Software Reliability in Space Applications – Facts, Trends and Challenges

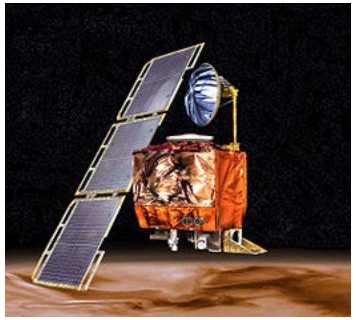
Ying Shi, NASA GSFC

Jan 28, 2020

Major Aerospace Failures due to Software

- Ariane 5 flight 501, 1996

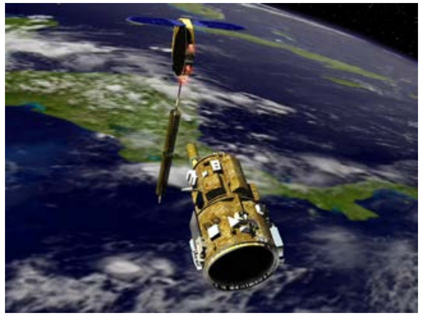
- MCO, 1998



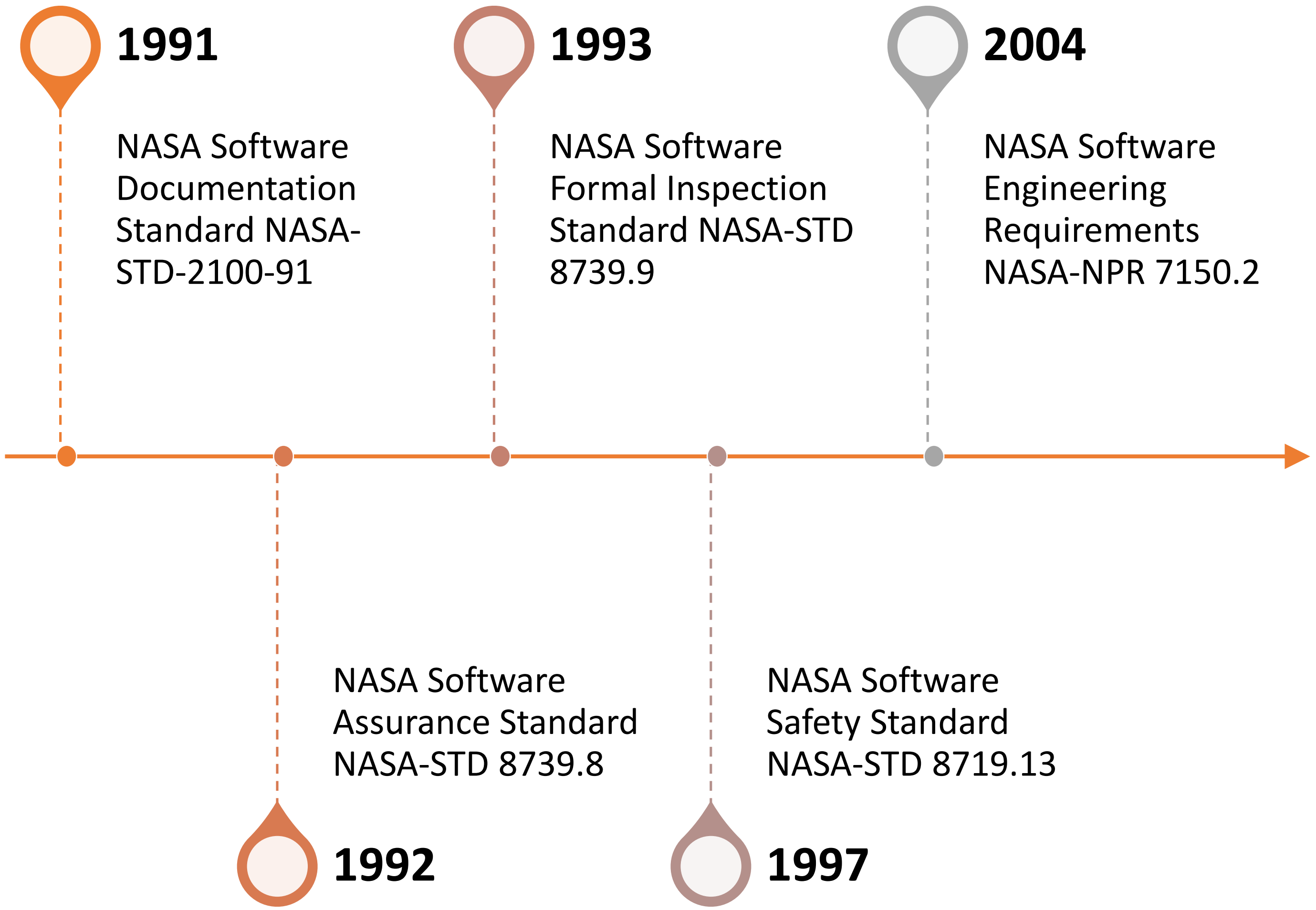
- MGS, 1996

- MPL, 1999

- DART, 2005



NASA Software Engineering & Assurance Pathway



1991

NASA Software Documentation Standard NASA-STD-2100-91

1992

NASA Software Assurance Standard NASA-STD 8739.8

1993

NASA Software Formal Inspection Standard NASA-STD 8739.9

1997

NASA Software Safety Standard NASA-STD 8719.13

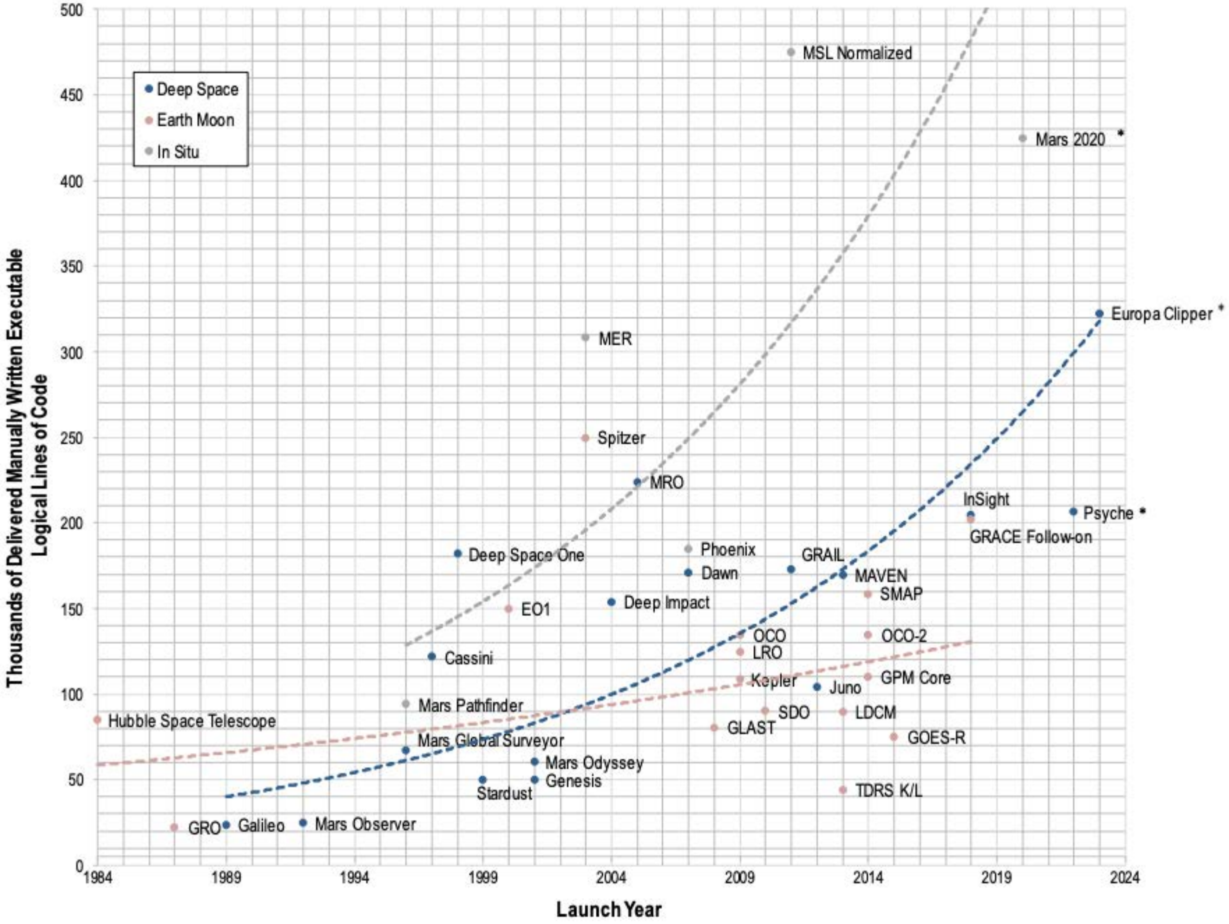
2004

NASA Software Engineering Requirements NASA-NPR 7150.2

