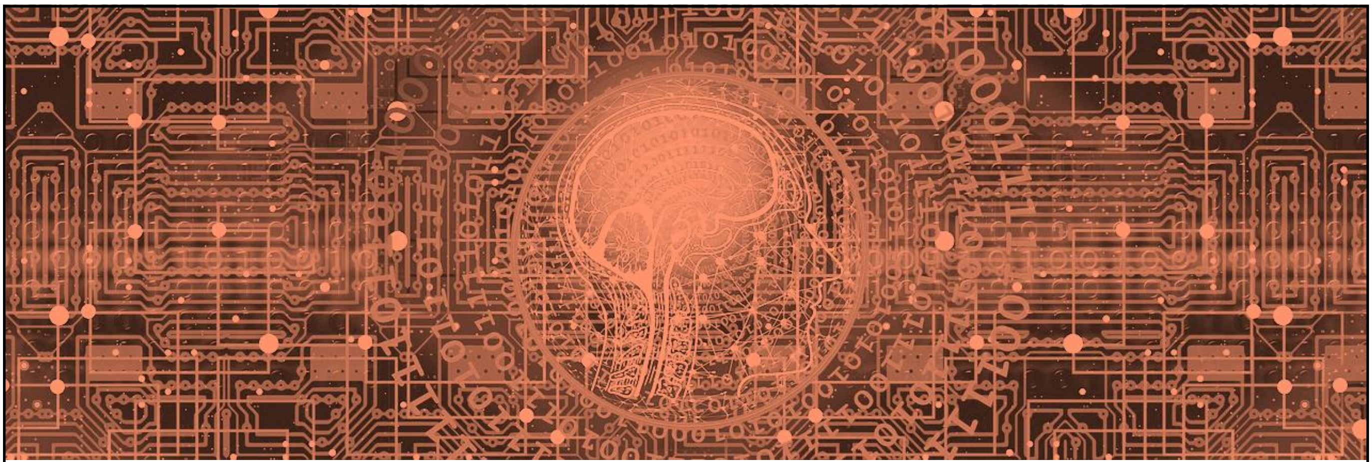


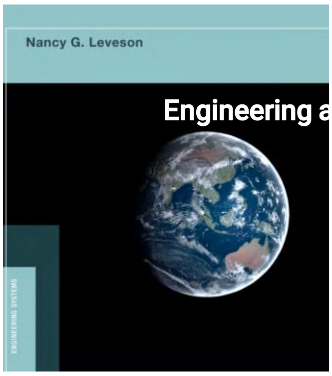
Human Robot Interaction I

Prof. Dr. Marian Daun

thws Technical University of Applied Sciences
Würzburg-Schweinfurt

Safety Engineering

Recommended Reading



https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjezrGE36H_AhVFCuwKHSewCdcQFnoECAoQAQ&url=https%3A%2F%2Flibrary.oapen.org%2Fbitstream%2F20.500.12657%2F26043%2F1%2F1004042.pdf&usg=AOvVaw2lziuP7bKPZ1UGIqV1BCEy

Fundamentals of Safety Engineering

Safety-Critical Software-Intensive Systems

Many systems nowadays take on safety-critical functionality. As with almost all innovations, they achieve their functionality mainly through the use of software.

Examples:

Automotive Electronic Stability Program
monitors speed and yaw rate and activates wheel individual brakes to ensure safe trajectory in curve while decelerating

Airborne Traffic Collisions Avoidance Systems
detects other aircraft on collision course and advises flight crew to increase separation altitude

Airborne Wing Control Software
adjusts flight surface parameters depending on "cargo" and fuel consumption

Industry 4.0 manufacturing cell supply chain monitoring software
monitors the supply of raw material, work products, and intrusion of foreign objects into the manufacturing cell

Safety – What is that?

"Safety" denotes the absence of the potential harm during operation for

human users
human non-users (e.g., "innocent bystanders," other stakeholders)
external systems
the system itself (sometimes… depending who you ask)
the environment (e.g., through pollution)

"Harm" could entail…
injury
death
destruction

Which can lead to **"risk"**, such as

- financial loss
- legal liability
- loss of reputation

Please note, that "risk" in this sense means "risk to the company making the product". Later, we will call "risk" the "probability of something bad happening."

Group Discussion

Why is there no absolute safety?

Technical University of Applied Sciences Würzburg-Schweinfurt

7

But…

The development process must ensure that during operation, the system will be **sufficiently safe**.

Technical University of Applied Sciences Würzburg-Schweinfurt

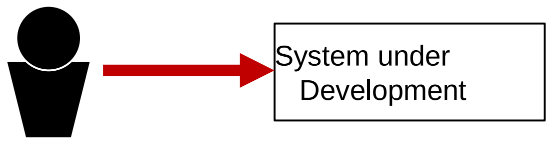
8

Group Discussion

What does sufficiently safe mean?

Difference between Safety and Security

Safety: System's behavior harms external entities by causing injury, death, or destruction.



Security: Malicious, external entities exploit system's behavior to control or obtain assets.

Safety Requirements Engineering Definitions

Hazard-Inducing Requirement
A functional safety-related requirement in the sense of [Firesmith 2004], which is the origin of a hazard during operation, given the occurrence of trigger conditions from the operational context of the system.

Hazard-Mitigating Requirement
A functional safety-related requirement in the sense of [Firesmith 2004], which, possibly together with other hazard-mitigating requirements, mitigates a hazard.

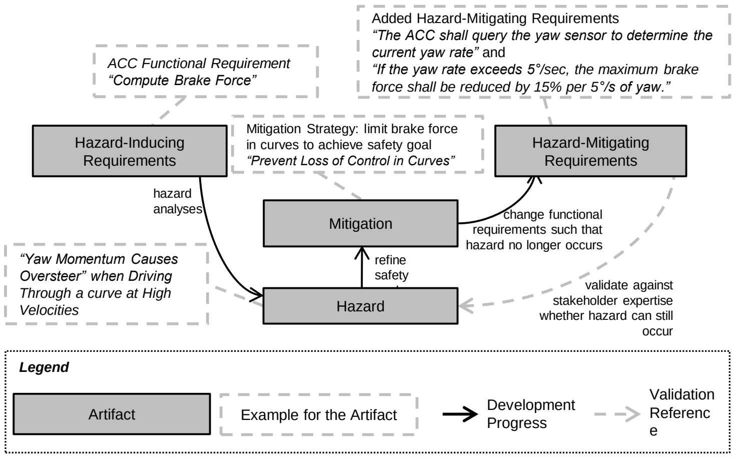
Hazard
An operational situation that – given disadvantageous triggering conditions from the operational context of the system – could lead or contribute to harm to come to humans or systems.

Trigger Condition
An operational or environmental condition, which may occur during operation such that a hazard is caused and must hence be avoided or rendered sufficiently unlikely to occur during operation for the hazard to be mitigated.

Safety Goal
A statement about the system's safety or specific safety property the system possesses or shall possess.

Mitigation
A set of hazard-mitigating requirements, which refine a safety goal into concrete, implementable measures that are intended to mitigate a specific hazard.

Example of the Relationship of these Terms



The Software Safety Development Process

Safety Assessment

Safety Assessment is the collection of all activities carried out during development to ensure that the system is sufficiently safe.

Safety Engineering

Safety Engineering is the academic study of the processes underlying safety assessment during development of safety-critical systems.

