists which reflect the latest information available to the DR.

In the Cold Plate example, an initial report from the Thermal Control System may tell the DR that Cold Plate 15 has a low flow rate. The DR places Cold Plate 15 in an initial Suspect List. Later, the Communications and Tracking System might tell the DR that High Data Rate Frame Multiplexer 2 has overheated and failed. The DR notes that the High Data Rate Frame Multiplexer is cooled by Cold Plate 15, based on the Resource Supply and Demand relationships and Locality relationships in the Component Model; however, at present, there is no indication that Cold Plate 15 is overheated or will overheat. Consequently, the DR creates a second Suspect List to explain the High Data Rate Frame Multiplexer failure. Finally, the Thermal Control System reports to the DR that Cold Plate 15's outgoing temperature is high. Based on behavioral information in the Component Model, the DR finds that a low flow rate in a Cold Plate causes a high outgoing temperature in a Cold Plate, and that high outgoing temperature causes the components cooled by that particular Cold Plate to overheat. The DR relates the two failures, and merges the two Suspect Lists into one with the root cause being Cold Plate 15's low flow rate.

Generating the Impact Sequence

The DR generates an Impact Sequence for each Suspect Group in a Suspect List. By instantiating the appropriate cause-and-effect sequences in the Impact Model, the DR dynamically constructs the cascade of effects expected to result with time from a given failure situation. The Impact Sequence is a tree-like structure made up of nodes, where the root node is an instantiation of the cause-and-effect sequence type related to the failure identified in the Suspect Group. Each node in the cascade of nodes to follow is also an instantiation of cause-and-effect sequence types that propagate from the effect identified in the particular node's parent.

The DR also attaches a severity value and an element of time to each node using severity and temporal factors in the Impact Model. The generation of the Impact Sequence is bounded by an envelope of time and severity. Given enough time, the most insignificant failure might become quite severe. Since we assume that no such failure will go unattended for a period of time sufficient to become a severe problem, the DR bounds the Impact Sequence generation by time. Similarly, generation is bounded by severity in that the DR projects the cascade of effects up to some predetermined severity value.

Conclusions

We completed the design of the DR and the Rx, and are currently demonstrating the first, stand-alone release of the DR and the Rx. Based on experience with this prototyping effort, which includes the DR and the Rx as well as earlier efforts from which the DR and the Rx evolved (Kelly, 1988; Kelly, 1989; Marsh 1989), we learned lessons in technology transition, software reuse, and prototype software life cycles.

The primary goal of the work represented by this series of prototypes is to influence Freedom's design and implementation. Its longer-term goal is to find other domains in which this technology and approach can be used. Both are aspects of technology transition (bringing technology innovations into operational use).

Failure management is a process that is needed in many domains. The kinds of problems we are solving, and the approach we are taking to solve them, are very general. We are designing the failure managment prototype to be useable in domains beyond space operations. Examples of other possible applicable domains are NASA's manned Mars mission, chemical processing plants, or nuclear power plants.

With respect to future plans for the DR and the Rx, the two independently functioning applications will be integrated to function cooperatively. We will also integrate the DR/Rx with PI. For the commands that PI issues to be followed, the prototype must be integrated with a simulation environment for Space Station *Freedom* systems. We are investigating the possibility of developing our own simulations of the systems on a machine linked to our development workstations to provide a more portable demonstration capability.

Acknowledgements

The work referenced in this paper was jointly sponsored by the Automation and Robotics Section under the direction of our Technical Monitor, Dennis Lawler, at NASA's Johnson Space Center (contract NAS9-18057), and by the MITRE corporation through the MITRE Sponsored Research Program. David G. Hammen and Christine M. Kelly of MITRE developed the Rx portion of the failure management prototype which is discussed in a companion article (Hammen, 1991).

References

Baker, Carolyn, D. Hammen, C. Kelly, C. Marsh (June, 1990), "A Space Station Failure Management Prototype," presented at 1990 Space Operations, Applications and Research Symposium, Albuquerque, NM.

Davis, R., "Diagnostic reasoning based on structure and behavior," *Artificial Intelligence* 24(3), pages 347-410, North-Holland, 1984.

Davis, R. and W. Hamscher, "Model-based Reasoning: Troubleshooting," *Exploring Artificial Intelligence*, pages 297-346, Morgan Kaufmann Publishers, Inc., 1988.

Holtzblatt, L. J., R. A. Marcotte and R.L. Piazza (October 1989), "Overcoming Limitations of Model-based Diagnostic Reasoning Systems," presented at the AIAA Computers in Aerospace VII Conference.

Ireson, W., and C. Coombs, Jr., *Handbook of Reliability Engineering and Management*, New York, New York: McGraw-Hill, 1988, pp. 13.1-13.36.

Kelly, Christine M. (1988), "Automated Space Station Procedure Execution," presented at AIAA 26th Aerospace Sciences Meeting, Reno NV.

Kelly, Christine M., (1989), *Space Station Freedom Program Advanced Automation, Volume II, Series 2, Integrated Test Beds: Lessons Learned from the Data Management System Test Bed*, MITRE Technical Report Number MTR-89W00271-02, The MITRE Corporation, Houston, TX.

Marsh, Christopher and Christine Kelly (1989), Operations Management Application Prototype, Fourth Artificial Intelligence and Simulation Workshop, Detroit, Michigan, pp 75-77.

NASA (November 1989), *Space Station Projects Description and Requirements Document*, JSC 31000 Revision E, Johnson Space Center Space Station Project Office, Houston TX.