FAQ-02
Are the problems with software development relevant within NASA?

Does NASA have Systemic Problems with Software?

Following with the trends in Industry, NASA is not void of software mishaps related to poorly defined requirements and design and integration problems.

According to a 2004 GOA Report -- Development cost estimates for more than half of the 27 programs that GAO reviewed have increased and for some programs this increase was significant—as much as 94 percent.

- Cost estimates changed for each of 10 programs that GAO reviewed in detail. For 8 of the 10 programs, the estimates increased.
- Although NASA cited specific reasons for the changes, such as technical problems and funding shortages, the variability in the cost estimates indicates that the programs lacked the sufficient knowledge needed to establish priorities, quantify risks, and make informed investment decisions, and thus predict costs.

Examples:

- **DART - 2005.** The Mishap Investigation Board (MIB) determined the root cause of the mishap was an inadequate Guidance, Navigation, and Control (GN&C) software development process (IV&V Facility performed an IA and source of error was not in scope).
- **Genesis – 2004.** The MIB said the likely cause was a design error that involves the orientation of gravity-switch devices (IV&V Facility performed an IA).

Does NASA have Systemic Problems with Software?

- **HESSI - 2000.** The damage was caused when the test device, called a "shaker," delivered approximately 20 G's, ten times the appropriate level for the test, to the spacecraft (IV&V Facility performed an Mission Readiness Assessment).
- **Mars Climate Orbiter – 1999.** The orbiter failed to move into a low circular orbit due to failure to use metric units in the coding of a ground software file.
- **Mars Polar Lander – 1999.** The flight software failed to terminate the engine thrust within 50 milliseconds after touchdown to avoid overturning the Lander (IV&V Facility performed an IA and source of error was not in scope).
- **Titan/Centaur/Milstar satellite – 1999.** An inadequate software development, testing, and quality assurance process for the Centaur upper stage. This led to the incorrect roll rate filter constant zeroed the roll rate data, resulting in the loss of the roll axis control and then yaw and pitch.

Does NASA have Systemic Problems with Software?

- **SOHO – 1998.** A series of errors in making the software changes along with errors in performing the calibration and momentum management maneuver and in recovering from an emergency safing mode let to the loss of telemetry.
- **Landsat-7** – Series of design and production problems, uncovered during system level testing, delayed launch and increased cost by more than $50M for the Landsat-7 program.
- **CLCS** was canceled by factors such as software development delays based on poorly defined requirement and design and integration problems (IV&V Facility performed IV&V).

Does NASA have Systemic Problems with Software? (Cont)

- ***Nancy G. Leveson -- "The Role of Software in Spacecraft Accidents."*** Almost all software-related aerospace accidents (and accidents in other industries) have been related to flawed requirements and misunderstanding about what the software should do. [1]
  - Although the details in each accident were different, very similar factors related to flaws were identified, which include:
    - Complacency and discounting of software risk, diffusion of responsibility and authority, limited communication channels and poor information flow.

The software had operated according to its requirements in most of these accidents. It was the requirements and the behaviors specified for the software that were wrong, which led to disastrous system behavior.

- Inadequate system and software engineering (poor or missing specifications, unnecessary complexity and software functionality, software reuse without appropriate safety analysis, violation of basic safety engineering practices in the digital components).

