Indicator-Based Inspections: A Risk-Oriented Quality Assurance Approach for Dependable Systems

Frank Elberzhager

Outline

- Motivation & Problem
- Idea
- Indicator-based Inspection with respect to safety
- Conclusion

Motivation

- Surrounded by evermore software and software-intensive systems
- Some Characteristics
  - Increasing functionality
  - Increasing complexity
  - Have to ensure certain non-functional requirements

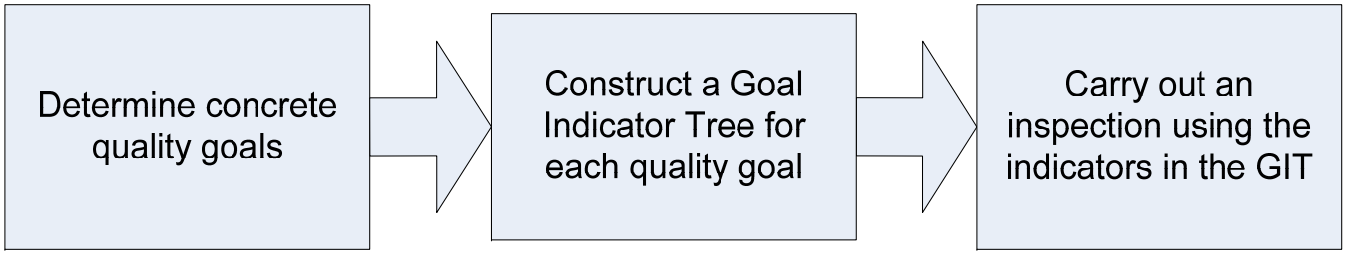
  → Software may cause harm to the environment (number of examples from every domain)
- Challenge: Perform suitable development and **quality assurance**
  - Number of different techniques exist

Motivation

- Some static quality assurance techniques to cover risk
  - Risk analysis (Safety)
    - FME(C)A: Identification of failure modes and assessment of criticality
    - FTA: Identification of causes
  - Reliability Block Diagrams
  - Attack Trees (Security)
  - Software Inspections

- Problems
  - Often too coarse-grained analysis
  - Little support to ensure certain quality properties
  - Many different techniques

