Contingency Management: Allocating Requirements Between Flight and Ground Software

Presented to
Workshop on Flight Software Effects on Ground Systems
2012 Ground Systems Architecture Workshop

Presented by
Myron Hecht
The Aerospace Corporation

February 29, 2012

Outline

- Objective
- Design Decisions
- Constraints
- Anomaly Descriptions
- Conclusions

Objective

- Show that design decisions for contingency and failure management should be made at the start of conceptual design – if not sooner

Design Decisions

Function	Contingency	Decision
Commanding	Defective or hazardous commands from ground to spacecraft	Allocation of command checking between ground and space; Extent of autonomy
Health Monitoring, Prognostics	Inadequate, uninformative, or misleading telemetry	What telemetry data and what intervals? Checks for multiple conditions
Uploads	Degradation or loss of functions due to improper or corrupted uploads	Allocation of upload checks between ground and space
Anomaly Mitigation	Errors of Omission or Commission making the situation worse	Autonomy, status, diverse alternatives, redundancy
Safe Mode Recovery	Spurious entry into safe mode, inability to exit safe mode	Autonomy, status, diagnostics

Constraints

- Orbit
  - *Vehicle visibility to ground*
- Communication channel
  - *Bandwidth*
  - *Channel availability*
  - *Security*
- On-board capabilities
  - *Computational capacity and architecture*
  - *Cross strapping design*
  - *Telemetry and diagnostics points*

Example 1a: No Command Checking

- Event: Vehicle enters extended sun safe mode,
- Cause: Very egregious typo, supposed to be 5 places to left of decimal point and 7 to the right, decimal point left out in command. Software not designed to detect that type of error.
- Impact: Vehicle outage
- Corrective Action
- Comment: Command syntax and semantics checking requirements should have been allocated to the FSW or the ground system (or both)

Source: The Aerospace Corporation SSED database (identifying data removed)

Example 1b: No Command Checking

- Symptom: An incorrect command load reversed configuration: Format 1 became High Gain, while Format 2 became Low Gain
- Cause: Operator error: Data must be processed with an Event-Specific file, cannot be processed by the default software
- Impact: Payload data was lost for 15 days
- Correction: The original gain settings were restored
- Comment: semantic command check in FSW or on ground would have prevented this failure (in at least one location)

Source: The Aerospace Corporation SSED database (identifying data removed)

Example 2: Insufficient or Misleading Diagnostics

- Event: A Status Valid flag went to false during the support.
- Cause: Reason unknown
- Impact: Not stated in report
- Corrective action: The status of this anomaly was changed to inactive due to the 6 month time limit.
- Comment: Had there been sufficient diagnostics, the reason (including possibly an incorrect condition for setting the flag) would have been known. One of many examples

Source: The Aerospace Corporation SSED database (identifying data removed)

Example 3: No Upload Checking

- Event: Upload broke four pages of code forcing the payload into an faulted condition.
- Cause: Non compliant upload in which the uplink antenna did not wait for an acknowledgement of receipt from the payload computer
- Impact: Could have caused complete lockup of payload computer
- Corrective Action: Corrected upload, restarted computer
- Comment: This failure mode would not have been caught on the ground. Upload checking and verification should have been done in the FSW

Source: The Aerospace Corporation SSED database (identifying data removed)

Example 4: Insufficient Autonomy

- Event: (a) Ground station down due to antenna gain and output power issues (b) picosat malfunction upon deployment
- Cause: Battery exhaustion, a delay in ejection, or a minor over-temperature
- Impact: transmissions lost from most or all of the picosatellites.
- Corrective action: Antenna issues resolved and operations returned to after 13 days (too late for picosats)
- Comment: Lack of autonomy functions in FSW combined with ground system unavailability caused loss of mission

Source: The Aerospace Corporation SSED database (identifying data removed)

Example 5: Failure to Exit from Safe Mode

- Event: Phobos Grunt Spacecraft rocket pack fails to fire
- Cause: Simultaneous reboot of both SC computers resulting in safe mode from which no exit was possible; rocket firing could not occur
- Impact: Orbit decays and vehicle lost
- Corrective action: review board
- Comment: An alternate path is to enable ground restart of normal processing if no autonomous exit from safe mode would have saved the mission

Source: Spaceflight Now, February 6, 2012, available online at http://www.spaceflightnow.com/news/n1202/06phobosgrunt/

Conclusions

- Contingency management is an easier problem to solve when there are fewer constraints
  - *Conceptual design is when there are the least constraints*
  - *This is the best time to allocate between space and ground software*
- Mistakes are easy to see in hindsight
  - *Methodologies are available to make them more visible in the course of development*