Safety Assessment during "Early" and "Late" Stages of Development

"Early" Stages of Development:

"left side of the V"

Activities before implementation

The further up in the V, the earlier

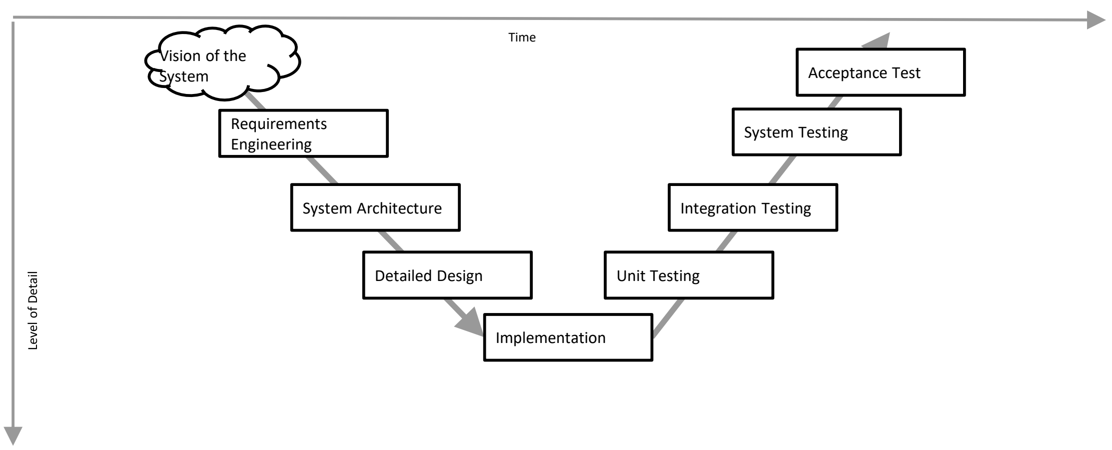
Purpose during Safety Assessment:
Find out which harm could occur.
Design the system to avoid harm or protect humans.

"Late" Stages of Development:

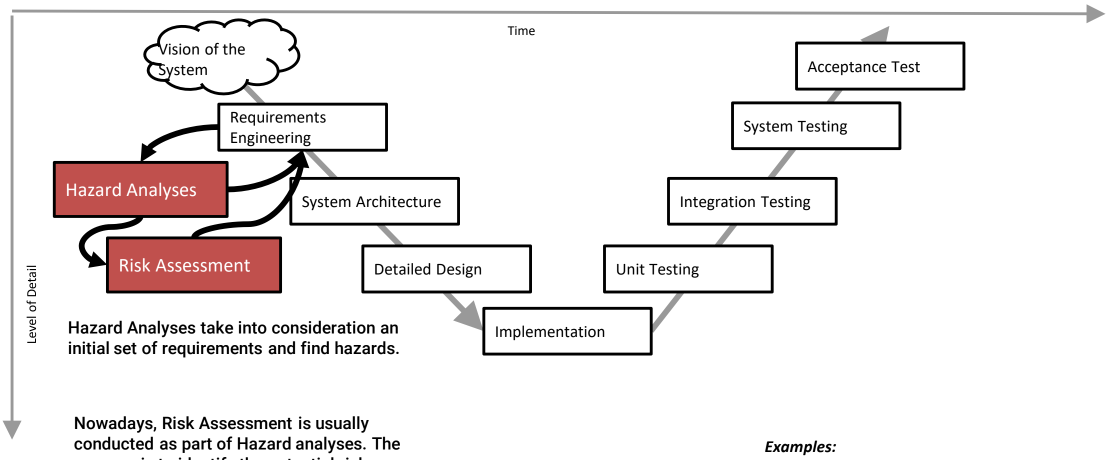
- "right side of the V"
- Activities after implementation
- The further up in the V, the later

Purpose during Safety Assessment:
Find out if the system was designed to avoid harm or protect humans well enough. Fix, if necessary.

The Software Development Process



The Software Safety Development Process

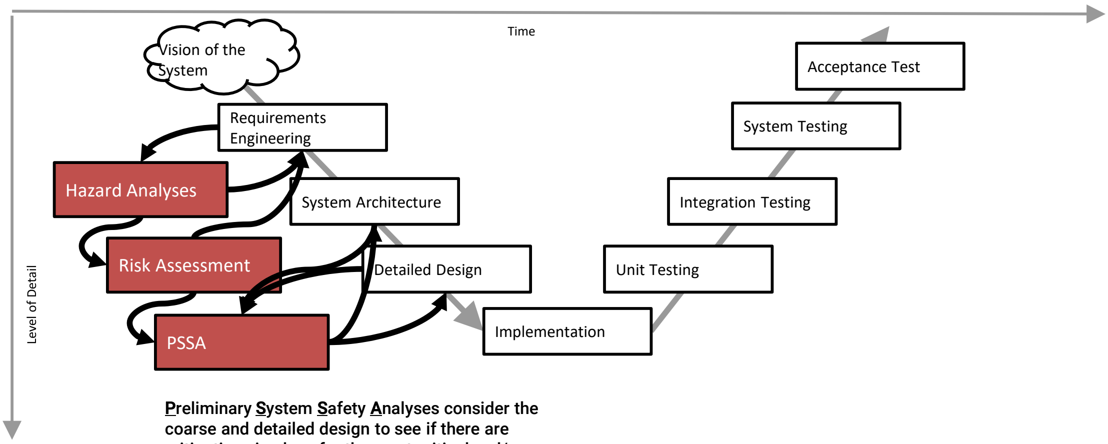


Hazard Analyses take into consideration an initial set of requirements and find hazards.

Nowadays, Risk Assessment is usually conducted as part of Hazard analyses. The purpose is to identify the potential risk criticality and probability of a hazard.

Examples:
- Functional Hazard Analysis (FHA)
- Hazard+Risk Analysis (HaRa, H+R)
- IEC61508 risk analysis

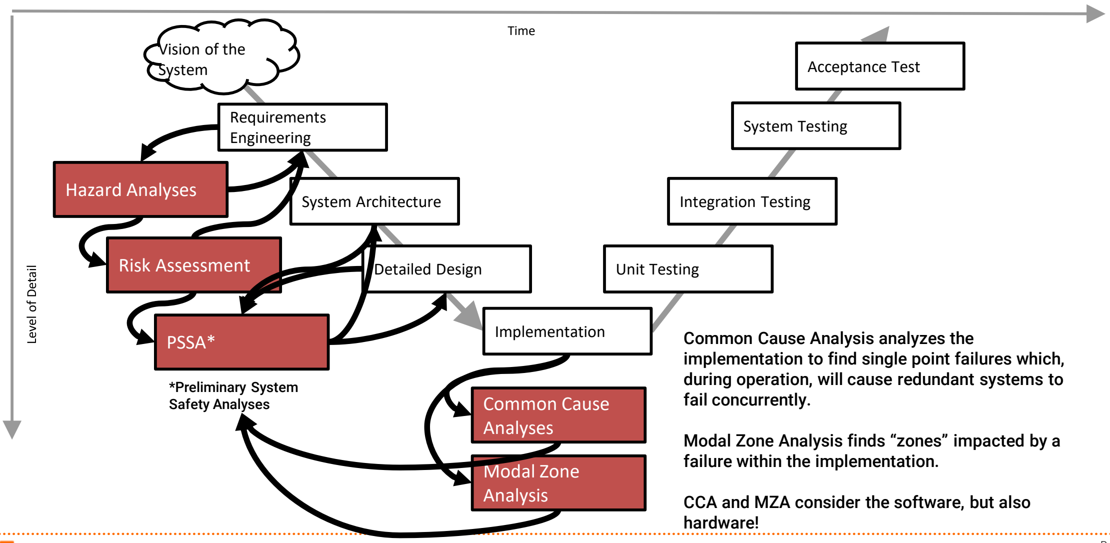
The Software Safety Development Process



Preliminary System Safety Analyses consider the coarse and detailed design to see if there are mitigations in place for the most critical and/or probable hazards.

