Systeme hoher Qualität und Sicherheit
Universität Bremen, WS 2013/14

Lecture 04 (11.11.2013)

Hazard Analysis Techniques

Christoph Lüth
Christian Liguda

Universität Bremen

Where are we?

- Lecture 01: Concepts of Quality
- Lecture 02: Concepts of Safety and Security, Norms and Standards
- Lecture 03: Quality of the Software Development Process
- Lecture 04: Requirements Analysis
- Lecture 05: High-Level Design & Formal Modelling
- Lecture 06: Detailed Specification
- Lecture 07: Testing
- Lecture 08: Program Analysis
- Lecture 09: Model-Checking
- Lecture 10 and 11: Software Verification (Hoare-Calculus)
- Lecture 12: Concurrency
- Lecture 13: Conclusions

Your Daily Menu

- Ariane-5: A cautionary tale
- Hazard Analysis:
  - What‘s that?
- Different forms of hazard analysis:
  - FMEA, Failure Trees, Event Trees.
- An extended example: OmniProtect

Ariane 5

- Ariane 5 exploded on its virgin flight (Ariane Flight 501) on 4.6.1996.

- How could that happen?

What Went Wrong With Ariane Flight 501?

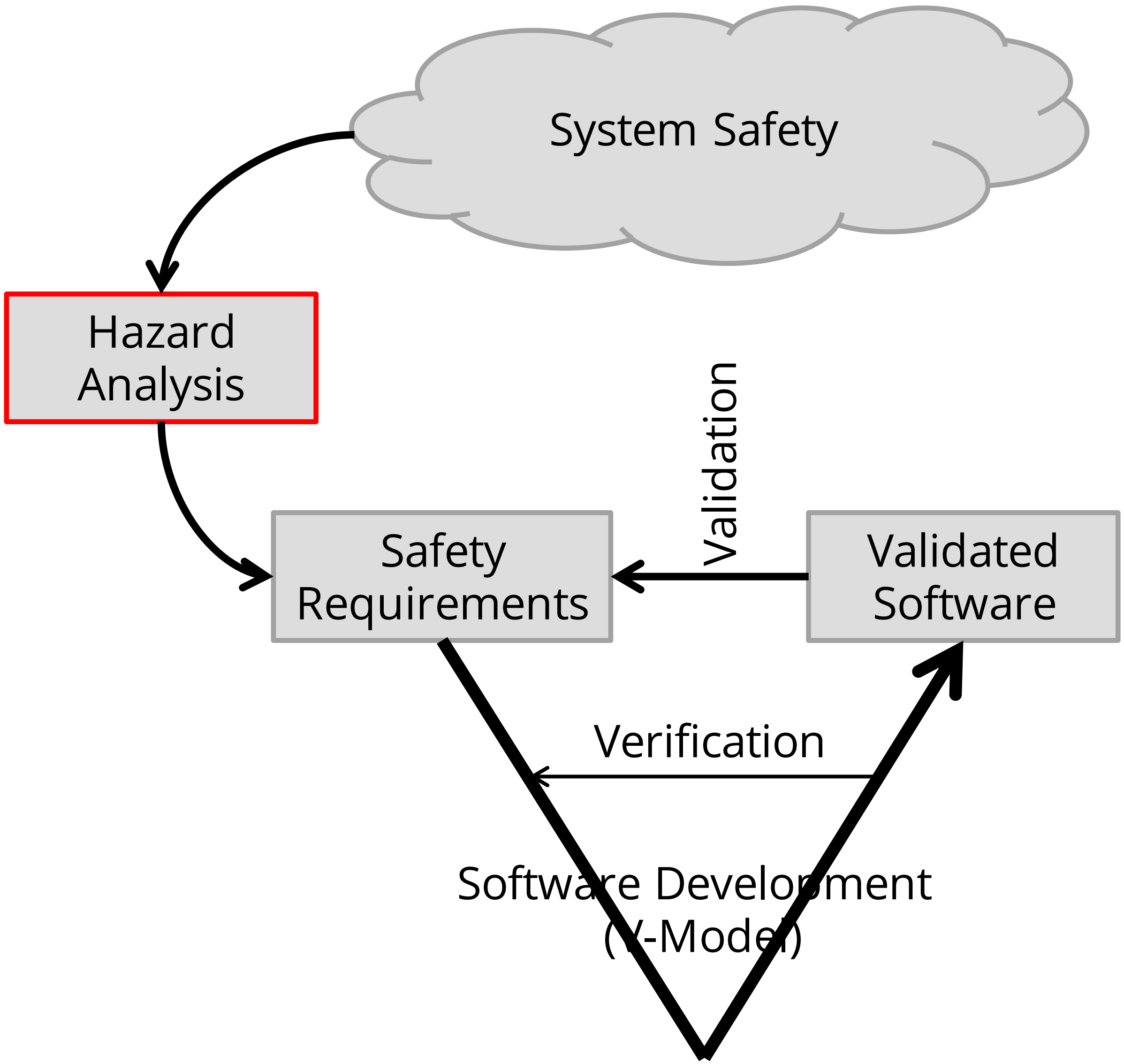
- Self-destruct triggered after 39 secs. due to inclination over 20 degr.
- OBC sent commands because it had incorrect data from IRS and tried to `adjust' trajectory.
- IRS sent wrong data because it had experienced software failure (overflow when converting 64 bit to 16 bit).
- Overflow occured when converting data to be sent to ground control (for test/monitoring purposes only).
- Overflow occured because
  - IRS was integrated as-is from Ariane 4, and
  - a particular variable (Horizontal Bias) held far higher values for the new model, and
  - the integer conversion was not protected because it was assumed that its values would never become too large.
  - This **assumption** was not **documented**.
- Because of its criticality, IRS had a backup system, but it ran the same software, so it failed as well (actually, 72 ms before the main one).

