

Software Safety Tutorial (Status Update)

Jeff Tian, tian@engr.smu.edu
CSE, SMU, Dallas, TX 75275

Topics

- Project Overview
- Software Safety Overview
- Project Tasks/Schedule/Progress

What Is Software Safety?

- *Software safety*: The property of being accident-free for (embedded) software systems.
 - ▷ Accident: failures with severe consequences
 - ▷ Hazard: condition for accident
 - ▷ Specialized techniques

- Software safety engineering (SSE):
 - ▷ Goal: to ensure software safety via
 - ▷ hazard identification/analysis techniques
 - ▷ hazard resolution alternatives
 - hazard elimination/reduction/control
 - (tracking/mitigation/control – NASA)
 - ▷ safety and risk assessment
 - ▷ safety and process improvement

- Qualitative focus, systematic approach

Project Overview

- Sponsor: Dennis Frailey (David Struble), Raytheon.

- Motivation:
 - ▷ DoD commitment to safety (personnel/system/property/environment)
 - ▷ DoD goal: 0 mishaps (accidents above)
 - ▷ DoD guidance: MIL-STD-882D (2000)

- Goal: Software safety should become a core competency for real-time software engineers.

- Project team:
 - ▷ Jeff Tian (SMU): Basics of SSE
 - ▷ D.T. Huynh and Eric Wong (UTD): related research and extensions

Overall Approach

- Basics about software safety (Tian):
 - ▷ Basic definitions and concepts
 - ▷ Hazard identification/analysis techniques, primarily fault-/event-tree analyses
 - ▷ Design for safety via hazard elimination/reduction/control
 - ▷ Leveson's SSP (software safety program) and STAMP (sys. theoretic accident modeling and processes)
 - ▷ Formal verification of safety
 - ▷ New applications and development

- Tailoring to meet project sponsor goals:
 - ▷ Include DoD MIL-STD-882D (Tian)
 - ▷ Testing for safety (Wong)
 - ▷ Formal methods for safety (Huynh)

Comparison: Lutz/NASA

- Software safety: 7 directions:
 - 1 integration of informal/formal methods
 - 2 safe reuse
 - 3 testing/evaluation of SCS
 - 4 runtime monitoring
 - 5 education
 - 6 certification and standards
 - 7 collaboration with related fields

- Coverage in our SST:
 - ▷ 1/3/5: existing research/course
 - ▷ 6: focus on DoD MIL-STD-882D
 - ▷ 2/4/7: existing expertise

Basic Definitions

- Accident or mishap:
 - ▷ unplanned (series of) events
 - ▷ leading to unacceptable loss
 - death, injury, illness
 - equip./property/environment damage
 - ▷ excess energy/dangerous substance
 - ▷ computers relatively safe
 - ▷ but computer control \Rightarrow accidents

- Hazard:
 - ▷ a set of conditions leading to accidents under certain environmental conditions
 - ▷ e.g.: guard gates at rail-crossing
 - ▷ safety focus: control factors (vs. env. factors beyond control)
 - ▷ analysis and resolution \Rightarrow SSE

Basic Definitions

- Risk: function of 3 elements
 - ▷ likelihood(hazard)
 - ▷ likelihood(hazard \Rightarrow accident)
 - ▷ worst possible loss due to accident
(compare to expected loss)

- (System) safety engineering:
 - ▷ ensuring acceptable risk
 - ▷ scientific/management/engineering
 - ▷ reducing risk factors
 - ▷ context for software safety
 - ▷ hazard identification, assessment,
analysis, and resolution

Safety Analysis & Resolution

- Hazard analysis:
 - ▷ Fault trees: (static) logical conditions
 - ▷ Event trees: dynamic sequences
 - ▷ Combined and other analyses
 - ▷ Generally qualitative
 - ▷ Related: accident analysis and risk assessment

- Hazard resolution (pre-accident)
 - ▷ Negate/block/mitigate/etc.
 - ▷ Hazard elimination/reduction/control

