DIGITAL ENGINEERING
/ SYSTEMS ENGINEERING

Automated System/Software Safety Analysis Using Model-Based Systems Engineering

Gregory P. Czerniak
Rodolfo Proenza

Objective

- Develop a system/software safety analysis tool that facilitates digital safety engineering for the Army.
- Have the tool provide Authoritative Sources of Truth (ASoT) about safety per the Department of Defense Digital Engineering Fundamentals [1].
- Have the tool trace systems, functions, requirements, mitigations, etc. against existing Authoritative Sources of Truth in other Digital Engineering Systems.
- Assist Agile development efforts by auto-generating documentation.

[1] Department of Defense, “Department of defense (dod) digital engineering fundamentals,” Tech. Rep. [Online]. Available: https://ac.cto.mil/wp-content/uploads/2022/03/ DE-Fundamentals-2022.pdf.

Challenge

- Digital Engineering generally requires formal and machine-understandable definitions to work.

- The Army uses three main standards/documents for safety:
  - MIL-STD-882E
  - Joint Software Systems Safety Engineering Handbook
  - Joint Services – Software Safety Authorities Implementation Guide

- The definitions and processes in these documents are written in plain English, and therefore require more formalization and structure to operate under a digital engineering platform.

Formalizing Hazards

- The fundamental piece of information in system/software safety is the **hazard**, as evidenced by safety data systems frequently being referred to as Hazard Tracking Systems, and most analyses having “Hazard Analysis” in the name.

- MIL-STD-882E’s definition of a hazard is “A real or potential condition that could lead to an unplanned event or series of events (i.e. mishap) resulting in death, injury, occupational illness, damage to or loss of equipment or property, or damage to the environment.”

- This definition is not precise enough to produce the structure required for digital engineering.

Quiz

Which of these are hazards?

- Person is electrocuted
- Big accelerating truck
- Software fails to stop vehicle
- Fuel spill

Quiz Answer

Which of these are hazards? **None of them!**

- Person is electrocuted (Mishap)
- Big accelerating truck (Source of kinetic energy)
- Software fails to stop vehicle (Cause)
- Fuel spill (Mishap)

Special Thanks to Hunter Austegard of HCRQ for the idea of this quiz.

Formalizing Hazards (cont.)

Per Clifton Ericson's "Hazard Analysis Techniques for System Safety," a hazard is defined with three elements:

- **Threat Outcome** (or Mishap): A bad thing you do not want to happen.
- **Hazard Source**: The thing in the system that causes the harm (usually a source of energy such as kinetic, potential, chemical, etc but may also be toxins)
- **Initiating Mechanisms**: A set of events that must occur simultaneously for there to be a hazard.

These elements can usually be ordered to form a complete sentence (see next slide).

Note: Having a hazard occur does not mean that the mishap has occurred. If any of the initiating mechanisms or the hazard source cease to exist, then the hazard is over.