Hazard Analysis...

- provides the basic foundations for system safety.
- is Performed to identify hazards, hazard effects, and hazard causal factors.
- is used to determine system risk, to determine the signifigance of hazards, and to etablish design measures that will eliminate or mitigate the identified hazards.
- is used to **systematically** examine systems, subsystems, facilities, components, software, personnel, and their interrelationships.

Clifton Ericson: *Hazard Analysis Techniques for System Safety*. Wiley-Interscience, 2005.

Hazard Analysis i/t Development Process



Hazard Analysis systematically determines a list of **safety requirements**.

The realisation of the safety requirements by the software product must be **verified**.

The product must be **validated** wrt the safety requirements.

Classification of Requirements

- Requirements to ensure
  - Safety
  - Security

- Requirements for
  - Hardware
  - Software

- Characteristics / classification of requirements
  - according to the type of a property

Classification of Hazard Analysis

- **Top-down methods** start with an anticipated hazard and work back from the hazard event to potential causes for the hazard
  - Good for finding causes for hazard
  - Good for avoiding the investigation of “non-relevant” errors
  - Bad for detection of missing hazards

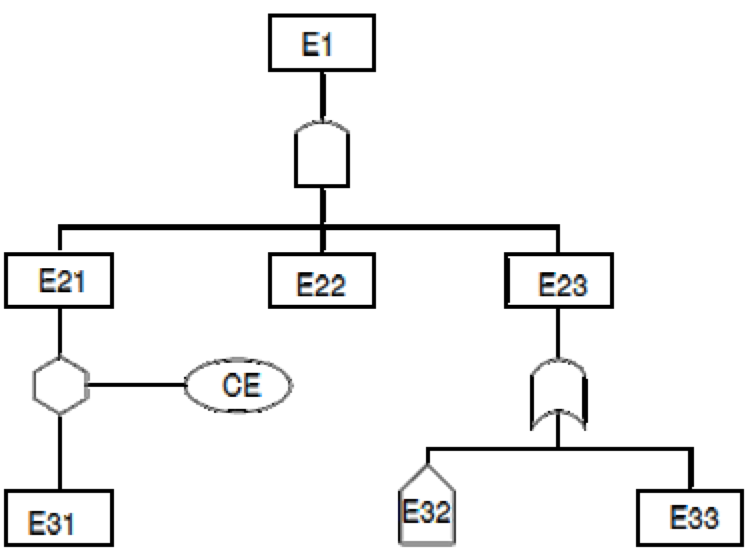
- **Bottom-up methods** consider “arbitrary” faults and resulting errors of the system, and investigate whether they may finally cause a hazard
  - Properties are complementary to FTA properties