Examples:
- (Component) Fault Tree Analysis
- Markov Analysis

The Software Safety Development Process



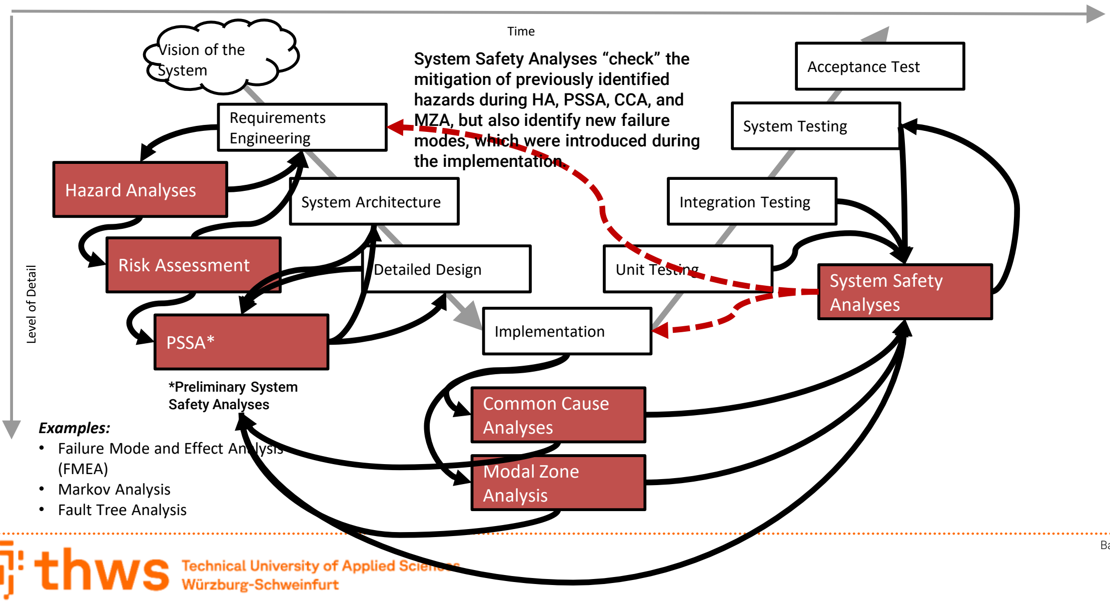
Common Cause Analysis analyzes the implementation to find single point failures which, during operation, will cause redundant systems to fail concurrently.

Modal Zone Analysis finds "zones" impacted by a failure within the implementation.

CCA and MZA consider the software, but also hardware!

Based on [Pumfrey 1999]

The Software Safety Development Process

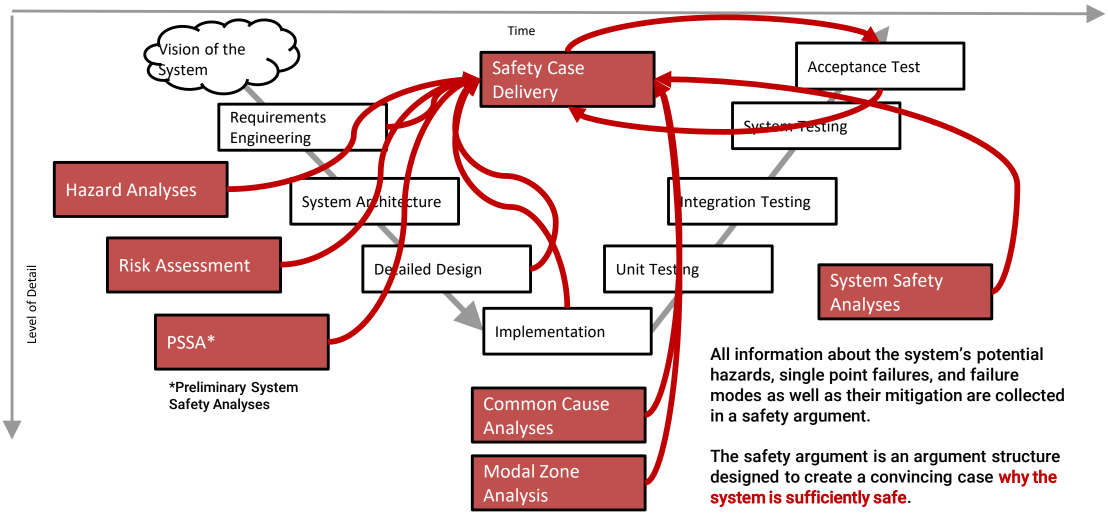


System Safety Analyses "check" the mitigation of previously identified hazards during HA, PSSA, CCA, and MZA, but also identify new failure modes, which were introduced during the implementation

Examples:
- Failure Mode and Effect Analysis (FMEA)
- Markov Analysis
- Fault Tree Analysis

Based on [Pumfrey 1999]

The Software Safety Development Process



All information about the system's potential hazards, single point failures, and failure modes as well as their mitigation are collected in a safety argument.

The safety argument is an argument structure designed to create a convincing case **why the system is sufficiently safe**.

Based on [Pumfrey 1999]

Lifecycle of Safety Requirements

Requirements

Requirements Types

Functional Requirements

Quality Requirements

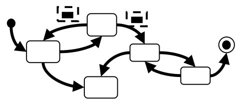
Constraints

Requirements Artifact Types

- Goals
- Scenarios
- Solution-oriented Requirements

Documentation Format

Natural-Language Requirements



Model-based Requirements

Group Discussion

What is a Safety Requirement?

Safety Requirements

Depending on whom you ask, the term "safety requirement" means something different.

Safety requirements could be...

Quality ("non-functional") requirements

– safety is a type of quality property of the system

Goals in goal-based requirements engineering

- see the GRL learning materials

Solution-oriented requirements in the Functional Perspective

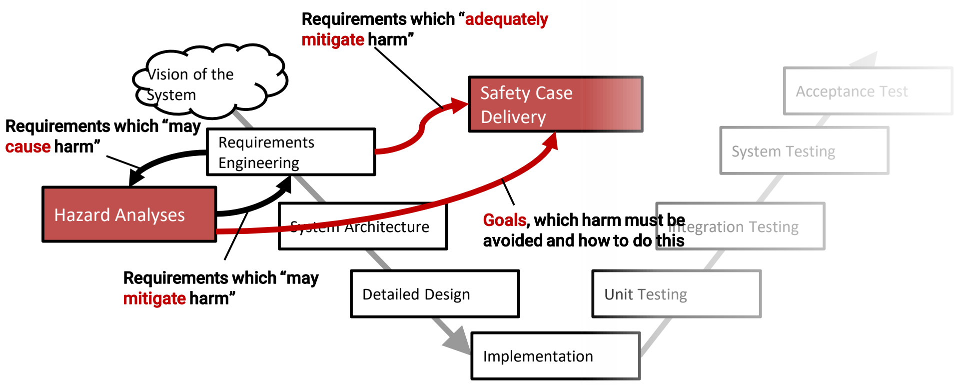
- for example, function "deploy airbag"

Solution-oriented requirements in the Behavioral Perspective

- for example, function "deploy airbag"

The Role of Safety Requirements

It becomes even more complicated when you look at this:



Summary

Safety Engineering is the study of processes underlying safety assessment.