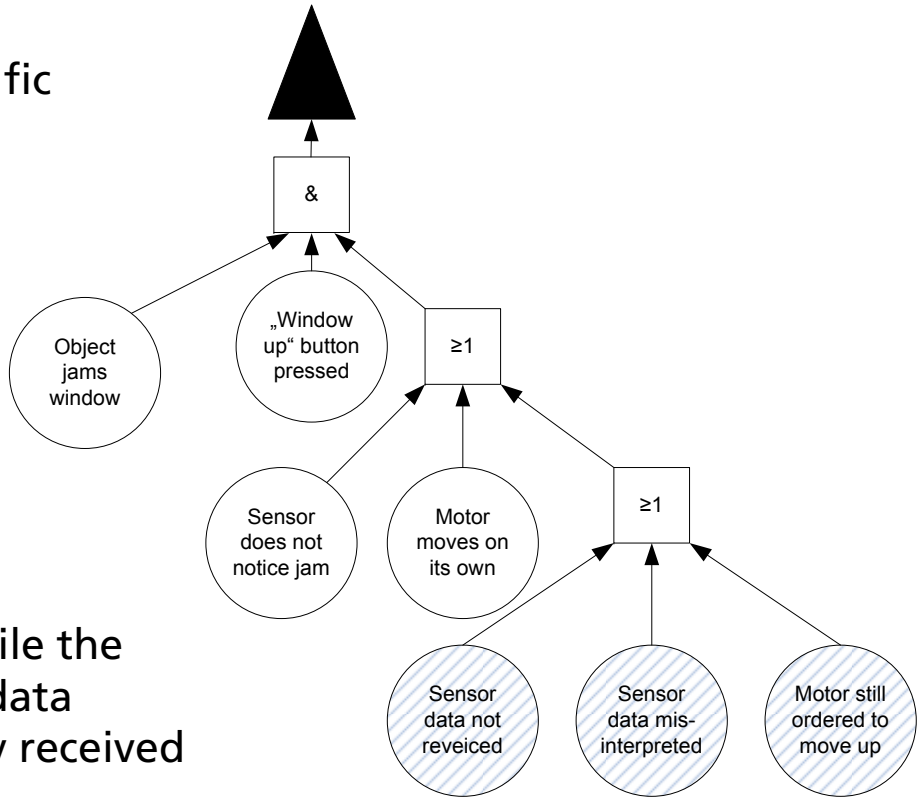
Idea



- Example: Control unit for an electrically-powered car window
  - Focus on demonstration that all **safety**-related non-acceptable risks have been reduced to an acceptable level

1. Determine concrete quality goals

- Quality goals are often application-specific
- For safety, FMECA and FTA can be used to identify safety hazards and causes (high-level analysis)

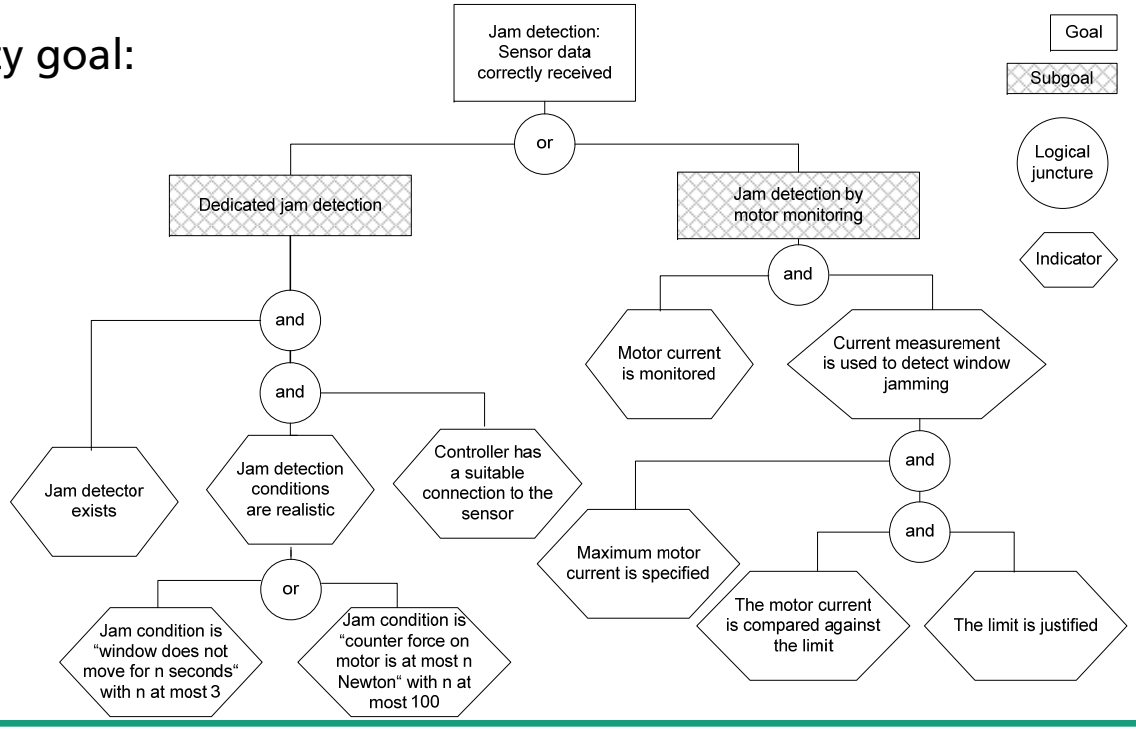


- FTA enables determination of minimal cut sets which can be used to derive quality goals, e.g.:
  - When an object jams the window while the “window-up” button is pressed, the data from the jam sensor must be correctly received

2. Construct a Goal-Indicator Tree for each quality goal

- Determination of possible correct implementations
- Identification of indicators that ensure a correct implementation

