Model-based design inspection based on traceability information models and design slicing

Shiva Nejati

April 15, 2015

About Me:

- Research Scientist at SnT Centre, U of Luxembourg since 2012
- Previous experience: Researcher at Simula Research Lab, Norway (2009-2012), PhD in Computer Science from U of Toronto (2008)

- Research Interests: Testing and verification of embedded real-time systems (aka cyber physical systems) using a combination of mete-heuristic techniques, formal methods, model-driven engineering techniques and empirical methods
- Research related to traceability: an industry-driven research project on certification and verification of safety-critical embedded systems using model-based software engineering techniques

Context of this Talk

Software Engineering for Safety-Critical Embedded Applications in the Maritime and Energy domain

Purpose:

- Supporting safety certification

Example application:

- Monitoring applications (fire and gas detection and emergency and process shutdown)

Software complexities

- Interaction with physical environment
- Real-time and concurrent
- Highly configurable
- Distributed and networked

Industry-driven project:

- ModelME! (2009-2011)

Problem Identification

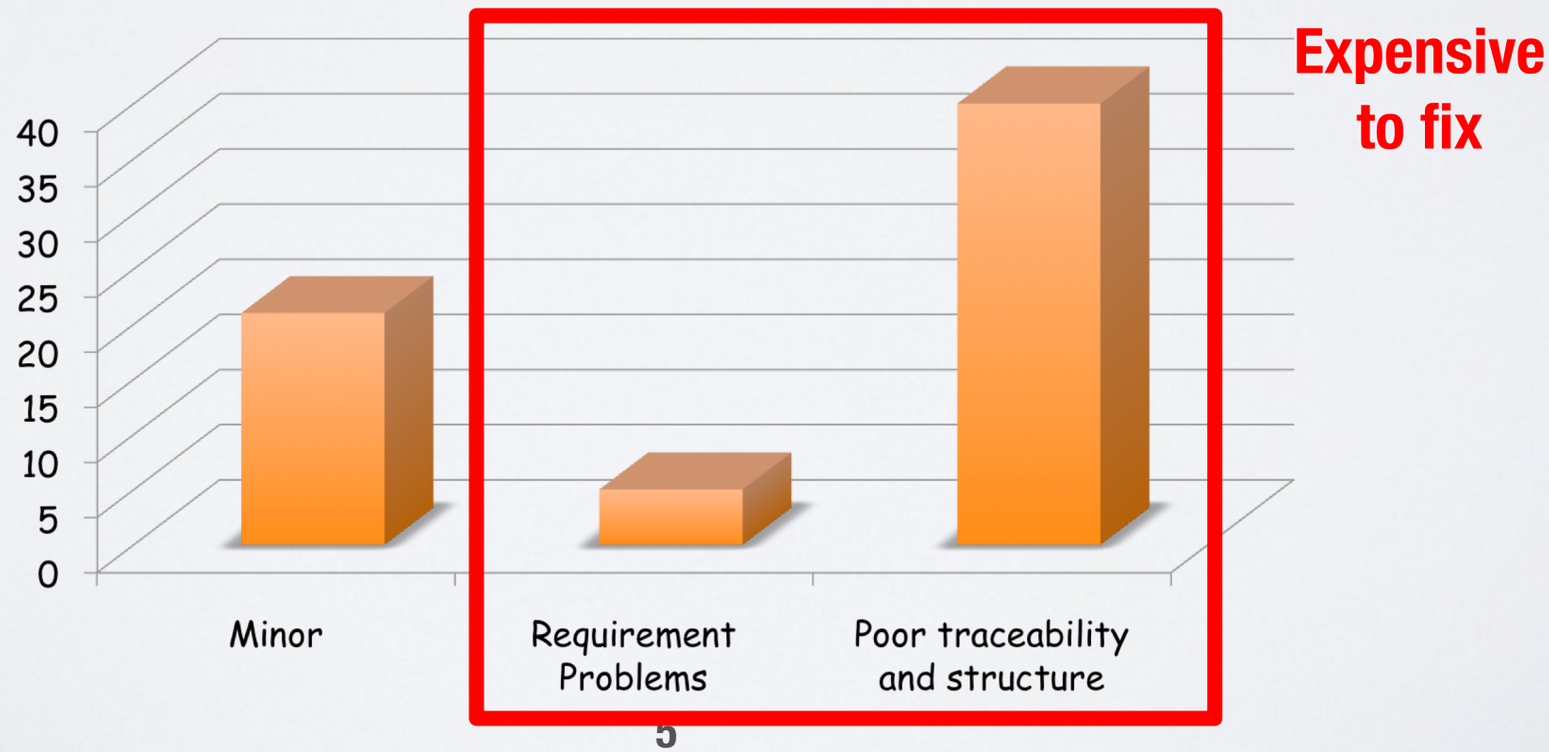
- Systematic study of the certification process and documents

1. Assessing and discussing software requirements, design/architecture and implementation documents (Several meetings taking place over 6 to 18 months)

2. Testing the devices in the field (A one week visit at the production site)

Observations from Safety Certification Projects

- Analyzed 66 distinct classification issues:
  - Issues collected through observing certification meetings at different suppliers of maritime and energy systems
- Meetings focused on requirements, architecture, and design documents



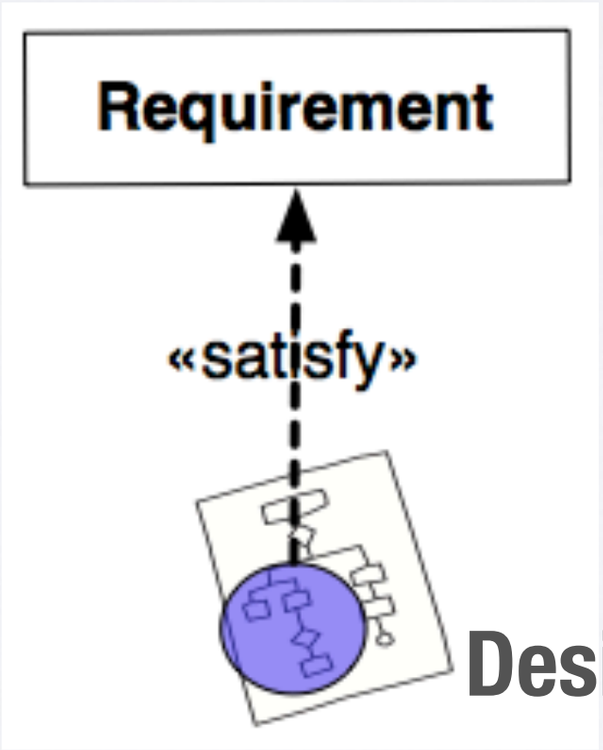
Effective Inspection of Embedded System Designs

- A key aspect of safety assessment is to inspect the design of safety-critical embedded systems and ensure that the design satisfies the system's safety requirements
- Design of embedded systems is often model-based
- Inspecting large design models is expensive and error-prone
- Question: How can we narrow the focus of design inspections to model fragments that are relevant to safety?