Facts and Trends – Flight Software Size & Complexity

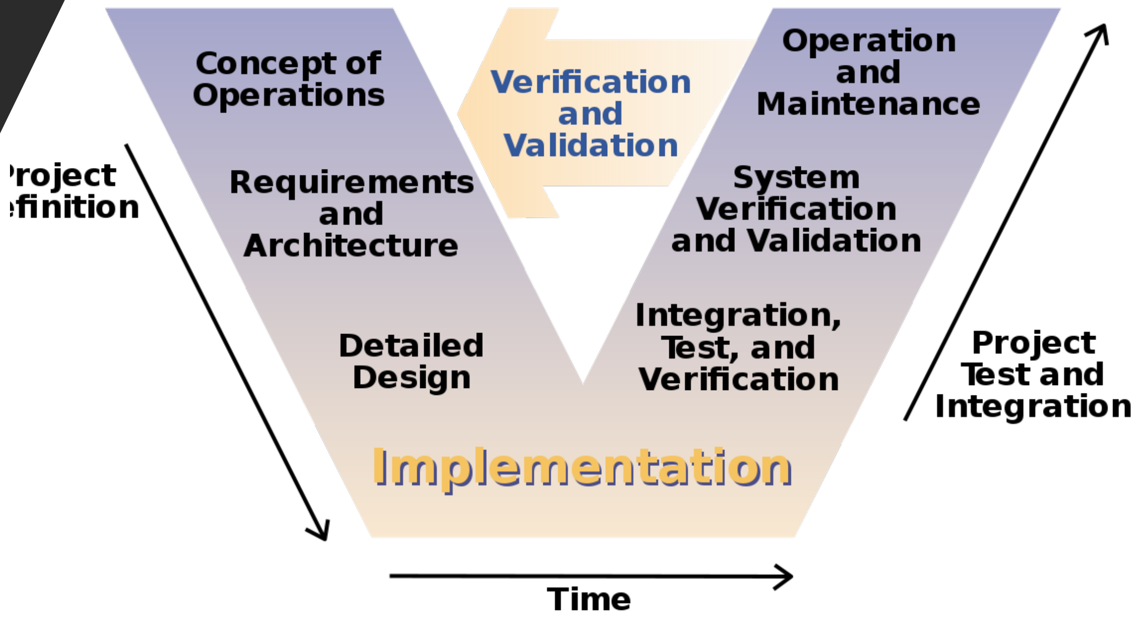
- **Estimated Size:**
  - Car: 100 MLOC
  - F-35: 22 MLOC
  - Orion: >1 MLOC
  - JWST: > 200 KLOC

- **Complexity**: How hard something is to understand or verify. -Dan Dvorak, JPL



Facts and Trends – Software Engineering

- C is still the primary language used in flight software; ground software uses a wider range of languages (C, C++, JAVA, Python etc.)
- Still follow modified Waterfall/Incremental development model
- Software reuse has increased dramatically over the last 10 years
- Increasing use of auto-generated code
- Growing interest in open source software
  - cFS at GSFC
  - F’ and FCPL at JPL