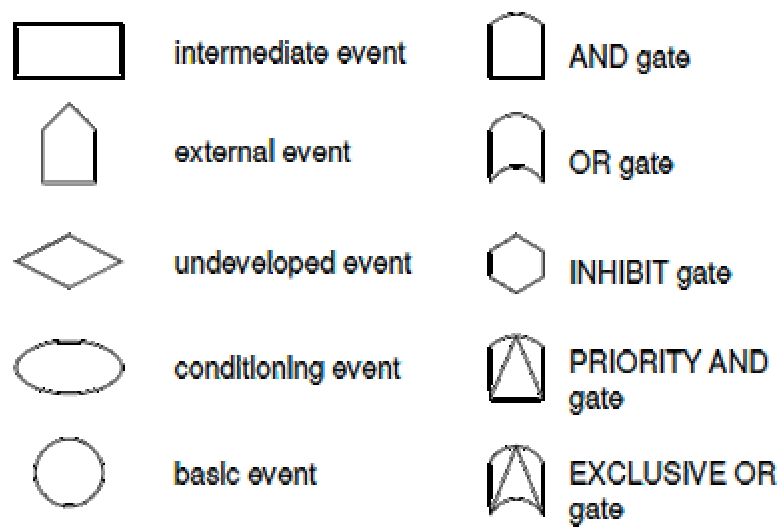
Hazard Analysis Methods

- Fault Tree Analysis (FTA) – top-down
- Failure Modes and Effects Analysis (FMEA) – bottom up
- Event Tree Analysis – bottom-up
- Cause Consequence Analysis – bottom up
- HAZOP Analysis – bottom up