Research Objective

- Generate a sound and yet minimal **design slice** for a given safety requirement

- Slices constructed based on traceability links established between safety requirements and design



Design Slice

Research Approach

Traceability Methodology

to relate safety Requirements to design

Slicing Algorithm

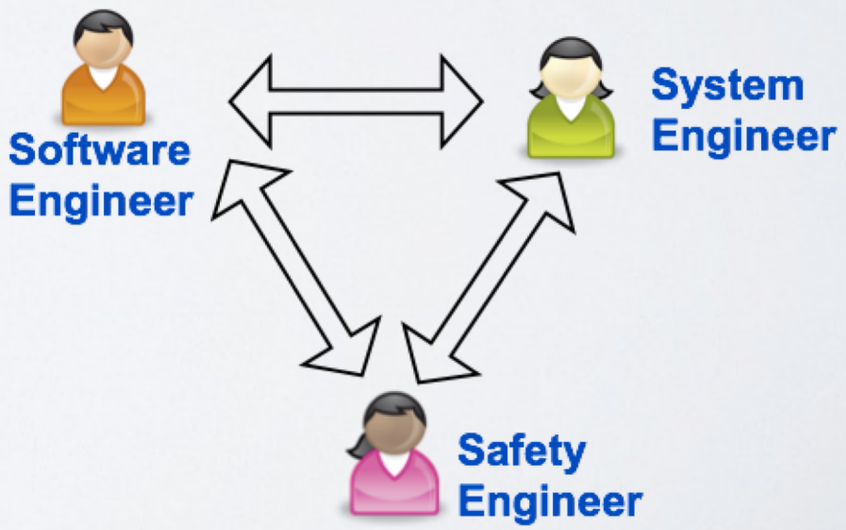
to extract a design slice relevant to a given safety requirement

Model Driven Engineering (MDE) is the enabler

(IST Journal 2012)

MDE Language

- System Modeling Language (SysML)
  - A subset of UML extended with system engineering diagrams
  - A standard for system engineering
  - A preliminary support for requirement analysis and built-in traceability mechanism



Is SysML enough?

- Do we have proper guidelines for establishing traceability links between requirements and design?
  - SysML is only a notation and needs a methodology
- Are the built-in SysML traceability links capable of addressing certification traceability issues?

