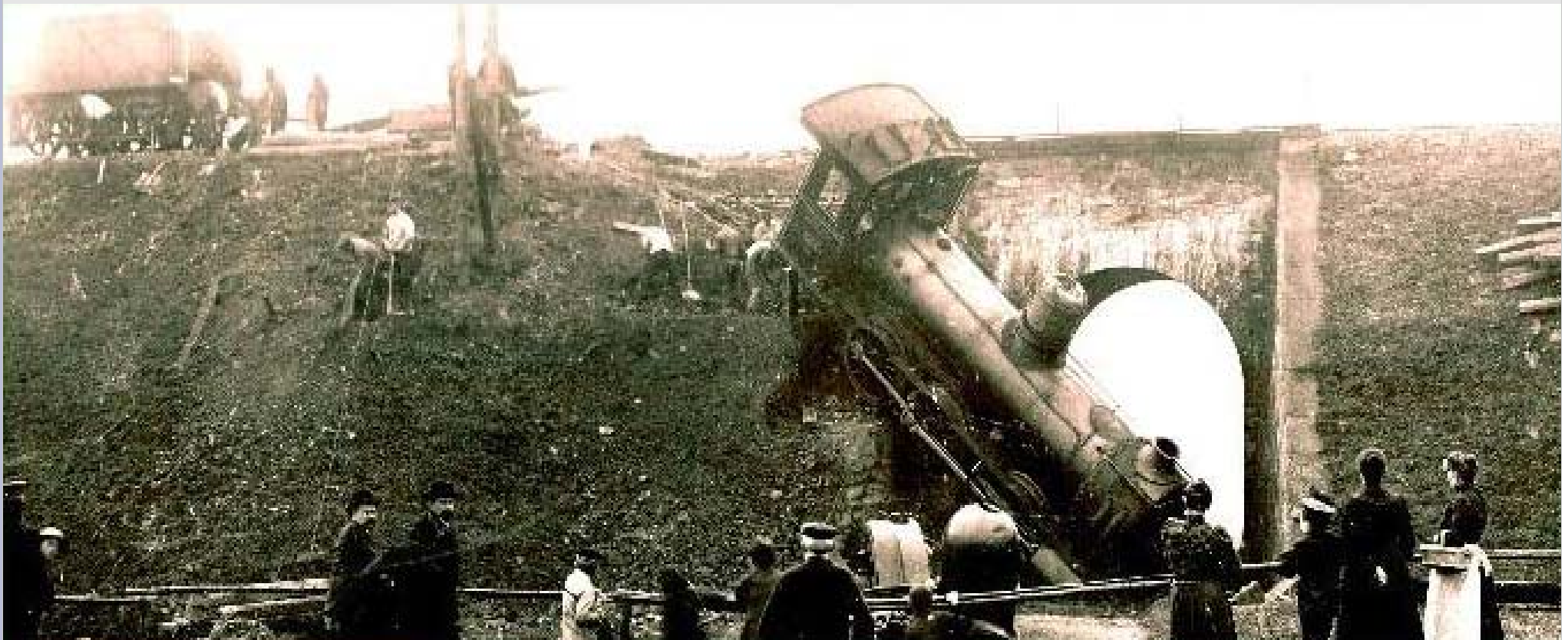


# Safety Certification - FAQ

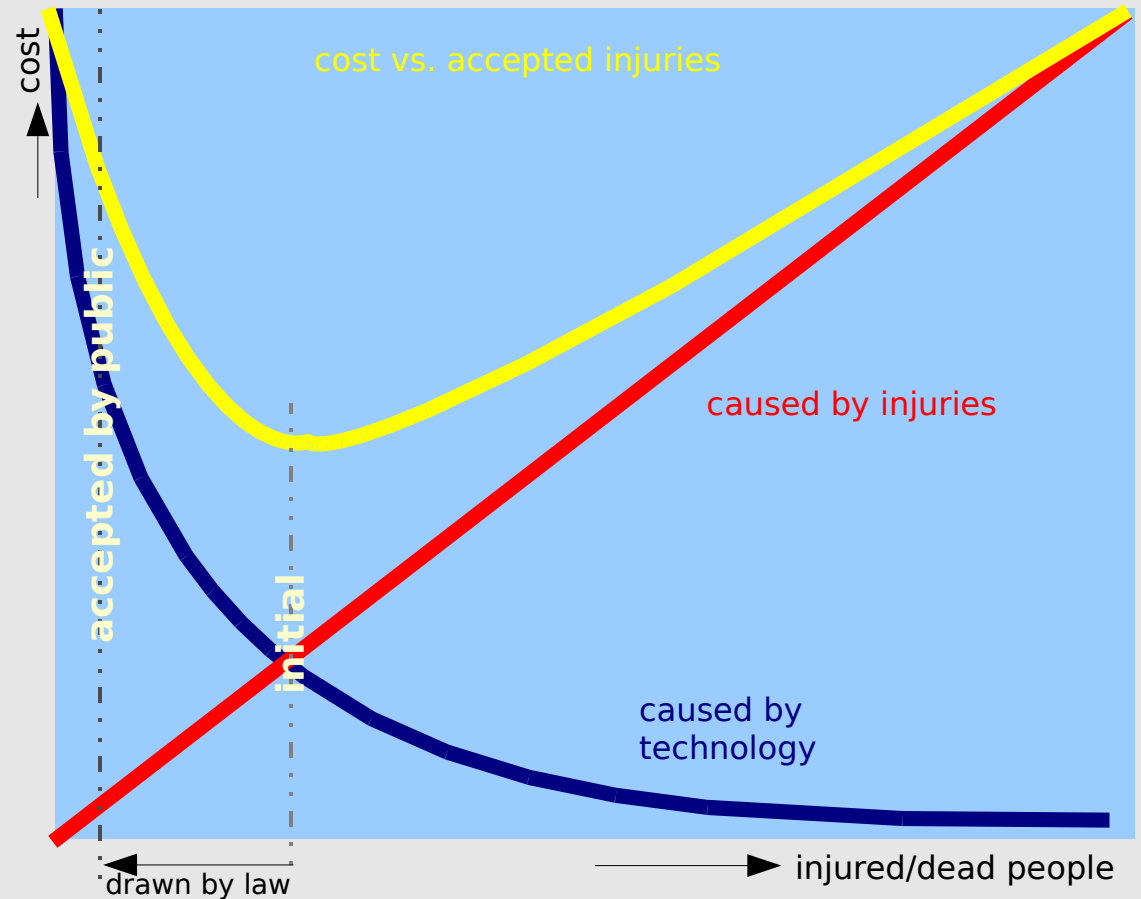
How does Software fit in?



# Who we are?

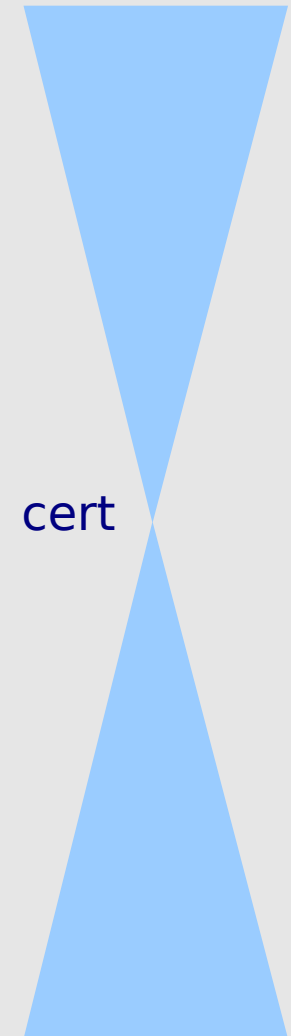


Rolf Schumacher  
rolf@august.de  
0049 177 7312001  
safety related RT-S/W projects since 1978  
independent safety assessor since 1998



# Who cares about safety?

- ◆ Vendors
  - Industries
  - Developers, Manufacturers, Operators
- ◆ Brokers / Mediators
  - Independent Safety Assossors
  - Safety Authorities
- ◆ Customers
  - Government and Public



# How Does Industry Care?

- ◆ Safety Aware Development & Production
- ◆ Verification & Certified Quality Control
- ◆ Safety Management
- ◆ Certified Safety Organization
- ◆ Independent Validation
- ◆ Process Paralleled Assessment



# How is Public Involved?



## ◆ Safety Authority

- Accrediting Safety Assessors
- Releasing Certifications

Deutscher  
Akkreditierungs  
Rat



## ◆ Government

- Converting Standards to Acceptable Laws

Deutscher  
Bundestag



## ◆ International Organization / EU

- Minimal Consensus: <http://www.era.eu.int/public/Safety/>
- Release of Internationally Binding Commitments based on Standards



