Attitude and Articulation Control for the Cassini Spacecraft: A Fault Tolerance Overview

G. Mark Brown, Dr. Douglas E. Bernard, Dr. Robert D. Rasmussen

Jet Propulsion Laboratory, California Institute of Technology
Pasadena, California

ABSTRACT

This paper describes how fault tolerance has been addressed in the design of the Attitude and Articulation Control Subsystem for the Saturn-bound Cassini spacecraft. Cassini's fault tolerance objectives have strongly influenced the subsystem's level of autonomy, and have motivated some significant improvements over the autonomous capabilities of previous interplanetary spacecraft. Autonomous fault tolerant behaviors have been embedded at several points in the object-oriented flight control software, including a dedicated set of failure detection, isolation, and recovery algorithms.

THE CASSINI MISSION AND SPACECRAFT DESIGN

The Saturn-bound Cassini spacecraft, shown in Figure 1, will be launched in October 1997 by a Titan IV launch vehicle with a Centaur upper stage. After an interplanetary cruise of almost seven years that includes several large propulsive maneuvers as well as gravity assists from Earth, Venus, and Jupiter, Cassini will reach Saturn in June 2004 and establish an orbit about the planet. The orbiter will tour the Saturnian system through June 2008, using its suite of twelve science instruments to acquire detailed images and sense the local environments. The Huygens probe will be released on the first or second of many planned encounters with Titan, the only moon in the solar system with a substantial atmosphere.

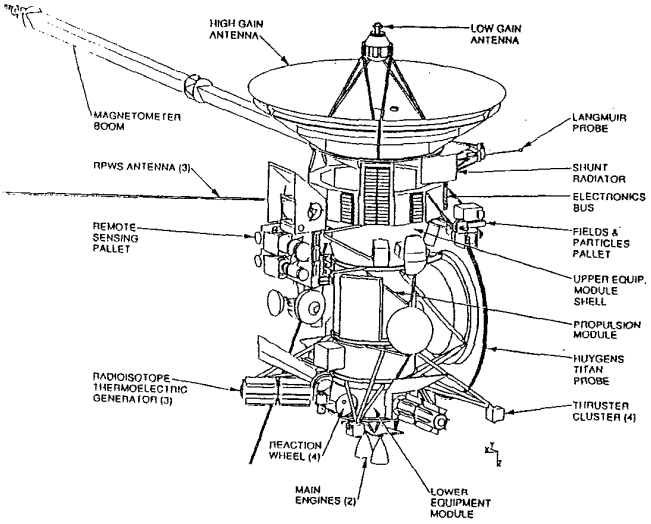

Figure 1: The Cassini Spacecraft

Cassini is a three-axis stabilized spacecraft with rigidly mounted antennas and rigidly mounted science instruments. At close solar distances, the spacecraft shades its thermally sensitive elements by pointing its four meter diameter High Gain Antenna directly at the Sun, while communicating with Earth via one of two available Low Gain Antennas. During the Saturn orbital tour, the spacecraft attitude is much less constrained, and can therefore can be dictated by scientific objectives for periods of up to sixteen hours per day. During the remaining eight hours per day, the spacecraft points its High Gain Antenna to Earth and "plays back" recently gathered scientific data from a four gigabit onboard recorder.

Three radioisotope thermoelectric generators (RTGs) provide approximately 825 watts at the beginning of the mission, decaying to 650 watts at the end of the mission. All onboard customers receive switched power from this source via a 30 volt power bus. The onboard avionics are distributed in several engineering subsystems, including:

The *Command and Data Subsystem* (CDS), which stores ground-provided command sequences, and then distributes stored commands at the appropriate times to the engineering subsystems and science instruments. CDS also coordinates all the onboard intercommunications, and collects and formats the science and engineering telemetry.

0-7803-3050-1/95 $4.00

The *Radio Frequency Subsystem* (RFS), which receives ground-transmitted commands for the CDS, and transmits CDS-formatted telemetry to the ground.