Research Approach

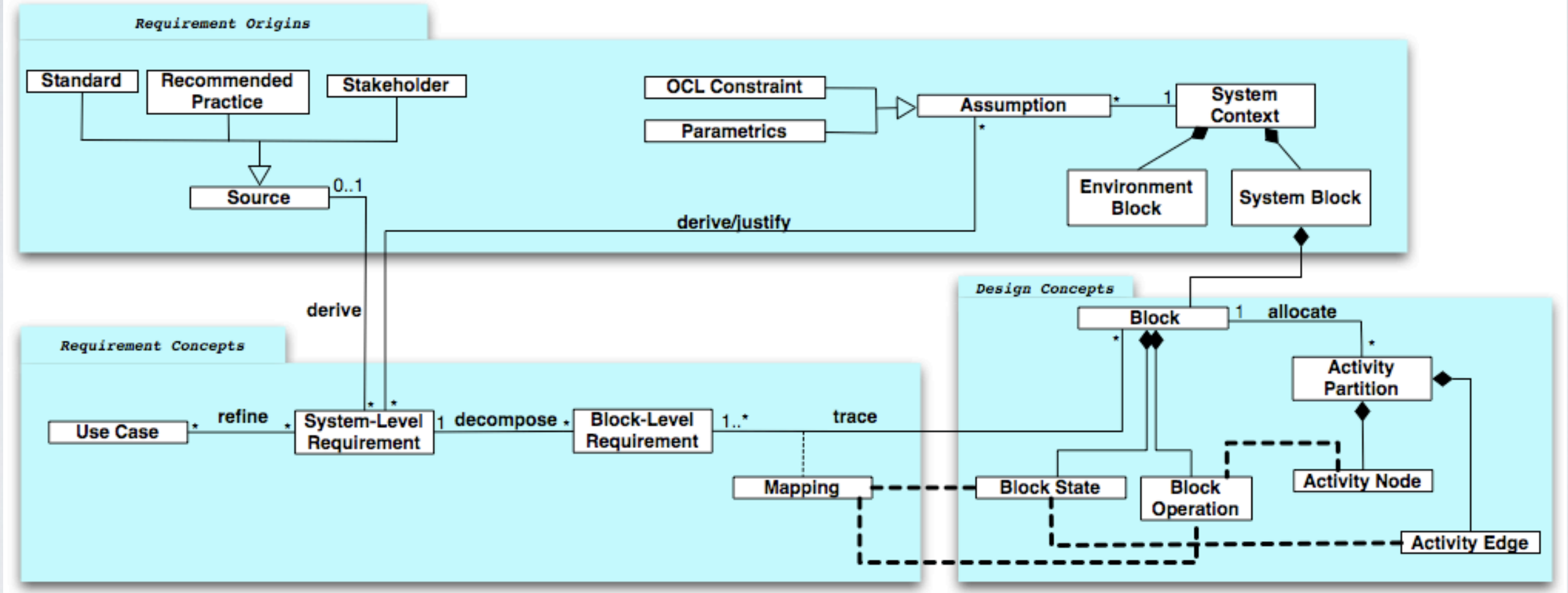
Traceability Methodology

to relate safety Requirements to design

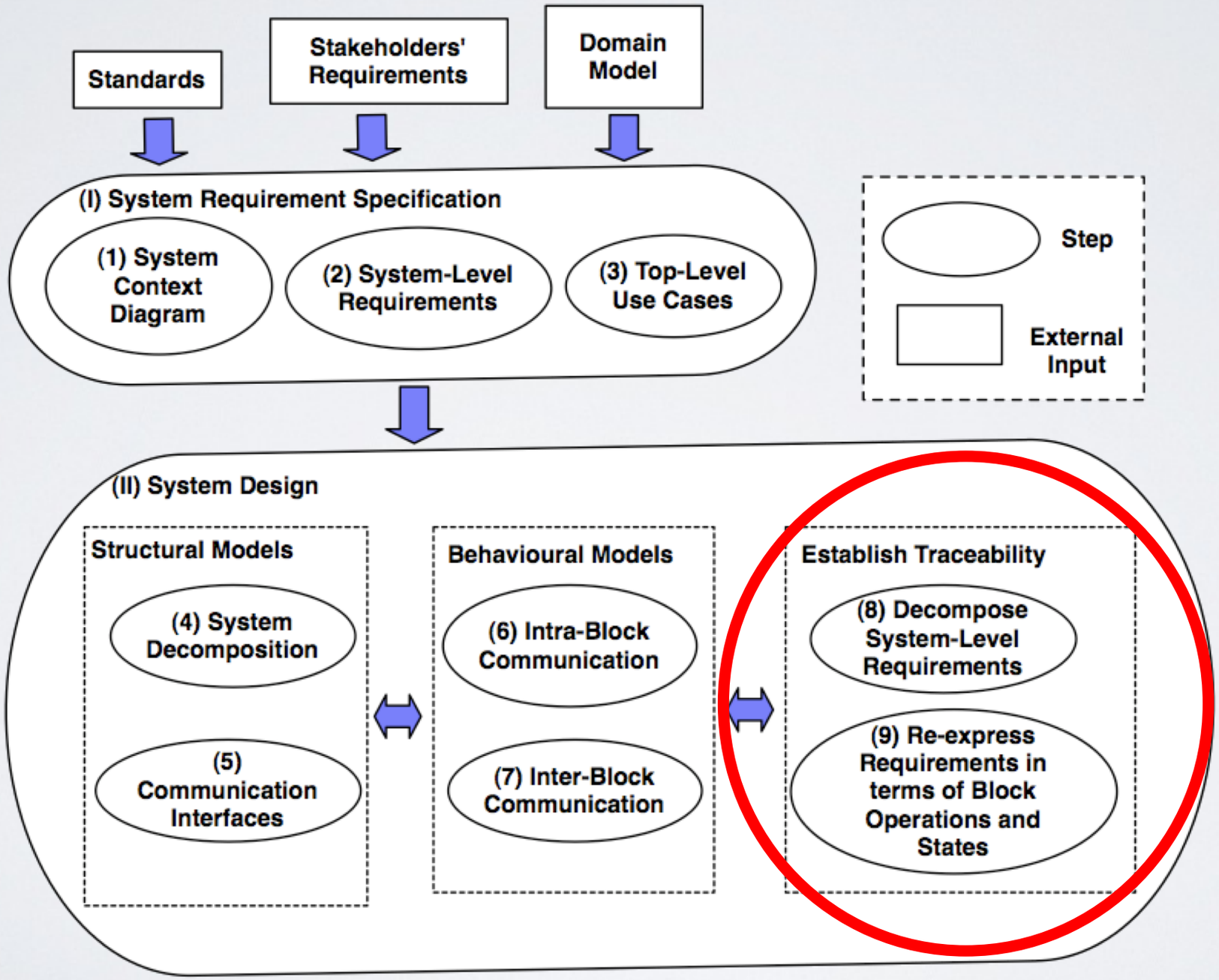
Slicing Algorithm

to extract a design slice relevant to a given safety requirement

Traceability Information Model



Our Methodology for Requirement to Design Traceability



Requirement to Design Traceability

Avoidance of falling metal blanks

decompose

The feed belt conveys a blank to table if the table is in load position

trace

trace

feedbelt.feed_table() causes "feed belt conveys a blank to table"

After executing table.go_load_position(), "table is in load position"

FeedBelt
-running:boolean -blankAtEnd:boolean
-initialize() -add_blank() -feed_table()

feedbelt ◀ **table**

1 **interact** **1**

Table
-pos:Table_Position -loaded:boolean
-initialize() -go_load_position() -go_unload_position()

Research Approach

Traceability Methodology

to relate safety Requirements to design

Slicing Algorithm

to extract a design slice relevant to a given safety requirement

Design Slicing

Avoidance of falling metal blanks

decompose

The feed belt conveys a blank to table if the table is in load position

feedbelt.feed_table() causes "feed belt conveys a blank to table"

trace

trace

After executing table.go_load_position(), "table is in load position"

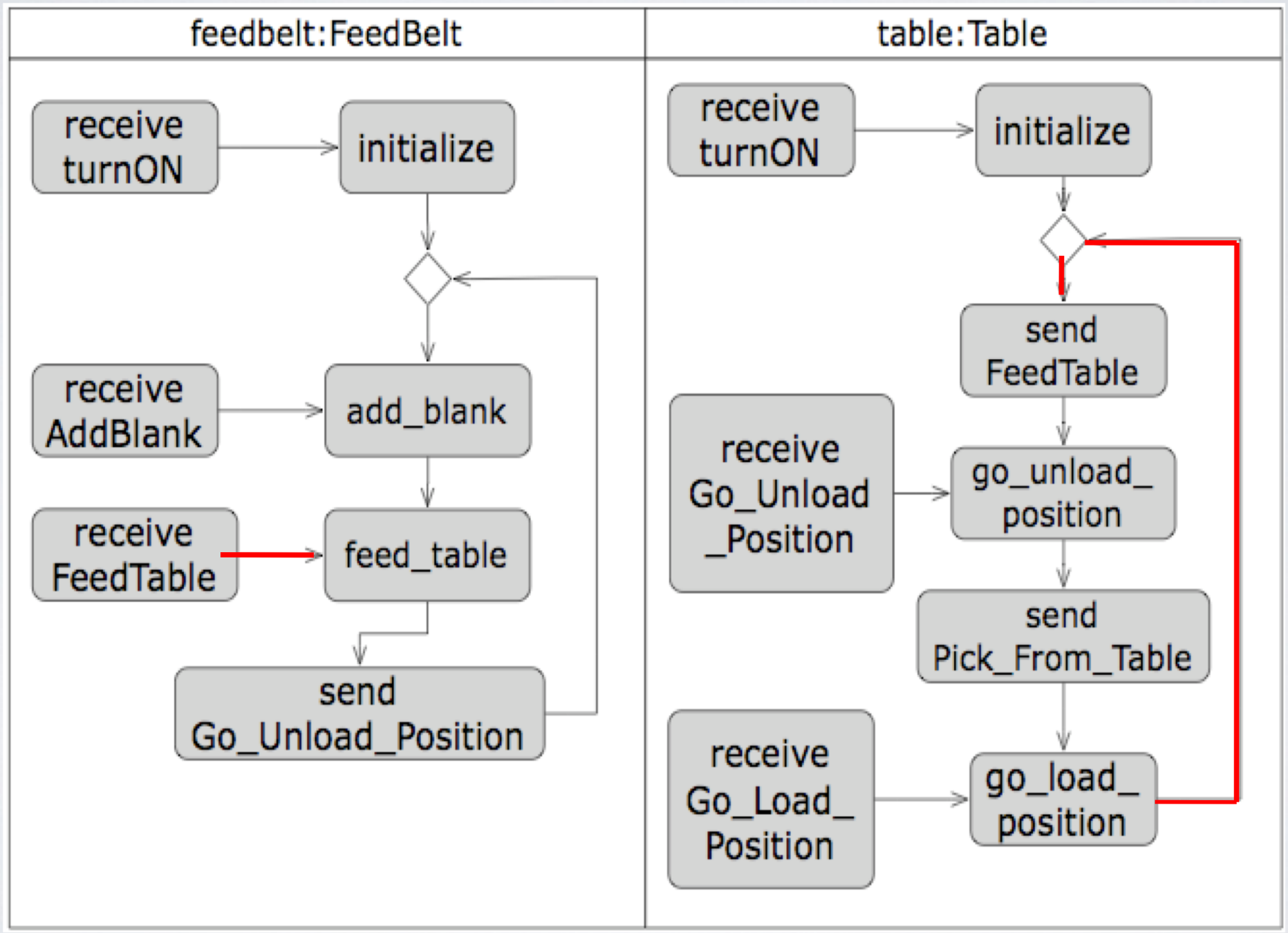
FeedBelt
-running:boolean -blankAtEnd:boolean
-initialize() -add_blank() -feed_table()

feedbelt ◀ **table**

1 **interact** **1**

Table
-pos:Table_Position -loaded:boolean
-initialize() -go_load_position() -go_unload_position()

