Michael Grant
Sikorsky Aircraft Corporation
A Lockheed Martin Company

Software Safety

What is it?

Approved for public release (SIK202202002)

Software Issues

- Software control of functions is evolving exponentially
- Software is an intangible element of design
- No methodology to predict specific “failure” probabilities
- Software defects can produce unpredictable outcomes
- Requirements differ between military and civil programs
- Testing will not identify all defects in highly complex systems
- Budget/schedule for exhaustive testing not available
- Even small changes can have catastrophic results

Objective

Implement a comprehensive software safety process at Sikorsky, that is standardized across programs and complies with all current standards and industry best practices, to ensure the inherent safety of our products that employ software in safety-critical applications.

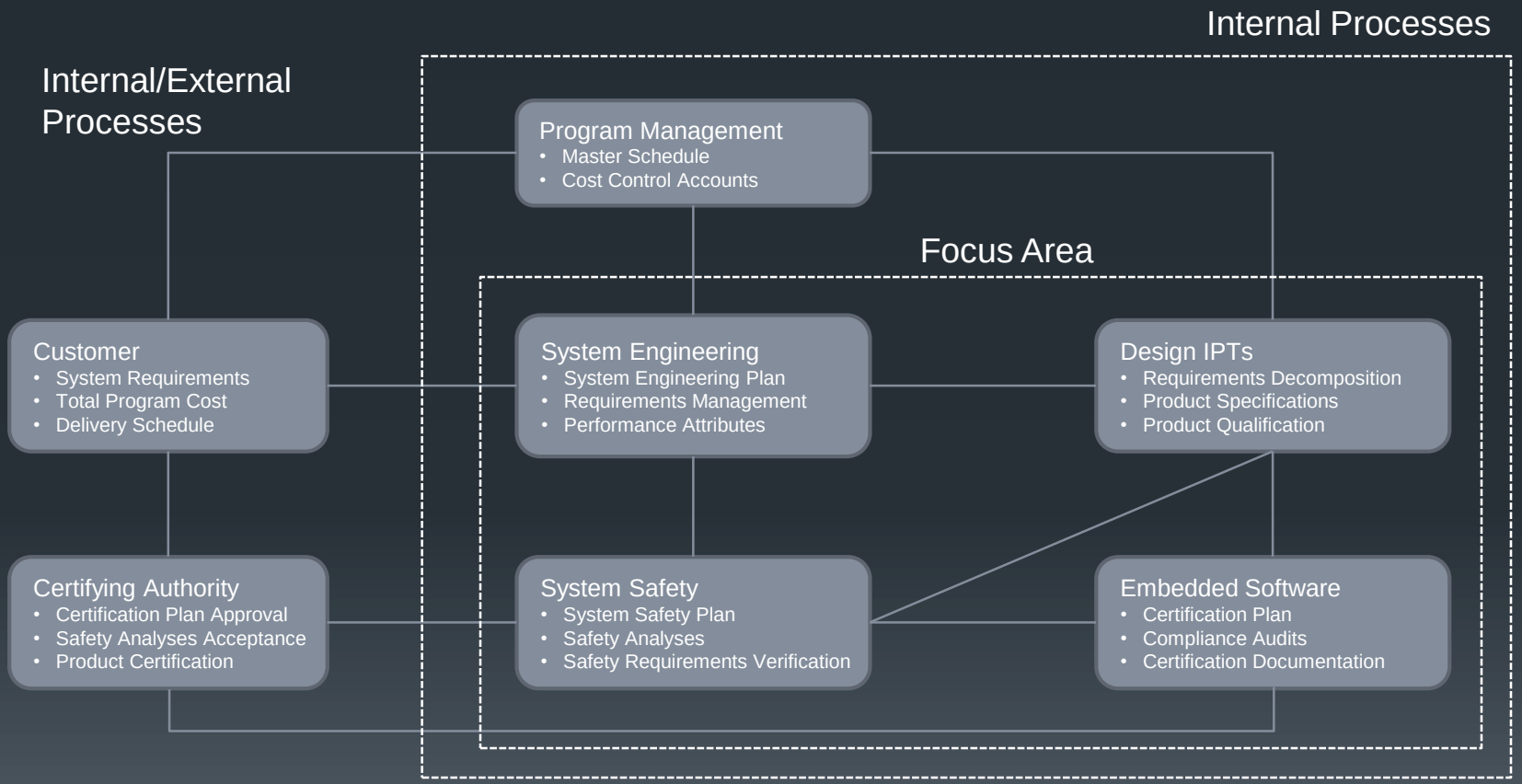
Software Safety Program Coordination Team Charter (2017)