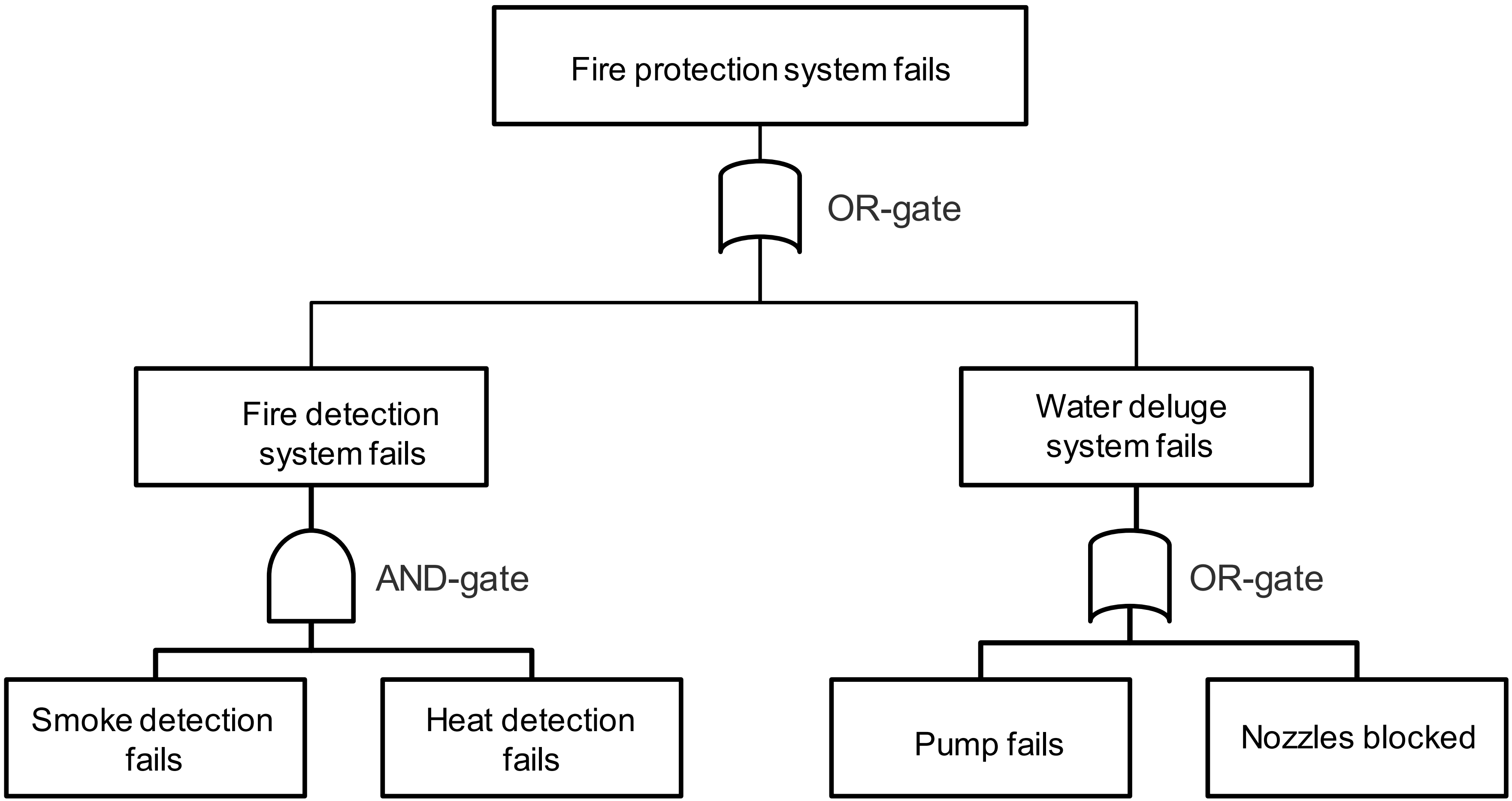
Fault Tree Analysis (FTA)

- Top-down deductive failure analysis (of undesired states)
  - Define undesired top-level event
  - Analyse all causes affecting an event to construct fault (sub)tree
  - Evaluate fault tree





Fault Tree Analysis: Example



Failure Modes and Effects Analysis (FMEA)

- Analytic approach to review potential failure modes and their causes.
- Three approaches: *functional*, *structural* or *hybrid.*
- Typically performed on hardware, but useful for software as well.
- It analyzes
  - the failure mode,
  - the failure cause,
  - the failure effect,
  - its criticality,
  - and the recommended action.

and presents them in a **standardized table**.

Software Failure Modes

Guide word	Deviation	Example Interpretation
omission	The system produces no output when it should. Applies to a single instance of a service, but may be repeated.	No output in response to change in input; periodic output missing.
commission	The system produces an output, when a perfect system would have produced none. One must consider cases with both, correct and incorrect data.	Same value sent twice in series; spurious output, when inputs have not changed.
early	Output produced before it should be.	Really only applies to periodic events; Output before input is meaningless in most systems.
late	Output produced after it should be.	Excessive latency (end-to-end delay) through the system; late periodic events.
value (detectable)	Value output is incorrect, but in a way, which can be detected by the recipient.	Out of range.
value (undetectable)	Value output is incorrect, but in a way, which cannot be detected.	Correct in range; but wrong value

Criticality Classes

- Risk as given by the *risk mishap index* (MIL-STD-882):

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

- Names vary, principle remains:
  - Catastrophic – single failure
  - Critical – two failures
  - Marginal – multiple failures/may contribute

FMEA Example: Airbag Control (Struct.)