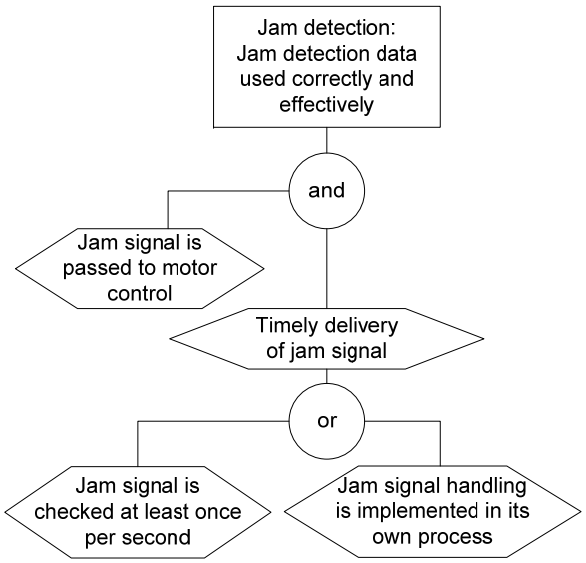
- First quality goal:

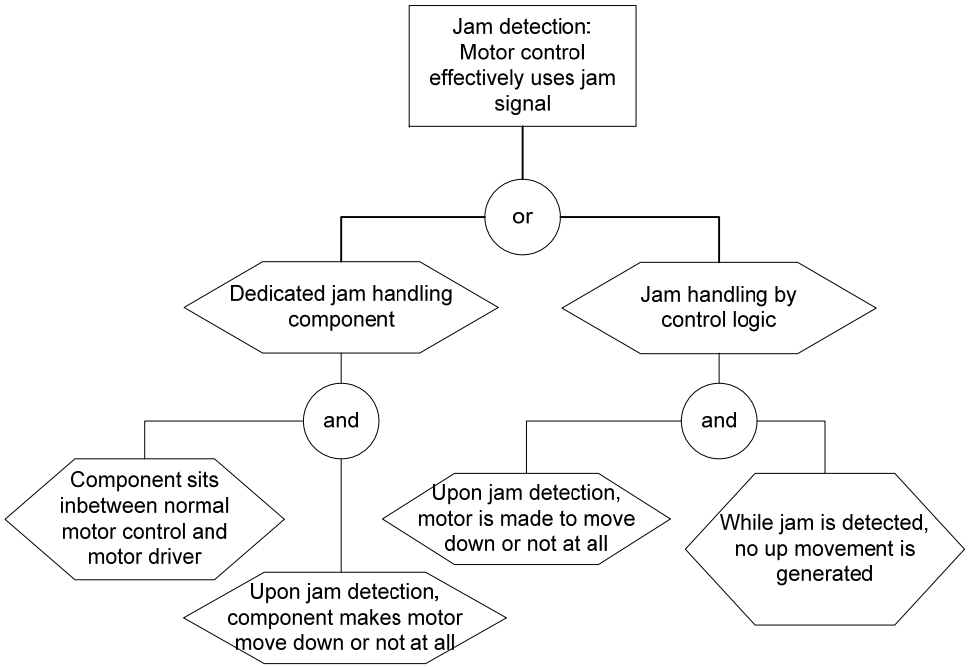


Fraunhofer IESE

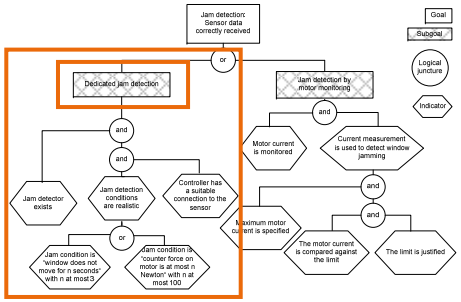
2. Construct a Goal-Indicator Tree for each quality goal

- Second and third quality goal:





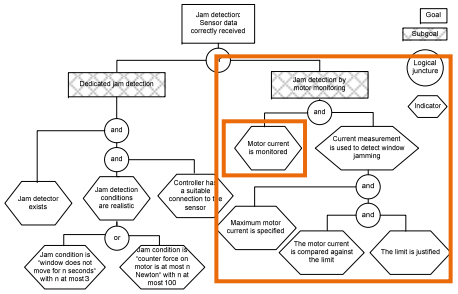
3. Carry out the inspection



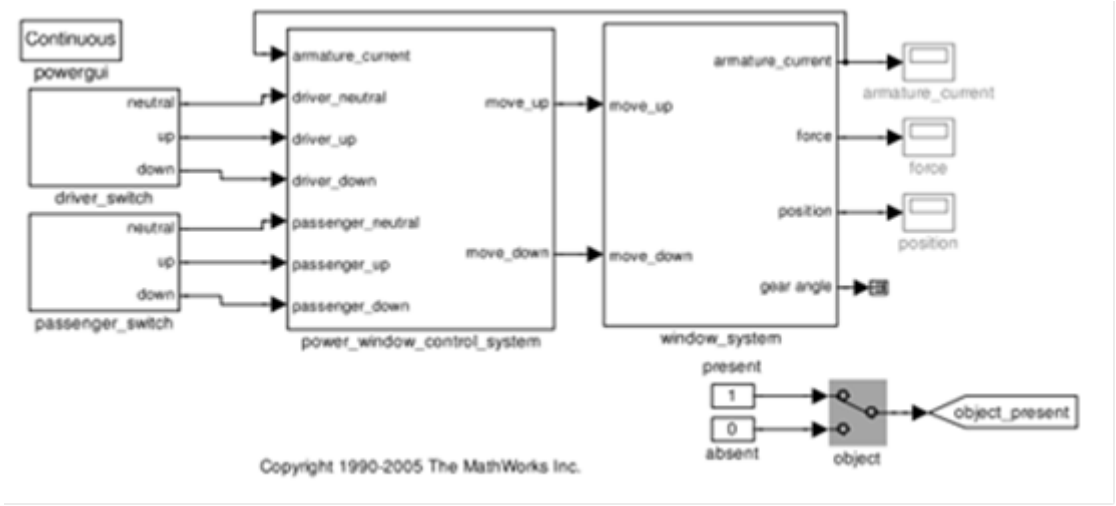
- Available artifacts: Requirements and Matlab Simulink model
- Use the GIT and check each indicator
  - Order: depth first, left to right

- First question: Does a dedicated piece of hardware exist for jam detection?
  - The requirements document describes another realization
  - → Question is answered with “no”
  - → Indicators below can be skipped

3. Carry out the inspection

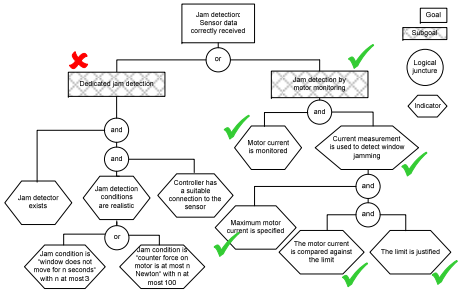


- Second subgoal: “Jam detection by motor monitored”
  - Question: Is the motor current monitored?
  - Answer 1: “…when a current is detected that is less than -2.5 A” (requirements)
  - Answer 2: Implementation shows connection from window system to the power window control system



Fraunhofer IESE

3. Carry out the inspection



- The remaining indicators are also fulfilled, either by a description in the requirements or by the realization in the Matlab Simulink model

- After performing the inspection, the fulfillment of the quality goal has to be checked:
  - Left subgoal “Dedicated jam detection” is not fulfilled
  - Right subgoal “Jam detection by motor current” is fulfilled
  - → Due to logical “or” connection, overall goal is fulfilled

- Remark: Enhancement of the reading support possible by derivation of a checklist which presents more detailed support for an inspector

Summary and Outlook

- Indicator-based inspection approach to ensure certain quality properties
  - Determination of quality goals
  - Construction of goal-indicator tree
  - Performing the inspection
- Benefit
  - Concrete and detailed support how to reduce risk
  - Improvement of quality
  - Knowledge transfer
- Outlook
  - Application to further quality properties
  - Details on construction
  - Usage of goal-indicator trees to select different quality assurance techniques

Thank you! Questions?

Frank Elberzhager (TAI)

+49 631 6800 2248

Frank.Elberzhager@iese.fraunhofer.de