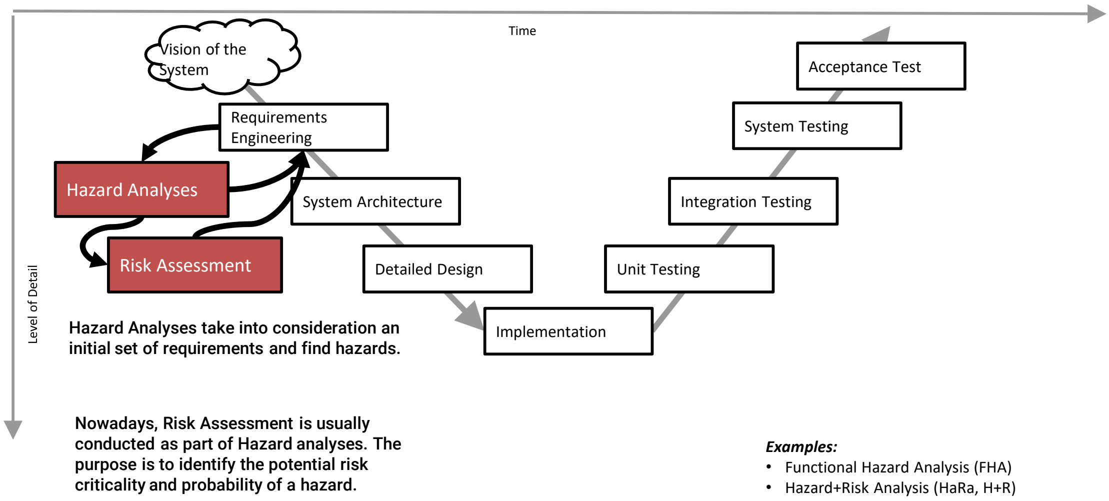
Safety Assessment describes the collection of processes that are parallel to the regular development process, which comprise all activities pertaining to the establishment of hazards, the design of mitigations, and the validation/verification that the mitigations are adequate.

Identified hazards, mitigations, and analysis results become part of a safety argument.

The safety argument documents that a system is sufficiently safe during operation.

Hazard Analyses

The Software Safety Development Process



Examples:
- Functional Hazard Analysis (FHA)
- Hazard+Risk Analysis (HaRa, H+R)
- IEC61508 risk analysis

The Purpose of Hazard Analyses

Hazard
An **operational situation** that – given disadvantageous triggering conditions from the operational context of the system – could lead or contribute to harm to come to humans or systems.

Hazard Analyses consider the functionality of the system and identify:

operational situations, in which harm could occur (i.e., *hazards*)

the conditions, under which the hazard occurs (i.e., the *trigger conditions*)

A principle strategy or desired property to mitigate the hazard (i.e., the *safety goal*)

Types of Hazard Analyses (Examples)

Functional Hazard Analysis (FHA)

Common in the aviation industry and US military applications

mandated by ARP 4761, FAA, US DoD → common in the avionics domain

Also includes a risk analysis component

Hazard and Risk Analysis (HaRa, H+R Analysis)

Common in the automotive industry

mandated by ISO26262 → common in the automotive domain

Also includes a risk analysis component

Almost the same thing as FHA

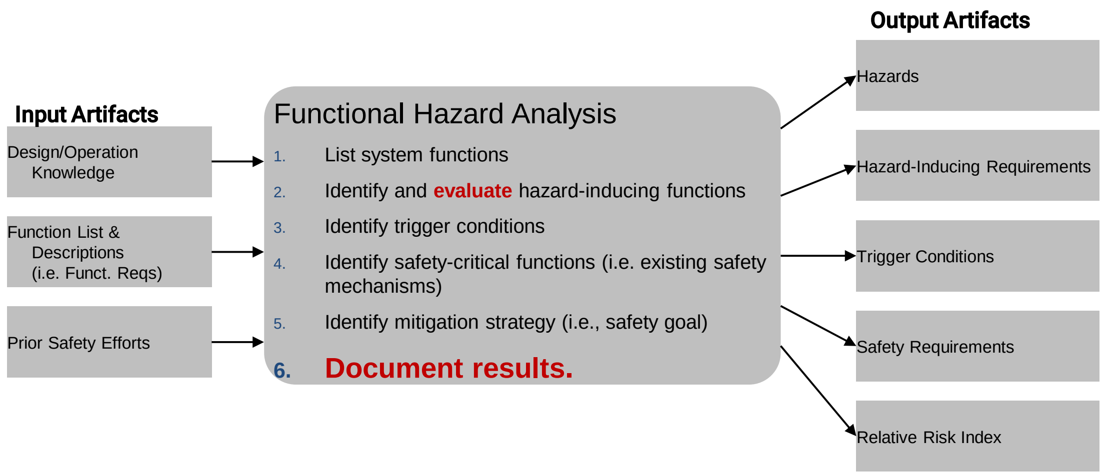
Hazard and Operability Studies (HAZOP)

Predecessor to FHA and H+R

mandated by (the now superseded) UK DoD MIL Std 882e

Focuses on exceptional behavior, not so much risk

FHA Process Overview



FHA Input Artifacts

Input Artifacts

- Design/Operation Knowledge
- Function List & Descriptions (i.e. Funct. Reqs)
- Prior Safety Efforts

Context of the System
& lots and lots of Experience

Functions within the Context,
Functional Solution-Oriented Requirements,
sometimes also Static-Structural SORs,
& lots and lots of Experience

Prior Hazard Anlayses and other kinds
of safety-relevant analyses and documents,
all documented in the **Context of the System**
& lots and lots of Experience

FHA Process Details

Functional Hazard Analysis

1. List system functions
2. Identify and **evaluate** hazard-inducing functions
3. Identify trigger conditions
4. Identify safety-critical functions (i.e. existing safety mechanisms)
5. Identify mitigation strategy (i.e., safety goal)
6. **Document results**.

