Some of the ideas described in this talk emerged as a result of projects funded by the US National Science Foundation under Grants CCF-0959924 and CCF-1265178.
Safety Critical Software

Software whose failure or malfunction may result in death or serious injury to people or in loss or severe damage to equipment or property.

Such systems cannot be fragile!
Certification standards prescribe the use of traceability to demonstrate compliance to regulatory codes and to show that all hazards have been mitigated.
The Traceability Gap

Based on over a decade of traceability engagements in industrial projects we have observed a traceability gap between what is prescribed and what is delivered:

Our study of the traceability components of Medical Device submissions to the FDA showed incomplete and sometimes entirely missing trace links, inaccurate, redundant traceability – delivered through a big bang solution.

A formal comparison of five safety-critical software systems against prescribed guidelines showed similar traceability problems.
The Traceability Gap

Mind the Gap: Assessing the Conformance of Software Traceability to Relevant Guidelines, Patrick Rempel, Patrick Mäder, Tobias Kuschke (TU Ilmenau), and Jane Cleland-Huang, ICSE 2014, Hyderabad, India
Injecting Safety into Agility

In this talk I’m going to share with you my own experiences and how I’ve woven safety critical practices into agile.

1. Preliminary Hazard Analysis
2. Continuous safety assessment
3. Continuous traceability analysis
4. Continuous Safety Assurance
**Safety Effort: Fit for purpose**

<table>
<thead>
<tr>
<th>DO-178b Criticality Levels:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Catastrophic</strong></td>
</tr>
<tr>
<td>Failure may cause a crash. Error or loss of critical function required to safely fly and land aircraft.</td>
</tr>
<tr>
<td><strong>Hazardous</strong></td>
</tr>
<tr>
<td>Failure has a large negative impact on safety or performance, or reduces the ability of the crew to operate the aircraft, or causes serious or fatal injuries among the passengers.</td>
</tr>
<tr>
<td><strong>Major</strong></td>
</tr>
<tr>
<td>Failure is significant, but has a lesser (for example, leads to passenger discomfort rather than injuries)</td>
</tr>
<tr>
<td><strong>Minor</strong></td>
</tr>
<tr>
<td>Failure is noticeable</td>
</tr>
<tr>
<td><strong>No Effect</strong></td>
</tr>
<tr>
<td>Failure has no impact on safety</td>
</tr>
</tbody>
</table>

How hazardous is my software system?
Safety Critical?

<table>
<thead>
<tr>
<th>Drones</th>
<th>E-Health</th>
<th>Environment</th>
<th>Mission-Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fleets of drones are coordinated to deliver medical supplies in a natural catastrophe and to use aerial reconnaissance for tracking.</td>
<td>A person recovering from a medical issue such as a heart attack, is safely monitored during his/her rehabilitation.</td>
<td>Crowd-sourced pollution detection in case of a chemical spill. (Also for environmental pollution).</td>
<td>Soldiers (or rescue team) on a mission. Monitor position &amp; health of team. Provide instructions and information via visor.</td>
</tr>
</tbody>
</table>

Examples of safety-critical systems that everyday developers might engage in...
Amazon’s vision for package delivery... **We will focus on the area of low-speed localized traffic**

https://www.wired.com/2015/08/really-want-amazons-drones-swarm-skies/
Integrating Safety into the Agile Process

Given a standard SCRUM or Kanban Process, where do safety activities fit?

Before answering this question – let’s look at several safety aspects:
- FMECAs
- Safety Stories
- Traceability
- Safety Cases

1. Inputs from Stakeholders, Customers, Users, Team
2. Product Owner
3. Ranked list of what is required – Features, User stories, Safety-Stories, EARS
4. Team selects user stories for the sprint
5. Sprint
6. Working Product
7. Scrum meeting
FMECA: Data Faults

Preliminary Hazard: Drone’s battery fails unexpectedly and the drone crashes.