roject
finition

Concept of Operations

Requirements and Architecture

Detailed Design

Implementation

Integration, Test, and Verification

System Verification and Validation

Operation and Maintenance

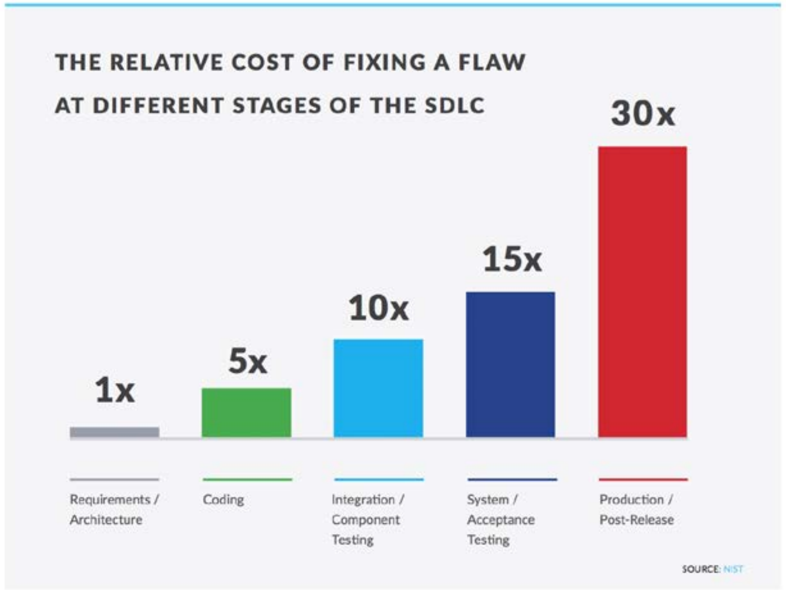
Verification and Validation

Project Test and Integration

Time

Facts and Trends – Defect Density & Fault Management

- Operations Defect Density continues to decrease
  - JPL (17 missions) : 0.38 defects/KLOC
  - GSFC(LRO): 0.46 defects/KLOC
  - Late 90’s: 0.6 defects/KLOC
  - Target: 0.1 Defect/KLOC
- Current Fault management is mainly designed for hardware faults
  - NASA Fault Management Handbook 2012
  - Needs to extend fault management to cover software faults