The *Attitude and Articulation Control Subsystem* (AACS), which is described in the following section.

ATTITUDE AND ARTICULATION CONTROL SUBSYSTEM

Cassini's Attitude and Articulation Control Subsystem (AACS) estimates and controls the spacecraft attitude, responding to ground-provided pointing goals for the spacecraft's science instruments and/or communications antennas with respect to targets of interest (e.g. Saturn, Earth). The AACS also executes ground-commanded spacecraft velocity changes.

Figure 2 is a functional block diagram of the Cassini AACS hardware. Each rounded box represents a separately powerable AACS assembly. Stacked groups of these boxes indicate the available redundancy. Circles indicate propulsion system elements that are controlled by the AACS.

The AACS supervisory, estimation, and control algorithms are executed by one of two block-redundant AACS Flight Computers (AFCs). Each AFC houses a 1750A microprocessor with 512 kwords of Random Access Memory (RAM). The object-oriented flight software is written in Ada; see Reference 1 for more information on the AACS flight software architecture.

The AFCs are remote terminals on the spacecraft's MIL-STD-1553B Command and Data Bus. Each AFC receives ground commands from CDS via this bus, and provides telemetry and other ancillary data to CDS for subsequent distribution to ground and onboard customers. For more details on Cassini's CDS and the Command and Data Bus, consult Reference 2.

The prime AFC coordinates the activities of all the AACS peripheral hardware via two-way transactions on a separately dedicated AACS Bus. The AACS Bus conforms to MIL-STD-1553B electrical standards, but uses a custom protocol that provides more than 32 remote terminal addresses and supports message lengths of more than 32 words.

