

# Diverse Design

# Diverse Design

## Case Studies and Topics for Discussion

- Definition
- LEM abort computer
- Skylab Lessons Learned
- Space Station - ISS
- Software
- Shuttle Computers
- Small Satellites, University of Surrey

# Diverse Design Definition

In diverse design redundancy two or more components of different design furnish the same service.

This has two advantages: it offers high protection against failures due to design deficiencies, and it can offer lower cost if the back-up unit is a "life-boat," with lower accuracy and functionality, but still adequate for the minimum mission needs. The installation of diverse units usually adds to logistic cost because of additional test specifications, fixtures, and spare parts. This form of redundancy is, therefore, economical primarily where the back-up unit comes from a previous satellite design, or where there is experience with it from another source. Where there is concern about the design integrity of a primary component, diverse design redundancy may have to be employed regardless of cost.

# Diverse Design

## Case Study - LEM Abort Guidance Computer

- Main computer
  - 15-bit AGC, common with the CSM
  - Single string
- Not enough resources for redundancy
- TRW produced a small computer
  - MARCO 4418
  - 8-bit
- Limited functionality
  - Put the LEM in lunar orbit

From: **Computers Take Flight: A History of NASA's Pioneering Digital Fly-by-Wire Project**, Dr. Tomayko

# Diverse Design

## Skylab Lessons Learned

When designing redundancies into systems, consider the use of nonidentical approaches for backup, alternate, and redundant items.

### Background:

A fundamental design deficiency can exist in both the prime and backup system if they are identical. For example, the rate gyros in the Skylab attitude control system were completely redundant systems, i.e., six rate gyros were available, two in each axis. However, the heater elements on all gyros were identical and had the same failure mode. Thus, there was no true redundancy and a separate set of gyros had to be sent up on Skylab 4 for an in-flight replacement.

SKYLAB LESSONS LEARNED AS APPLICABLE TO A LARGE SPACE STATION, A dissertation submitted to the faculty of The School of Engineering and Architecture Of the Catholic University of America For the Degree Doctor of Engineering by William C. Schneider, Washington, D.C., 1976.

# Diverse Design

## Case Study: Space Station

- No intentional diverse design, despite Skylab's lessons learned<sup>1</sup>. Very expensive.
- Overlap in functions between US and Russia provides some diversity in ISS.
- Russian side has some diversity more as a result of heritage than an objective.

<sup>1</sup>As far as I know.

# Diverse Design

## Topic for Discussion: Software

- Not widely applied in software
  - Difficult to quantify expected improvement
- N-version Programming
  - In hardware NMR, there are identical copies; in software NMR, independent coding.
  - Voted: Reference states “sufficiently similar.”
- Limitation: 50% of faults in software control systems are in the specification

# Diverse Design

## Software Voting

In the N-version programming approach a number of independently written programs for a given function are run simultaneously; results are obtained by voting upon the outputs from the individual programs. In general the requirement that the individual programs should provide identical outputs is extremely stringent. Therefore, in practice "sufficiently similar" output from each program is regarded as equivalent; however, this increases the complexity of the voters [4.54].

# Diverse Design

## Case Study: Space Shuttle Computers

- Five Identical Sets of Computer Hardware
  - 4 run the primary software (PASS)
    - Each computer sees all I/O
    - Displays status to crew
  - 1 runs the Backup Flight System (BFS)
    - Runs during critical stages but does not control I/O unless engaged by the crew
  - Voting is done at the actuators (dynamic)
  - Crew provides decision making on switching redundancy (static)

# Diverse Design

## Case Study: Space Shuttle Computers

DG. How do you make the system reliable?

As I mentioned, there is a fifth computer that runs the Backup Flight System (BFS). Early on, NASA was concerned about the possibility of a generic software problem in the PASS. What if there were a "bug" in the PASS that brought the entire primary system down? The way they alleviated their fears was by developing independent ascent and entry software from a subset of the requirements they had given us. This independent software was written by Rockwell International and resides in the fifth computer.

...

The decision to engage the VGS is totally a crew function. Their procedures identify certain situations for which the switch should be made: for instance, loss of control, multiple consecutive failures of PASS computers, or the infamous two-on-two split where the computers split up into two pairs (we've never seen this occur). To date the crew has never had to use the BFS during a mission..

# Diverse Design

## Case Study: Space Shuttle Computers

Some more information on this is available from Computers in Spaceflight - The NASA Experience, James E. Tomayko, Wichita State University:

At first the backup flight system computer was not considered to be a permanent fixture. When safety level requirements were lowered, some IBM and NASA people expected the fifth computer to be removed after the Approach and Landing Test phase of the Shuttle program and certainly after the flight test phase (STS-1 through 4). However, the utility of the backup system as insurance against a generic software error in the primary system outweighed considerations of the savings in weight, power, and complexity to be made by [104] eliminating it.

[104] A.D. Aldrich, "A Sixth GPC On-Orbit," Memorandum, Johnson Space Center, Houston, TX, October 13, 1978, JSC History Office.

# Diverse Design

## Case Study: Small Satellites/Surrey

- Components: risk inherent in the use of components which are not formally “space qualified”
- New technologies: employed alongside flight-proven technologies in a “layered architecture”
  - Top-layer systems use state-of-the-art high-performance device types
  - Lower-layer systems use device-types which have been flown and tested in previous spacecraft, and which are able to carry out most of the same functions, albeit with a possible loss of performance.
- Layered architecture protects against design faults.

# Diverse Design

## Case Study: Small Satellites/Surrey

From the "Design Philosophy" section:

Recognising the risk inherent in the use of components which are not formally “space qualified”, we use redundancy at many levels to reduce the risk of total mission failure. When adopting new technologies, we employ them alongside flight-proven technologies in order to reduce risk. Thus we build a “layered architecture”, in which each successive layer relies on different systems comprising increasingly well-proven technologies. The top-layer systems use state-of-the-art high-performance device types - often without flight-heritage - but which give a high degree of functionality. Whereas the lower-layer systems use device-types which have been flown and tested in previous spacecraft, and which are able to carry out most of the same functions, albeit with a possible loss of performance. In this way, problems caused by an inherent system design fault, or by the failure of a particular device-type, are not duplicated in the different layers.