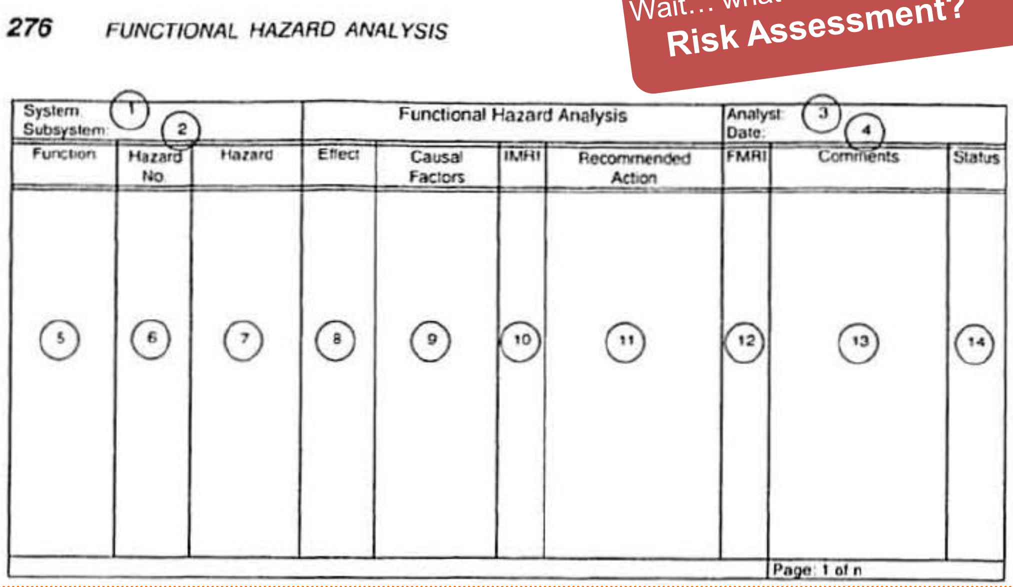
Write them all down, one by one.
Purpose: If you wrote them down, this means you thought about it and didn't forget.

Apply **guidewords**!
What happens, if the function/requirement…
1. … executes **too early**?
2. … executes **too late**?
3. … **fails** to execute?
4. … **executes, but shouldn't**?
5. … executes, but renders **wrong value**?

This about **when and why**.
You may want to look at:
- **Scenarios**
- **Behavioral** SORs
- **Conditions** in the Context

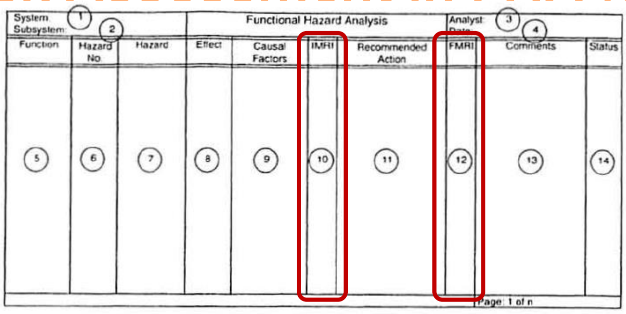
Document Results

Wait… what about
Risk Assessment?



From [Ericsson 2005], 1st Ed., p. 276

Risk Assessment in FHA (and many other HAs)



Initial and **Final**
Mishap Risk Indeces

This could go wrong,…

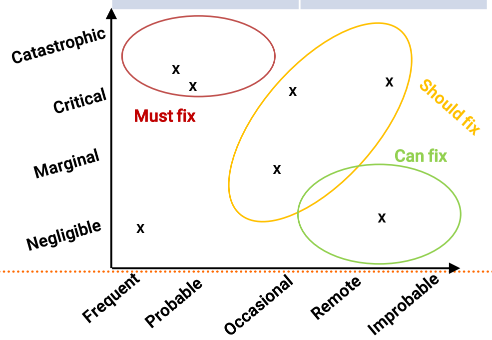
… which is approx. **this** likely,…

… but if we do **this**…

… It might only by **this** likely.

Assessing Risk:

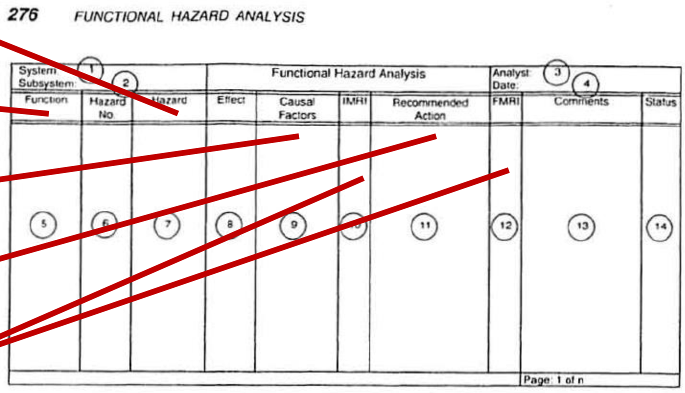
Severity	Probability
1: Catastrophic	A: Frequent
2: Critical	B: Probable
3: Marginal	C: Occasional
4: Negligible	D: Remote
	E: Improbable



FHA Output Artifacts

Output Artifacts

- Hazards
- Hazard-Inducing Requirements
- Trigger Conditions
- Safety Requirements
- Relative Risk Index



Hazard Mitigation Strategies

So…. We have the Safety Goals… Now what?

Hazard Analyses are but the first step in the safety development process.

Now, we need to find mitigations to implement the safety goals.

Mitigation
A **set of hazard-mitigating requirements**, which refine a safety goal into **concrete, implementable measures** that are intended to mitigate a specific hazard.

Mitigations

A mitigation must exist for each hazard.

Mitigations subsume requirements that establish safety. These are called hazard-mitigating requirements.

But, there is rarely a 1:1 correspondence…

One-to-One: One hazard-mitigating requirement exists for one hazard.
One-to-Many: One hazard-mitigating requirements exists for multiple hazards.
Many-to-One: A number of hazard-mitigating requirements exist to mitigate one hazard.
Many-to-Many: A number of hazard-mitigating requirements exist to mitigate multiple hazards.

Mitigation Strategies

Hazard Prevention.
A hazard is mitigated by preventing the hazard's trigger conditions from occurring during operation.

Hazard Reduction.
A hazard is mitigated by reducing the likelihood of the hazard to occur, e.g., by reducing the likelihood of the trigger conditions to occur.

Mitigation Strategies

Accident Prevention.
If a hazard cannot be prevented or sufficiently reduced, a mitigation can prevent the occurrence of a harmful accident due to a hazard.

Damage Control.
If a hazard can neither be prevented, nor sufficiently reduced, nor can an accident be prevented, a mitigation can aim to protect human users from injury, protect external systems from damage, or reduce the severity of such harm.

This could be achieved, for example, by means of **additional functionality** intended specifically for damage reduction.

Summary

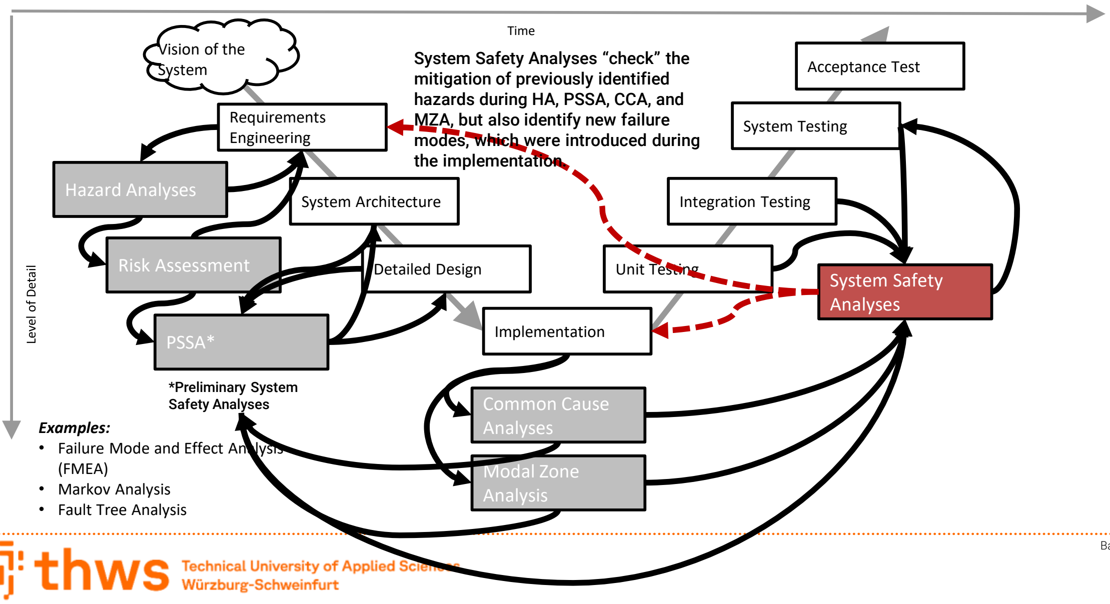
Hazard Analyses identify
- Hazards
- Trigger Conditions
- Safety Goals