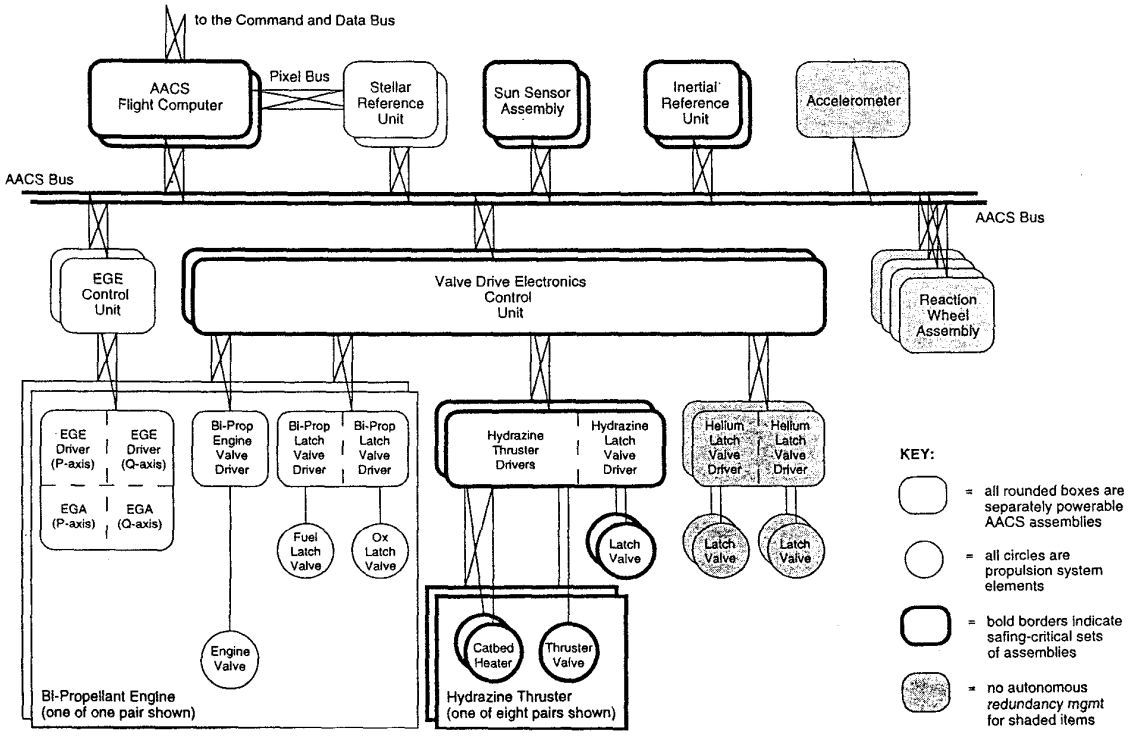

Figure 2: Cassini AACS Functional Block Diagram

Three-axis attitude sensing is provided by both block-redundant Inertial Reference Units (IRUs) and block-redundant Stellar Reference Units (SRUs). Each IRU contains four hemispheric resonator gyroscopes, arranged in an orthogonal-triad-plus-skew configuration. Each SRU is a 15-degree square field-of-view star tracker that can provide either AFC with up to 50,000 pixels per second through a dedicated Pixel Bus. AFC-resident software algorithms are able to establish and maintain stellar reference by comparing incoming pixel frames to an onboard catalog of approximately 5000 stars. Three to five stars are commonly tracked at any one time.

AACS acquires stellar reference by first locating the Sun and then Sun-pointing the High Gain Antenna using feedback from one of two available Sun Sensor Assemblies (SSAs). Each SSA provides two-axis digital gray-coded output over a 64-degree square field-of-view. Once stellar reference is acquired, AACS can maintain stellar reference while turning at slow angular rates; this allows the IRUs to be turned off and left off during most of the mission's seven-year interplanetary cruise phase.

The desired telecommunications rates during interplanetary cruise only require antenna pointing accuracies of a few milliradians. The AACS provides this level of pointing control via infrequent firings of 0.8 Newton hydrazine thrusters. The sixteen available thrusters and the other propulsion elements are all managed by one of two block-redundant Valve Drive Electronics (VDE) control units.

Scientific observations during the Saturn orbital tour require higher pointing accuracy and stability, as well as frequent spacecraft repositioning. During this mission phase, the AACS employs three orthogonally oriented Reaction Wheel Assemblies (RWAs), and the thrusters are used primarily for momentum mnagement. Turn rates during both the cruise and orbital phases are kept below 0.75 deg/sec.

Small velocity corrections are accomplished by timed firings of the hydrazine thrusters. For large velocity corrections such as Saturn Orbit Insertion (SOI), the AACS employs one of two available 450 Newton bi-propellant engines. Each engine nozzle is articulated by two physically-dedicated Engine Gimbal Actuators (EGAs) whose extensions are managed by either of two available Engine Gimbal Electronics (EGE) control units. Engine burns are terminated autonomously based on feedback from a single accelerometer (ACC).

CASSINI'S FAULT TOLERANCE OBJECTIVES

Cassini has two fundamental fault tolerance objectives that strongly influence its level of autonomy:

1) During all mission phases, the spacecraft must be able to ***fail safe*** by autonomously locating and isolating any single failure, recovering to a thermally safe and commandable attitude, and then waiting for further instructions from the ground. Since ground operators only attempt to make contact with the spacecraft once per week during low-activity mission phases, the spacecraft must be able to keep itself safe for at least two weeks.

2) During a few selected time-critical activities (e.g. Launch, SOI, and Probe Relay), the spacecraft cannot afford to simply isolate a failure and wait for ground assistance. At these times, the spacecraft must be able to ***fail operational*** by autonomously recovering a much larger set of its capabilities and then proceeding with a previously uplinked "critical" command sequence.

It is important to note that the ground-generated command sequences for time-critical activities contribute an important part of the spacecraft's capability to fail operational. Following an autonomously detected failure in a critical sequence, the CDS temporarily suspends the sequence, coordinates an autonomous fail-safe response, and then restarts the sequence from the last achieved "checkpoint". Each critical sequence has several such "checkpoints", each of which enforces any necessary differences between the fail-safe configuration and the operational configuration for subsequent critical sequence activities.