Standards/Guidance

"Tip of the Iceberg"

- ISO/IEC/IEEE 12207: Systems and software engineering — Software life cycle processes
- ISO/IEC/IEEE 15288: Systems and software engineering — System life cycle processes
- IEEE 1228: Standard for Software Safety Plans
- IEC-1508: Functional Safety: Safety-Related Systems
- UL 1998: Standard for Software in Programmable Components
- RTCA/DO-178: Software Considerations in Airborne Systems and Equipment Certification
- MIL-STD-882E: Military Standard, System Safety Program Requirements

Problem Workspace

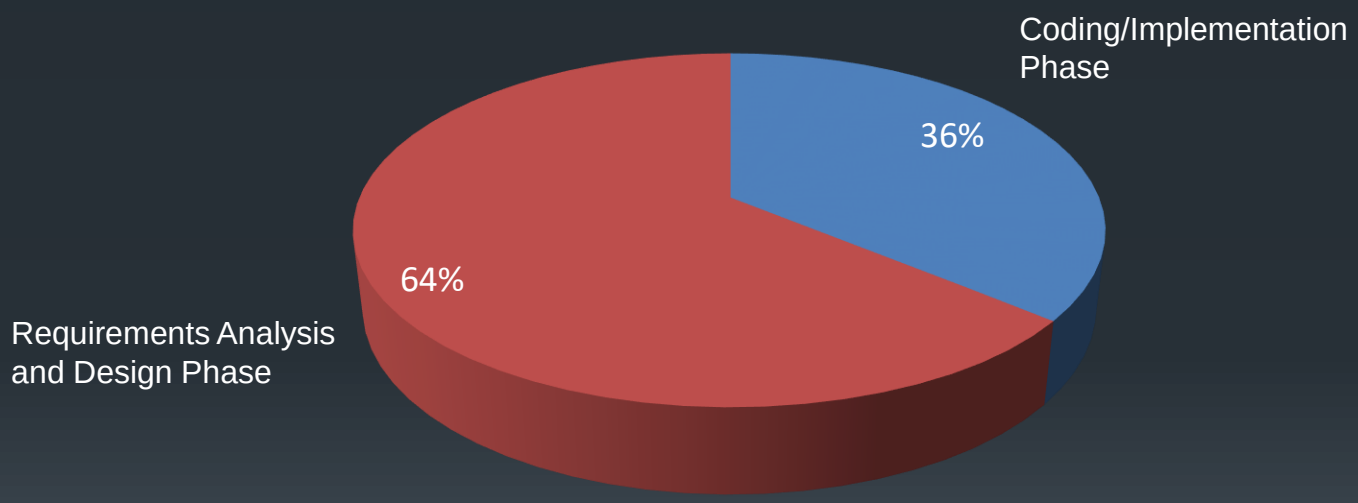


Responsibilities

- System Engineering – overall integration of design effort
- System Safety – hazard identification and classification to establish and verify software design requirements
- Design IPTs – product specifications and design documents to capture software design/qualification requirements
- Software Engineering – coordinate with certifying authority, provide certification plan and oversee software development

Software Defects

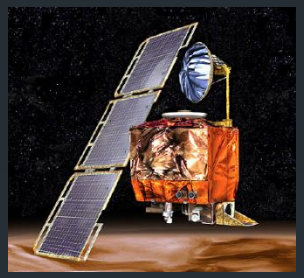
Origin of Software Defects



Source: Crosstalk, the Journal of Defense Software Engineering

Examples

Mars Polar Lander and Climate Orbiter



Climate Orbiter 1999

Polar Lander 1999

Both craft were lost due to requirements errors:

- The Mars Climate Orbiter was destroyed in the Mars atmosphere because the requirement for a supplier to use metric units was not verified.
- The Mars Polar Lander crashed because the requirement to inhibit touchdown sensor data until the craft was 12 meters above the surface was not flowed down.
- Total cost $328 million (not counting lost data and research opportunities).