1. The battery level detector fails to detect a low battery level.
2. Battery level indicator fails and an incorrect battery level is reported.
3. The software fails to check battery level in time to take responsive actions before the battery fails.
## FMECA: Data Faults

<table>
<thead>
<tr>
<th>ID</th>
<th>Data Item</th>
<th>Data Fault Type</th>
<th>Description</th>
<th>Effect</th>
<th>Criticality</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM-D2</td>
<td>Battery level</td>
<td>Faulty error</td>
<td>Low battery level is not detected.</td>
<td>EF-4 Drone runs out of power and lands in</td>
<td>Critical</td>
</tr>
<tr>
<td></td>
<td>indicator</td>
<td>detection</td>
<td></td>
<td>an uncontrolled way.</td>
<td></td>
</tr>
<tr>
<td>FM-D3</td>
<td>Battery level</td>
<td>Faulty data</td>
<td>Battery level indicator depicts incorrect power availability.</td>
<td>EF-5 Drone runs out of power and lands in</td>
<td>Critical</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>an uncontrolled way.</td>
<td></td>
</tr>
<tr>
<td>FM-D4</td>
<td>Drone health</td>
<td>Missing data</td>
<td>Drone fails to communicate its location</td>
<td>EF-6 Mission control can not accurately</td>
<td>Critical</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>track the drone, potentially causing</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>accidents such as drone crashes.</td>
<td></td>
</tr>
<tr>
<td>FM-D5</td>
<td>Altitude level</td>
<td>Faulty error</td>
<td>Altitude reading is lower than the actual altitude of the drone.</td>
<td>EF-7 Drone flies too high potentially</td>
<td>Critical</td>
</tr>
<tr>
<td></td>
<td></td>
<td>detection</td>
<td></td>
<td>entering the flight path of an airplane.</td>
<td></td>
</tr>
<tr>
<td>FM-D1</td>
<td>Landed status</td>
<td>Faulty status</td>
<td>On ground status = true even though drone is still in the air.</td>
<td>EF-8 Propellers stop prematurely and</td>
<td>Critical</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>crashes</td>
<td></td>
</tr>
</tbody>
</table>
# FMECA: Event Faults

<table>
<thead>
<tr>
<th>Event</th>
<th>Event Type</th>
<th>Event Fault Type</th>
<th>Description</th>
<th>Effect</th>
<th>Criticality</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM-E1</td>
<td>Route miscalculated</td>
<td>Faulty Algorithm</td>
<td>Drone's route is miscalculated</td>
<td>EF-8 Drone flies into prohibited airspace</td>
<td>Catastrophic</td>
</tr>
<tr>
<td>FM-E2</td>
<td>Minimum separation distance violated</td>
<td>Faulty Algorithm</td>
<td>Minimum separation distance is not preserved between drones during flight.</td>
<td>EF-9 Drones crash in midflight</td>
<td>Catastrophic</td>
</tr>
<tr>
<td>FM-E3</td>
<td>Payload exceeds manufacturer's limits</td>
<td>Faulty functionality</td>
<td>Drone's payload is too heavy for sustained flight.</td>
<td>EF-10 Drone crashes or runs out of battery power</td>
<td>Critical</td>
</tr>
<tr>
<td>FM-E4</td>
<td>Windspeeds exceed manufacturer's limits.</td>
<td>Faulty functionality</td>
<td>Drone flies in windspeeds greater than specified by the manufacturer.</td>
<td>EF-11 Drone crashes or runs out of battery power prematurely.</td>
<td>Critical</td>
</tr>
</tbody>
</table>
Using EARS to write Safety-Stories

- Simple structure adds rigor & clarity
- System response describes what the system must actually do that is visible at the boundary of the system

1. **Ubiquitous**
   Requirement is always active

2. **Event-driven** (keyword When)
   Required response to a triggering event

3. **State-driven** (keyword While)
   Required response in a specified state

4. **Option** (keyword Where)
   Applicable only if feature is included

5. **Hybrid:**
   Use combinations of when, while and where for requirements with complex conditional clauses.