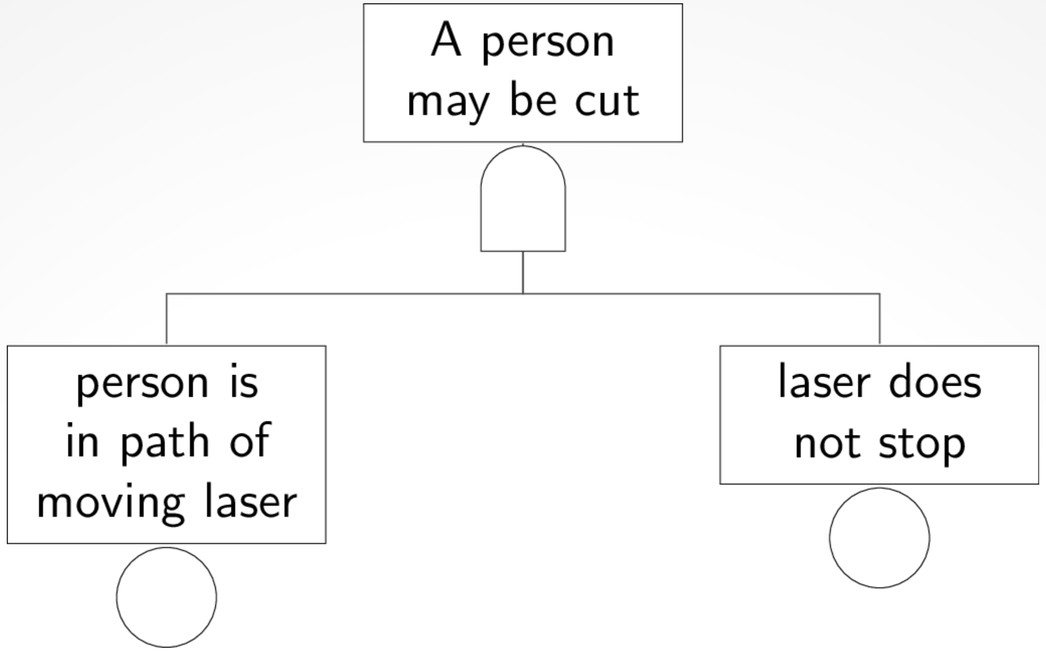
Hazard Example

Personnel may be cut	Threat Outcome (Mishap)
By a Class IV laser product	Hazard Source
Because personnel is in path of moving laser	Initiating Mechanism
And laser does not stop	Initiating Mechanism

Photo from “The Simpsons”

Advantages of Hazard Formalization

- There is now clear structure in the definition.
- The definition naturally leads to the start of a fault tree (see example)



- The analyst can then flesh out this mini fault tree into a complete representation of how the hazard could happen.

Functional Hazard Analysis

- As usual, the analyst must perform a full functional decomposition of the system.
- Once done, the analyst goes through each functionality and analyses how the function can fail and the impact of the failure.
- **However**, in this tool this is done by **associating the failure of a functionality (unavailable, malfunction, out of sequence, too early, too late, etc) with boxes in the previously-made fault trees**.
- This inherently links functions with hazards by a soft data link.

FHA through Cutsets

- The tool then performs “cut set analysis” on all hazard fault trees. This flattens all fault trees into a group of “cut-sets”, in which each cut-set is a set of lowest-level events that must simultaneously occur for the hazard to happen. Example of a cut-set:

Description	Probability
A person is in front of the vehicle	1
Drive-by-wire software malfunctions	SCC1
Software-based emergency stop fails	SCC1

- For each software functionality in the system, the system looks at cut-sets that involve boxes linked to the function’s failure, and it counts how many hardware interlocks, human actions, etc that could prevent the software from contributing to the hazard.
- From this, it produces an approximate Software Control Category for the function.

Advantages of FHA through Cutsets

- Eliminates guesswork and debate about the Software Control Category of software safety-significant functions.
- Reduces deferring to “engineering judgment.”
- Reduces the chance of missing unusual/hard-to-imagine cases during the determination of Software Control Categories.

Mitigation Traceability

- The tool also allows the analyst to list mitigations for hazards.
- Per HCRQ's "Fault Tree Analyses – When to Accept, When to Reject", fault trees shall include/represent potential failures of the mitigations.
- To achieve this, the analyst can also produce soft links between mitigations and boxes within hazard fault trees.
- When this is done, the mitigation is automatically listed in the hazard report's list of mitigations, and the box that gets linked will show an "M" in the lower left corner of the fault tree to indicate that it is a mitigation.

Document Generation

The system auto-generates two main documents, which are generated as LaTeX code:

- **Hazard Tracking System Dump**: Usually embedded in the Safety Assessment Report. Contains:
  - Hazard List
  - Functional Hazard Analysis
  - Fault Tree Analyses
  - Requirements Traceability Matrices
  - HAZMAT List
  - Mitigation Lists
  - High-Level Hazard Risk Counts
- **Deliberate Risk Assessment Worksheet**: Needed for performing range tests.

Projects

- The tool has seen active use in multiple Ground Vehicle Systems Center projects.
- Projects include producing a Safety Assessment Report (SAR) and Deliberate Risk Assessment Worksheet (DRAW) for a test event involving an optionally-manned vehicle and a SAR for the test of an electrified vehicle.
- In both projects, the tool allowed for producing a SAR and DRAW within weeks.
- Both projects were approved for testing by the independent accreditor.

API Access

- Version 2 of this tool provides Application Programming Interface (API) access to the underlying data stored in the program's database.
- With the appropriate "glue code," this should allow for traceability against other tools with APIs, such as SysML tools, requirements trackers, Jira, etc.
- The easiest "bridge" between this tool and others is requirement IDs.
- We have an internal tool for tracing requirements to individual lines of code, and the goal is to eventually achieve bidirectional traceability from lines of code to the hazards involved with those lines.

Thank You!

This concludes the presentation. Any questions?