AACS AUTONOMOUS FAULT TOLERANCE ALGORITHMS

Figure 3 illustrates the architecture of the AFC-resident flight software algorithms that perform autonomous detection, isolation, and recovery from failures of AACS equipment and the AACS-controlled propulsion elements. The primary architectural goals, all of which are motivated by recognized shortcomings of past interplanetary spacecraft, are to:

1) improve the diagnostic accuracy of the algorithms by creating and making use of explicit "goodness" indications in addition to the typically available "badness" indications

2) improve the operability of the spacecraft by using knowledge of the subsystem's present goals to choose an appropriate level of response

3) decompose the autonomous tasks in a natural manner to facilitate near-term analysis and testing, as well as long-term comprehension and augmentation.

The primary architectural components are:

Error Monitors, which test local performance measures against expectations, apply discriminating filters, and then output a color-coded opinion. The available output colors and their interpretations are:

Black:	no opinion
Green:	performance meets expectations
Yellow:	performance is unexpected, but does not merit autonomous response
Red:	performance is anomalous, and merits immediate autonomous response

Activation Rules, which evaluate subsets of the color-coded error monitor outputs in the context of the subsystem's current hardware configuration and activity goals, diagnose the most likely causes of anomalous behaviors, and then activate one or more appropriate response scripts.

Response Scripts, which isolate failed equipment and recover a desired level of subsystem functionality. They do so by issuing commands directly to the subsystem, directing the activities of autonomous repair managers, and/or requesting external assistance via CDS-collected alert messages.

Repair Managers, which track the success or failure of past corrective measures for each piece of equipment, and then determine the most appropriate corrective measure to try next. Repair Managers use the same command interface as Response Scripts, and thus they can be thought of as special-purpose subroutines that are exercised by the Response Scripts.

All four of the above-described components are exercised during each AACS computation cycle (i.e. every 125 msec). The Error Monitors are distributed throughout the flight software, usually at the earliest possible test point. Tests are performed and opinions are generated throughout each computation cycle.