## ◆ Standards Organizations

- Release of Safety Aware State-of-the-Art Standards



# How do we all Communicate?

- ◆ We Share a Common Language
- ◆ Industry Releases *Safety Cases*
- ◆ Project Attendant Assessment
  - Access to Resources and People
  - Checks against Standards, Fully Documented
  - Independently Released Final Judgement
- ◆ Safety Authority gets *Assessments*
  - Releases Certification based upon Assessments
  - Gives Allowence to Operate in Public

# Key Terms of Safety Language?

## ◆ *Safety Integrity*

- Probability to adhere to Safety Functions
  - Complexity increases problems to give evidence for Safety Integrity

## ◆ *Tolerable Hazard Rate*

- caused by residual undetected *Dynamic Hardware Errors*
- e.g. not more than  $10^{-9}$ /h, or 1 hazardous failure in 100000 product-years

## ◆ *Safety Target*

- Hazard Rate that can be/shall be demonstrated to hold, upfront

## ◆ *Safety Integrity Levels*

- Mapping of Hazard Rate Ranges to the State-of-the-Art in Avoidance of *Systematic Errors*

# Essentials of a Safety Case?

- (1) Determine the System or Product
- (2) Give Evidence for Quality Management
- (3) Give Evidence for Safety Management
- (4) Demonstrate Technical Safety
- (5) Refer to Related Safety Cases
- (6) Give Concluding Summery

# Impact on Software Lifecycle?

- ◆ **Give Evidence for Safety-Rq-Completeness**
  - ... and trace requirements at all stages
- ◆ **Demonstrate Software-SIL**
  - how did you avoid systematic errors?
  - at any phase, involve the assessor
- ◆ **Detect any Single Random Hardware Fault**
  - implement detection algorithms in software
- ◆ **Communicate / Interact with Safe Systems**
  - across unsafe channels by means of safe protocols
- ◆ provide *Safety Application Conditions*

# What's more to Know about Safety Case?

- testing, error classes, the Validation Report
- detailed
  - Quality Management and its Report
  - Safety Management and its Report
  - Technical Safety and its Report
  - Impact of Safety Application Conditions
- examples
  - redundant/diverse processor architecture
  - communicating safe systems
  - systems covering different SIL
  - safety case of compatibility
- re-use modules that had been part of a safe product
- consider the prove-in-use approach
  - SIL2 and then SIL4
- ...

# What Classes of Errors to Find? In what Test Phase?

- ♦ module test:
  - (1) programming errors like  $\pm 1$ , true/false, array bounds
  - **(2) random errors in code (checksums, per module under test?)**
- ♦ s/w integration:
  - (3) incomplete interface specs between modules
  - (4) diff. in interpretation of interface specs between modules
- ♦ hw/sw integration:
  - (5) incomplete lower layer interface spec (e.g. TAS Platform)
  - (6) differences in interpretation of lower layer interface specs
  - (7) unexpected, unforeseen behavior of hardware/system-software
  - (8) performance issues
  - (9) unexpected behaviour in functionality, esp. safety
- ♦ independent validation
  - (10) differences in interpretation of requirements
  - (11) doubts about functionality, esp. safety (give evidence)
  - (12) doubts about non-func. requirements (give evidence)
  - (13) incomplete module-, s/w-, hw/sw-integr. tests
- ♦ system tests, field trials, qualification periods
  - (14) incomplete external interface specs
  - (15) differences in interpretation of external interface specs
  - **(16) unexpected behavior or reaction to faults of real external hard-/software**

# How is Testing Related to Safety?

- ◆ Testing: just another 4-Eyes-Principle
  - Risk Mitigation of Design, Implementation Errors
- ◆ one Test Coverage Statement by Error Class
- ◆ No Evidence on Fault Absence by Testing
- ◆ Quality and Safety are Build-In *not Tested-In*
- ◆ Safety Management Separate from Testing

# What's in Validation Report

- ◆ Statement of Independence, including Safety Management
- ◆ Evidence for Configuration Management
- ◆ Evidence for each Verification Step
- ◆ Evidence for each Single Requirement
- ◆ with Emphasis on Safety Requirements
- ◆ Justification
  - for each Open Issue
  - of Manuals, Installation, Operation, Maintenance
- ◆ Final Justification for taking Safety Responsibility by the Product

# What's in Quality Management Report?

- ◆ Give Evidence of
  - Configuration Management
  - ISO 9001, 9000-3 Processes
  - Competent Professionals
  - Verification, Reviews, and Inspections
  - Defect and Change Management
  - Known Open Issues