Original Diagrams

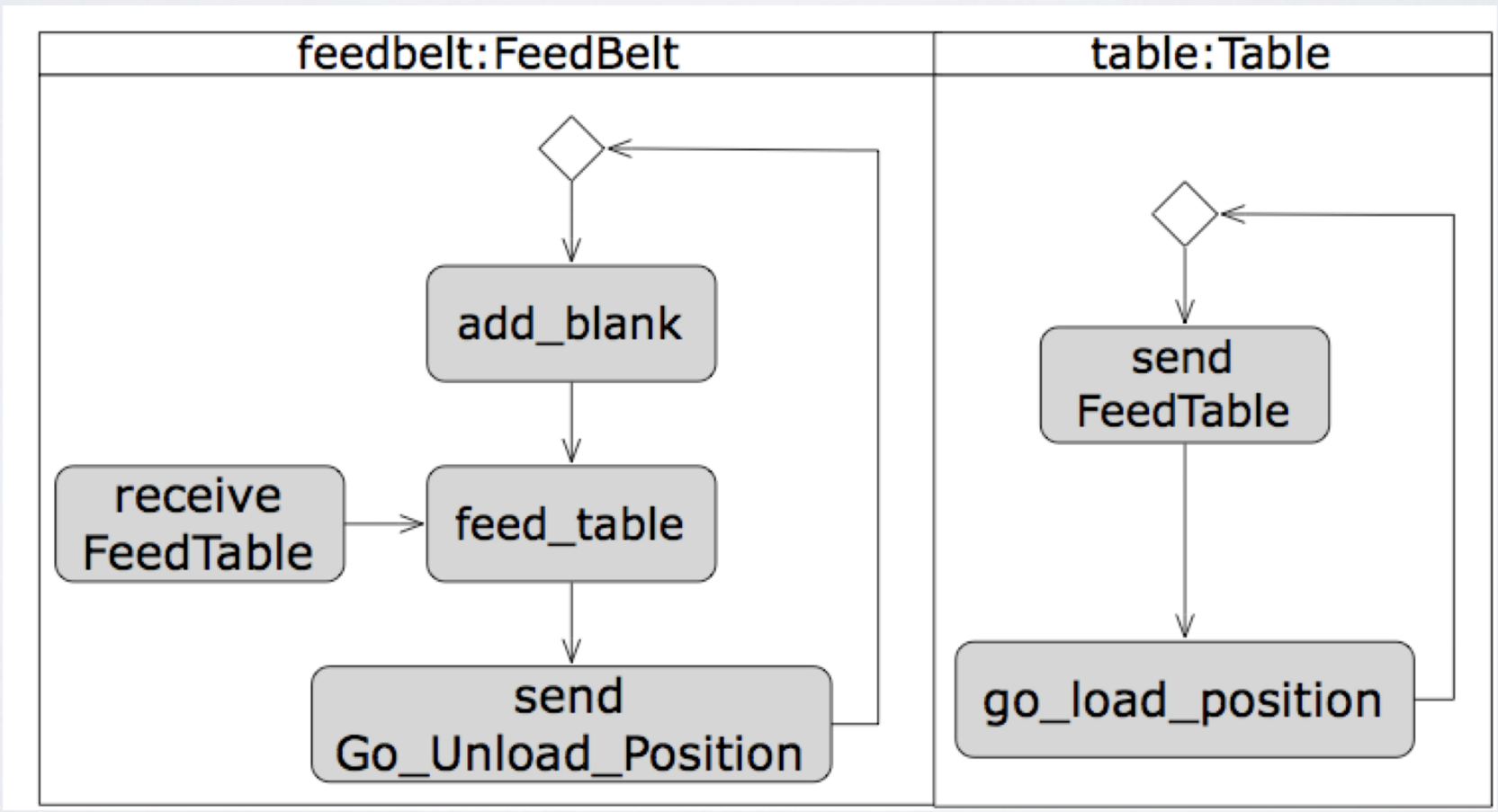


Slices

Avoidance of falling metal blanks

decompose

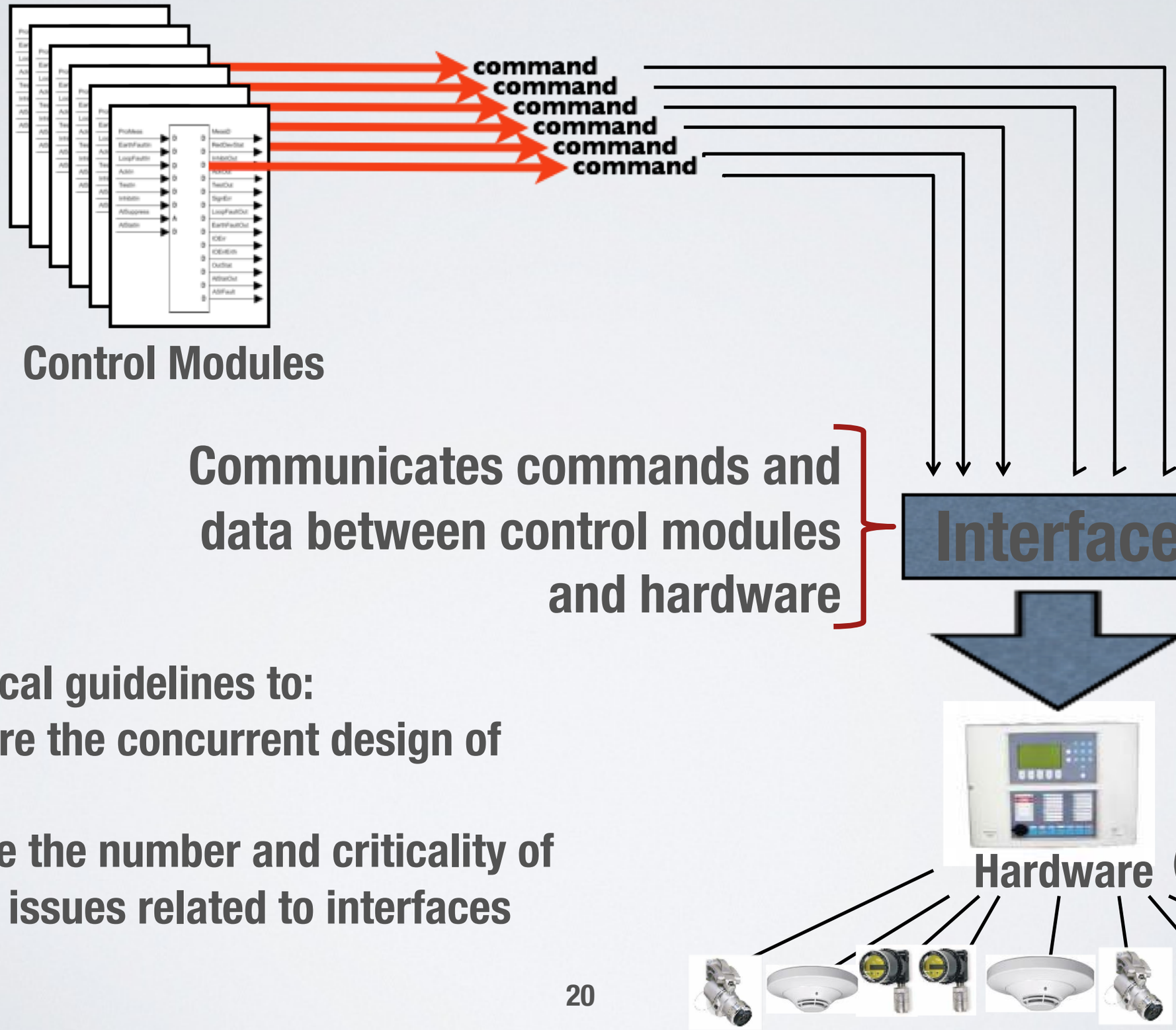
The feed belt conveys a blank to table if the table is in load position