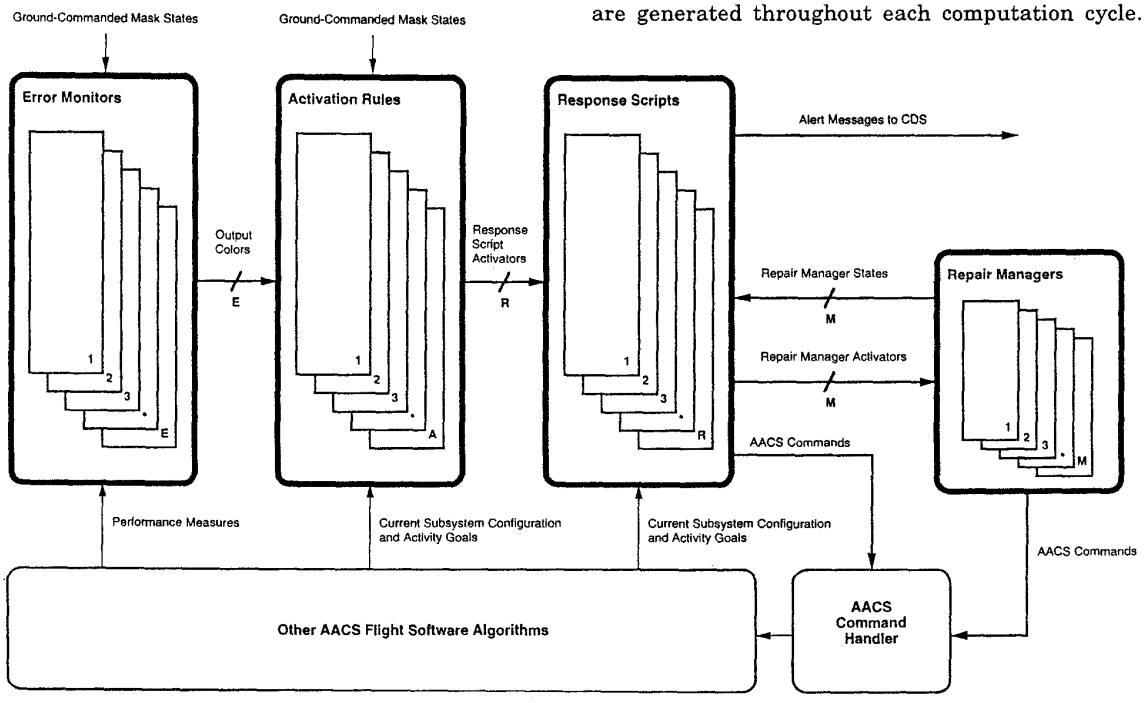

Figure 3: Architecture of AACS Fault Tolerance Algorithms

The Activation Rules, Response Scripts, and Repair Managers are all centralized and are executed during reserved time slices at the end of each computation cycle. Response Scripts are prioritized via a simple ordered list. Two or more Response Scripts can be simultaneously active; however the actions of lower priority scripts are delayed when higher priority scripts consume all of the available computation time.

If necessary, ground operators can tailor the behavior of these algorithms through two sets of commandable mask states. Error monitors can be individually masked to prevent autonomous responses to particular physical assemblies. For instance, ground operators could choose to mask the monitor that tests the magnitude of the tachometer output from RWA2, while leaving unmasked the similar monitors for RWAs 1, 3, and 4. When an error monitor is masked, its output color immediately becomes black. Activation rules can also be individually masked, preventing autonomous responses to particular diagnoses. For instance, ground operators could choose to prevent activation of responses to all apparently stuck-closed bi-propellant latch valves, while continuing to allow activation of a response to a related diagnosis such as an underperforming bi-propellant engine.

AACS AUTONOMOUS FAULT TOLERANCE CAPABILITIES

This section summarizes the autonomous fault tolerance capabilities that the Cassini AACS provides through the behaviors of its dedicated fault detection, isolation, and recovery algorithms, in addition to the autonomous behaviors that have been embedded at other appropriate places in the AACS flight software.

Failing Safe

The AACS is an active participant in Cassini's spacecraft-wide fail-safe response, which is autonomously directed by the CDS. Upon receiving the designated fail-safe command from the CDS, the AACS activates a specially-dedicated Response Script; this script establishes a minimum-power AACS hardware configuration, enforces a benign propulsion system state, and starts acting on a CDS-provided fail-safe attitude goal.

Returning to Figure 2, the "safing-critical" AACS hardware sets are indicated by bold borders. At least one member of each "safing-critical" set must be functioning properly in order for the AACS to maintain a thermally safe, commandable attitude. The AACS autonomously detects failures of prime "safing-critical" hardware and initiates replacement of a failed prime with its block-redundant unit.

Onboard vector propagation algorithms are used to periodically recalculate the direction to the fail-safe pointing target (e.g. the Earth). This allows AACS to continue providing a safe commandable attitude for an indefinite period while waiting for the ground to uplink new pointing commands.

Failing Operational

As previously described, Cassini must be able to fail operational during Launch, Saturn Orbit Insertion (SOI), and Probe Relay. Since the AACS "safing-critical" hardware (see Figure 2) supports both the Launch and Probe Relay activities, it is not difficult for the AACS to resume these activities following a fail-safe response. SOI, however, requires the AACS to resume and complete a commanded velocity change. This motivates additional AACS autonomy above and beyond the fail-safe capabilities, including:

- Autonomous reconfiguration of the propulsion system, in order to support a second burn attempt on the backup bi-propellant engine.

- Autonomous adjustment of the ΔV-to-go, in order to account for the partial ΔV achieved during a prior burn attempt, and the time delay associated with the later burn attempt.

