Efficient Safety Analysis through Combination of Methods

Bettina Buth

Fakultät TI, Department Informatik
HAW
Hamburg University of Applied Sciences

Bieleschweig 2007, Hamburg

Overview

- Motivation
  - Current Status
  - Efficient Safety Analysis
- An Example: Combining FTA and FMEA
  - Approach
  - Example Application
  - Results
- Conclusion
  - Experiences
  - Suggestion

Situation with Critical Systems to Date

- Increasing complexity of systems
- Integrated use of HW and SW in embedded systems
- High demands concerning safety and dependability
- Particularly: avionics - space - railways - automobile - medical technology
- Constraints: Cost effectiveness - Automation
- QA for all development phases
  Analysis - Simulation - Test specifically: Safety Analysis
- But with minimal budget

$\Rightarrow$ Effective QA planning required

Standards Perspective

- E.g. ECSS, CENELEC 50128, DO 178B
- QA as supporting process
- Applicable Methods
  - Static analysis - Dynamic analysis (Test)
  - Methods for error identification (reviews, inspection, debugging)
  - Mostly manual application (checklists)
  - Few tools
- Suggested Alternative: Formal Methods ?
  British MoD - NASA
- Little support for method selections
  e.g. CENELEC 50128 decision tables
  e.g. ECSS Dependability Handbook
- Tailoring (e.g. ECSS)
  - Drivers: SW engineering approach - Criticality - Customer requirements
  - Constraints: conformance to SW processes - adequacy for safety / dependability level

Strategic Planning of Analysis

The Idea

- Thorough approach not possible due to
  - Time constraints
  - Cost constraints
- Desirable: focus on critical components / failure scenarios
- Organize SA according to priorities
  - Systematic selection of analysis targets
  - Prioritised test approach
- Combine methods!
  May not be possible unless
  - Interdependencies of sub-components are clearly visible
  - Architecture is adequate (Modularity)

