Formalized Pilot Study of Safety-Critical Software Anomalies

Robyn Lutz and Carmen Mikulski (JPL)

NASA Code Q Software Program Center Initiative UPN 323-08; Kenneth McGill, Research Lead

OSMA Software Assurance Symposium
Sept 5-7, 2001
Topics

- Overview
- Preliminary Results
  - Quantitative analysis
  - Evolution of requirements
  - Visualization tools
- Work-in-progress
- Benefits
Overview: Goal

To reduce the number of safety-critical software anomalies that occur during flight by providing a quantitative analysis of previous anomalies as a foundation for process improvement.
Overview: Approach

- Analyzed anomaly data using Orthogonal Defect Classification (ODC) method
  - Developed at IBM; widely used by industry
  - Quantitative approach
  - Used here to detect patterns in anomaly data
- Evaluated ODC using Formalized Pilot Study
  - R. Glass ['97] detailed rigorous process to get valid results
  - 35 steps divided into 5 phases
  - Used here to evaluate ODC for NASA use
Overview: Status

- Year 2 of planned 3-year study
  Plan → Design → Conduct → Evaluate → Use
- Adapted ODC categories to operational spacecraft software:
  - Activity: what was taking place when anomaly occurred?
  - Trigger: what was the catalyst?
  - Target: what was fixed?
  - Type: what kind of fix was done?
Preliminary Results: Quantitative Analysis

- Analyzed 189 Incident/Surprise/Anomaly reports (ISAs) of highest criticality
  - 7 spacecraft: Cassini, Deep Space 1, Mars Global Surveyor, Galileo, Mars Polar Lander, Mars Climate Orbiter, Stardust
- Institutional defect database → Access database of data of interest → Excel spreadsheet with ODC categories → Pivot tables with multiple views of data
- 1-D and 2-D frequency counts of Activity, Trigger, Target, Type, Trigger within Activity, Type within Target, etc.
Preliminary Results: 
Quantitative Analysis

- User-selectable representation of analysis results: tables, pie charts, bar graphs
- User-selectable sets of spacecraft for comparisons
- Provides rapid quantification of data
- Assists in detecting unexpected patterns, confirming expected patterns
Preliminary Results: Quantitative Analysis

Target Distribution

- Information Development: 24%
- Ground Software: 23%
- Flight Software: 30%
- None/Unknown: 2%
- Hardware: 2%
Preliminary Results: Quantitative Analysis

Distribution of Triggers within Activity

Data Access/Delivery
Hardware Failure
Normal Activity
Recovery
Special Procedure
Cmd Seq Test
Hardware Configuration
Inspection/Review
Software Configuration
Start/Restart/Shutdown
Unknown
Preliminary Results: Quantitative Analysis
Preliminary Results:

Evolution of Safety-Critical Requirements Post-Launch

- Anomalies sometimes result in changes to software requirements
  - Looked at 86 critical ISAs from 3 spacecraft (MGS, DS-1, Cassini)
  - 17 of 86 had Target (what was fixed) = Flight Software
  - 8 of 17 changed code only; 1 was incorrect patch; 1 used contingency command
  - Focused on remaining 7 with new software requirements as a result of critical anomaly
Preliminary Results:

*Evolution of Safety-Critical Requirements Post-Launch*

- Found that requirements changes are *not* due to earlier requirement errors
- Instead, requirements changes are due to:
  - Need to handle rare event or scenario (4; software adds fault tolerance)
  - Need to compensate for hardware failure or limitations (3; software adds robustness)
Preliminary Results:

*Evolution of Safety-Critical Requirements Post-Launch*

- Confirms value of requirements completeness for fault tolerance
- Confirms value of contingency planning to speed change
- Contradicts assumption that “what breaks is what gets fixed”
- Suggests need for better requirements engineering for maintenance
Preliminary Results: 
Evolution of Safety-Critical Requirements 
Post-Launch
Preliminary Results: 
Web-based Visualization Tool

- Results of Peter Neubauer (ASU), Caltech/JPL Summer Undergraduate Research Fellow, 2001
- Developed alternate visualizations of data results to support users’ analyses
- Web-based tool assists distributed users
- Sophisticated tool architecture builds on existing freeware
- Demo at QA Section Manager’s meeting (FAQ: Would this work for our project?)
- Demo to D. Potter’s JPL group developing next-generation Failure Anomaly Management System
Preliminary Results: Web-based Visualization Tool

Objective: Investigate and characterize the common causes of safety-critical, in-flight software anomalies on spacecraft. The work uses a defect-analysis technology called Orthogonal Defect Classification, developed at IBM. A rigorous pilot study approach using the Glass criteria is currently underway.

7 space missions: 189 defects classified; chart shows one of the 6 possible 2-way views into this information.

Large number of defects seen during sending commands to / receiving data from spacecraft.

Of these, many were responded to by changing operational procedures or software on the ground.

For other defects, changes to flight software more prevalent.

"Trigger" – what was happening to cause defect to be noticed

"Target" – what was changed to respond to defect

RRL/SAS 9/01
Work-in-progress

- Several patterns noted but not yet quantified
  - Ex: Procedures often implicated
- Profile by mission phase
  - Ex: Cruise, orbit insertion, entry, landing
- Better way to disseminate “mini-LL’s”? 
  - Ex: Corrective action sometimes notes need for similar action on a future mission
- Incorporate standardized ODC classifications in next-generation database to support automation and visualization
Benefits

- Data mining of historical and current databases of incidents / surprises / anomalies
- Uses metrics information to identify and focus on problem areas
- Provides a quantitative foundation for process improvement
- Equips us with a methodology to continue to learn as projects and processes evolve