Slicing Algorithm

- A slicing algorithm that for the given traceability links generates block and activity diagram slices
- Properties
  - If a requirement holds over a design slice, it should also hold over the original design (soundness)
    - Proven analytically (formal proof)
  - If a requirement holds over the original design, then the design slice created for that requirement should conclusively satisfy that requirement (completeness)
    - Evaluated empirically (Case studies and experiments)

Case Study: SW/HW Interfaces



Goal: Practical guidelines to:

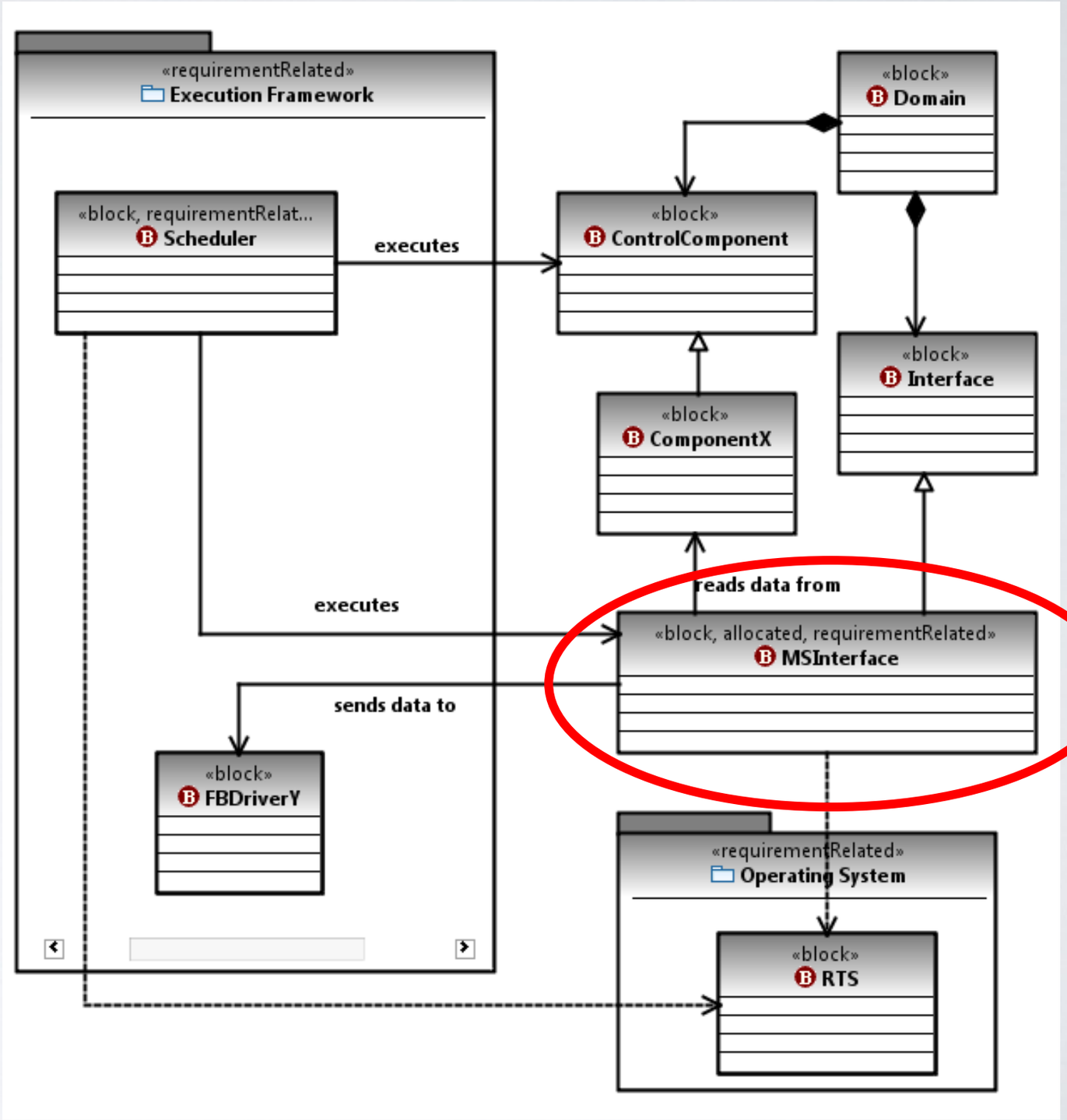
(1) Capture the concurrent design of interfaces

(2) Reduce the number and criticality of certification issues related to interfaces

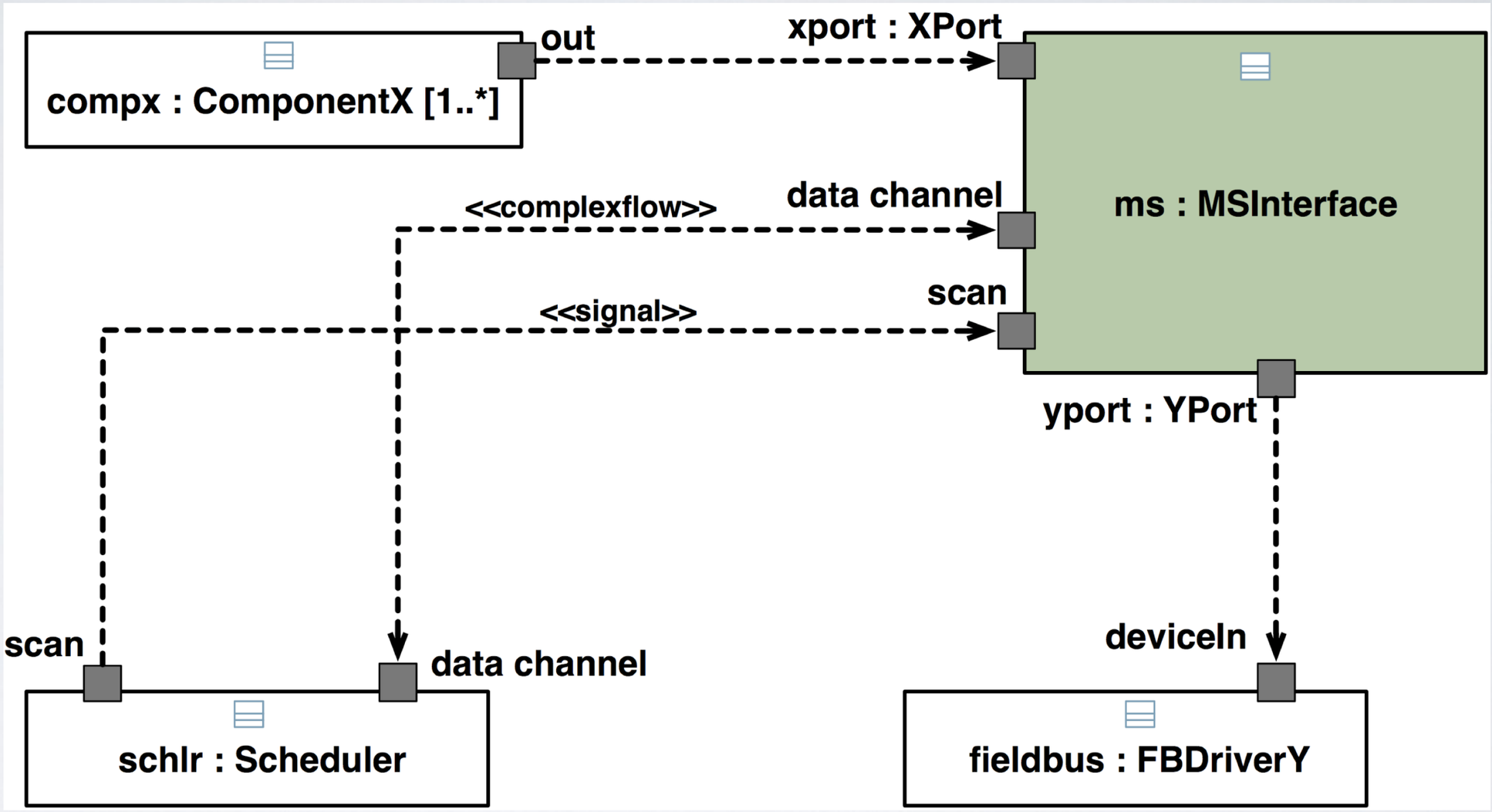
(HASE 2011)

