Weapons Systems Software Safety Criticality and Level of Rigor (LoR)

5 August 2014

Naval Ordnance Safety and Security Activity (NOSSA)
Indian Head, MD

Douglas J. Bower
douglas.j.bower@navy.mil
(301) 744-6069

Opening Video

Class Objectives

- Functional Hazard Analysis (FHA)
  - Understand and Apply a Methodology for Conducting an FHA
- Software Safety Criticality / Level of Rigor
  - Determine the Criticality of Safety Significant Functions
  - Define the Level of Rigor (LOR)
  - Document the application of the LOR

Using an example missile system developed by M&M Missile Company

Functional Hazard Analysis (FHA)

Purpose: The FHA is primarily used to identify and classify the system functions and the safety consequences of functional failure or malfunction, (i.e. hazards (MIL-STD-882E).

- These consequences will be classified in terms of severity for the purpose of identifying the safety-critical functions (SCFs), safety-critical item (SCIs), safety-related functions (SRFs), and safety-related items (SRIs) of the system.
- SCFs, SCIs, SRFs, and SRIs will be allocated or mapped to the system design architecture in terms of hardware, software, and human interfaces to the system.
- The FHA is also used to identify environmental and health related consequences of functional failure or malfunction.
- The initial FHA should be accomplished as early as possible in the Systems Engineering (SE) process to enable the engineer to quickly account for the physical and functional elements of the system for hazard analysis purposes; identify and document SCFs, SCIs, SRFs, and SRIs; allocate and partition SCFs and SRFs in the software design architecture; and identify requirements and constraints to the design team.

FHA Methodology

FHA considers the following to identify and evaluate functions within a system:

- Decomposition of the system and its related subsystems to the major component level.
- Functional description of each subsystem and component identified.
- Functional description of interfaces between subsystems and components. Interfaces should be assessed in terms of connectivity and functional inputs and outputs.
- Hazards associated with loss of function, degraded function or malfunction, or functioning out of time or out of sequence for the subsystems, components, and interfaces. The list of hazards should consider the next effect in a possible mishap sequence and the final mishap outcome.
- An assessment of the risk associated with each identified failure of a function, subsystem, or component.

FHA Methodology (continued)

- An assessment of whether the functions identified are to be implemented in the design hardware, software, or human control interfaces. This assessment should map the functions to their implementing hardware or software components.
- Functions allocated to software should be mapped to the lowest level of technical design or configuration item prior to coding (e.g., implementing modules or use cases).
- An assessment of Software Control Category (SCC) for each Safety-significant Software Function (SSSF). Assign a Software Criticality Index (SwCI) for each SSSF mapped to the software design architecture.
- A list of requirements and constraints (to be included in the specifications) that, when successfully implemented, will eliminate the hazard or reduce the risk. These requirements could be in the form of fault tolerance, detection, isolation, annunciation, or recovery. (i.e. Derived Requirements).

Example Missile System

- Application
  - Examples and practical exercises using Robin Hood Missile System (RHMS)

The RHMS is a software intensive system: all functions related to the pointing and firing of missiles are under the control of software. The RHMS consists of three major subsystems: the Bow Launcher, the Arrow missile, and the Archer Fire Control System (AFCS).

Practical Exercise #1

- Identify five Top-Level Mishaps associated with the Robin Hood Missile System (RHMS)
  - Arrow Missile
  - Bow Launcher
  - Archer Fire Control System

Practical Exercise #1 - Possible Answer

- Identify five Top-Level Mishaps associated with the RHMS (Missile & Launcher)
  1. Inadvertent/Early Ignition of Rocket Motor
  2. Inadvertent Warhead Detonation
  3. Loss of Flight Control
  4. Missile Engages Incorrect Target
  5. Launch Abort/Restrained Firing

  … There are may be others.

Software Safety Criticality

- Degree to which the software has influence on the safety related aspects of a system
  - Level of Control
    - Considers what other interlocks (both hardware and separate independent software) exist in the system
    - The ability of the software to assert the safety critical actions of the system
  - Mishap Severity
    - Determined in the same manner as hardware or system mishap severity
    - Level of personnel injury and/or equipment damage
- Drives the level of rigor of analysis and testing which needs to be applied

