Safety Analysis Approach

Paul Albertella, Codethink
Chair of OSEP working group

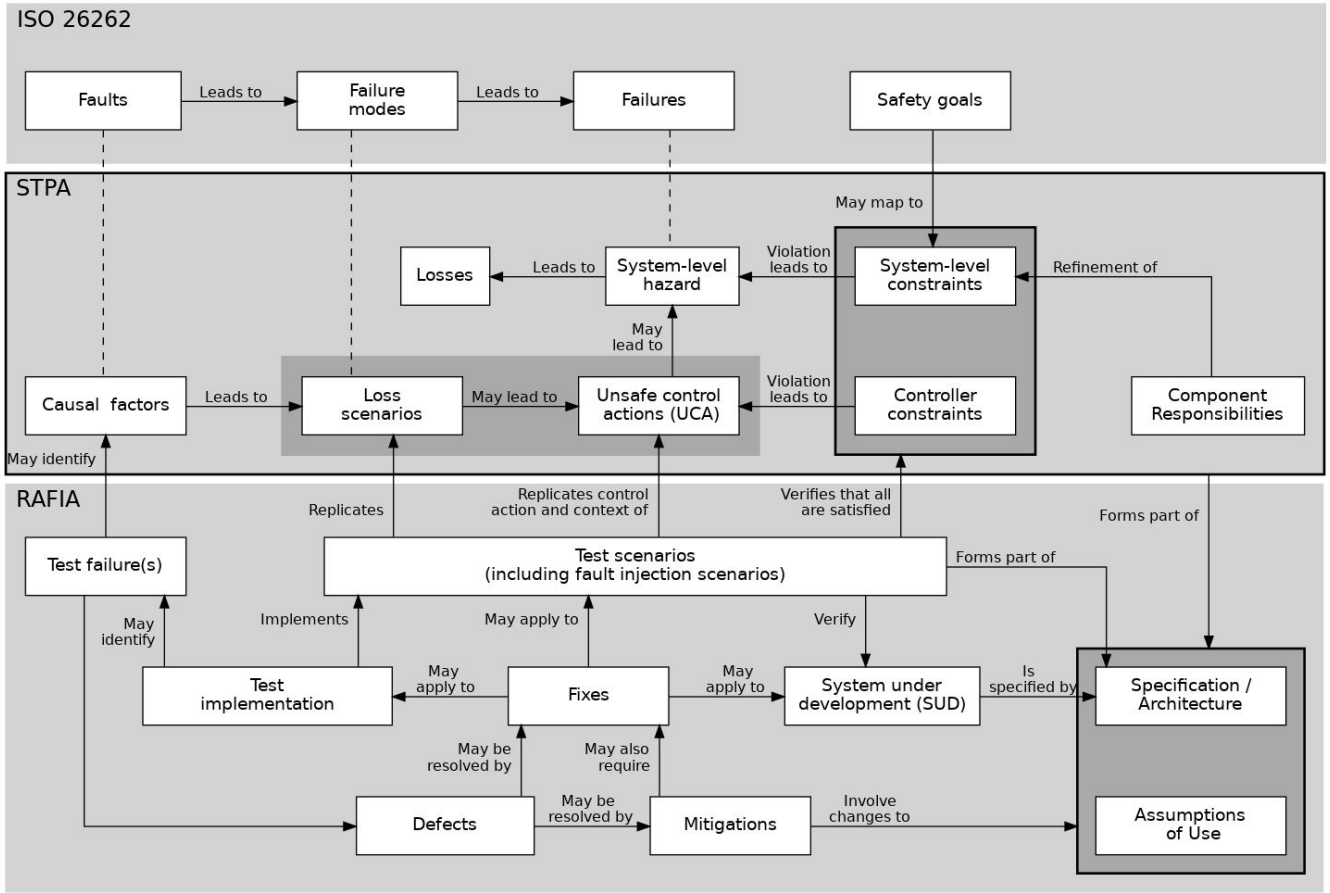
Background

- Safety as a system property
  - Achieving safety goals depends on many components working in concert
  - Focus on component-level properties can mean that problems are missed or discovered late
  - Incompatible assumptions for ‘safe’ components lead to integration problems and recalls
- Open source software does not follow the standards
  - Evidence of good engineering practice, but projects are organised on diverse principles
  - Often value breadth and immediacy of use, rapid iteration and adaptability
  - Formal requirements and designs specifications are frequently absent
- Increasing use of complex software in safety-related systems
  - Makes use of ‘traditional’ methods infeasible
    - Predicated on small, purpose-built and exhaustively-specified software components
  - Requires ongoing maintenance of software and both build and runtime dependencies
  - Demands greater focus on system integration and verification

RAFIA: Risk Analysis, Fault Injection and Automation

- Software engineering process developed by Codethink to address these challenges
  - Introduced and discussed at previous ELISA workshops
    - https://elisaworkshopmay2021.sched.com/event/j3T7
    - https://elisaworkshopspring2022.sched.com/event/za1z
- Risk analysis using STPA to describe, explore and verify system design
  - Requirements defined by STPA *system-level* and *controller constraints*
  - Test cases derived from *unsafe control actions* and *loss scenarios*
- Automated construction and system-level testing to verify requirements
  - Including soak and stress testing to identify unspecified behaviour
- Fault injection to confirm that tests *and* system-level safety mechanisms work
  - Break software to force *loss scenarios* and verify that tests fail and report results as expected
  - Use to extend test scenarios to verify *exception handling* behaviour and *safety mechanisms*
- Iteration and refinement based on the results of these processes

RAFIA



STPA and RAFIA concepts and their relationship to ISO 26262 terms

ELISA WORKSHOPS

ELISA Safety Analysis: Objectives

- Applying the *Risk Analysis* approach of RAFIA
  - Examining example roles of Linux as part of a safety-related system
- Specify a *system context*
  - Concrete system design, or an abstraction representing a class of system designs