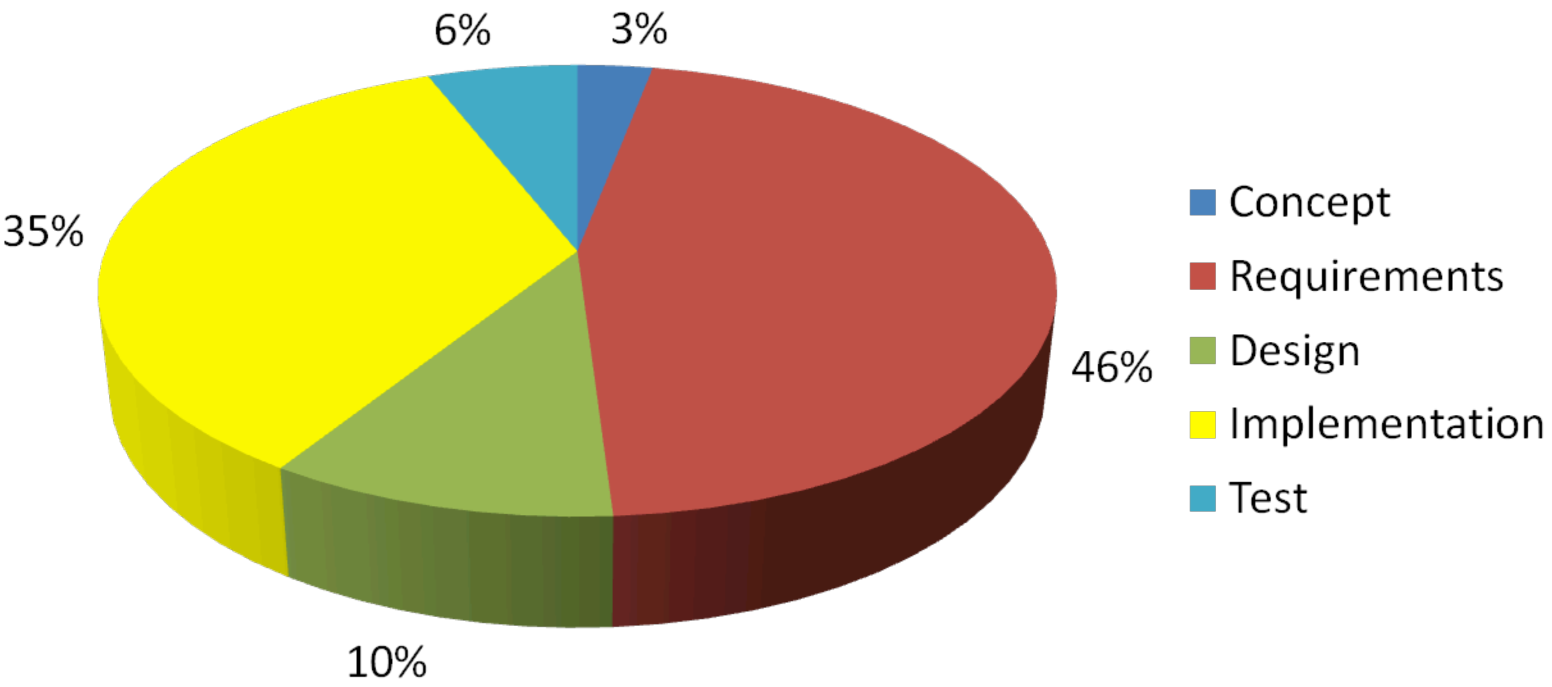
Causes for NASA accidents have been traced back to inadequate validation of requirements and flawed flow down from system requirements to software requirements.

- Inadequate review activities, ineffective system safety engineering, flawed test and simulation environments, and inadequate human factors engineering.

Does NASA have Systemic Problems with Software? (Cont.)

