Safety Performance Indicators and Continuous Improvement Feedback

Prof. Philip Koopman

SEAMS 2022 KEYNOTE

Carnegie Mellon University

SEAMS 2022
17th International Symposium on Software Engineering for Adaptive and Self-Managing Systems

Overview

- Lifecycle approach to Autonomous Vehicle safety
  - Historically we assume perfectly safe production release
  - Need move to lifecycle adaptation model
    - Operational metrics used as basis for continuous improvement

- Safety Performance Indicators (SPIs)
  - Beyond “vehicle is acting unsafely”
  - Beyond dynamic risk management
  - Beyond run-time safety monitors
  - ANSI/UL 4600 SPIs monitor safety case soundness
Big Changes In Safety Engineering for AVs

Conventional software safety engineering
- Do hazard and risk analysis (e.g., ISO 26262)
- Mitigate hazards; achieve acceptable risk
- Assume “perfect” for safety when deployed
  - Human driver intervention to clean up loose ends

Autonomous system safety is about change
- Machine learning-based validation is immature
- Open, imperfectly understood environment
  - Unknown unknowns, gaps in requirements, etc.
  - Keep up with a constantly evolving real world
- System monitoring ➔ safety/security updates

Source: Tartan Rescue’s CHIMP in 2015

https://goo.gl/dBdSDM
Safety Engineering: Hazards & Risks

- Hazard and Risk Analysis for conventional systems
  - List all applicable hazards
  - Characterize the resultant risk
  - Mitigate risk as needed, e.g., update design
  - Iterate until all risks acceptably mitigated

- Use various techniques to create hazard list
  - Lessons learned from previous projects; industry standards
  - Brainstorming & analysis techniques
    - FMEA, Fault Trees, HAZOP, .... bring your own favorite approach ...

- Presumption all hazards covered before deployment
  - Fully characterized operating environment
Hazard Analysis for Novel, Open World Systems

- Operating in the open world
  - All hazards aren’t known at first
  - Test, test, test until you have uncovered enough hazards

- Safety Of The Intended Function (SOTIF)
  - Operate in the real world
  - Unknowns manifest “triggering events” (ISO 21448 terminology)
  - Mitigate newly discovered hazards caused by triggering events
  - Repeat until you stop seeing triggering events

- Limitation: residual unknown unknowns (requirements gaps)
  - Hypothesize you can find enough of the unknowns
Driver Assistance Feedback Model

- Driver does dynamic risk mitigation
- Useful fiction: systems safe forever when released
  - Driver expected to help mitigate risks & surprises
  - Recalls for defects drivers can’t handle – not supposed to happen

![Diagram showing the feedback model of design, testing, and deployment with hazard analysis, SOTIF triggering events, driver experience, and recalls.](image-url)
Reaction To Incidents and Loss Events

Conventional systems (in practice) too often:
- Ignore if not reproducible
- Blame it on the operator
- Educate operators on workarounds
- Try again to blame it on the operator
- VERY reluctantly do a software update

This persists across domains:
- Power imbalance between victims and system designers
- Normalization of #MoralCrumpleZone strategies [https://bit.ly/3qX2D92]
- Poor adoption of software engineering practices
- The fact that the feedback loop is called a “recall”
How Is The Recall Approach Working Out?

Small sampling of NHTSA recalls (confirmed defects)

- 22V-169 and many others: Backup camera & display failures
- 21V-972: Parking lock system error leads to vans rolling away when parked
- **21V-873 and MANY others: Airbags disabled**
- 21V-846: Phantom braking due to inconsistent software state after power up
- 21V-109: Battery controller reset disconnects electric drive motor power
- 20V-748: Improper fail-safe logic degrades brake performance
- 20V-771: Malfunctions of wipers, windows, lights, etc. due to comms failure
- **20V-557 and others: Airbags deploy too forcefully or when they should not**
- 17V-713: Engine does not reduce power due to ESP software defect
- 15V-569: Unexpected steering motion causes loss of control
- 15V-145: Unattended vehicle starts engine → carbon monoxide poisoning

