AIAA /NASA

AIAA-89-5070
Maintainability of Unmanned Planetary Spacecraft: A JPL Perspective

P. Kobele,
Jet Propulsion Laboratory,
Pasadena, CA

AIAA/NASA SYMPOSIUM ON THE MAINTAINABILITY OF AEROSPACE SYSTEMS

26-27 July 1989/Anaheim, CA

Maintainability of Unmanned Planetary Spacecraft: A JPL Perspective

A89-48179

Abstract

The requirements for mission success in unattended environments which do not allow direct repair of spacecraft faults have posed significant challenges in the areas of spacecraft design and mission operations. These challenges have resulted in innovative design requirements and implementation approaches intended to maximize the likelihood of being able to reconfigure the spacecraft to accommodate any of a myriad of spacecraft faults. Autonomous fault detection and correction algorithms and the mission operations elements of recent JPL interplanetary projects have been able to utilize these design features in their operational strategies to recover the spacecraft from what might have been mission terminating occurrences and to allow continuation of essentially undegraded missions.

INTRODUCTION

The maintainability of unmanned planetary spacecraft is essential to the achievement of prime mission objectives, as well as for the additional science return which has been possible from both the Viking Orbiter (VO) and Voyager (VGR) spacecraft during the so-called extended missions. Traditionally, the spacecraft development process has attempted to utilize prac-

tices which minimize the likelihood of the occurrence of faults: the design process itself includes exhaustive analyses which lead to design features intended to reduce the occurrence of failures and to enhance robustness and limit the impact of faults, should they occur; part selection and qualification processes are intended to provide parts and materials which are not expected to fail during the mission; thorough quality control during fabrication and assembly is provided; software design, development and testing proceeds under rigidly enforced standards; assembly, unit, and system level testing is as exhaustive as schedule and resource limits allow, as is the training of the flight teams and preparation of the spacecraft flight sequences and various contingency plans. Yet, all of these practices have been unable to assure the absence of in-flight anomalies: on-board fault responses and flight teams have been required to interact with the spacecraft under circumstances which can only be described as "repair and/or maintenance".

REDUNDANCY PHILOSOPHY

A fundamental design driver for recent JPL interplanetary spacecraft has been the requirement that no single fault occurrence would result in mission loss. Exceptions have generally been allowed only for mechanical/structural elements of the spacecraft for which overdesign to standards which are significantly beyond worst-case requirements can be demonstrated, and for other components for which Failure Mode Effects and Criticality Analyses (FMECA) have shown that the risk of failure is extremely remote. This requirement also applies to the proper execution of mission critical sequences: that is, the sequences themselves must be designed to allow the spacecraft to perform the critical mission events in the presence of any realistically credible fault scenario.

The single point failure policy is generally expressed as a design requirement as follows: "No single point failure shall result in the loss of more than one science instrument or the loss of more than one-half of the engineering data." This has resulted in implementations in which

virtually all of the engineering subsystems are fully redundant (utilizing either block redundancy or selected functional redundancy) while the science instruments are not. It is generally accepted that the loss of a single science instrument out of a complement of ten or more (on VGR and Galileo (GLL)), while obviously still of major concern, does not represent a magnitude of loss to the science objectives which is great enough to warrant expenditure of the resources required to implement redundancy.

The most commonly utilized method of protecting against the failure of a component is the inclusion of block redundant elements which can be switched in to replace primary elements which have either failed or degraded beyond some tolerable level. Redundancy is provided to a level of two: resource constraints have precluded additional levels except in extremely isolated instances. JPL's implementations have provided for the individual or cross-strapped, rather than grouped, selection of redundant elements down to below subsystem levels in such a manner that the spacecraft design retains essentially full capability even in the presence of multiple component failures, so long as the failures do not include both elements of a redundant pair. This cross-strapping requires more complex switching and interfacing implementations which have their own unique failure modes, but studies have shown that the overall reliability of the spacecraft is nevertheless increased.