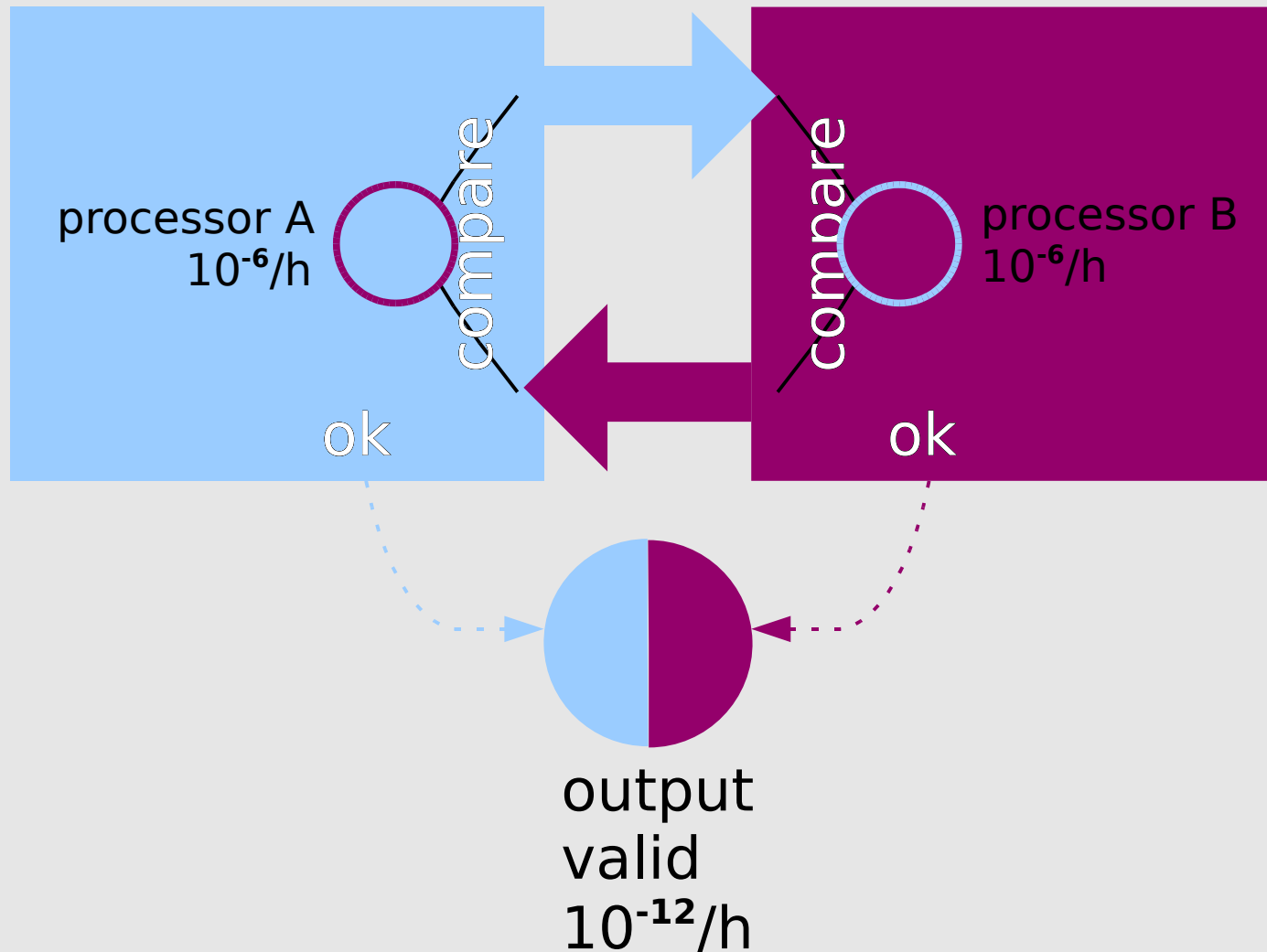
# What's in a Safety Management Report?

- ◆ Evidence of
  - Hazard Log and Analysis, Safety-Rq-Verification
  - (Software) Failure Mode and Effect Analysis
  - Audits on
    - Technical Safety Concept, State-of-the-Art
    - Adherence to Chosen Techniques and Measures
      - including Software-Maintenance
    - Rules for Operation and Maintenance
  - Achieved Safety and Availability Targets

# What's in a Technical Safety Report?

- ◆ Assurance of Safety Coverage in Application
  - show Architecture together with Safety Cases to be Applied to Parts of Application
- ◆ Assurance of Correct Functional Operation
- ◆ Effects of Faults, including Multiple Faults
- ◆ Operation with External Influences
- ◆ Detailed Safety Application Conditions