See: https://betterembsw.blogspot.com/p/potentially-deadly-automotive-software.html
Machine Learning (ML) only learns things it has seen
- Learns by example
- Can be brittle; generalization is limited
- Spectacular failures for the unexpected

ML complicates safety engineering
- Safety engineering assumes “V” model
- Prone to brittleness to unexpected data variations
- Were there biases or gaps in training data?
- Assurance for rare objects and events in the real world?
  - Safety tends to be limited by rare, high-consequence events
Incomplete Open World Requirements

- Unusual road obstacles & conditions
- Strange behaviors
- Subtle clues

https://dailym.ai/2K7kNS8
https://en.wikipedia.org/wiki/Magic_Roundabout_(Swindon)
https://goo.gl/J3SSyu
The Real World: Heavy Tail Distribution

Common Things Seen In Testing

Edge Cases Not Seen In Testing

Random Independent Arrival Rate (exponential)

Power Law Arrival Rate (80/20 rule) (Heavy Tail Distribution)

Many Different, Infrequent Scenarios Total Area is the same!
Why The Heavy Tail Matters

- Where will you be after 1 Billion miles of testing?
  - At 100M miles per fatality, need perhaps 1 billion miles
- Assume 1 Million miles between unsafe “surprises”
  - Example #1:
    100 “surprises” @ 100M miles / surprise
  - Example #2:
    100,000 “surprises” @ 100B miles / surprise
    - Only 1% of surprises seen during 1B mile testing
    - SOTIF fixes of triggering events don’t really help
- “Perfect when deployed” no longer a useful fiction
  - We’re going to need feedback measurements from deployment

https://goo.gl/3dzguf
Key Performance Indicator (KPI) approach is typical:
- Deviation from intended vehicle path
- Ride smoothness
- Hard braking incidents
- Disengagements during testing
- Coverage of defined scenario catalog
- Risk metrics such as Time to Collision

But how do we predict operational safety?
- Are KPIs good leading metrics for loss events?
- Does a particular KPI set cover all aspects of safety?
- How can we select KPIs for traceability to safety?
Safety Performance Indicator (SPI)

SPI (per ANSI/UL 4600):
- Measurement used to measure or predict safety

Lagging SPI metrics (how it turned out):
- Arrival rate of adverse events compared to a risk budget
  - Example: Loss events (crashes) per hour
- Incidents (could have been a loss event)
  - Example: Running a red light, wrong lane direction

Also need leading metrics to predict safety
- We can do that by linking to a safety case
Safety Cases for Autonomous Vehicles

- Claim – a property of the system
  - “System avoids hitting pedestrians”
- Argument – why this is true
  - “Detect & maneuver to avoid”
- Evidence – supports argument
  - Tests, analysis, simulations, ...
- Sub-claims/arguments address complexity
  - “Detects pedestrians” // evidence
  - “Maneuvers around detected pedestrians” // evidence
  - “Stops if can’t maneuver” // evidence
SPIs Instrument a Safety Case

- SPIs monitor the validity of safety case claims

```
LAGGING METRICS

Vehicle is Safe

SPI

CLAIMS-ONLY VIEW OF SAFETY CASE

Avoids Crashes

SPI

DETECTS OBJECTS

SPI

LEADING METRICS

Sensors Effective

SPI

Data Fusion Effective

SPI

Sensor Cleaning

SPI

SW Quality

SPI

Test Coverage

SPI
```
Example SPIs

■ System Level SPIs:
  ● Road test incidents caught by safety driver in testing
  ● Simulator (SIL/HIL) incidents

■ Subsystem SPIs:
  ● Vehicle Controls: compromised vehicle stability
  ● Path Planning: insufficient clearance to object
  ● Perception: false negative (non-detection)
  ● Prediction: unexpected object behavior

■ Lifecycle SPIs:
  ● Maintenance errors
  ● Invalid configuration installed
An SPI is a metric supported by evidence that uses a threshold comparison to condition a safety case claim.