The spacecraft can also be protected against the failure of selected elements by functional, rather than block, redundancy. What this means is that functionally similar, but not identical, components can be switched to take over the functions of a failed element. Some examples of this type of implementation are the gyros on VGR, and the thrusters on GLL. The gyro package on VGR contains three two-axis gyros oriented such that their spin axes are orthogonal: thus, one gyro can provide references for the pitch and yaw axes, another for pitch and roll, and the third for roll and yaw. In this manner, the failure of any one of the three gyros can be accommodated without compromise to spacecraft capability by appropriate reconfiguration of the attitude control software. On GLL, the thrust vectors of individual control thrusters have been optimized for spin control, lateral thrust, angular momentum vector re-orientation, and

axial Δ−V maneuvers. As a consequence, some of the thrusters did not need to be replicated, as their functions could be assumed by others with some tolerable loss of efficiency. It is also of interest to note that, until very recently, the axial 10N thrusters were to have been available as a backup to the 400N main engine for the Jupiter Orbit Insertion (JOI) maneuver.

There also exist certain spacecraft elements which are not amenable to on-line protection through either block or functional redundancy, yet which still have credible failure modes which make waiving the single point failure requirement inappropriate. An example of an element falling into this category is the pressure regulator of the propulsion subsystem for the VO spacecraft. This propulsion subsystem is a pressure-fed bi-propellant system, which relies on helium as a pressurant to maintain both the fuel and oxidizer within an acceptable operating pressure range. Due to the relatively large volumes of the fuel and oxidizer tanks which the pressurant must eventually nearly fill, the helium is initially stored at a much higher pressure in a separate tank and fed into the other tanks, as their ullage increases due to the consumption of propellant, through a pressure regulator which is intended to limit the pressures in these tanks to within the allowable operating range. Rather than utilizing redundancy, protection against a pressure regulator failure is provided through a ladder network of pyrotechnically activated valves which alternately enable and disable pressurant flow to the propellant tanks. The actual use of this type of protection requires special interactions between the spacecraft and the ground operations elements. A specific example of a pressure regulation fault is discussed in a subsequent section of this paper.

REQUIREMENTS FOR AUTONOMOUS SPACECRAFT OPERATIONS

It has been determined that a high degree of spacecraft autonomy is required for the conduct of deep-space missions. Round-trip light times are such that, even if the spacecraft were to be continuously observed by the operations staff, several hours could elapse before any com-

mands transmitted by the ground would arrive at the spacecraft. When one considers the realities associated with Deep Space Network (DSN) availability for tracking a particular spacecraft, the time required for telemetry analysis to detect fault conditions and to determine proper fault responses by the ground, the time required to generate the proper commands, the often lengthy process of command validation, verification, and obtaining transmission approval, one finds that the spacecraft is entirely on its own for many hours, often days, following the occurrence of an on-board fault.

While it is true that the spacecraft can be configured to a safe, quiescent state following a fault condition, certain processes must continue without compromise if the mission is to be preserved. For these deep-space projects, certain activities are essentially one-time, "make-or-break" mission events. For example, for a planetary fly-by mission, the sequences which have been prepared for the science acquisition during the close approach phase must continue to be executed properly, otherwise, the major mission objectives would not be attained. For orbiter missions, the orbit insertion maneuver must be executed under all possible circumstances, otherwise the mission is completely lost. Even during those mission phases which do not contain the aforementioned critical events, continuation of certain processes is essential: spacecraft attitude maintenance requirements cannot be relaxed significantly beyond nominal values if there is to be a usable downlink, component thermal control limits must be actively maintained, and consumables must continue to be protected, among others. For these reasons, the autonomous fault detection and correction capability of the spacecraft has evolved into an integral element of the spacecraft and Mission Operations System (MOS) designs.

For some functions, the proper operation of an element is required at all times with virtually no interruption in order for the spacecraft to perform even the most rudimentary of activities. This type of failure condition can be protected against in one of two ways. First, the selection of the redundant element can be performed via an autonomous hardwired logic circuit which is immediately activated by the failure of the primary element. As an example, the failure of the primary 2.4 kHz inverter on GLL would cause an interruption of power to the cen-

tral computer, which then obviously could not perform the processing required to detect the fault and could not initiate the switching to select the redundant unit. For this reason, such an event will cause the selection of the redundant inverter via hardware logic circuits designed specifically for that purpose.