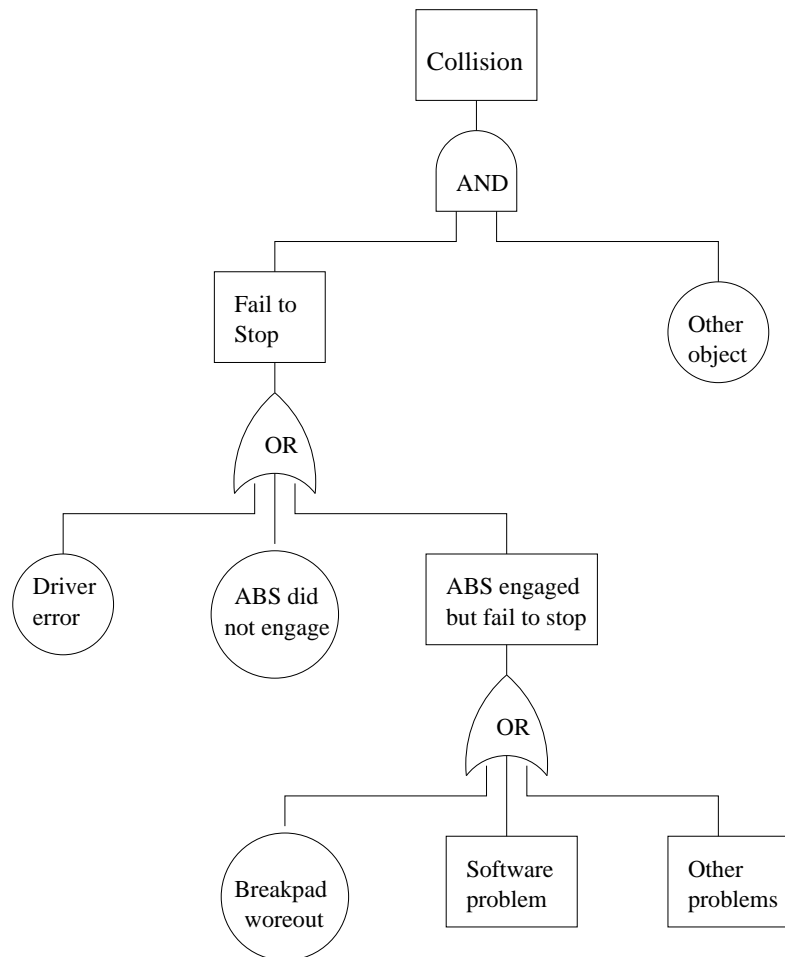
- Damage reduction (post-accident)

Hazard Analysis: FTA

- Fault tree idea:
 - ▷ Top event (accident)
 - ▷ Intermediate events/conditions
 - ▷ Basic or primary events/conditions
 - ▷ Logical connections
 - ▷ Form a tree structure

- Elements of a fault tree:
 - ▷ Nodes: conditions and sub-conditions
 - terminal vs. no terminal
 - ▷ Logical relations among sub-conditions
 - AND, OR, NOT
 - ▷ Other types/extensions possible

Hazard Analysis: FTA Example



- Example FTA for an automobile accident

Hazard Analysis: FTA

- FTA construction:
 - ▷ Starts with top event/accident
 - ▷ Decomposition of events or conditions
 - ▷ Stop when further development not required or not possible (atomic)
 - ▷ Focus on controllable events/elements

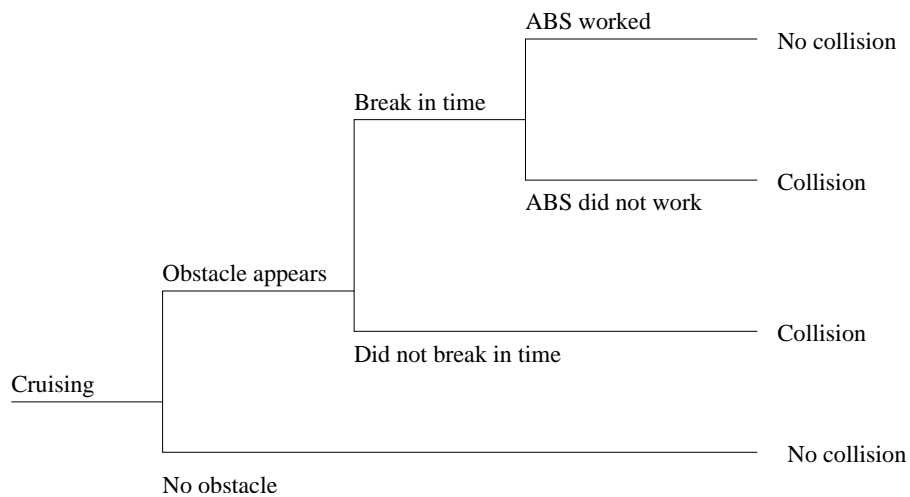
- Using FTA:
 - ▷ Hazard identification
 - *logical* composition
 - (vs. *temporal* composition in ETA)
 - ▷ Hazard resolution (more later)
 - component replacement etc.
 - focused safety verification
 - negate logical relation

Hazard Analysis: ETA

- ETA: Why?
 - ▷ FTA: focus on static analysis
 - (static) logical conditions
 - ▷ Dynamic aspect of accidents
 - ▷ Timing and temporal relations
 - ▷ Real-time control systems

- Search space/strategy concerns:
 - ▷ Contrast ETA with FTA:
 - FTA: backward search
 - ETA: forward search
 - ▷ May yield different path/info.
 - ▷ ETA provide additional info.