Development Programs

Primary Customer Specified Software Process Drivers

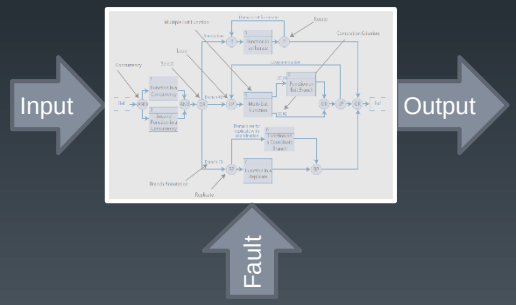
- ARP4754A – The most comprehensive of guidance material for aircraft system development and preferred FAA system safety process for aircraft certification.
- MIL-STD-882E – Not aircraft or software specific. Describes various system safety artifacts but does not provide any framework for implementation. Not recognized by civil certification authorities.
- MIL-HDBK-516C – Handbook for developing Airworthiness Qualification Plans (AQPs) that is very broad, fixed-wing oriented, and incorporates many other documents by reference.
- RTCA DO-178 – Defines software development assurance process and certification artifacts based on software hazard severity (de facto international standard).

Software

- Software is all or part of the programs, procedures, rules, and associated documentation of an information processing system. [ISO/IEC 2382-1:1993]
- Software is not an independent entity – by itself, it does nothing.
- In design, software should be treated as any other component part to include its potential to induce system faults or failures.
- In System Safety, hazard severity is based on the loss or corruption of a function in specific scenarios, not what caused the event.
- The source of the initiating hazardous event (causal factor) directly influences the method of mitigation.

Types of Analyses

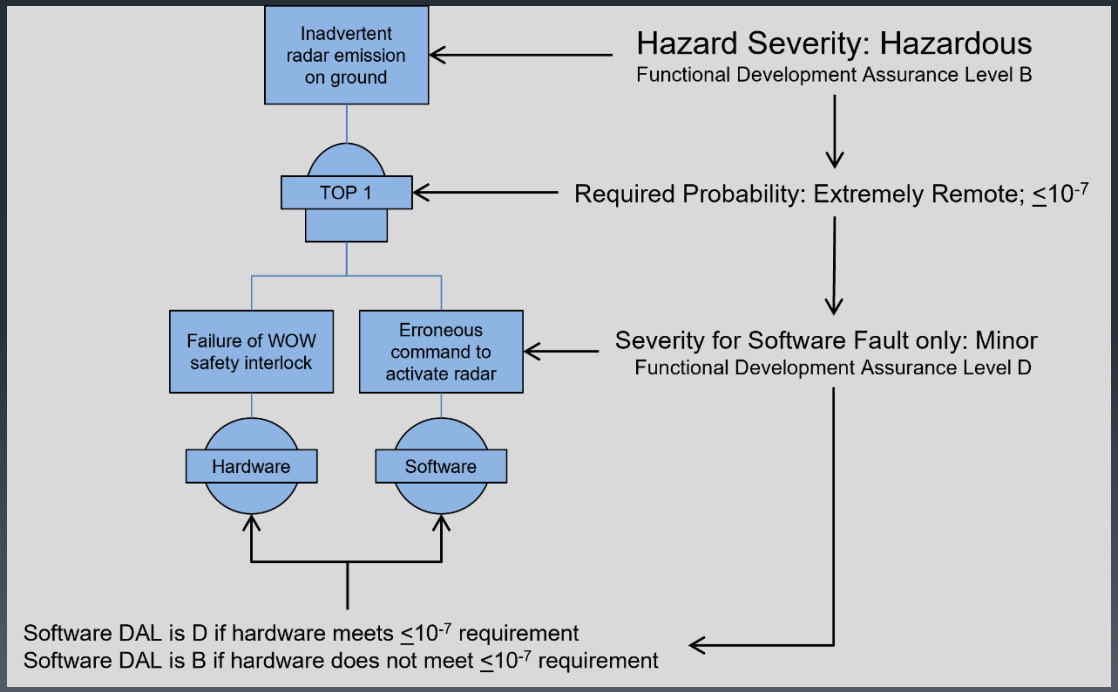
- Black Box - Analysis or test performed without having any knowledge of the interior workings of the component. The analysis simply evaluates the potential hazards from loss or corruption of the functions provided. Testing includes such items as removing power or disconnecting signal inputs/outputs.



- Grey/White Box – Analysis or test performed with limited (grey) or detailed (white) knowledge of the component inner workings. Testing includes such items as verifying a defined output given a defined input or injecting faults into software code to verify fault handling routines.

Black Box Analysis

- Safety analysis starts from the Black Box view assessing the severity for the loss or corruption of the intended function.
- Hazard mitigation should be implemented external to the device producing the hazardous function (if possible) to provide functional and physical independence of the causal and mitigation functions.