Context Diagram (Case Study)

Usage in certification: showing the context in which the interface operates



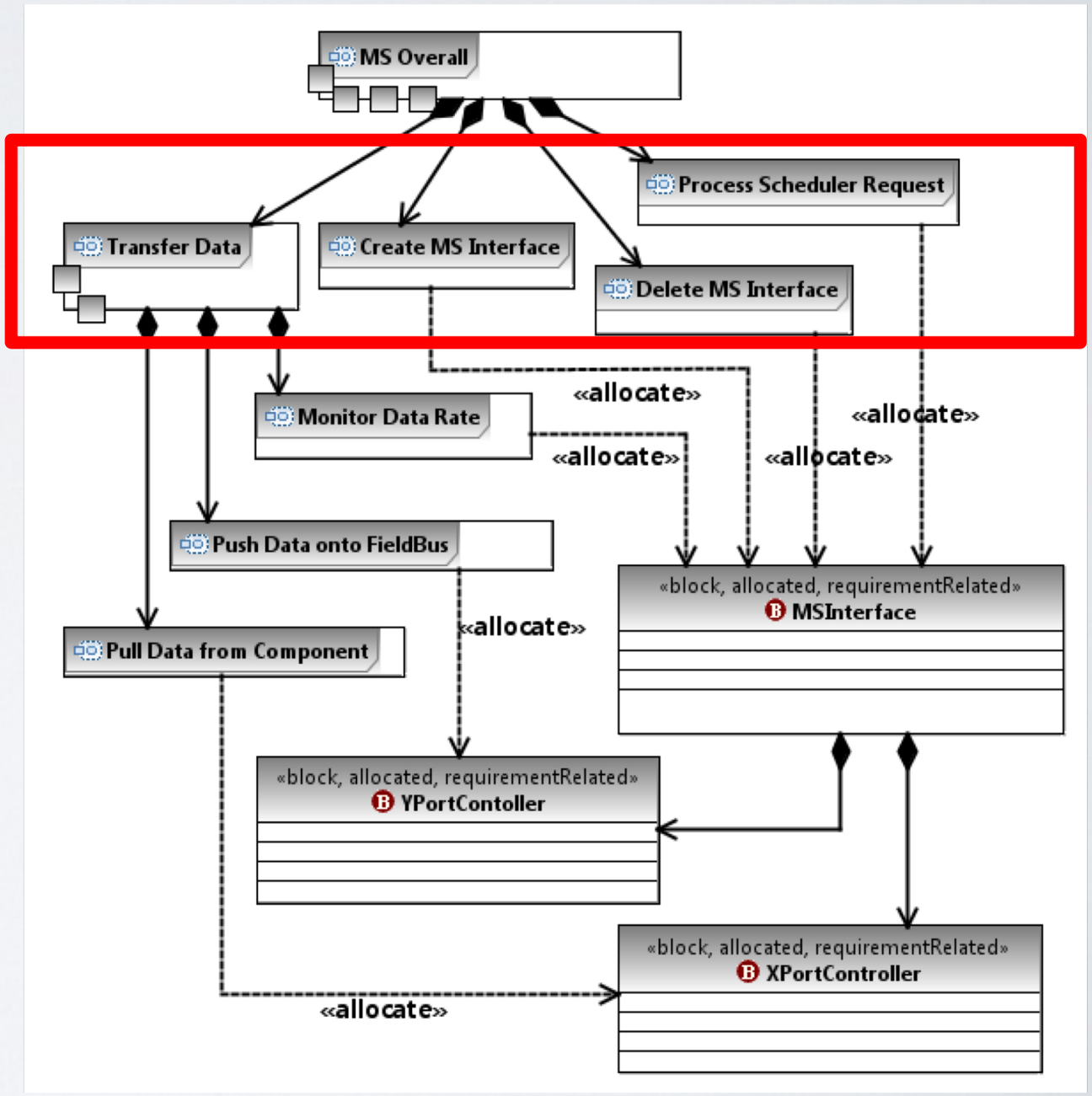
Architecture Diagram (Case Study)