Hazard Analysis: ETA Example



- Example ETA for an automobile accident
- Compare/contrast with FTA a few slides back.

Hazard Analysis: ETA

- Event trees:
 - ▷ Temporal/cause-effect diagram
 - ▷ (Primary) event and consequences
 - ▷ Stages and (simple) propagation
 - not exact time interval
 - logical stages and decisions

- Event tree analysis (ETA):
 - ▷ Recreate accident sequence/scenario
 - ▷ Critical path analysis
 - ▷ Used in hazard resolution (more later)
 - esp. in hazard reduction/control
 - e.g. creating barriers
 - isolation and containment

Design for Safety

- **Eliminate** identified hazard sources in material/component/software/etc.

- **Reduce** hazard likelihood/severity via:
 - ▷ Creating hazard barriers,
 - ▷ Minimizing failure probability, etc.

- **Control** hazard (after detection) via:
 - ▷ Isolation and containment,
 - ▷ Fail-safe design, etc.

- **Reduce** damage (post-accident, as compared to pre-accident for the above)

Hazard Elimination

- Hazard sources identification \Rightarrow elimination
(Some specific faults prevented or removed.)

- Traditional QA (but with hazard focus):
 - ▷ Fault prevention activities:
 - education/process/technology/etc
 - formal specification & verification
 - ▷ Fault removal activities:
 - rigorous testing/inspection/analyses

- “Safe” design: More specialized techniques:
 - ▷ Substitution, simplification, decoupling.
 - ▷ Human error elimination.
 - ▷ Hazardous material/conditions↓.

Hazard Reduction

- Hazard identification \Rightarrow reduction
(Some specific system failures prevented or tolerated.)

- Traditional QA (but with hazard focus):
 - ▷ Fault tolerance
 - ▷ Other redundancy

- “Safe” design: More specialized techniques:
 - ▷ Creating hazard barriers
 - ▷ Safety margins and safety constraints
 - ▷ Locking devices
 - ▷ Reducing hazard likelihood
 - ▷ Minimizing failure probability
 - ▷ Mostly “passive” or “reactive”

Hazard Control

- Hazard detection \Rightarrow control
 - ▷ Key: failure severity reduction.
 - ▷ Post-failure actions.
 - ▷ Failure-accident link weakened.
 - ▷ Traditional QA: not much, but good design principles may help.

- “Safe” design: More specialized techniques:
 - ▷ Isolation and containment
 - ▷ Fail-safe design & hazard scope↓
 - ▷ Protection system
 - ▷ More “active” than “passive”
 - ▷ Similar techniques to hazard reduction,
 - but focus on post-failure severity↓
 - vs. pre-failure hazard likelihood↓.

Accident Analysis & Damage Control

- Accident analysis:
 - ▷ Accident scenario recreation/analysis
 - possible accidents and damage areas
 - ▷ Generally simpler than hazard analysis
 - ▷ Based on good domain knowledge (not much software specifics involved)

- Damage reduction or damage control
 - ▷ Post-accident vs. pre-accident hazard resolution
 - ▷ Accident severity reduced
 - ▷ Escape route
 - ▷ Safe abandonment of material/product/etc.
 - ▷ Device for limiting damages

Software Safety Program (SSP)

- Leveson's approach (Leveson, 1995)
 - Software safety program (SSP)

- Process and technology integration
 - ▷ Limited goals
 - ▷ Formal verification/inspection based
 - ▷ But restricted to safety risks
 - ▷ Based on hazard analyses results
 - ▷ Safety analysis and hazard resolution
 - ▷ Safety verification:
 - few things carried over

- In overall development process:
 - ▷ Safety as part of the requirement
 - ▷ Safety constraints at different levels/phases
 - ▷ Verification/refinement activities
 - ▷ Distribution over the whole process

Tasks/Schedule/Progress

- Major tasks:
 - 1 state-of-the-art survey
 - 2 literature/research survey
 - 3 tutorial
 - 4 annotated bibliography

- Schedule (6 months in summary table):
 - ▷ 3 months: interim review
 - ▷ 6/9/12 months: draft/semi-/final tutorial

- Progress to date:
 - ▷ Basis/extensions for all tasks identified
 - ▷ Personnel/responsibility specified
 - ▷ draft in 3 months