  - May have both safety and non-safety functions
- Specify *safety goals* for the system and context
  - System-level criteria that must be satisfied to avoid specific negative outcomes
  - May relate to more than one safety function
- Specify *safety responsibilities* for components of the system
  - Behaviour or properties required to avoid violating the safety goals for the system context
  - Focus is on responsibilities assigned to Linux, but specifying our assumptions about the assumed responsibilities of other components is just as important

ELISA Safety Analysis: Motivations

- Establish and document a common process
  - For use across ELISA and by other FOSS projects
  - Provide a *lingua franca* for recording safety analysis to aid component integration
- Example analyses as building blocks
  - Common system context definitions and approach
  - Accumulate analysis focusing on different aspects of Linux
  - Build on and refine earlier analysis
- Basis for cooperation between working groups
  - LFSCS - investigation of Linux features to support analysis
  - Medical Devices - analysis of OpenAPS using STPA
  - Safety Architecture - STPA-like analysis of kernel
  - Automotive - Telltale use case
  - Systems - define a common reference system context for analyses?
  - Aerospace - define a use case as a future subject of analysis?
  - Tools - identify supporting tools?

OSEP: Initial work

- Call for input and participation on ELISA blog
  - https://elisa.tech/blog/2023/05/17/elisa-project-safety-analysis-approach-using-stpa/
- Plan to document process and examples in OSEP GitHub repository
  - Structured and peer-reviewed documentation in the main repository
  - Informal journal of discussions in parallel in Wiki
    - https://github.com/elisa-tech/wg-osep/wiki/Telltale-Safety-Analysis
- Starting to apply the approach to Automotive WG use case
  - https://github.com/elisa-tech/wg-osep/pull/18
  - Would welcome input and participation from the Automotive WG!

Discussion and next steps

Licensing of Workshop Results

All work created during the workshop is licensed under *Creative Commons Attribution 4.0 International (CC-BY-4.0)* [https://creativecommons.org/licenses/by/4.0/] by default, or under another suitable open-source license, e.g., GPL-2.0 for kernel code contributions.

You are free to:

- **Share** — copy and redistribute the material in any medium or format
- **Adapt** — remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

ELISA WORKSHOPS