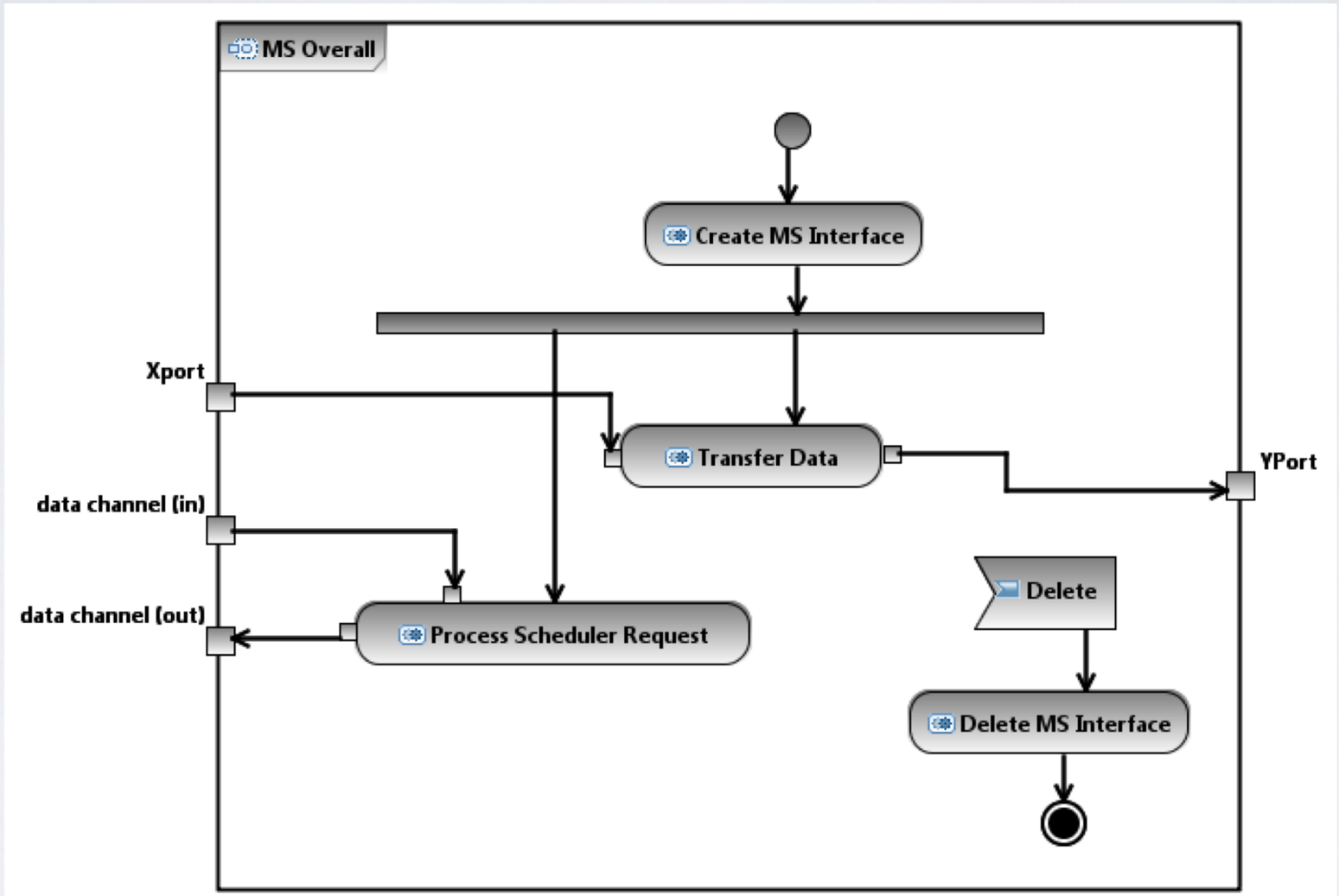
Usage in certification: showing the communication ports, signals, multiplicities for interacting components

Activity Decomposition (Case Study)

Usage in certification: maintaining traceability between high-level activities and blocks



Internal Behaviour (Case Study)

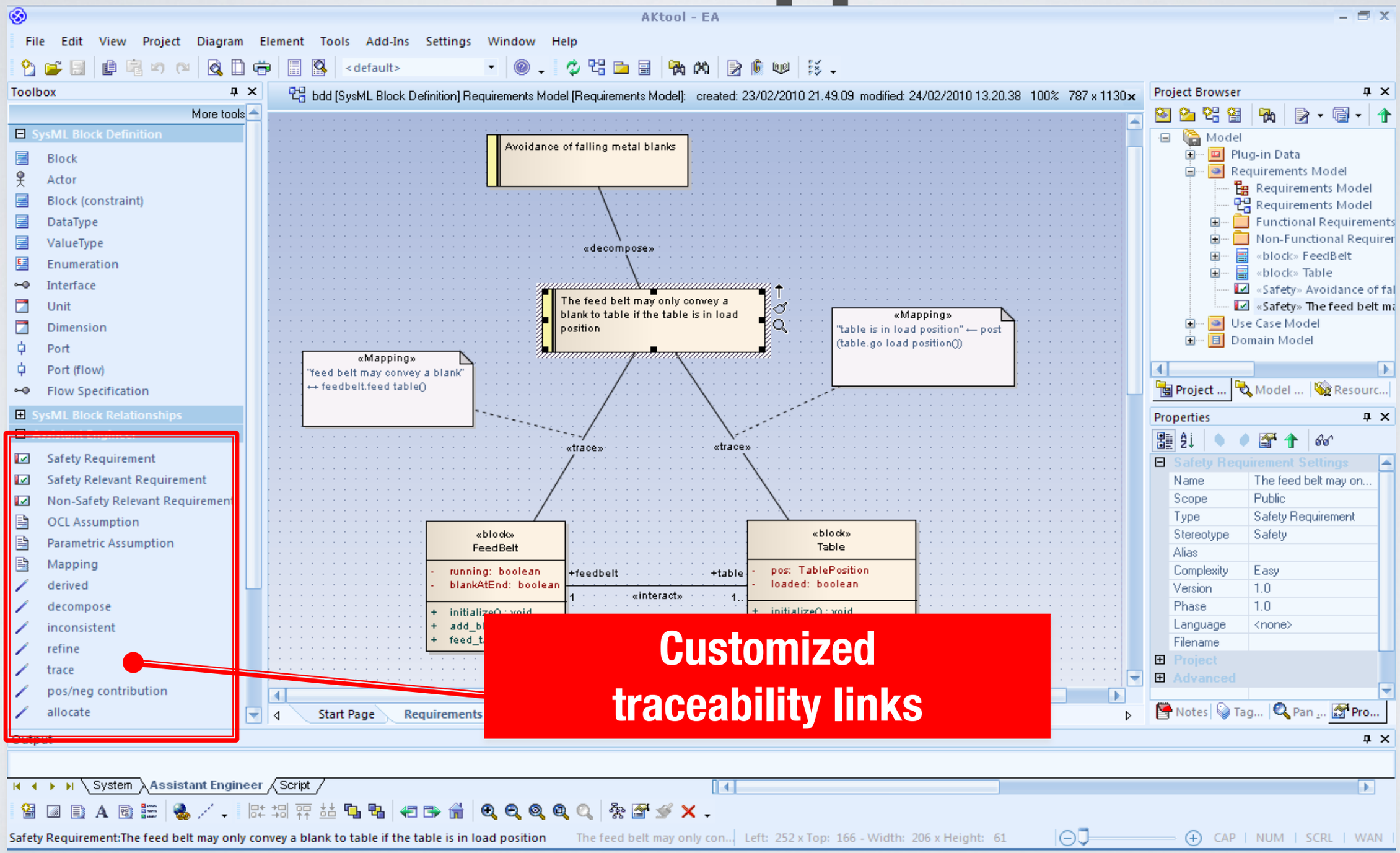


Usage in certification: systematic expression of behavioral design at different levels of abstraction

Results

- Created design models with traceability to requirements
    - One context diagram (BDD), One architecture diagram (IBD), One detailed structure diagram (BDD), One activity decomposition diagram (BDD), One overall activity diagram, 19 detailed activity diagrams
    - Created 65 traceability links for 30 safety-relevant requirements
    - Modeling effort was approximately 40 person-hours
- Model Slicing
    - Extracted 34 block slices and 31 activity slices
    - Slicing reduced the number of block operations by 70% and the number of activity nodes by 50%

Tool Support



(FSE 2011, Tool)