Take into account functional requirements → hazard-inducing requirements

Aim: find mitigation strategies and hazard-mitigating requirements for the most hazards with highest severity and probability

Failure Mode and Effects Analysis

The Software Safety Development Process



Based on [Pumfrey 1999]

The Purpose of Safety Analyses

Safety Analyses consider the **implemented functionality** of the system and identify:

possible failures, which could lead to harm (i.e., ***hazards***)

the conditions, under which the failures occur (i.e., the ***trigger conditions***)

the state the system and the context is in when a failure occurs (i.e., the ***failure mode***)

the local and systemic impact of the failures (i.e., the ***effects***)

a principle strategy or desired property to mitigate the hazard (i.e., the ***safety goal***)

Subtypes of FMEA (Examples)

Failure Mode and Effects Analysis (FMEA)

Common in the automotive, aviation, and US military applications
mandated by ARP 4761, ISO26262, and international agencies
Also includes a risk analysis component

Failure Mode, Effects, and Criticality Analysis (FMECA)

Also includes a risk analysis component
Sometimes also includes a reliability analysis component

Causal Chain of Errors, Faults, and Failures

Hazard
Operational Condition leading to Harm
Example: Airbag doesn't protect passenger, passenger suffers head trauma.

may cause

Failure
Deviation of the System's Functionality from Required Behavior
Example: Airbag didn't deploy

may cause

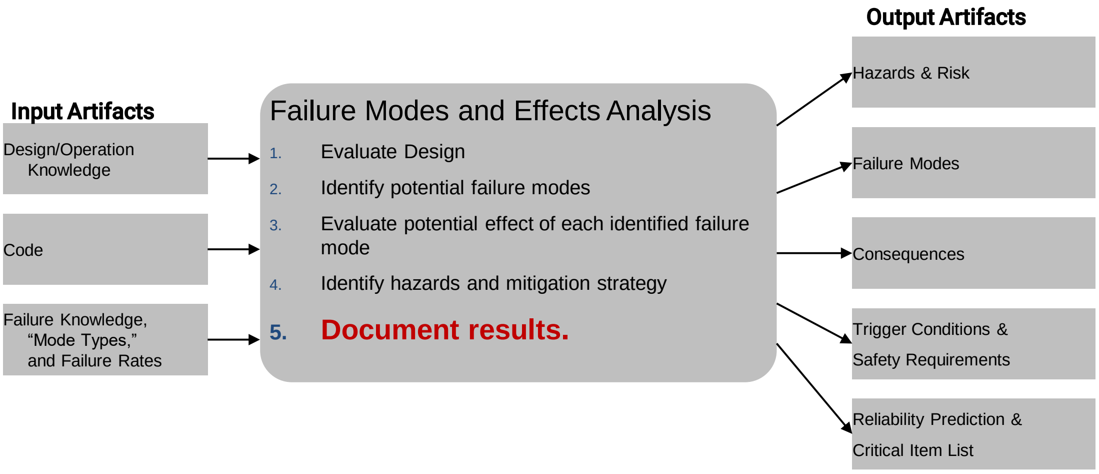
Fault
Defect in a software artifact (e.g., code, requirements)
Example: Method "`calculateTrigger(int impactForce)`" didn't trigger airbag

causes

Error
Mistake by the developer
Example: Programmer thought "`int impactForce`" is measured in PSI, but was in fact kg*m/sec

based on [Spillner & Linz, 2005]

FMEA Process Overview



FMEA Input Artifacts

Input Artifacts

Design/Operation Knowledge

This basically includes every single artifact produced up to this point. For example:
- Context Models
- Requirements Models
- Design Artifacts

& lots and lots of Experience

Code

This also includes Test Cases!
After all, these contain information about failures

Failure Knowledge, "Mode Types," and Failure Rates

Prior knowledge about the statistical failure probability
& lots and lots of Experience

FMEA Process Details

Failure Modes and Effects Analysis

1. Evaluate Design
2. Identify potential failure modes
3. Evaluate potential effect of each identified failure mode
4. Identify hazards and mitigation strategy
5. **Document results.**

Think about:

- System purpose
- System's components
- Intended interactions
- **Possible exceptions and alternatives**

Apply **guidewords**!

What happens, if the function/component…

1. … executes **too early**?
2. … executes **too late**?
3. … **fails** to execute?
4. … **executes, but shouldn't**?
5. … executes, but renders **wrong value**?

This largely **overlaps with software quality assurance**:

- What is the **failure**?
- What will happen to the component?
- What will happen to the system?
- What will happen to a human user or external system?
- What is the **fault**?
- What could be the **error** made by a human?

Document Results

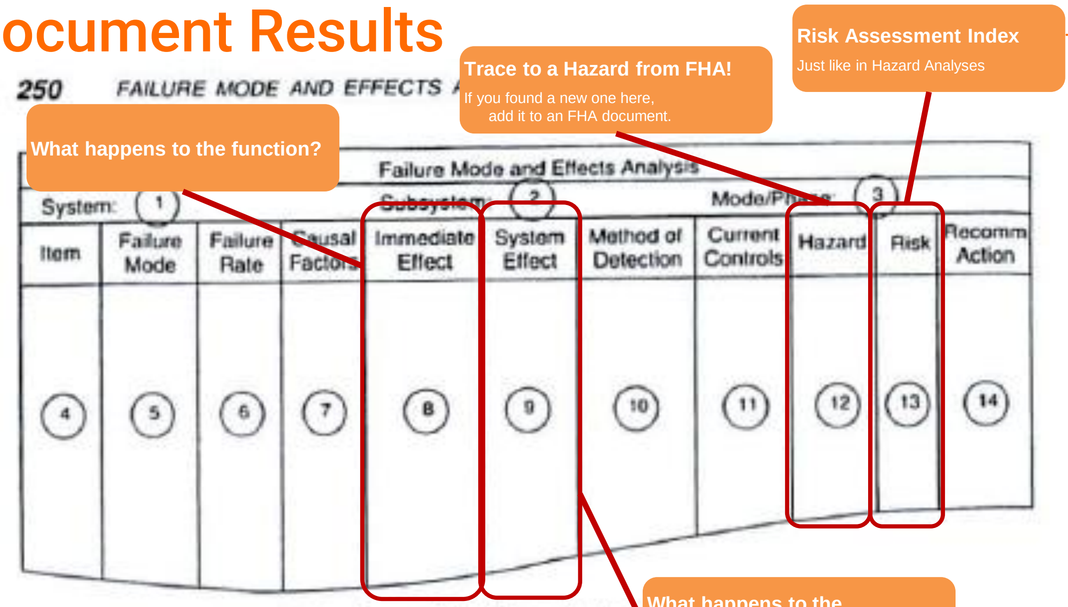


Figure 13.8 Example FMEA worksheet

SEVERITY / PROBABILITY	Catastrophic (1)	Critical (2)	Marginal (3)	Negligible (4)
Frequent (A)	High	High	Serious	Medium
Probable (B)	High	High	Serious	Medium
Occasional (C)	High	Serious	Medium	Low
Remote (D)	Serious	Medium	Medium	Low
Improbable (E)	Medium	Medium	Medium	Low
Eliminated (F)	Eliminated			

from [Sojka 2017]