Here: Combining FTA and FMEA for SA of Software

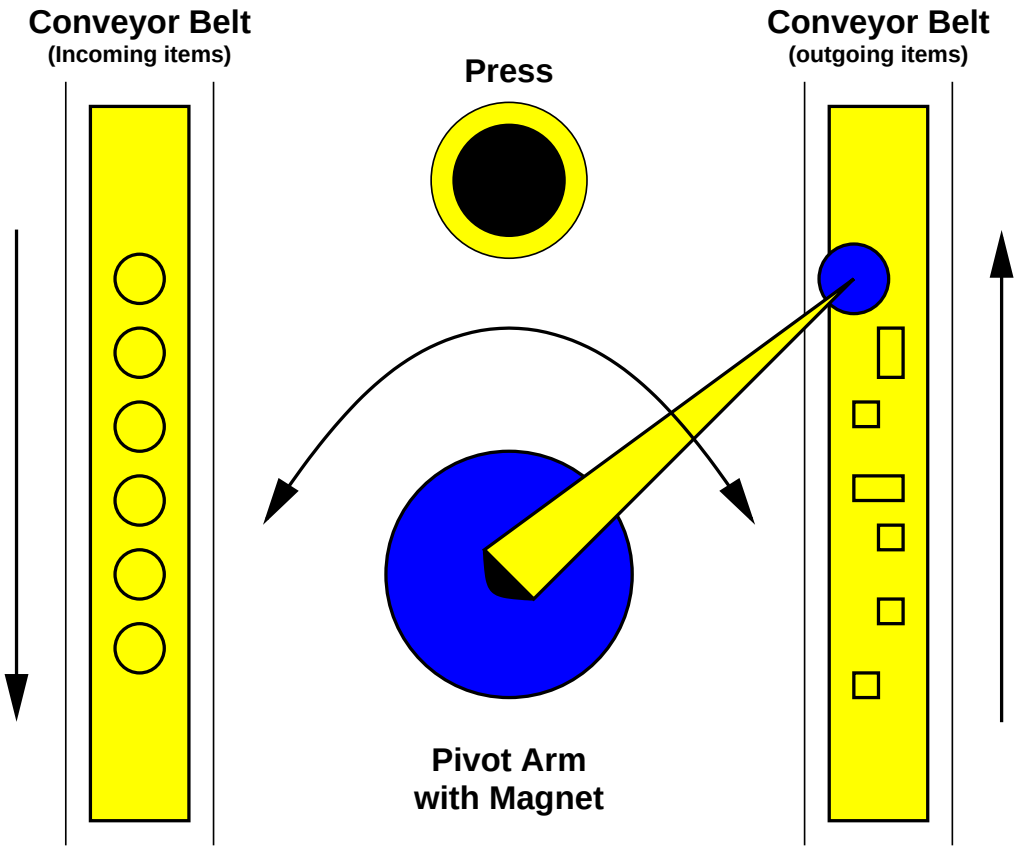
SW FTA and SW FMEA

- FTA
  - Top-Down Analysis
  - Starting from identified hazards
  - Investigation of potential causes (hierarchical approach, simple causal connectives)
  - calculation of minimal cut sets
  - extensions and tools available
- FMEA
  - Bottom-Up Analysis
  - Based on identified failure modes / categories
  - identification of error effects (local / global)
- FTA and FMEA for SW
  - not practical due to high complexity
  - no causal / temporal dependencies for traditional FTA
  - evaluation of FMEA depends on design knowledge
  - no guarantee for completeness

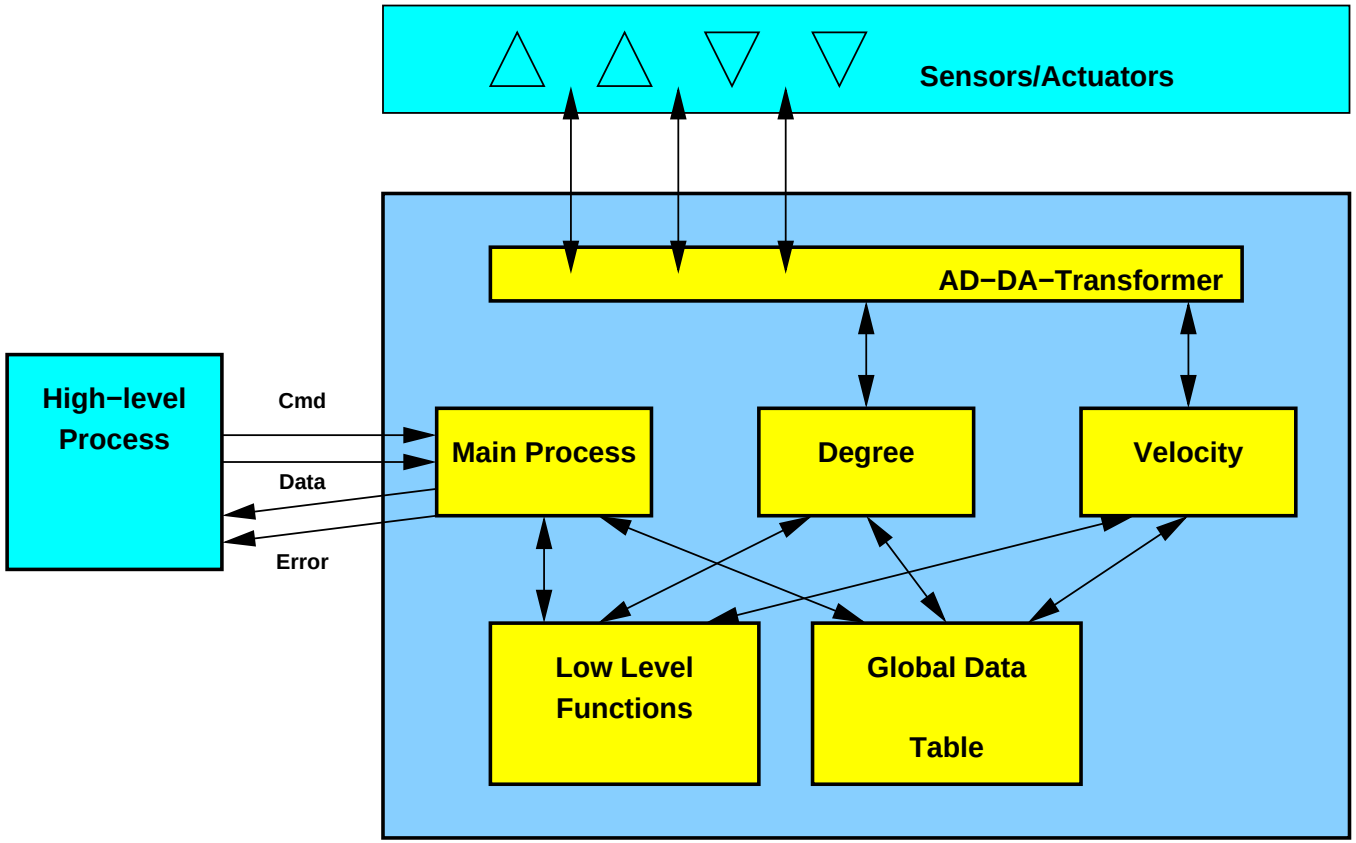
Combining SW FTA and SW FMEA - The Approach