- Metric: measurement of performance, design quality, process quality, operational procedure conformance, etc.
- Threshold: acceptance test on metric value
  - Often statistical (e.g., fewer than X events per billion miles)
- Evidence: data used to compute the metric
- Condition a claim: threshold violation falsifies a specific claim
  - Argument for claim is (potentially) proven false by SPI
- Anything that does not meet all criteria is a KPI, not an SPI

SPI violation: part of a safety case has been falsified
SPIs and Lifecycle Feedback

- SPI: direct measurement of claim failure
  - Independent of reasoning ("claim is X ... yet here is ~X")
  - Partial measurement(s) OK; multiple SPIs for a claim OK

- A falsified safety case claim:
  - Not (necessarily) imminent loss event
  - Safety case has some defect

- Root cause analysis might reveal:
  - Product or process defect
  - Invalid safety argument
  - Issue with supporting evidence
  - Assumption error, ...

```
CLAIM

ARGUMENT 1
    Sub-CLAIM 1A
        Sub-ARGUMENT 1A
            EVIDENCE 1A

    Sub-CLAIM 1B

SPI: {Metric, Threshold}

Is Claim False?
```
SPI-Based Feedback Approach

- Safety Case argues acceptable risk
  - SPIs monitor validity of safety case
SPIs Go Beyond Overt Dangerous Behavior

- “Acts dangerously” is only one dimension of SPIs
  - Violation rate of pedestrian buffer zones
  - Time spent closer than safe following distance

- Components meet safety related requirements
  - False negative/positive detection rates
  - Correlated multi-sensor failure rates

- Design & Lifecycle considerations
  - Design process quality defect rates
  - Maintenance & inspection defect rates

- Is it relevant to safety? ➔ Safety Case ➔ SPIs
Quality vs. Runtime Monitor vs. SPI

- Functionality (KPIs):
  - Are all the features implemented?
  - Does each feature work as intended?
  - Is testing progress on track per schedule?

- Runtime safety monitors:
  - Triggers risk reduction during run time

- Safety Feedback (SPIs):
  - Did runtime safety monitor miss something?
  - Are there dangerous gaps in the Operational Design Domain?
  - Are there problems with requirements, design, upkeep, etc.?
  - Are there dangerous gaps in fault responses?

Following Distance Example

Responsibility-Sensitive Safety (RSS) Scenario:

- KPI: is average following distance appropriate for driving conditions
- Runtime monitor: force an increase of following distance if too close
- SPIs: situation more dangerous than expected (e.g., ODD issues)
  - Spent more time in too-dense traffic than expected
  - Lead/own vehicle brake violate expectations (too often; too aggressive)
  - Spent too long to recover from lead vehicle cut-in
AV is safe enough to deploy because:

- We’ve followed industry safety standards
  - ISO 26262, ISO 21448, ANSI/UL 4600, ...
  - Safety culture is robust
- Known hazards have been mitigated
  - Residual risk is acceptable at system level
- Arrival rate of unknowns is low
  - Incidents which do not trigger runtime safing
- Safety case has good SPI coverage
  - SPIs usually detect unknowns without an actual crash
  - System is fixed to mitigate unknowns before likely reoccurrence

[Sketch of an AV Safety Argument](https://shutr.bz/3LyTr2H)
Removing human drivers makes safety much harder
- Tactical: run-time safety monitoring in vehicle
- Strategic: SPI monitoring across fleet
- Field feedback as lifecycle adaptation

SPIs predict and monitor system safety
- KPIs: “how well do we drive?”
- SPIs: “how often are safety claims falsified?”
- SPIs can detect safety problems with no crash

SPIs: are you as safe as you think you are?
- See ANSI/UL 4600 Chapter 16 for SPI guidance
- Field feedback via SPIs provides lifecycle safety adaptation