A second method of accommodating this situation is by having both of the block redundant elements being operational at all times, with sophisticated fault detection mechanisms ensuring that a failed element will not compromise the operation of the remaining one. The central spacecraft computer is handled in this manner. This creates the obvious quandary of attempting to have two separate elements operating independently enough to assure that the failure of one does not bring down the other, yet with a sufficient degree of control over the spacecraft to prevent an active failure of one from damaging the spacecraft. If the two computers disagree about the results of any health check on the other, the vote on which one is in error is very likely to be a tie.

The method chosen to minimize the mission risk on VO and VGR involved the use of two modes of operation of the central computers, the Computer Command Subsystem (CCS). The results of the command generation activity of each CCS processor was transferred to an output unit (OU), which actually transferred the command data to the recipient subsystems. The two OUs were cross-strapped to the two CCS processors such that they could receive command data from both. In the "individual" and "parallel" modes, each OU transferred the command data to the destinations from its processor without checking the data from the other processor. In the "tandem" mode, each OU compared the data received from each CCS processor, and only if the data was bit for bit identical and received within a very tight time window from both processors did it transfer the indicated commands to the recipient subsystems. For most of the mission, each CCS half operated in the "individual" mode. Critical sequences such as the VO Mars Orbit Insertion (MOI) maneuver were designed such that both processors had to agree to initiate the engine burn prior to some nominal time, after which either processor could initiate the burn individually in accordance with its internal timing. For the burn termination, both processors

had to agree to terminate the burn prior to some pre-determined minimum burn duration necessary to achieve orbit. After the minimum burn duration had elapsed, either processor could terminate the engine firing individually. In this manner, the critical MOI sequence was protected against both a runaway processor issuing commands spuriously, as well as a processor which "died" quiescently. The architecture of the GLL central computer, the Command and Data Subsystem (CDS), does not allow this particular type of implementation for the JOI maneuver; however, the fact that a third computer, in the Attitude and Articulation Control Computer Subsystem (AACS), is in the engine firing loop would allow some form of majority vote implementation, if so indicated.

AUTONOMOUS FAULT DETECTION AND CORRECTION

In addition to the hardware controlled switching of redundant elements, the capabilities of the spacecraft computers have been utilized for fault detection and correction algorithms. These have evolved in complexity with each new mission. They originally were intended to provide the minimum required actions necessary to protect against spacecraft and/or mission loss and to ensure commandability; they now also provide protection against the loss of individual instruments and take actions designed solely to reduce the impact of faults on the ground recovery processes. There also has been a marked change in philosophy from VGR to GLL relative to the independence of faults and their impact on the rest of the spacecraft. On VGR, it was an explicitly stated design goal that so long as faults and their autonomous recovery processes could be isolated within a particular area, other spacecraft operations were to continue without impact. Although a large number of spacecraft faults on GLL are autonomously handled with little or no impact to continued spacecraft operations, many of the fault protection responses which resulted in continued operations on VGR will cause the cessation of sequenced activity and result in placing the spacecraft in a safe and relatively quiescent state. This philosophy change was driven by several factors. Primarily, the design of the overall GLL spacecraft

system has resulted in a much greater interdependence of subsystem states which would be extremely computer resource intensive to autonomously separate with on-board software. The simplest approach to many of the faults was a generalized safing strategy. Second, since GLL is an orbital mission with numerous encounters with the Galileian satellites, the thinking is that the loss of data from one particular satellite encounter due to a fault represents only a small fraction of the science return from the overall mission. Since the risk of continuing with the execution of the on-board sequences cannot be a priori demonstrated to be zero, this data loss is considered an acceptable trade for preserving the integrity of subsequent orbital encounters.