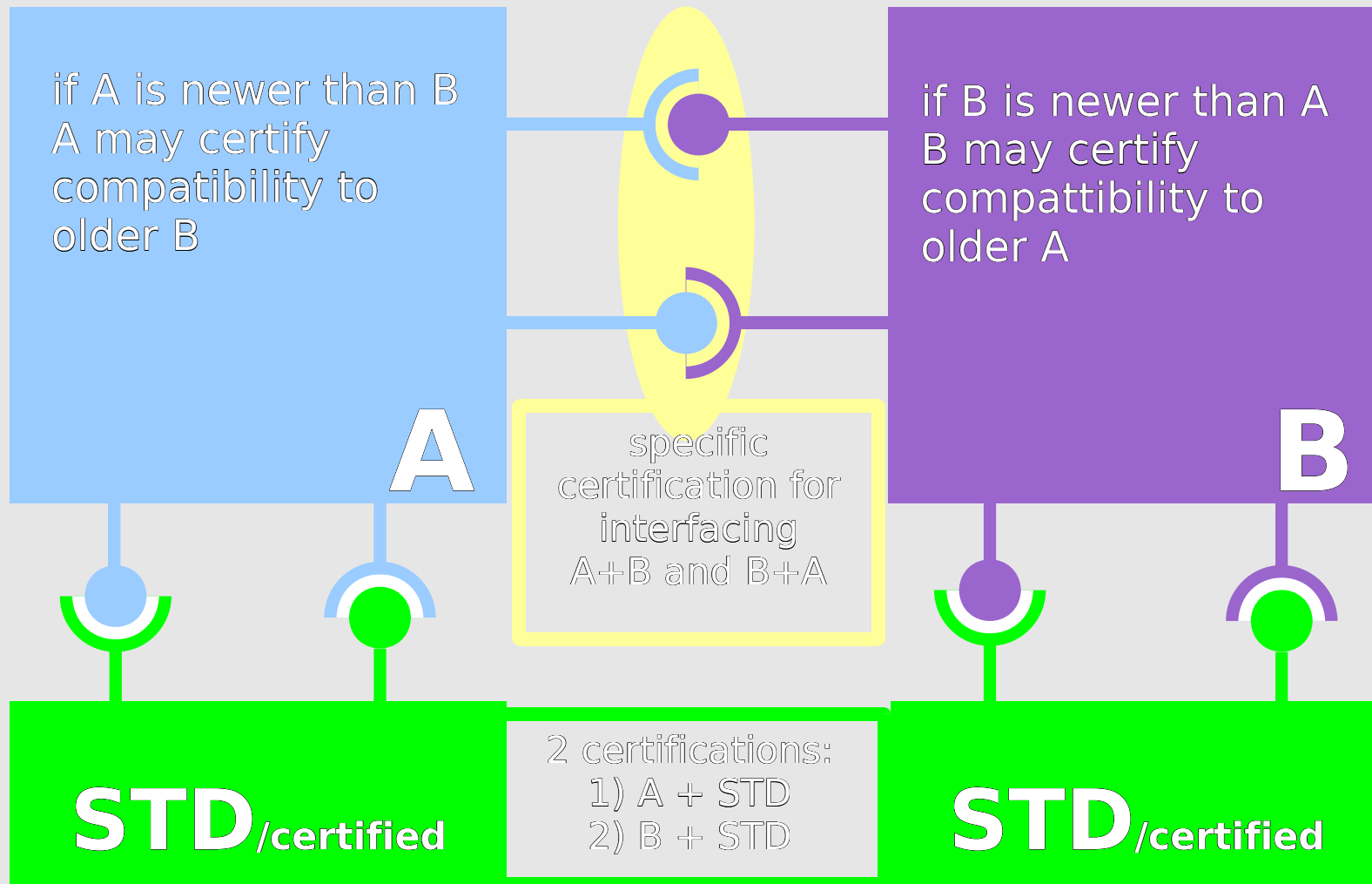
- ♦ **Example Redundant/Diverse Architecture**



- ◆ communicating safe systems

- ◆ systems covering different SIL

# ♦ safety case for compatibility



- ◆ **re-use modules**

- ◆ ... that had been part of a safe product

- ◆ **consider the proven-in-use approach**

- ◆ **SIL2**

- independent validation
- no need for evidence of systematic failure avoidance

- ◆ **... and then SIL4**

- 100 product-year long continuous logging
  - variety of operations
  - not a single near-safety critical event

- ◆ **you may persuade ISA and safety authority**