Grey/White Box Analysis

- Detailed analysis of component inner workings to substantiate hazard mitigation is very resource intensive (cost & schedule).
- Safety Critical Function Thread Analysis (MIL-HDBK-516) type analyses duplicate design description documents and provide little, if any, objective evidence that the safety criteria has been met.
- The identification of requirements that mitigate specific hazards and are traceable from the aircraft level to the box (hardware) and software requirements specification (SRS) levels provide objective evidence of hazard mitigation.
- Verification of those requirements provides objective evidence that the prescribed level of safety has been achieved.

Software Requirements

- Development Assurance (Systems)
  - Defined in DO-178 for aviation (guidance, not regulatory)
  - Applies only to software (does not attempt to define firmware)
  - Provides five software levels that correlate to hazard severity
  - Evaluates the process – not the product
- Design Assurance (Hardware)
  - Defined in DO-254 for aviation (guidance, not regulatory)
  - Applies to electronic hardware and firmware
  - Does not attempt to define firmware (mirrors DO-178)
  - Provides five design assurance levels that correlate to hazard severity
  - Evaluates the process – not the product
- DAL – generic term used to identify software levels (not used in DO-178/254)
- SwCI – Software Criticality Index, used in MIL-STD-882E for software levels

DO-178 Severity Levels

- Catastrophic Failure Conditions, which would result in multiple fatalities, usually with the loss of the airplane.
- Hazardous Failure Conditions, which would reduce the capability of the airplane or the ability of the flight crew to cope with adverse operating conditions to the extent that there would be: A large reduction in safety margins or functional capabilities; Physical distress or excessive workload such that the flight crew cannot be relied upon to perform their tasks accurately or completely, or Serious or fatal injury to a relatively small number of the occupants other than the flight crew.
- Major Failure Conditions which would reduce the capability of the airplane or the ability of the crew to cope with adverse operating conditions to the extent that there would be, for example, a significant reduction in safety margins or functional capabilities, a significant increase in crew workload or in conditions impairing crew efficiency, or discomfort to the flight crew, or physical distress to passengers or cabin crew, possibly including injuries.
- Minor Failure Conditions which would not significantly reduce airplane safety, and which involve crew actions that are well within their capabilities. Minor Failure Conditions may include, for example, a slight reduction in safety margins or functional capabilities, a slight increase in crew workload, such as routine flight plan changes, or some physical discomfort to passengers or cabin crew.
- No Safety Effect Failure Conditions that would have no effect on safety; for example, Failure Conditions that would not affect the operational capability of the airplane or increase crew workload.

DO-178 Software Levels

- Level A: Software whose anomalous behavior, as shown by the system safety assessment process, would cause or contribute to a failure of system function resulting in a catastrophic failure condition for the aircraft.
- Level B: Software whose anomalous behavior, as shown by the system safety assessment process, would cause or contribute to a failure of system function resulting in a hazardous failure condition for the aircraft.
- Level C: Software whose anomalous behavior, as shown by the system safety assessment process, would cause or contribute to a failure of system function resulting in a major failure condition for the aircraft.
- Level D: Software whose anomalous behavior, as shown by the system safety assessment process, would cause or contribute to a failure of system function resulting in a minor failure condition for the aircraft.
- Level E: Software whose anomalous behavior, as shown by the system safety assessment process, would cause or contribute to a failure of system function with no effect on aircraft operational capability or pilot workload. If a software component is determined to be Level E and this is confirmed by the certification authority, no further guidance contained in this document applies.

DAL Assignment

DAL assignment is made at the level (functional/physical) in the architecture where the effect of the fault/failure is being assessed as follows:

- Catastrophic effect – DAL A
- Hazardous effect – DAL B
- Major effect – DAL C
- Minor effect – DAL D
- No Safety Effect – DAL E