The design of the autonomous fault protection system has been implemented such that, wherever reasonably feasible, the detection of a fault and the required responses thereto take place at the lowest possible level, in an attempt to limit the propagation of effects to the system as a whole. On VGR and GLL, for example, the AACS is designed to handle, on its own, virtually all faults of its subordinate elements so long as they do not impair the operation of the processor, with the central spacecraft computer (CDS) being required only for the actual switching of power to the AACS elements. Required reconfiguration of internal AACS algorithmic processes is done entirely within AACS. Only in those instances where the AACS is unable to handle the fault, or if the fault condition poses system-level risk, does the CDS respond to internal AACS faults. On GLL, some of the science instruments will, to a point, perform internal reconfiguration independently, requesting CDS safing actions only if the conditions are of such a magnitude that they cannot be handled within the science subsystems. Power subsystem faults are handled entirely within that subsystem, the only fault responses required of the CDS are those system level reconfigurations of spacecraft state required to accommodate the new state which the power subsystem recovery process has created.

With the proliferation of computers in spacecraft and the utilization of a distributed data system on GLL, the system fault protection related software within the CDS subsystem has also become distributed. Rather than being implemented within the central spacecraft computer, many of the fault detection and correction functions have been allocated to other pro-

cessors. This has resulted in much of the system oriented fault protection software being executed by processors within CDS which are subordinate to the "main" CDS computer. While this was not done by choice, but was forced by the architecture selected for the CDS, it does not appear to have resulted in any reduction in fault protection capability.

SELECTED IN-FLIGHT EXPERIENCES

Certain examples of fault occurrences experienced during the VO and VGR missions are of interest in order to note the methods by which the responses of the autonomous fault protection functions and/or the flight team utilized the spacecraft design features to prevent mission loss or degradation. In some instances, flight team ingenuity resulted in the preservation of the spacecraft without utilizing any of the features designed specifically for spacecraft fault recovery.

VO-1 Pressure Regulator Failure

Following the second pressurization of the VO-1 propulsion subsystem prior to the MOI maneuver, the pressure regulator leaked helium pressurant into the propellant lines and tanks at a rate sufficient to cause a tank pressure rise of roughly .7 psi/hr. The CCS had a "Pressurant Regulator Failure Routine", designed specifically for a failure of this type, which would have fired a normally open pyro valve to isolate the pressurant from the propellant prior to the attainment of undesirable overpressure conditions which would compromise the mission due to increased risk of loss of engine cooling, and loss of pressurant and potentially some propellants due to automatic actuation of the pressure relief devices installed to protect the tanks from burst pressure levels. The ladder-like arrangement of normally open and normally closed pyro valves would then

allow repressurization of the system when necessary for proper execution of the MOI burn. Given that the leak rate was so low, it was decided to not allow firing of the pyro valve, but to significantly increase the maneuver duration of the previously scheduled near-Mars trajectory correction maneuver (AMC-1), in order to attempt to obviate the problem by increasing the tank ullage, and also to try to unseat any contaminant in the regulator seal which could have been responsible for the leak. Later, following further leak rate assessment, an unscheduled trajectory correction maneuver (AMC-2) was also performed prior to the MOI.

This strategy proved successful in preventing any spacecraft damage or state change which would have placed MOI, and hence the success of the mission, in series with proper operation of the single remaining pyro valve which would have been used to repressurize the propellant tanks. The final mission result of this decision was that after the attainment of Mars orbit, the period deviated from what was originally desired and required unplanned corrections to rectify. The compromise to the overall science return of the mission was apparently negligible.

Even though the design specifically provided protection against a fault of this type, the characteristics of this particular fault allowed for implementation of an operational workaround which was much preferable to the initially designed "solution" to the fault. Of necessity, the autonomous fault response was a generalized response, designed to handle the worst-case situation in a "brute-force" manner, with the immediate safety of the spacecraft as its only objective. The response by the entire VO flight team did not compromise the immediate safety of the spacecraft, yet avoided potential compromise to later mission events. This particular example is useful in illustrating a very important point. No matter how well a system is designed and how well protected it is against faults, there is no substitute for the talent and ingenuity of a well-trained flight team when it comes to responding to fault conditions by taking into account the specific details of each situation.