FMEA Output Artifacts

Output Artifacts

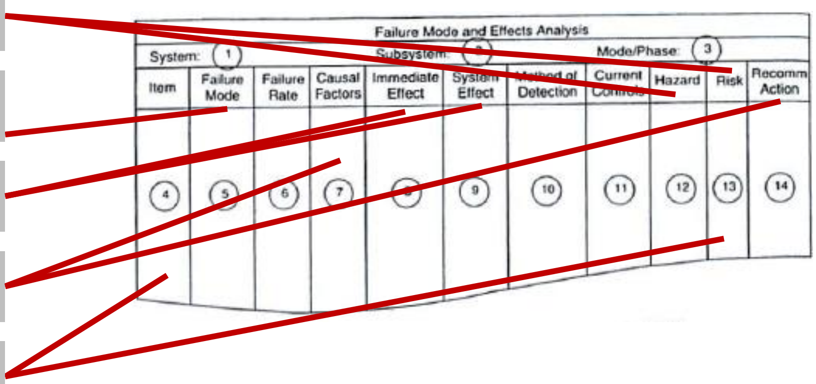
Hazards & Risk

Failure Modes

Consequences

Trigger Conditions & Safety Requirements

Reliability Prediction & Critical Item List



Summary

Safety Analyses identify

- Safety critical deviations, which could lead to harm
- The failure modes, in which these deviations occur
- Trigger Conditions
- Safety Goals

Take into account implemented functionality → code

Aim: find mitigation strategies and hazard-mitigating requirements for the failures with highest severity and probability which lead to hazards

Safety Argumentation

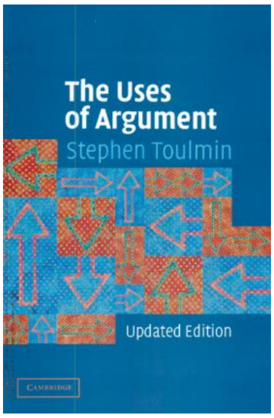
Recommended Reading

This unit is mainly based on the work by

Prof. Dr. Tim Kelly

University of York, https://www-users.cs.york.ac.uk/tpk/

Which in turn is based on
The Uses of Argument by Stephen Toulmin
Cambridge University Press, 1958



You may also want to look at:
A remarkably well-done tutorial on GSN: http://modeling-languages.com/goal-structuring-notation-introduction/
The GSN Standard Website: http://www.goalstructuringnotation.info/

The Purpose of Safety Argumentation

The purpose of safety argumentation is to establish, maintain, and provide a defensible argument about the system's safety.

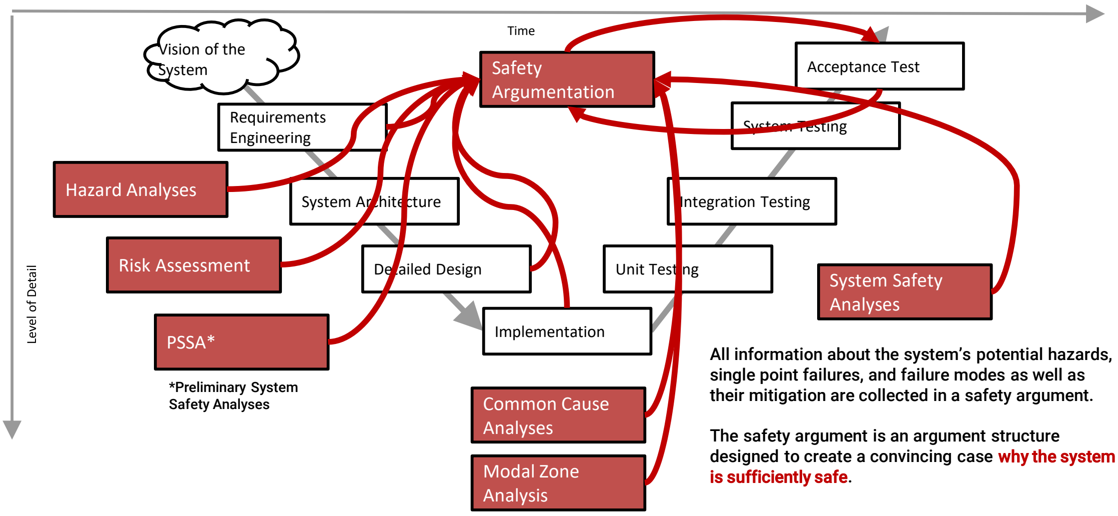
[Bishop et al. 2004], [Kelly 2007], [Tenbergen et al, 2015]

A safety argument is…

… established throughout development. You can't just "add on safety."
… maintained by systematically gathering evidence about a safety claim.
… provided to some type of authority for the purpose of repudiation of liability or certification.

A safety argument is not something you have. It's something you do.

The Software Safety Development Process

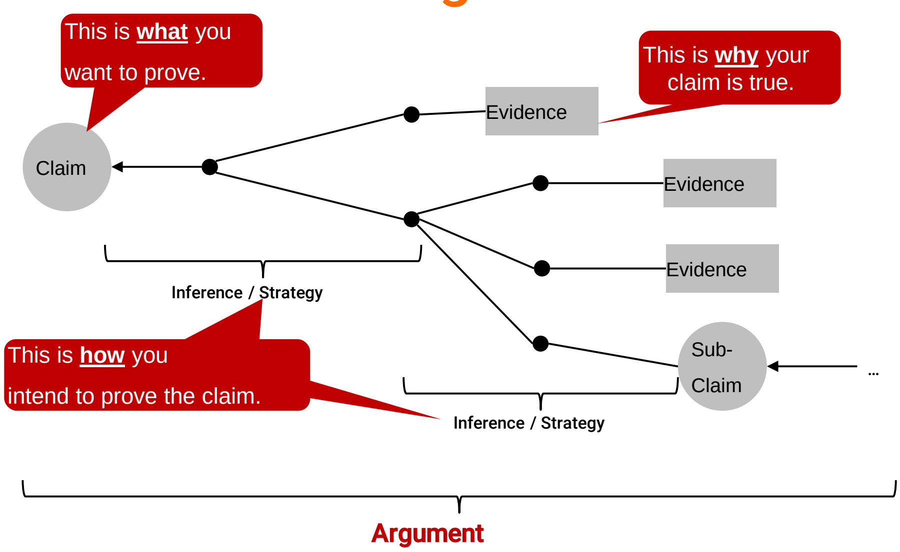


*Preliminary System Safety Analyses

All information about the system's potential hazards, single point failures, and failure modes as well as their mitigation are collected in a safety argument.