Controlled Experiment for evaluation of our approach

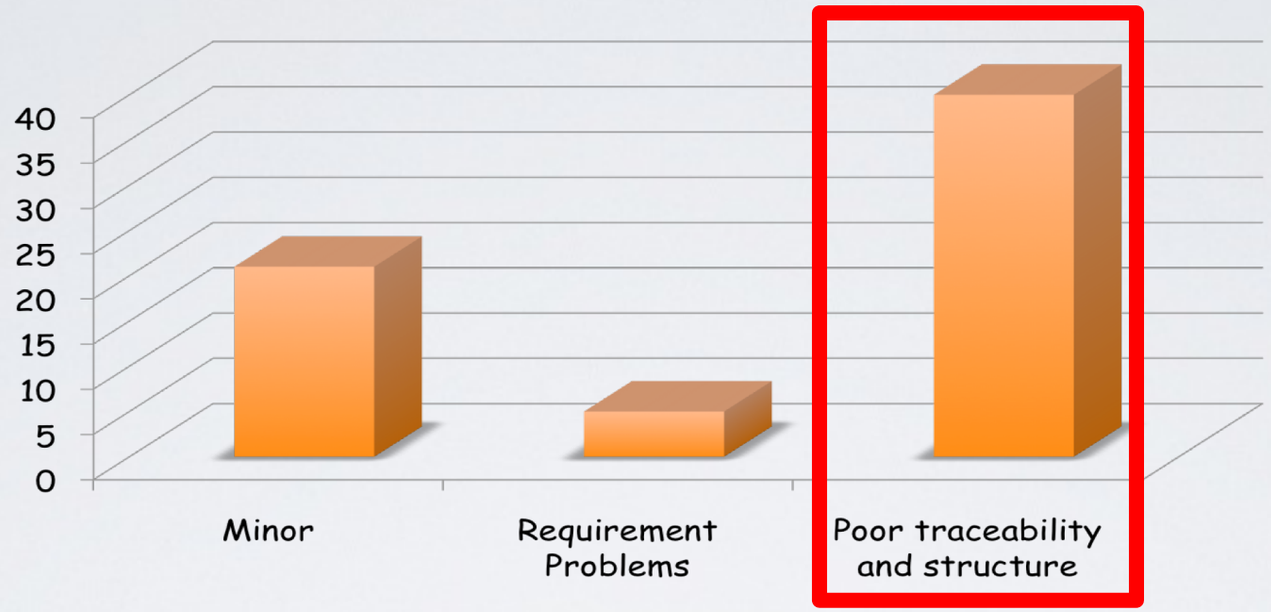
- conducted in a laboratory setting with master students
- Overall design
  - Seeded faults into the design
    - Incorrect behaviour and structure
  - Divided the subjects into two groups
    - One group gets the design without slices
    - One group gets the design plus the relevant slices

Experiment Results

- Main question: Do slices improve the identification of design faults?

- Slices show strong benefits in terms of:
  - Increasing the correctness of inspection decisions
  - Decreasing the proportion of uncertain decisions
  - Reducing the effort of inspections

Summary



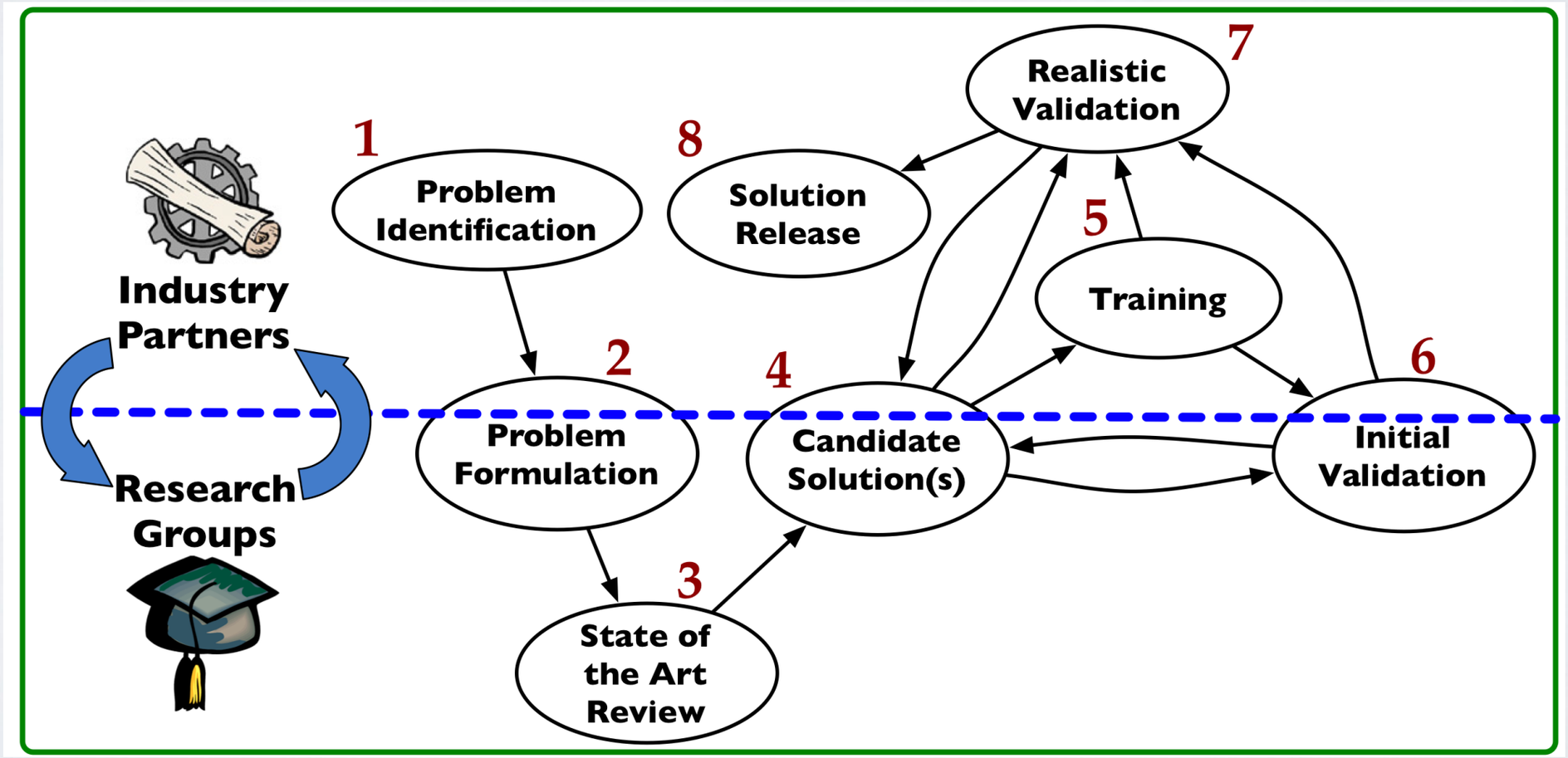
- Our methodology avoids large majority of traceability and structuring issues by:
  - Prescribing what kind of diagrams to create
  - Providing heuristics about the level of details in diagrams
  - Giving guidelines for creating traceability links

Acknowledgement

- Lionel Briand
- Mehrdad Sabetzadeh
- Davide Falessi
- Antonio Messina
- Anne-Heidi Evensen Mills
- Thierry Coq

Industry-Driven Project

- Mode of collaboration with industry
  - A refinement of the collaboration model by (Gorschek et al 2006)



Traceability in Certification

- Traceability mandated by many certification standards
  - IEC 61508, DO-178B, and the forthcoming Do-178C and ISO 26262
- Main problems observed:
  - Traceability links were either missing or not easily identifiable due to poor structuring
  - Most pressing issues concerned the link between safety requirements and software design
    - Does my design satisfy the given safety requirements?