VGR-2 Launch Anomaly

Although there were several anomalies which occurred early in the VGR mission, the one most traumatic to the flight team occurred immediately after the separation from the propulsion module (PM) used to inject the spacecraft into the desired trajectory. At PM jettison, an anomalous yaw angular rate of approximately -1 degree per second was experienced by the spacecraft, possibly due to the failure of the science boom to fully latch at deployment. In order to protect against the possibility of pyro shock induced state changes which might have compromised the ability of AACS to fire thrusters, the launch sequence in the CCS commanded the AACS to reset the propulsion isolation valves to the state required for proper thruster operation following the separation event. To protect against CCS failures, each half of CCS was sequenced to issue the reset command individually. As designed, the AACS responded to both of these commands, the response activity taking roughly eight seconds, during which time all attitude control thruster firing was inhibited. After this time, the AACS immediately began to fire the thrusters required to reduce the angular rate, but before the rate could be appreciably reduced, the angular error exceeded the allowable error as determined by the thruster fault algorithm which had been enabled in the interim. The algorithm, operating under the assumption that the attitude error had occurred due to thruster failure, requested the redundant thruster branch to be selected, which resulted in the cessation of the rate reduction until the redundant thruster branch could be brought on line. Following the switchover, the AACS again began to fire the thrusters as required, but by then, the secondary angular limit had been exceeded, resulting in the assumption that the problem was in the interface circuits between the processor and the thruster drive electronics. The hardware switch was requested by AACS and performed by CCS. It must be noted that every hardware switch was accompanied by an interruption in the thruster firing until the switch could be completed, and also, that the angular rate still persisted, causing ever increasing yaw position error. By this

time, no amount of hardware switching could have reduced the accumulated angular error within the times allowed by the algorithm, and, after several more hardware reconfigurations and a relay bounce which further delayed the eventual recovery, the AACS processor was switched by CCS. The processor switch obviously resulted in the re-initialization of the gyro position errors, allowing the nominal thruster operations to reduce the rate to zero within a very short time. Due to the gyro position re-initialization, the commanded turns in the sequence resulted in the attainment of an incorrect spacecraft attitude by the time the sun acquisition command was issued. The spacecraft executed the normal fault response which eventually resulted in the establishment of nominal spacecraft attitude and acquisition of celestial references.

This occurrence was the result of the spacecraft and the fault protection performing exactly as they were designed, but with the launch sequence and the fault protection designs not having been properly integrated to accommodate such a worst-case type of situation.

VGR-2 Receiver Failure

On VGR-2, there were two receiver failures on the spacecraft. This particular situation illustrates several important points very well: the required interactions of the flight team with an autonomous spacecraft, the adaptability required to operate with a severely degraded element, performing preventive maintenance on flight hardware by the flight team, and planning for total loss of an uplink capability.

During a period of major activity with VGR-1, the flight team did not transmit any commands to VGR-2 for a time duration greater than what had been specified as the limit for the Command Loss (CMDLOS) fault protection algorithm activation.

Consistent with the algorithm design, the spacecraft began to reconfigure its uplink elements under the assumption that the non-receipt of ground commands within the specified time interval was indicative of a fault in the uplink hardware of the spacecraft. Following the selection of the block redundant receiver, it was found that it was unable to lock onto the uplink signal transmitted by the DSN, apparently due to a failure in the receiver tracking loop. (The receiver tracking loop has a nominal bandwidth of ±250 kHz; post-fault analysis has determined that the current tracking loop bandwidth is approximately ±100 Hz. For a point of reference, the variations of the uplink frequency just as a function of the Doppler effects due to the rotation of the Earth are on the order of 3 kHz over a DSN pass.) At this point , it was decided that this receiver was unusable, and to wait until CMDLOS again selected the original, unfailed receiver. When this happened, however, this receiver failed catastrophically due to circumstances which are still not conclusively identified. Due to the interactions of the commanding attempts and the algorithm activation, CMDLOS remained inactive for one week, at which time it again started its hardware switching process. During that week, the Anomaly Team diagnosed the tracking loop fault and developed command procedures by which individual commands could be repetitively transmitted at ramped uplink frequencies with a very high likelihood of at least one being accepted by the spacecraft receiver.