---

Using EARS to write Safety-Stories

Ubiquitous
The <component name> shall <response>
The drone shall maintain a minimum-separation distance at all times.

Event Driven.
When <trigger> the <system name> shall <system response>
When the drone is within X centimeters of minimum separation distance from another drone, the collision avoidance system shall provide directives to all drones in the vicinity.

State Driven
While <in a specific state> the <system name> shall <system response>
While in landing mode the drone shall descend vertically until it reaches the ground.

Option
Where <feature is included> the <system name> shall <system response>
Where parachute mode is enabled and a drop is initiated the drone shall scan the dropzone for obstacles.

Unwanted Behavior
If <optional preconditions> <trigger>, then the <system name> shall <system response>
If wind gusts exceed desired wind velocity but are below the maximum wind velocity, the drone shall return to base.
Assumptions

It is essential to understand your assumptions!

Wheels are turning if, and only if, plane is on the runway.

Led to an accident when a plane failed to brake because runway was wet and hydroplaning occurred.
Assumptions and Safety Requirements

Assumption: The Model-X Drone is able to transmit messages at distances up to 1000 meters.

Safety Story: The drone must remain within 900 meters of a ground control station at all times.

User Story: As a drone dispatcher I will dispatch drones to deliver supplies.
### Activity: Localization Related Faults

**Where am I?**

Can you think of Data and Event faults related to localization?

<table>
<thead>
<tr>
<th>Assump.</th>
<th>Assumption Description</th>
<th>External System</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-6</td>
<td>The operating range (i.e. maximum distance that a drone can communicate with its base station) is 1 Km.</td>
<td>Telemetry</td>
</tr>
<tr>
<td>A-13</td>
<td>Communicable distance between the base station and the drone can be significantly reduced by environment factors such as radio interference.</td>
<td>Telemetry</td>
</tr>
<tr>
<td>A-20</td>
<td>The maximum flight time of the Iris 3DR drones is 22 minutes.</td>
<td>Iris 3DR</td>
</tr>
<tr>
<td>A-12</td>
<td>Iris 3DR GPS accuracy is within 8 meters.</td>
<td>GPS</td>
</tr>
<tr>
<td>A-29</td>
<td>The wing span of the 3DR drone is 55 cm.</td>
<td>Iris 3DR</td>
</tr>
<tr>
<td>A-21</td>
<td>The maximum airspeed of the Iris 3DR drone is 22 meters per second.</td>
<td>Iris 3DR</td>
</tr>
<tr>
<td>A-24</td>
<td>When a lock is acquired on 3 satellites, GPS accuracy will be within 5 meters.</td>
<td>GPS</td>
</tr>
<tr>
<td>A-40</td>
<td>The locator beacon can broadcast reliably over a distance of 400 feet.</td>
<td>Loc8tor beacon</td>
</tr>
<tr>
<td>A-41</td>
<td>The locator beacon is accurate to within one inch.</td>
<td>Loc8tor beacon</td>
</tr>
</tbody>
</table>
Making connections..

Hazard
Fault
FMECA?

Goal

Project Glossary
Context Diagram

UseCase
Acceptance Test

tests

Environmental Assumptions

JIRA

User Story

depends on

depends on

mitigates

describes

Project

Glossary

GitHub

Source Code

Package

Occurs in

Fault

Design

influences

Test

Unit Test

Test

Tests

Trace Matrix in excel.
Trace links embedded in tool
Artifact in tool, spreadsheet, word doc etc.

Threat Model

Documents security vulnerabilities

Safety Assurance Case

Compiles evidence for safety
Build a Safety Argument

Construct a systematic convincing argument, supported by evidence, to demonstrate that the system is safe for use.
Goal Structured Notation (GSN)

**G1:** UAV's approximation of its current location is bound by the maximum potential error it reports.

**C2:** Hazard FM-D1 means that position coordinates fail to reflect actual position of drone.

**S2:** Extensive tests are conducted following industry best practices.

**G2:** Current GPS coordinates of UAV are accurate.

**S3:** RTK (Real Time Kinematics) provides GPS accuracy to within 2.1 horizontal centimeters and 6.3 vertical centimeters when within 1 Km of a base station.