SOURCE: NIST

Challenges

Inadequate Reliability

- Increasing size and complexity
- Ineffective defect tracking and fault management
- Inadequate use of static Code analysis and coding rules

Schedule Overruns

- Tight schedule, not enough testing time and resources
- Estimation is overly optimistic at front

Education of Software Engineers

- Insufficient collaboration between System Engineers, Software Engineers and Software Assurance
- Suggest to include operations engineers in development

Unmet Requirements

New SW requirements

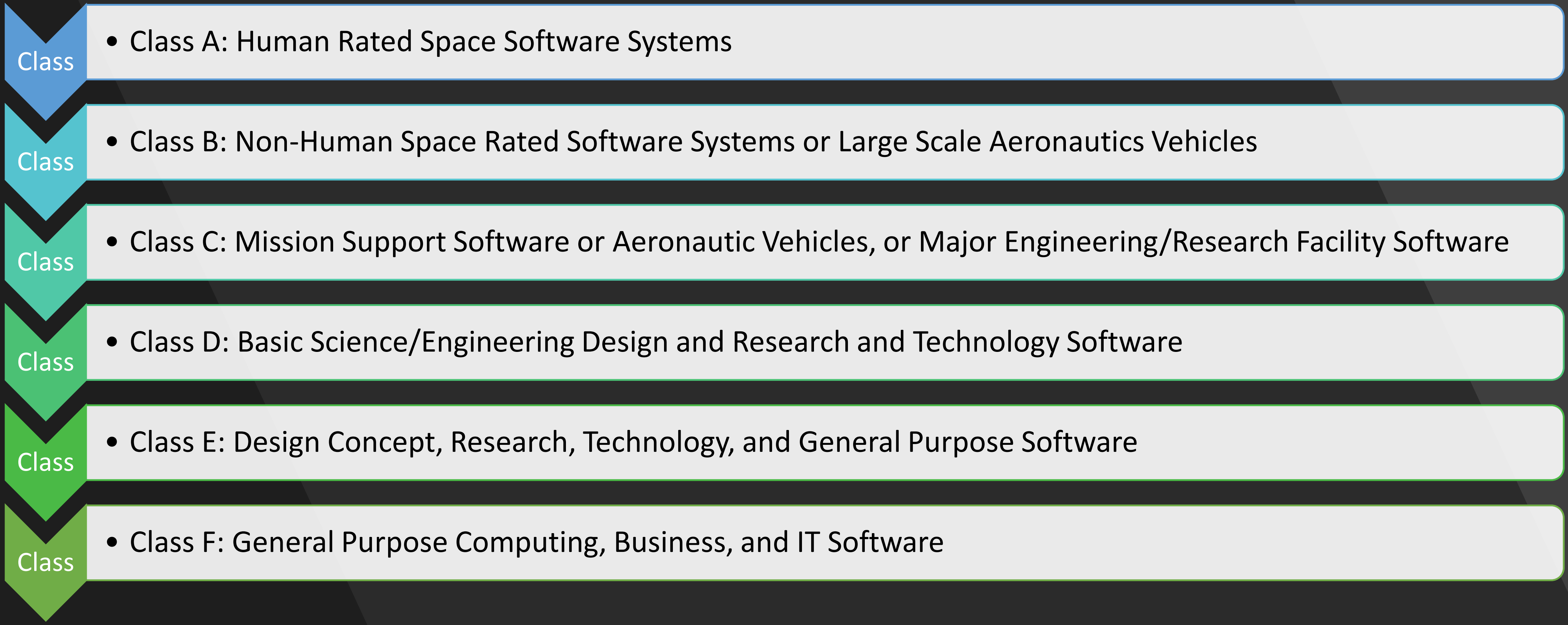
- On –board automatous detection and reaction
- On –board data processing, modeling, situation awareness

Software from multiple sources, test and integration?

Data collected but lack of data analysis

Backup

NASA Software Classifications – NPR 7150.2C



Class

- Class A: Human Rated Space Software Systems

Class

- Class B: Non-Human Space Rated Software Systems or Large Scale Aeronautics Vehicles

Class

- Class C: Mission Support Software or Aeronautic Vehicles, or Major Engineering/Research Facility Software

Class

- Class D: Basic Science/Engineering Design and Research and Technology Software

Class

- Class E: Design Concept, Research, Technology, and General Purpose Software

Class

- Class F: General Purpose Computing, Business, and IT Software