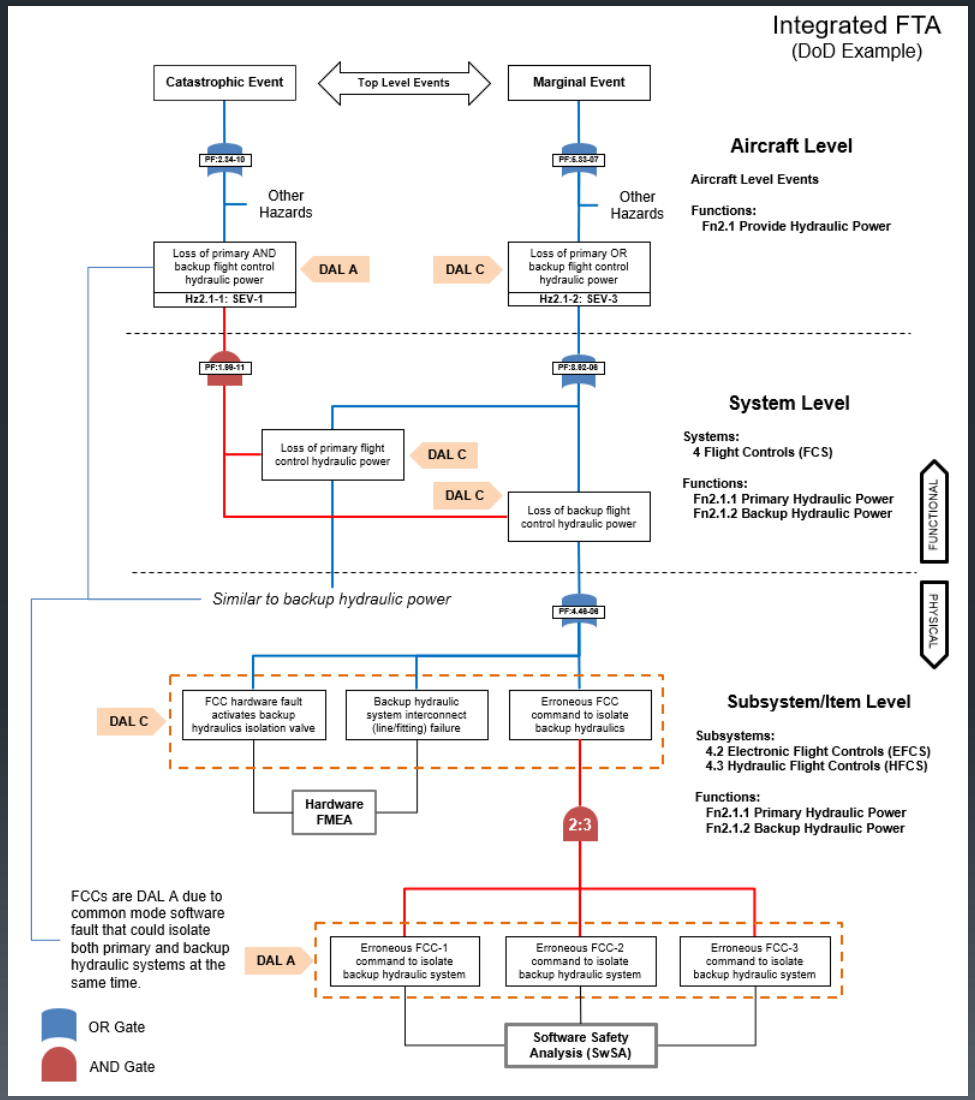
Design decisions have a direct impact on DAL assignment and should be documented in the design description documents.

Design Influence

Top level events

System Decomposition

Detailed Design



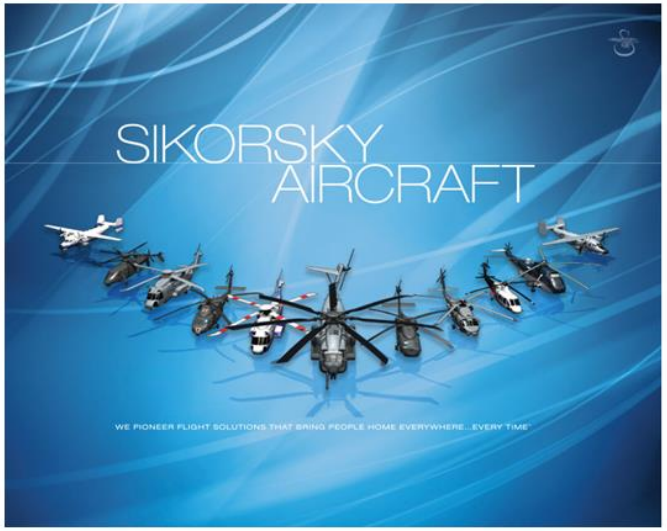
Unified Process

- 1 Introduction
- 2 System Safety Process
- 3 System Safety Analyses
- 4 Requirements Verification
- 5 Software Development

Provides a standard framework that implements the ARP4754A process!