ID	Mode	Cause	Effect	Crit.	Appraisal
1	Omission	Gas cartridge empty	Airbag not released in emergency situation	C1	SR-56.3
2	Omission	Cover does not detach	Airbag not released fully in emergency situation.	C1	SR-57.9
3	Omission	Trigger signal not present in emergency.	Airbag not released in emergency situation	C1	Ref. To SW-FMEA
4	Comm.	Trigger signal present in non-emergency	Airbag released during normal vehicle operation	C2	Ref. To SW-FMEA

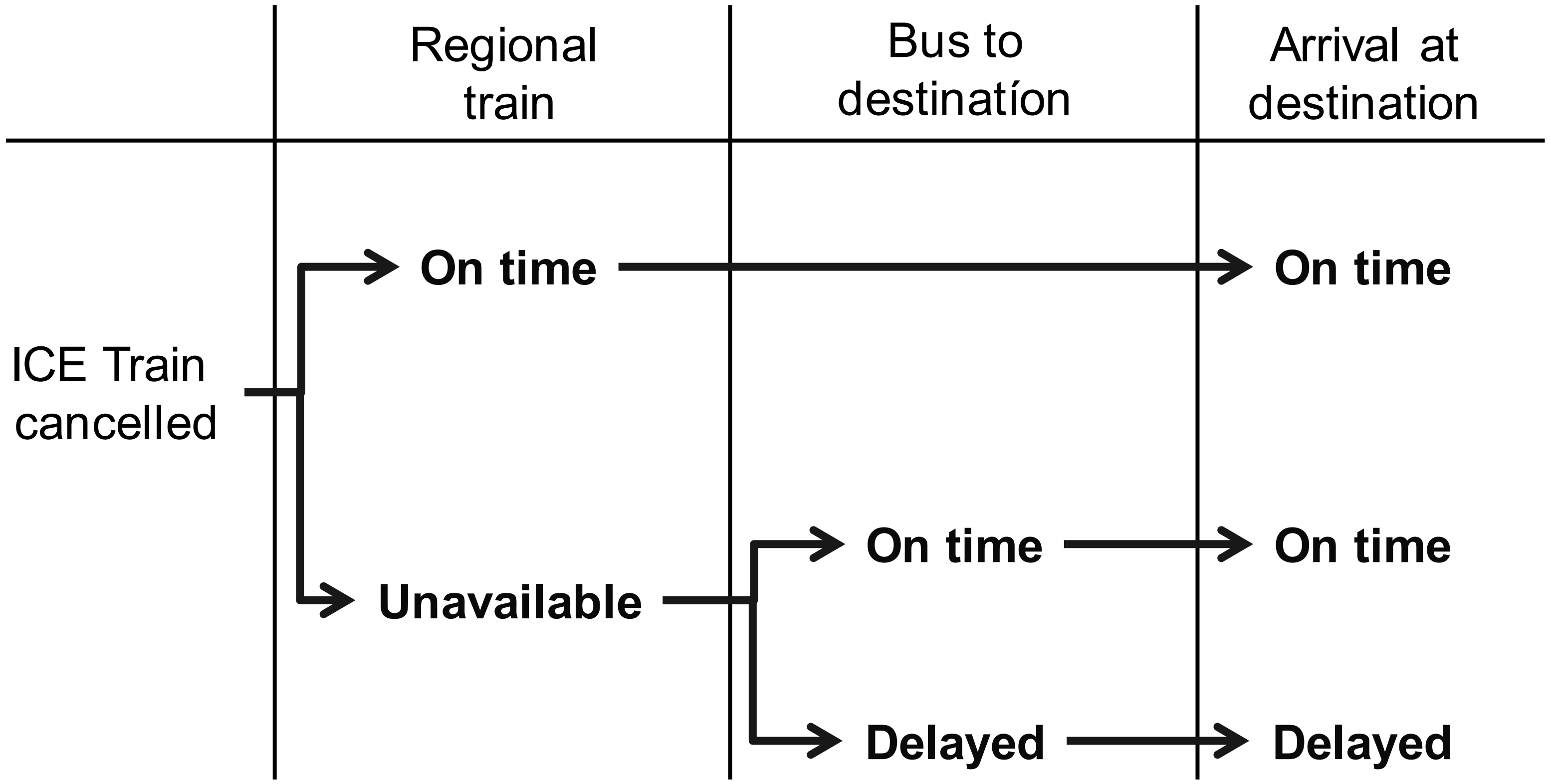
FMEA Example: Airbag Control (Funct.)