- Objectives:
  - reduce overall effort
  - Provide input for test / verification / validation planning
- Top-Down and Bottom-Up
  - FTA down to level of components
    - E.g. refinement of system FT
    - Identification of critical components
  - FMEA up from level of components / functions / procedures
    - Starting from typical failure-modes
    - Identify their local and global effects
    - de facto a systematic code inspection
    - for data related and control flow related problems
    - fault categories depend on programming language
- Combine the results

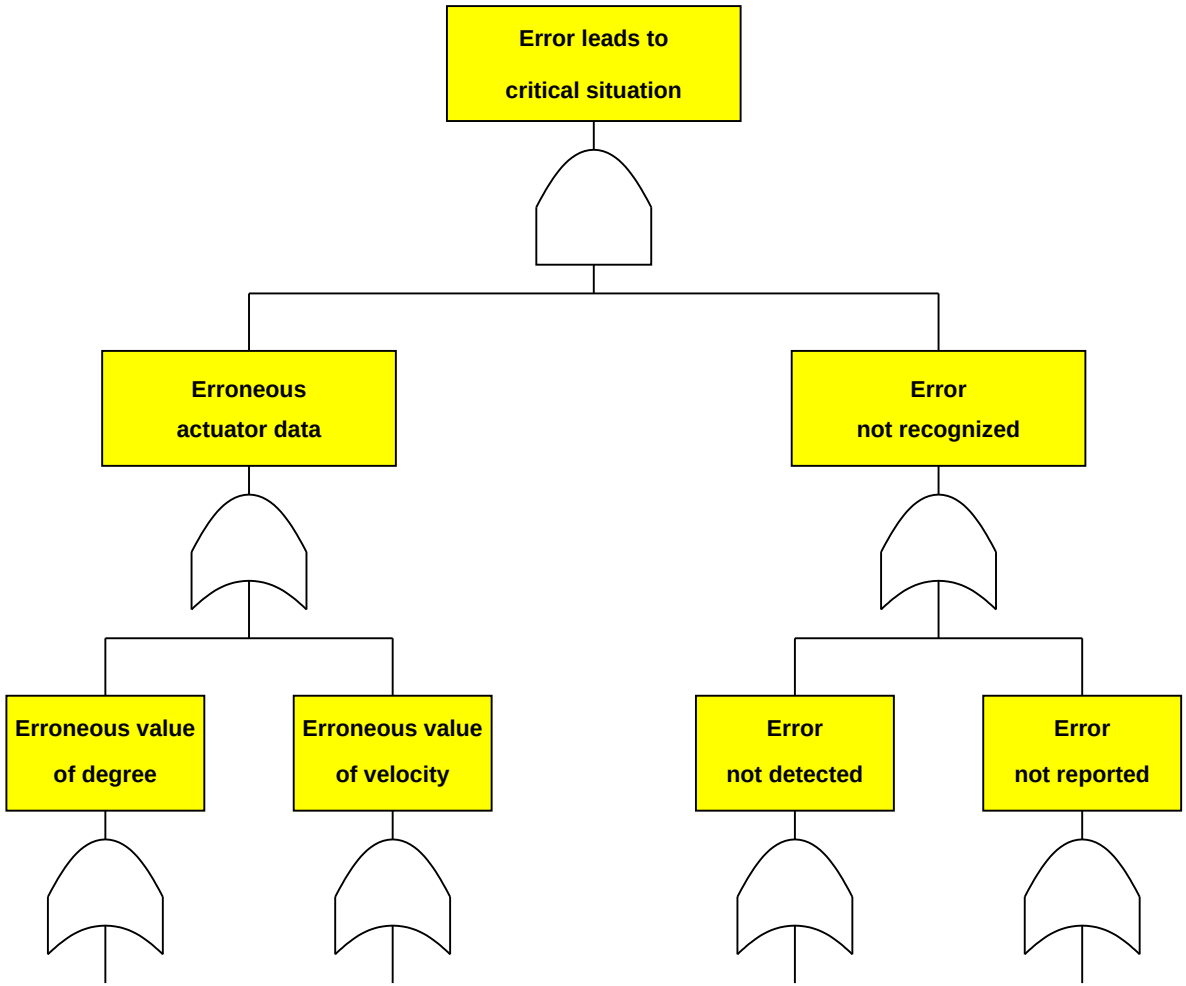
An Abstract Example - Pivot Arm



Example Pivot Arm - Architecture



Example Pivot Arm - FTA (1)



Example Pivot Arm - FTA (2)

Textual Representation

Erroneous actuator data lead to critical situation

- AND 1 Erroneous input data for actuator
  - OR 1.1 incorrect calculation of degree values
  - OR 1.2 Incorrect calculation of velocity values
- AND 2 Higher level process does not recognize error
  - OR 2.1 Error not recognized
    - OR 2.1.1 Error detection faulty
    - OR 2.1.2 No error detection performed
  - OR 2.2 Error not reported
    - OR 2.2.1 Wrong error class
    - OR 2.2.2 Error message overwritten

...

SW FMEA for Data-Related Faults

```
Speed (X,Y,Z) = {
    ... newSpeed = Speed_In;
    ... newSpeed = f(newSpeed, X, Y, Z);
    ... Return newSpeed;
}

Main = {... MotorSpeed = da_convert(Speed (current_arc, current_pos, final_pos)) ...}
```

Function Speed Data Related Failures						
Data Item	Failue Type	Description	Local Effects	Global Effects	Note	Id
Speed_In (global)	Absent	Used without init	Arbitrary value of newSpeed	Incorrect calculation MotorSpeed	!	1
	incorrect	Used in wrong way	incorrect value newSpeed	incorrect calculation MotorSpeed	-	2
	Wrong timing				NA	
	Duplicate				NA	
newSpeed (local)	Absent	Used without init			NA	
	incorrect	calculated in wrong way	incorrect value newSpeed, return	Incorrect Value MotorSpeed	✓	3
	Wrong timing				NA	
	Duplicate				NA	
X,Y, Z	Absent				NA	