**G3:** Ground speed is measured accurately.

**S4:** Time elapsed since last accurate GPS reading is accurately reported.

**G4:** Drone position is computed using redundant techniques of GPS, inertial navigation system, and beacon references.

**G5:** Upper bound of potential location error is computed and reported accurately.

**C2:** Hazard FM-D2 means that the reported GPS position is greater than the computed error.

**S5:** Inertial Navigation System (INS) uses dead reckoning to compute GPS coordinates.

**S6:** Accelerometers and gyroscopes are calibrated frequently as part of a standard maintenance process.

---

**Fit for purpose?**

**Catastrophic:** Failure may cause a crash.

**Hazardous:** Failure has a large negative impact on safety or performance, ... or causes serious or fatal injuries among the passengers.

**Major:** Failure is significant, but has a lesser impact such as discomfort.

**Minor:** Failure is noticeable but not dangerous.

**No Effect:** Failure has no impact on safety.
Construct the FMECA incrementally

How do we prioritize safety stories into the backlog? What is the impact on Safety of various prioritization decisions?

1. Inputs from Stakeholders, Customers, Users, Team
2. Product Owner
3. Safety Analysts
4. Preliminary Hazard Analysis
5. Preliminary FMECA (Fault Mode Criticality Analysis)
6. FMECA for stories placed into sprint backlog and for new feature interactions
7. Team selects user stories for the sprint
8. Sprint
9. Scrum meeting
10. Incrementally Constructed Safety Assurance Case
11. Working Product
12. Is it currently safe?

Ranked list of what is required – Features, User stories, Safety-Stories
Carefully Track Safety Dependencies

User stories with safety dependencies are tracked as they move through the system.

The user stories turn “green” once all safety dependencies are completed.
Goal – to build a demonstrably safe system

<table>
<thead>
<tr>
<th>ID</th>
<th>Fault</th>
<th>Description</th>
<th>Consequence</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM-D1</td>
<td>Faulty data</td>
<td>GPS coordinates do not reflect actual position of drone.</td>
<td>Drones fail to maintain minimum separation distance, crash in midair, and debris falls to the ground.</td>
<td>Critical</td>
</tr>
<tr>
<td>FM-E1</td>
<td>Faulty Algorithm</td>
<td>Drone's route is miscalculated</td>
<td>Drone flies into prohibited airspace</td>
<td>Catastrophic</td>
</tr>
</tbody>
</table>

A1: UAV’s approximation of its current location is bound by the maximum potential error it reports.

C2: Hazard FM-D1 means that position coordinates fail to reflect actual position of drone.

G2: Current GPS coordinates of UAV are accurate.

S2: Extensive tests are conducted following industry best practices.

S3: RTK (Real Time Kinematics) provides GPS accuracy to within 2.1 horizontal centimeters and 6.3 vertical centimeters when within 1 Km of a base station.

S4: Time elapsed since last accurate GPS reading is accurately reported.

S5: Inertial Navigation System (INS) uses dead reckoning to compute GPS coordinates.

S6: Accelerometers and gyroscopes are calibrated frequently as part of a standard maintenance process.

G3: Ground speed is measured accurately.

G4: Drone position is computed using redundant techniques of GPS, inertial navigation system, and beacon references.

G5: Upper bound of potential location error is computed and reported accurately.
In Closing

- Safety Critical Systems are different.

- Think about the degree of criticality.

- Incrementally identify potential hazards and faults – and define safety stories to mitigate them.

- Track user stories with safety dependencies through the system so that you always understand the degree to which safety has been achieved in each end-of-sprint deployment.
Some of the ideas described in this talk emerged as a result of projects funded by the US National Science Foundation under Grants CCF-0959924 and CCF-1265178.