Since that time, the flight team has had to contend with conducting the entire mission with the severely degraded receiver which was originally considered to be totally unusable. Complicating matters is the fact that the receiver best lock frequency (BLF) around which the 200 Hz tracking bandwidth is centered, drifts as a function of receiver temperature at a rate of roughly 400 Hz per degree Celsius; unfortunately, temperature variations of several degrees are not uncommon as spacecraft power state changes occur due to nominally sequenced load switching. This has required the implementation of frequent BLF calibrations, the development of adaptive tracking procedures, and the implementation of true preventive maintenance on the receiver to protect

against total failure. It is surmised that the failure mode is a resistive short in the tracking loop capacitor due to particle migration through the dielectric material. It has been demonstrated that particles which have not yet completed their migration can be destroyed when the capacitor charge is maximized by periodically pulling the uplink signal to the extremes of the loop bandwidth.

Another action performed by the flight team following the receiver failure was the preparation of so-called "Backup Mission Loads" in anticipation of the potential total loss of the second receiver. A major effort was undertaken to develop a sequence stored within the CCS to be executed by the spacecraft upon final loss of uplink. This sequence was a highly optimized, essentially hand-coded, encounter sequence tolerant of trajectory inaccuracies and many spacecraft faults, intended to maximize the science return in the event the nominal stored sequences could not be uplinked. It was required to be updated following each major trajectory correction maneuver, and was eventually redesigned and uplinked numerous times for each of the planetary encounters.

VGR Flight Data Subsystem (FDS) Memory Failures

There have been several instances of memory failure on the VGR FDSs. The FDS provides the control for the science instruments and provides all of the data collection, processing, and formatting functions necessary to return science and engineering data to the ground. As the Earth-spacecraft distances have increased, the development of new flight software which contains new data modes and rates has been necessary to accommodate a reduced telecommunications bandwidth. Obviously, any memory failures, whether to a single bit or to larger areas, can result in major compromise to the data return functions. Although the FDS includes redundant processors and memories, extended mission operations have required the utilization of the FDS elements in a non-redundant manner. One of the major challenges in the FDS software development area has

been the requirement to be able to adapt to failed memory locations without relying on the redundancy available to the other computers on the spacecraft. To date, the continued FDS software development and maintenance activities have been one of the major contributors to the success of the VGR extended mission to Uranus and Neptune.

OBSERVATIONS ON LESSONS LEARNED

The experience gained during the conduct of previous flight projects and the development of the GLL spacecraft and mission operations system have provided many insights which should prove useful in the designs of potential future missions in the areas relating to autonomous operations and assurance for mission success. Although these insights cannot always be translated into specific fault detection and correction system design requirements without placing them into the context of the specific mission and spacecraft requirements, they are valuable as guidelines in the formulation of requirements.

Continuity of Development, Test, and Operations: Perhaps the most important lesson is that the design features related to fault tolerance and autonomous fault detection and correction capability must be integrated with the spacecraft design and operations elements throughout the development process. On VGR, although the development of the fault protection algorithms were incorporated into the spacecraft design activities, exhaustively tested during the subsystem and system test process, and well-documented in memoranda and official project documentation, many of the elements of the flight team were not adequately aware of the capabilities, functions, and operation of these algorithms. For this reason, the spacecraft anomalies encountered during the early phases of the VGR mission caused much greater perturbation to the flight team than was necessary. What had occurred on VGR was that as the spacecraft development and system test phases were completed, the responsibility for spacecraft operations was handed over to flight team personnel, a large number of whom had not been involved in the de-

velopment and test activities, and thus were not extremely familiar with the spacecraft. This problem was rectified on VGR by transferring key individuals from the development and test phases to the flight team to handle the anomaly situations. These design and development personnel remained in the operations environment until the rest of the flight team became sufficiently familiar with the spacecraft. The problem on GLL was recognized from the outset, and the mission operations planning has included many of the development and test personnel as possible as part of the post-launch flight team. For this reason, the flight team as presently constituted has the familiarity with the spacecraft and its capabilities that is necessary to perform any of the maintenance and recovery activities that may be required due to in-flight anomalies.