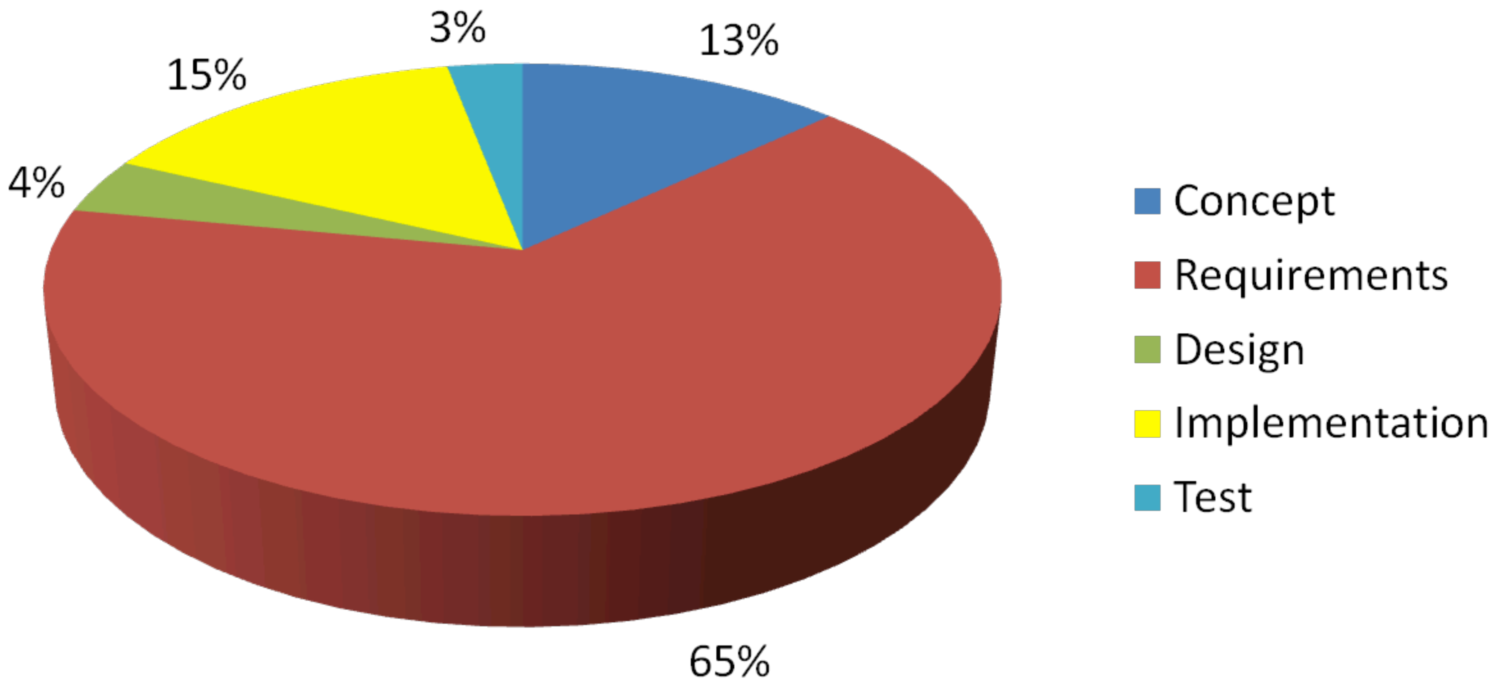
- Lutz examined 387 software errors uncovered during integration and system testing of the Voyager and Galileo spacecraft. [2]
- Lutz showed that for these two spacecraft, the safety-related software errors arose most commonly from:
  - Discrepancies between the documented requirements specifications and the requirements needed for correct functioning of the system
  - Misunderstandings about the software’s interface with the rest of the system.
- This experiential evidence points to a need for better specification review and analysis.