Example: Using the Results

- FTA leaves control FMEA
  - Example: explicit investigation of error detection and reporting
  - Selection criteria for individual FMEA
- Error situations in FMEA are leaves of FTA
  - Example: function da_convert delivers wrong value
  - Feedback for completeness!
- FTA provides severity category for FMEA errors
  - Selection criteria for further analysis/tests
- FTA allows to correlate FMEA results

  Partially ...
- Positive experiences
  - NASA: on architectural design level for re-use
  - Safety analysis for tilting control in train systems

Benefits of the Approach

- FTA provides
  - Basis for design decisions (de-coupling of components)
  - Selection criteria for FMEA and other analysis / testing
  - Error scenarios for robustness testing
- FMEA provides
  - Feedback on completeness of FTA
  - Systematic approach to code inspections
  - Input for unit test steps (data and control flow)
- Combined results provide
  - Input for analysis of interdependencies (causal / temporal)
  - Justification for prioritisation of verification / validation / test
- Systematic approach from System down to SW Subsystems

Without compromising overall safety / dependability !

Additional Concerns

Analysis of Dependencies

- Additional Analysis of Error Propagation
  Possible Approaches
  - FTA down to detailed design - inpractical
  - dependency analysis using code slicing techniques
- Analysis of Causal / Temporal Dependencies
  Objective: exclude error scenarios
  - use simulation models
  - use Sequence Charts (UML/SysML)
  - use formal models, model checking

Suggestions

- HAZOP or System-FT as starting point
- SW-FTA: Identify critical components in SW architecture
  - Start from external failures sources
  - Map to local properties
- SW-FMEA: detailed analysis of SW components
- Combine results:
  - Analyse component interaction
  - Analyse dependencies between data related failures
- Use results as basis for
  - Robustness test
  - Test case selection on compoennt tests