Tolerating Multiple Failures

Cassini is not required to survive all possible second failures following a prior failure, since doing so would require triply redundant equipment. But multiple recoverable failures are certainly possible on a mission of this duration, and therefore the AACS is designed to autonomously find an operational configuration when one still exists. If necessary, the Repair Managers can autonomously exercise several independent degrees of freedom that are provided by the hardware and software architecture, including:

Independent prime selections. The prime selection for each type of assembly can be made independent of other prime selections. This allows the AACS to autonomously recover from a failure of SRU_A by simply making SRU_B the prime SRU,

while continuing to use the remaining healthy A side hardware (e.g. SSA_A, IRU_A,etc.).

Independent bus selections. The AFC can operate the two AACS Busses in a time-multiplexed manner, communicating with different subsets of peripherals on different busses during the same computation cycle. Each assembly's bus selection can be made independent of other bus selections. For instance, following a prior failure of SRU_A, AACS can autonomously respond to a failure of SRU_B's transceiver on AACS Bus A by changing SRU_B's bus selection to AACS Bus B, while continuing to use AACS Bus A for all the other peripherals.

Composite prime sets. Where feasible, the AACS can mix subsets of equipment from block-replaceable units to form a composite prime set. For example, following a failure of one gyro in IRU_A and one gyro in IRU_B, the AACS can autonomously construct a composite "prime IRU" using three gyros from IRU_A and one gyro from IRU_B. A composite set of "prime thrusters" can also be constructed using subsets of thrusters from the two available thruster branches.

After successfully responding to an anomaly, the AACS supports ground-based troubleshooting by allowing ground operators to power any or all of the backup AACS assemblies, send commands to them and receive telemetry from them while keeping them in an offline standby mode. If a particular assembly has indeed failed and can no longer be used, ground operators can inform the AACS flight software of this conclusion by changing the "health state" of that assembly. An autonomous Response Script will always check the "health state" of a backup assembly before attempting to bring it online as a replacement for an apparently failing prime.

Tolerating Single Event Upsets

Ionizing radiation can induce currents in solid state materials that are sufficient to cause logic state changes, known as Single Event Upsets (SEUs). Primary sources of ionizing radiation in the space environment are galactic cosmic rays and the solar wind. For an excellent overview of radiation sources and radiation tolerance issues for spacecraft computers, consult References 3 and 4.

SEUs are transient effects that can be corrected by re-commanding the desired state of the affected logic circuit. The challenge with SEUs is to minimize the frequency at which they occur, and to detect them and locate them properly when they do occur. The Cassini AACS approach is to use parts that won't upset too frequently in the mission's "worst-case" radiation environment (i.e. a solar flare at 0.6 AU), and to then provide the following autonomous safeguards against their effects:

AFC Error Detection and Correction (EDAC). The AFC automatically detects and corrects all the single-bit errors that it encounters in its RAM. The AFC also detects multiple-bit errors, which can occur when two or more SEUs affect bits within the same word. Upon encountering a multiple-bit error, the AFC immediately resets. Resetting does not correct the error, but gives CDS the opportunity to either autonomously re-load the corrupted AFC's memory, or prime-select the other AFC.

Message Checksums. All the critical information that enters or exits the AACS via the Command and Data Bus is transferred via checksummed messages. All AACS Bus transactions in both directions are also checksummed.

State Enforcement. Every commandable state of every AACS peripheral assembly is explicitly enforced via AACS Bus commands during each computation cycle. There are no "set and forget" commands.

Reset Recovery of Peripherals. When activated, an autonomous Repair Manager for a peripheral AACS assembly proceeds initially on the hypothesis that the assembly has experienced an SEU or some other type of reset-recoverable failure. Each Repair Manager attempts a minimum of two resets before proceeding to isolate and/or replace an apparently failed assembly. Each Repair Manager also recognizes that SEUs can occur repeatedly, and is willing to continue resetting an assembly as long as the subsystem can tolerate that reset frequency. If an assembly responds to an autonomously requested reset with error-free behavior for a sufficiently long interval, the Repair Manager "times out" and forgets all prior resets. The assembly is isolated and/or replaced only after several closely spaced resets have failed to correct the faulty behavior.