ID	Mode	Cause	Effect	Crit.	Appraisal
5-1	Omission	Software terminates abnormally	Airbag not released in emergency.	C1	See 1.1, 1.2.
5-1.1	Omission	- Division by 0	See 1	C1	SR-47.3 Static Analysis
5-1.2	Omission	- Memory fault	See 1	C1	SR-47.4 Static Analysis
5-2	Omision	Software does not terminate	Airbag not released in emergency.	C1	SR-47.5 Static Analysis
5-3	Late	Computation takes too long.	Airbag not released in emergency.	C1	SR-47.6
5-4	Comm.	Spurious signal generated	Airbag released in non-emergency	C2	SR-49.3
5-5	Value (u)	Software computes wrong result	Either of 5-1 or 5-4.	C1	SR-12.1 Formal Verification

Event Tree Analysis

- Applies to a chain of cooperating activities
- Investigates the effect of activities failing while the chain is processed
- Depicted as binary tree; each node has two leaving edges:
  - Activity operates correctly
  - Activity fails
- Useful for calculating risks by assigning probabilities to edges
- O(2^n) complexity

Event Tree Analysis



Hazard Analysis as a Reachability Problem

The analysis whether "finally something bad happens" is well-known from **property checking** methods

- Create a model describing everything (desired or undesired) which might happen in the system under consideration
- Specify a logical property *P* describing the undesired situations
- Check the model whether a path – that is, a sequence of state transitions – exists such that *P* is fulfilled on this path
- Specify as safety requirement that mechanisms shall exist preventing paths leading to *P* from being taken

The Seven Principles of Hazard Analysis

Ericson (2005)

1) Hazards, mishaps and risk are not chance events.
2) Hazards are created during design.
3) Hazards are comprised of three components.
4) Hazards and mishap risk is the core safety process.
5) Hazard analysis is the key element of hazard and mishap risk management.
6) Hazard management involves seven key hazard analysis types.
7) Hazard analysis primarily encompasses seven hazard analysis techniques.

Verifying Requirements

- **Testing**
  - Executable specification (i.e. sort of implementation)
  - Covering individual cases
  - Functional requirements
  - Decidable

- **(Static / Dynamic) Program Analysis**
  - Executable specification
  - Covering all cases
  - Selected functional and non-functional requirements
  - Decidable (but typically not complete)

Verifying Requirements II

- **Model Checking**
  - Formal specification
  - Covering all cases
  - Functional and non-functional properties (in finite domains)
  - Decidable (in finite domains)

- **Formal Verification**
  - Formal specification
  - Covering all cases
  - All types of requirements
  - (Usually) undecidable

Our Running Example: OmniProtect

- OmniProtect is a safety module for an omnidirectional AGV such as the Kuka OmniMove.
  - *Demonstration project only.*
- It calculates **a safety zone** (the area needed for breaking until standstill).
- Documents produced:
  - Document plan
  - Concept paper
  - Fault Tree Analysis
  - Safety Requirements
  - .... more to come.

Summary

- Hazard Analysis is the **start** of the formal development.
- It produces **safety requirements**.
- Adherence to safety requirements has to be **verified** during development, and **validated** at the end.
- We distinguish different types of analysis:
  - Top-Down analysis (Fault Trees)
  - Bottom-up (FMEAs, Event Trees)
- Hazard Analysis is a creative process, as it takes an informal input („system safety“) and produces a formal outout (safety requirements). Its results cannot be formally proven, merely checked and reviewed.
- Next week: High-Level Specification.