Level of Control

SOFTWARE CONTROL CATEGORIES		
Level	**Name**	**Description**
1	**Autonomous (AT)**	• Software functionality that exercises autonomous control authority over potentially safety-significant hardware systems, subsystems, or components without the possibility of predetermined safe detection and intervention by a control entity to preclude the occurrence of a mishap or hazard. *(This definition includes complex system/software functionality with multiple subsystems, interacting parallel processors, multiple interfaces, and safety-critical functions that are time critical.)*
2	**Semi-Autonomous (SAT)**	• Software functionality that exercises control authority over potentially safety-significant hardware systems, subsystems, or components, allowing time for predetermined safe detection and intervention by independent safety mechanisms to mitigate or control the mishap or hazard. *(This definition includes the control of moderately complex system/software functionality, no parallel processing, or few interfaces, but other safety systems/mechanisms can partially mitigate. System and software fault detection and annunciation notifies the control entity of the need for required safety actions.)* • Software item that displays safety-significant information requiring immediate operator entity to execute a predetermined action for mitigation or control over a mishap or hazard. Software exception, failure, fault, or delay will allow, or fail to prevent, mishap occurrence. *(This definition assumes that the safety-critical display information may be time-critical, but the time available does not exceed the time required for adequate control entity response and hazard control.)*
3	**Redundant Fault Tolerant (RFT)**	• Software functionality that issues commands over safety-significant hardware systems, subsystems, or components requiring a control entity to complete the command function. The system detection and functional reaction includes redundant, independent fault tolerant mechanisms for each defined hazardous condition. *(This definition assumes that there is adequate fault detection, annunciation, tolerance, and system recovery to prevent the hazard occurrence if software fails, malfunctions, or degrades. There are redundant sources of safety-significant information, and mitigating functionality can respond within any time-critical period.)* • Software that generates information of a safety-critical nature used to make critical decisions. The system includes several redundant, independent fault tolerant mechanisms for each hazardous condition, detection and display.
4	**Influential**	• Software generates information of a safety-related nature used to make decisions by the operator, but does not require operator action to avoid a mishap.
5	**No Safety Impact (NSI)**	• Software functionality that does not possess command or control authority over safety-significant hardware systems, subsystems, or components and does not provide safety-significant information. Software does not provide safety-significant or time sensitive data or information that requires control entity interaction. Software does not transport or resolve communication of safety-significant or time sensitive data.

MIL-STD-882E: Table V.

Mishap Severity

SEVERITY CATEGORIES		
Description	**Severity Category**	**Mishap Result Criteria**
Catastrophic	**1**	Could result in one or more of the following: death, permanent total disability, irreversible significant environmental impact, or monetary loss equal to or exceeding $10M.
Critical	**2**	Could result in one or more of the following: permanent partial disability,injuries or occupational illness that may result in hospitalization of at least three personnel, reversible significant environmental impact, or monetary loss equal to or exceeding $1M but less than $10M.
Marginal	**3**	Could result in one or more of the following: injury or occupational illness resulting in one or more lost work day(s), reversible moderate environmental impact, or monetary loss equal to or exceeding $100K but less than $1M.
Negligible	**4**	Could result in one or more of the following: injury or occupational illness not resulting in a lost work day, minimal environmental impact, or monetary loss less than $100K.

MIL-STD-882E: Table I.

Software Safety Criticality – Level of Rigor

SOFTWARE SAFETY CRITICALITY MATRIX				
	SEVERITY CATEGORY			
SOFTWARE CONTROL CATEGORY	Catastrophic (1)	Critical (2)	Marginal (3)	Negligible (4)
1	SwCI 1	SwCI 1	SwCI 3	SwCI 4
2	SwCI 1	SwCI 2	SwCI 3	SwCI 4
3	SwCI 2	SwCI 3	SwCI 4	SwCI 4
4	SwCI 3	SwCI 4	SwCI 4	SwCI 4
5	SwCI 5	SwCI 5	SwCI 5	SwCI 5

SwCI	Level of Rigor Tasks
SwCI 1	Program shall perform analysis of requirements, architecture, design, and code; and conduct in-depth safety-specific testing.
SwCI 2	Program shall perform analysis of requirements, architecture, and design; and conduct in-depth safety-specific testing.
SwCI 3	Program shall perform analysis of requirements and architecture; and conduct in-depth safety-specific testing.
SwCI 4	Program shall conduct safety-specific testing.
SwCI 5	Once assessed by safety engineering as Not Safety, then no safety specific analysis or verification is required.

MIL-STD-882E: Table IV.

Practical Exercise #2

- Perform a preliminary Functional Hazard Analysis (FHA) of the Arrow Missile in the context of the Robin Hood Missile System (RHMS)
- Use the information provided in the handouts
- You may work in small groups
- Be prepared to present after the break

Example: Function – Hazard – SCI Relationship

- Function: Boost Phase Autopilot
- Hazard: Software design logic or coding error results in loss of flight control before safe separation
- SCI: Catastrophic / Autonomous – **1**

Example: Function – Hazard – SCI Relationship

- Function: Mid-Course Autopilot
- Hazard: Software design logic or coding error results in loss of flight control after safe separation
- **SwCI: Catastrophic / Autonomous – 1,**
- SwCI: Marginal / Semi-Autonomous – 3, or
- SwCI: No Safety Impact?

Practical Exercise #3

- For these two examples determine the specific types of analysis and testing to be performed

Practical Exercise #3 - Discussion

- For these two examples determine the specific types of analysis and testing to be performed
  - After completing actions in PE #3, how would the results of the application of the LoR be documented

Final Exam / Discussion

Why is important to complete a FHA as early as possible in the development of a system?

Should SwCI and LoR be determined for Hazards, Causal Factors, or Functions?

Is it ever too late to complete an FHA on a system?