The autonomous Response Scripts never "give up" on any of the assemblies, no matter how many prior assembly failures are detected, or the frequency with which such failures are observed. If SEUs occur at unexpectedly high rates across a set of block-replaceable units, they can only cause unnecessary

power cycling and/or resetting of the affected units as they are repeatedly isolated and replaced in the operational configuration.

Tolerating Failures of Other Subsystems

As outlined previously, the Cassini AACS is dependent on several other avionics subsystems for power, status information, and directions. Failures of other subsystems could cause an interruption or loss of one or more interface services. The AACS design helps contain the effects of failures in other subsystems by tolerating:

Interruption of Power. All of Cassini's engineering subsystems, including the AACS, are required to tolerate infrequent power outages of up to 37 msec in duration. There are no credible faults in the power supply and distribution system that cause longer-duration interruptions. The Cassini AACS meets this requirement via an integrated hardware and software solution:

1) Capacitive power storage in each AFC's power supply keeps the RAM flight software program intact across a brief power outage.

2) A ROM-resident "boot" program recognizes the difference between a "warmboot" (e.g. from a brief power interruption or a flight-software-initiated reset) and a "coldboot" (e.g. from a previously unpowered state). The ROM responds to a "coldboot" by performing a CDS-assisted memory load, but can autonomously respond to a "warmboot" by quickly validating and transferring execution back to its unaffected RAM program.

Loss of Status Information. Other subsystems provide AACS with power switch status, bi-propellant engine temperatures and chamber pressures via the CDS-controlled Command and Data Bus. The CDS also provides AACS with spacecraft time and synchronization signals. AACS is required to tolerate loss or interruption of all this information, which could occur during CDS autonomous responses to failures affecting the Command and Data Bus. AACS employs either of two primary strategies, depending on the criticality of the information to the subsystem:

1) Some information, such as the spacecraft time, is essential to AACS flight software processing; in order to tolerate loss of this data, AACS must provide a backup source. In the absence of fresh valid time transmissions from CDS, AACS propagates its own internal time, and uses that time instead. When and if fresh valid time transmissions resume, the AACS monitors the difference between the internally-propagated time and the new CDS-transmitted time. If a significant discontinuity exists, a Response Script is activated that suspends all time-sensitive activities, and then synchronizes to the new time.

2) Other information, such as the engine temperature and pressure measurements, are used only for fault detection purposes. This information is not essential to the AACS, and therefore a backup source is not required. In the absence of fresh valid engine temperature and pressure measurements, AACS immediately sets the output colors of the affected error monitors to black (i.e. "no opinion"), and continues to evaluate the engine performance using the remaining available information. For instance, loss of an engine's chamber pressure measurement would prevent that measurement from contributing to an engine failure diagnosis, but would not prevent AACS from arriving at the same diagnosis via accelerometer measurements.

Corruption of Status Information. All incoming status information is tested for freshness using either an explicitly provided time-tag, or a monotonically increasing message counter. All critical information is also checksummed.

Failure of the CDS. Although the Cassini CDS is a critical engineering subsystem that must be fault tolerant (see Reference 2), the AACS is expected to tolerate a long-term loss of CDS services, and to respond by establishing a thermally safe, commandable attitude that can support ground-in-the-loop CDS recovery activities. The AACS meets this requirement by monitoring the length of time that certain fundamental Command and Data Bus transactions (e.g. the spacecraft time broadcast) have been absent, and then activating an "unsupported safing" Response Script. This Response Script recognizes that CDS is not available to coordinate AACS-requested changes in the AACS hardware configuration, and so it simply initiates a turn to an internally-stored "default" safing attitude goal (e.g. Low Gain Antenna 1 to the Sun) using the currently available sensors and actuators.

Tolerating Operator Errors

Previous interplanetary spacecraft have not been required to tolerate operator errors in a graceful manner. Instead, previous missions have relied heavily on ground-based constraint checking, and their spacecraft have been programmed to execute their fail-safe response upon encountering anything resembling an errant command. Previous interplanetary spacecraft have also been equipped with a CDS-resident "Command Loss" response that safeguards against serious operator errors by enforcing the fail-safe configuration following an extended absence of ground contact.