Well-defined Project Level Requirements: The development of the autonomous fault detection and correction algorithms on the spacecraft must proceed according to well-defined and consistent requirements and criteria. The design, implementation, and test of these algorithms is a very resource intensive process, and the computer resources consumed by the algorithms on the spacecraft constitute a major fraction of the total resources available. The design and implementation approach for a fault detection and correction scheme requiring only immediately necessary safing actions and waiting for ground intervention for the recovery process is significantly different from that required of a scheme designed to perform all the steps of a recovery which allows essentially unimpaired continued operation of the spacecraft without ground involvement. If it is necessary to change the overall approach from one to the other in the middle of the design process, much of the work will need to be restarted, with the attendant development resource loss: it is extremely difficult to modify a partially completed design. In addition, any initial computer resource allocations made for one particular design approach become essentially meaningless for a different approach.

Function Oriented System Level FMECA: It is important to the design process that, at the system level, one does not fall into the trap of over-reliance on the traditional subsystem

type of FMECAs. If one takes the approach that all of the potential failure modes identified in all of the subsystem FMECAs must be protected against, the overall problem quickly becomes nearly intractable, as each individual component has many failure modes at each particular output which can occur in many different combinations. The standard FMECAs are invaluable for their particular purpose: critical assessment of subsystem susceptibility to piece part or component failures. They can be used to assess levels of fault tolerance within subsystems and to validate assembly and subsystem designs. In order to perform the design (and eventually the validation) of a fault detection and correction implementation at the system level, a functional FMECA must be performed. While a standard FMECA starts at the piece part level and defines impacts to higher level assemblies and functions, a functional FMECA needs to start at the highest possible functional level and work to lower levels. The reason for this is that, during flight, the visibility into the performance of lower level assemblies is of necessity less than in the development and test laboratories on the ground. Very often the first indication of a fault in a component or assembly is the anomalous performance by some higher level spacecraft function. For each of the major functions that the spacecraft must perform, one can determine which subfunctions, subsystems and/or components are required to be involved and assess the effect of faults affecting these lower level elements on the integrity of the higher level function. If one performs this "reverse" FMECA down to the lowest level element which is reconfigurable or replaceable, a finite set of potential fault responses can be identified for all of the major functions.

Early Design Involvement: Even though the most visible elements of the autonomous fault detection and correction design are the software algorithms in the spacecraft computers, a good overall design encompasses hardware, software, and procedural aspects. It is essential that fault protection be considered early enough in the development process that it can still influence spacecraft system and subsystem hardware designs and not so late in the process that the spacecraft hardware is given as a set of boundary conditions within which the fault protection elements must be designed. A relatively modest hardware design requirement, if articulated early enough that it can be implemented without major resource cost, can lead to significant

design, development, and implementation resource savings in the design of the software. For example, the definition of subsystem power-on-reset (POR) state and relative trip voltage requirements as well as decisions relating to the placement of various engineering subsystems on the switched or unswitched power buses made the design and implementation of the power fault recovery algorithms on VGR and GLL much simpler than they otherwise would have been.

Critical Sequence and Fault Protection Design Integration: The development of flight sequences and sequence components must be undertaken with the full participation of the personnel designing the fault protection system. Especially in the design of the critical sequences, every sequenced command must be defined and located in the sequence with complete understanding of how it might interact with any of the fault algorithms if the sequence is to execute properly even in the presence of faults. In addition, the fault protection responses must also be designed to be able to accommodate the the required state transitions defined in the sequence. This type of sequence is an example of how the sequence design can drive the fault protection design and vice-versa. For example, the GLL JOI sequence sets specific indicator flags accessible to the fault protection algorithms to indicate the state to which these algorithms must configure the spacecraft as a function of how far the sequence has progressed without the occurrence of faults. In the sequence, the timing between certain commands is selected to allow sufficient time for state transitions induced by various fault responses to complete. In this manner, given that the sequences are thoroughly tested with selected fault conditions being pseudo-randomly injected into the tests, one can have a reasonably high level of confidence in the integrity of the sequence even in the presence of various faults.