The safety argument is an argument structure designed to create a convincing case **why the system is sufficiently safe.**

Structure of an Argument



Building a Strong Safety Argument

The strength, defensibility, and irrefutability of a safety argument depends on three things:

How strong is your inference?
What do have to argue?
How do you argue?

How strong is your evidence?
What type of evidence to you have?
How objective is the evidence?
How subjective is the evidence?

How confident are you in your safety case?
What did you do to be sure that your safety claims are accurate?

Building an Argumentation Strategy (Inference)

What do you have to argue? How do you argue?

1. Start with the top level claim: "The system is safe."

   Notice, that it does not say "**must be**"!

2. Argue by means of identified hazards:
   What hazards were identified? What were their IMRIs?
   What mitigations were conceived for the hazards? What are their FMRIs?
3. Provide evidence:
   Hazard Analyses worksheet
   Hazard-Mitigating Requirements
   Validation Results / other analyses
4. Argue by means of identified failure modes.
   What failures were identified? What are their effects?
   How are these effects hazardous? What are their IMRIs?
   What mitigations were conceived for the hazards?
   What are their FMRIs?
5. Provide evidence:
   FMEA worksheet
   Hazard-Mitigating Requirements
   Implemented Changes in Code
   Validation Results / other analyses

Repeat until:
- All hazards are "argued"
- All analyses demanded by safety standards are complete
- Until you are **confident**
- Until the certification authority is satisfied

Evidence in your Safety Argument

There are several types of evidence, produced by different analyses.
Different types of evidence have a different argumentative strengths.

Type of Evidence	Source	Strength
Facts	Prior knowledge, human experience & wisdom	Objective, if provable
Assumptions	Prior knowledge, human experience & wisdom	Subjective, must be proven!
Sub-Claims	Arise from the engineering process and must be refined during continuous engineering	Depends on further refinement
Deterministic	Formal proofs (using mathematical means), formal analyses (e.g., Markov analyses)	Very objective
Probabilistic	Some form of quantitative statistical reasoning	Depends on the reasoning methodology and input data.
Qualitative	Compliance with rules that have an indirect link to the desired safety goal Opinions of "experts"	Subjective

based on [Bishop and Bloomfield 1998]

Confidence in your Safety Argument

How certain are you in the adequacy of your safety claim?
Confidence depends on two things:

Types and strength of evidence
An argumentation that you are confident in your safety case

But how?

Make a confidence case!

Just like the argument on safety, a confidence case argues about the confidence in the argument about safety.

Safety Cases and Confidence Cases

Safety Argument Artifacts

The safety argument hence consists of two complimentary artifacts:

Safety Case:
Argument structure containing claims and evidence about safety properties

Confidence Case:
Argument structure containing claims and evidence about adequacy of evidence in the safety case

safety argument := safety case + confidence case

Goal Structuring Notation (GSN)

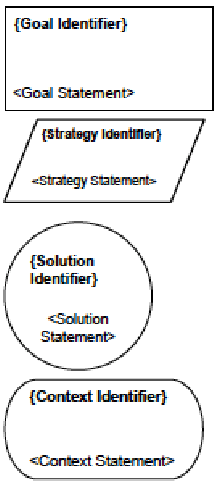
To document safety and confidence cases, Kelly's Goal Structuring Notation has been widely adopted:

UK Ministry of Defence
US Department of Defense
US Federal Aviation Administration
US National Transportation Safety Board
US Food and Drug Administration
US Federal Energy Regulatory Commission

don't require, but strongly recommend safety arguments to be GSN-based.

They also each have a set of standards, which describe how to argue safety and what must be done during development.

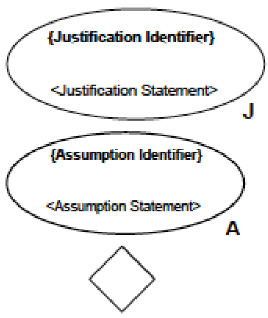
GSN Notation



A ***goal***, rendered as a rectangle, presents a claim forming part of the argument.

A ***strategy***, rendered as a parallelogram, describes the nature of the inference that exists between a *goal* and its supporting *goal*(s).
A ***solution***, rendered as a circle, presents a reference to an evidence item or items.

A ***context***, rendered as shown left, presents a contextual artefact. This can be a reference to contextual information, or a statement.



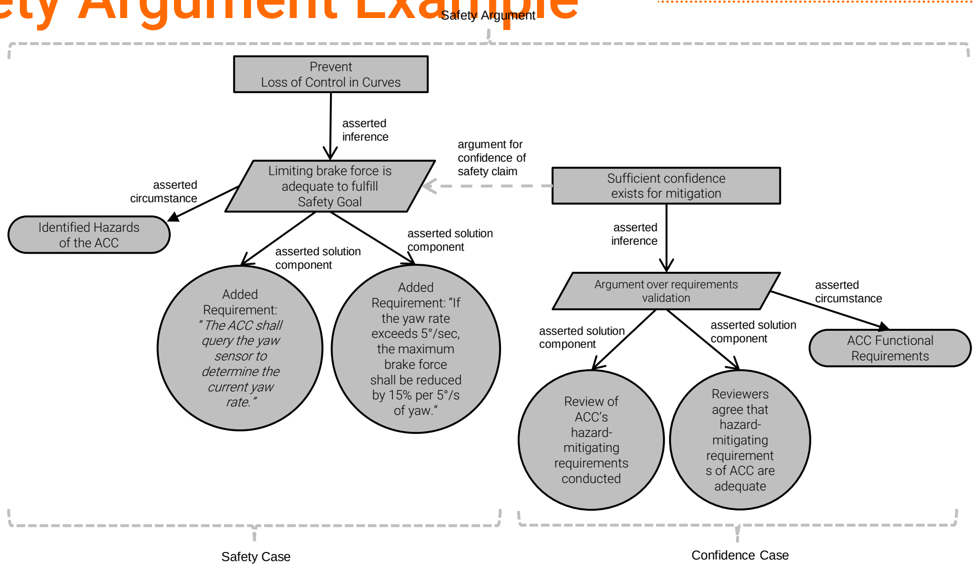
A ***justification***, rendered as an oval with the letter 'J' at the bottom-right, presents a statement of rationale.

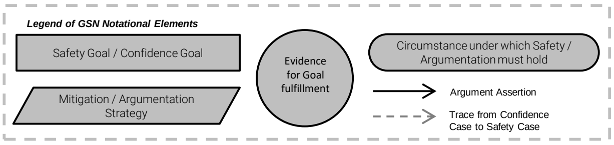
An ***assumption***, rendered as an oval with the letter 'A' at the bottom-right, presents an intentionally unsubstantiated statement.

Undeveloped entity, rendered as a hollow diamond applied to the centre of an element, indicates that a line of argument has not been developed. It can apply to goals (as below) and strategies.

http://www.goalstructuringnotation.info/documents/GSN_Standard.pdf p. 8

GSN Safety Argument Example





More examples on:
http://modeling-languages.com/goal-structuring-notation-introduction/

Summary

Safety Argumentation is concerned with establishing, maintaining, and providing a defensible argument about the system's safety

We use a typical argument structure consisting of claims, inferences, and evidence.

We prefer quantitative and objective evidence!

Document all your claims!
Then build a safety case and a confidence case. Together, these make up your safety argument.

The safety argument will be scrutinized by a (certification) authority or a court of law (in case of liability questions). Don't panic.