Cassini also has a CDS-resident "Command Loss" response, and the Cassini AACS is not required to tolerate operator errors. Admittedly, it is very difficult to autonomously guard against all the possible errors of both commision and omission. However, the Cassini AACS has been designed to tolerate those types of operator errors that it can recognize and disposition in a straightforward manner. Autonomous features at the user interface include:

Syntax Validation. The AACS flight software verifies that all incoming commands have the proper checksum and proper length. In addition, the arguments of each command are verified to be within pre-determined legal limits.

Configuration Safeguards. The AACS flight software recognizes that some operating configurations are fundamentally unsafe, and refuses to execute any command that would yield such a configuration. Many of these safeguards apply to configuration of the propulsion system. For example, AACS recognizes that it is unsafe to have any of the propulsion drivers powered without having any control or knowledge of their behavior. Therefore, AACS rejects commands to power any of these drivers when the prime VDE control unit is powered off.

Resource Requirements. The AACS flight software guards against errors of omission by autonomously requesting and verifying the availability of all the required resources for each activity before even attempting to begin that activity. As an example, when the AACS is commanded to reaction wheel control mode, it first requests power for the three prime reaction wheels (these requests may be redundant with the pre-existing configuration), and then verifies that the three reaction wheels are communicating via AACS Bus transactions. The flight software also guards against errors of commission by rejecting any commands that would remove a required resource. For instance, when the AACS is in reaction wheel control mode, it will reject any commands to power off any of the three prime reaction wheels.

Modal Constraints. The AACS flight software recognizes that certain AACS commands cannot be executed in certain AACS Modes, and explicitly rejects those commands that are incompatible with the current mode. For instance, the AACS will reject user commands to set the EGA extensions during engine burns, since the EGA extensions are under autonomous control at that time.

Attitude Constraints. The AACS flight software enforces ground-prescribed attitude constraints, autonomouly substituting a "close as possible" attitude goal until the commanded attitude once again satisfies constraints. This feature not only protects against outright operator errors, but also handles unforeseeable interactions between operator-designed command sequences and the autonomous recovery from a failure-induced attitude excursion. For more details on Cassini's treatment of attitude constraints, consult Reference 5.

AACS provides the CDS with a running count of the number of commands accepted for execution. When AACS rejects a command for any of the above-mentioned reasons, the count is not incremented. Sequences can choose to monitor this command counter and respond to deviations from the expected counter value.

SUMMARY

Fault tolerance considerations have motivated several advancements in the autonomous capabilities of the Cassini AACS. These advancements support Cassini's fundamental objectives to conditionally fail safe or fail operational, and also allow the AACS to more gracefully tolerate single event upsets, multiple failures, failures in other subsystems, and operator errors.

ACKNOWLEDGMENT

This work was performed at the Jet Propulsion Laboratory, California Instititue of Technology, under contract with the National Aeronautics and Space Administration.

REFERENCES

1) J.C. Hackney, D. E. Bernard, and R. D. Rasmussen, "The Cassini Spacecraft: Object Oriented Flight Control Software", 16th Annual AAS Guidance and Control Conference, February 1993.

2) T. K. Brown and J. A. Donaldson, "Fault Protection Design of the Command and Data Subsystem on the Cassini Spacecraft", 13th Annual Digital Avionics Systems Conference, October 1994.

3) R. D. Rasmussen, "Spacecraft Electronics Design for Radiation Tolerance", Proceedings of the IEEE, Vol. 76, No. 11, November 1988.

4) R. M. Manning, "Low Cost Spacecraft Computers: Oxymoron or Future Trend?", 16th Annual AAS Guidance and Control Conference, February 1993.

5) R. D. Rasmussen et al, "Behavioral Model Pointing on Cassini Using Target Vectors", 18th Annual AAS Guidance and Control Conference, February 1995.