Presented at the 2013 System Safety Conference in Boston

Development Programs Safety Process Guidebook



LOCKHEED MARTIN COMPANY
Rotary and Mission Systems – Sikorsky Aircraft Corporation
Aviation & Product Safety – System Safety Engineering

August 25, 2021

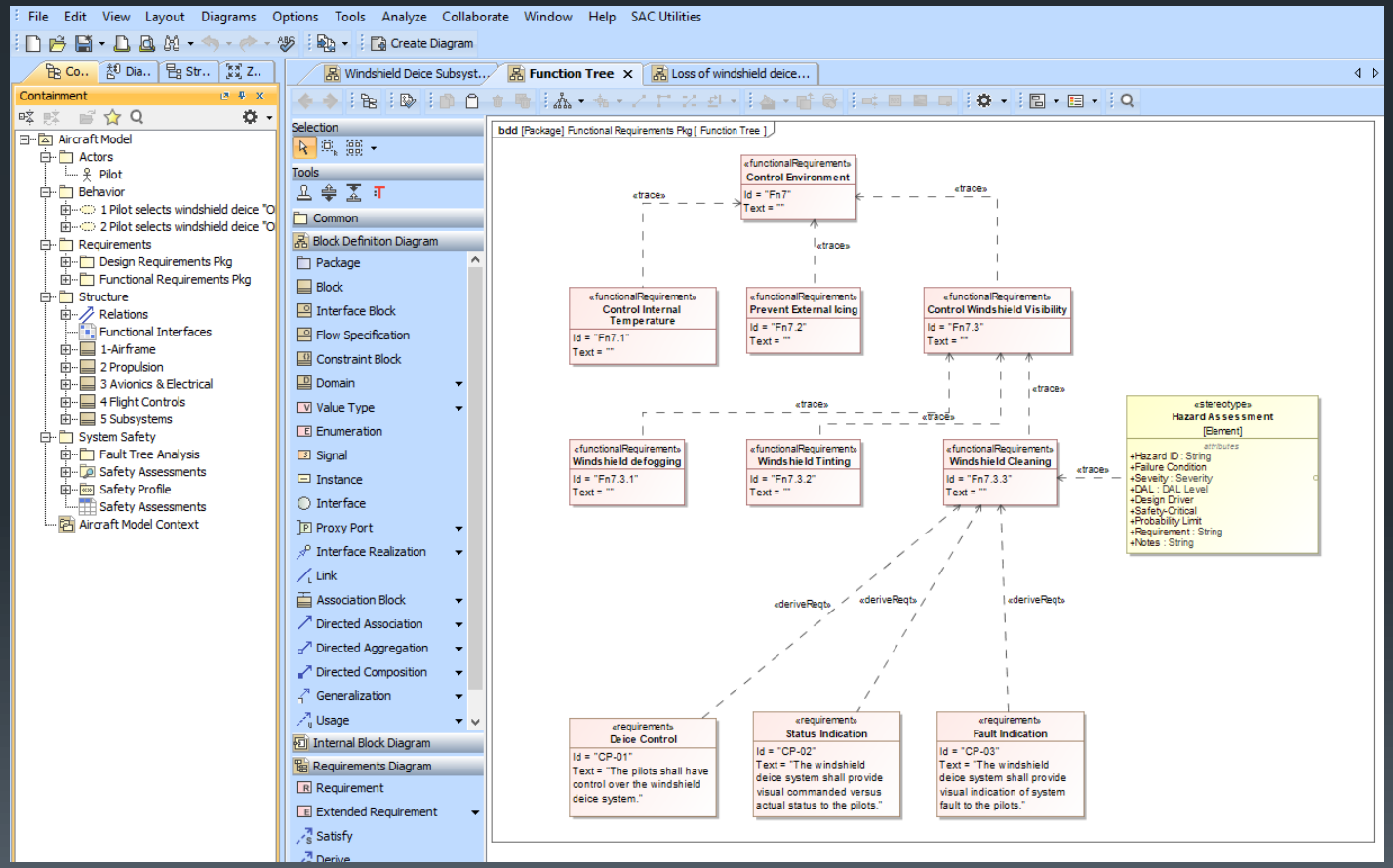
Document No.: APSSM-076-1

Key Points

- Software safety is not a software-centric issue, it is a system safety issue and must be approached as such
- Aircraft and system level hazards are assessed from the functional perspective, not in terms of hardware or software
- Functional hazards include loss of the function as well as corruption of the function (too early, too late, etc.)
- Detailed hazard analysis occurs at the subsystem/item level and assesses hardware/software potential to cause loss of or corruption of a specific function
- The vast majority of software related incidents and accidents have been traced to deficiencies in requirements development and management, not coding errors

Future

Model Based System Engineering (MBSE)



Questions