Flexibility of Responses: One lesson which has been delivered many times over in many applications but has still not been accommodated adequately is that fault protection responses cannot be designed as invariant. By its very nature, fault protection must be considered as a dynamic entity, requiring flexibility to accommodate changing conditions. The design requirements for the GLL fault protection algorithms specified the utilization of parameterized vari-

able and command data specifically for the purpose of accommodating the differing characteristics of the various mission phases. However, attempts to minimize the consumption of CDS computer resources caused many other portions of the fault protection algorithms to be implemented in such a manner that modifications to the initially specified responses are extremely difficult. The structure of the responses may need to be modified after the initial design for many reasons: mission redesign, differences in the characteristics of actually delivered hardware from that which was expected, late analysis results, subsystem and system level idiosyncrasies not detected until late in the test programs, and new information received from external sources, among others.

Fault Conditions versus Unexpected Performance: Another valuable lesson is the recognition that there is a large conceptual difference between performance which is unacceptable and that which is merely outside the initially predicted range. The design of complex systems such as spacecraft requires considerable analyses and simulations related to performance of constituent elements as well as of the system as a whole. Very often these simulations and analyses involve considerable time and effort and the talents of a large number of people in the development of the tools, their application, and the analysis of the results. For this reason, there is a tendency to accept the results of such analyses as definitive criteria for proper operation during the execution of spacecraft activities. This acceptance often leads to the establishment of these analysis results as the thresholds for proper performance in the fault protection algorithms. It is important to recognize that these analyses make simplifying assumptions, they often use idealizations for hardware performance, and assume environmental conditions which are themselves based on analyses and assumptions derived from ground test results extrapolated to the actual operating environment. Of course, these analyses are the best that can be made given the constraints and available resources, and are extremely valuable to the design process. However, the function of fault protection is just that: to protect against faults. The thresholds of the algorithms must initially be based on standards of unacceptable performance due to fault conditions, and not necessarily just on conditions which are different from those predicted by analysis. Only after observation of spacecraft operations and characterization of performance

in the actual flight environment should the fault protection thresholds be adjusted to tighter standards, as necessary. Based on the early VGR mission experiences, it has been an explicit GLL fault protection design requirement to select thresholds based on standards of unacceptable performance rather than just on deviations from expected performance.

Fault Testing and Contingency Planning: Familiarization of the flight team personnel with potential anomalies, spacecraft behavior under anomalous conditions, actual telemetry signatures of that behavior, resources available for the resolution of the anomaly, and potential ground responses can be extremely valuable in maximizing timeliness of ground responses to actual in-flight anomalies, if and when they occur. This familiarization should take place by injecting faults into the test process as well as by requiring the preparation of contingency plans. Although it has been often demonstrated on VGR that even the best of contingency plans do not apply exactly to the actual anomaly conditions which occur on the spacecraft, the experience gained in the development of contingency plans is an excellent preparation for the recovery actions necessary to respond to an actual fault. Contingency planning is a means of forcing critical thought processes due to the "what if..." nature of questions that need to be answered, and, even though the actual anomalies will often be different from what was anticipated, it will provide guidelines and uncover various approaches to problem resolution when actual in-flight anomalies occur.

CONCLUSION

Although an unmanned planetary spacecraft is physically inaccessible following its launch, it will often require repair and maintenance to be performed in order to accomplish mission objectives. The characteristics of these interplanetary missions require unique design features to maximize the likelihood of success over mission lifetimes on the order of a decade or more. The vast distances involved require a greater emphasis on autonomous fault detection

and correction than most near Earth missions, but the presence of a capable flight team is still required to handle each situation as it arises in an optimal manner to enhance the overall mission return.

ACKNOWLEDGEMENTS

This paper presents one phase of research carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract NAS7-100, sponsored by the National Aeronautics and Space Administration.