Distribution of Severity 1&2 Issues: NASA Robotics Systems



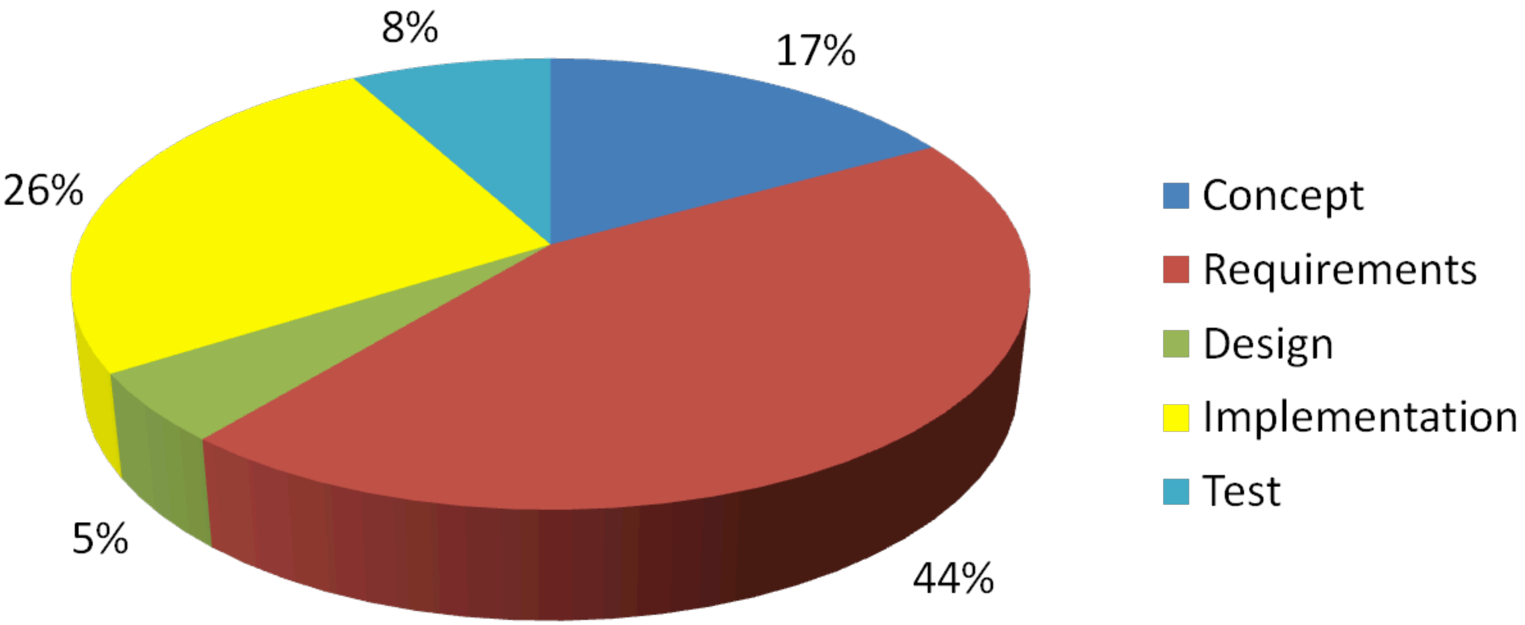
NASA IV&V found 46% of all Severity 1&2 issues in Requirements during IV&V of NASA Robotics Systems 3

Distribution of Severity 1&1N Issues: NASA Human-Rated Systems



NASA IV&V found 65% of all Severity 1&1N issues in Requirements during IV&V of NASA Human-Rated Systems [3]

Distribution of Severity 2&2N Issues: NASA Human-Rated Systems



NASA IV&V found 44% of all Severity 2&2N issues in Requirements during IV&V of NASA Human-Rated Systems [3]

07 January 2008

Summary of NASA Software Problems

- From NASA IV&V work, requirements problems ranged from 44% to 65% of all software problems. This is consistent with industry report of requirements problems being 56% of all software problems.

- If NASA is to build systems that meet the desired intent within budget, on schedule, and with all the desired features, then the NASA development community – to include NASA IV&V - must address the requirements problems during the requirements phase.

Backup Material

Robotics Severity Definitions

- Severity 1

  a) Prevent the accomplishment of an essential capability
  b) Jeopardize safety, security, or other requirement designated critical

- Severity 2

  a) Adversely affect the accomplishment of an essential capability and no work-around solution is known
  b) Adversely affect technical, cost or schedule risks to the project or life cycle support of the system, and no work-around solution is known

Human-Rated Severity Definitions

- Severity 1: A failure which could result in the loss of the human-rated system, the loss of flight or ground personnel, or a permanently disabling personnel injury.
- Severity 1N: A failure which would otherwise be Severity 1 but where an established mission procedure precludes any operational scenario in which the problem might occur, or the number of detectable failures necessary to result in the problem exceeds requirements.
- Severity 2: A failure which could result in loss of critical mission support capability.
- Severity 2N: A failure which would otherwise be Severity 2 but where an established mission procedure precludes any operational scenario in which the problem might occur or the number of detectable failures necessary to result in the problem exceeds requirements.

Bibliography

1. *Leveson, N.G. “The Role of Software in Spacecraft Accidents “ *AIAA J. Spacecraft and Rockets,* July 2004
2. *Reference: Lutz, Robyn, “Analyzing Software Errors in Safety-Critical, Embedded Systems”, JPL California Institute of Technology, February 24, 1994
3. NASA Independent Verification and Validation IV&V Program Metrics Progress Report - DEC06