Three Generations of Steam Boiler Models in SCCharts

Reinhard von Hanxleden, Alexander Schulz-Rosengarten, Jette Petzold

{rvh,als,jep}@informatik.uni-kiel.de
Kiel University

The Steam Boiler Specification of 1994

Steam-boiler control specification problem

Jean-Raymond Abrial

August 10, 1994

Abstract

The following specification problem is suggested to the participants of the Dagstuhl Meeting *Methods for Semantics and Specification*, organized jointly with Egon Börger (Pisa) and Hans Langmaack (Kiel) for the week from June 4-9, 1995.

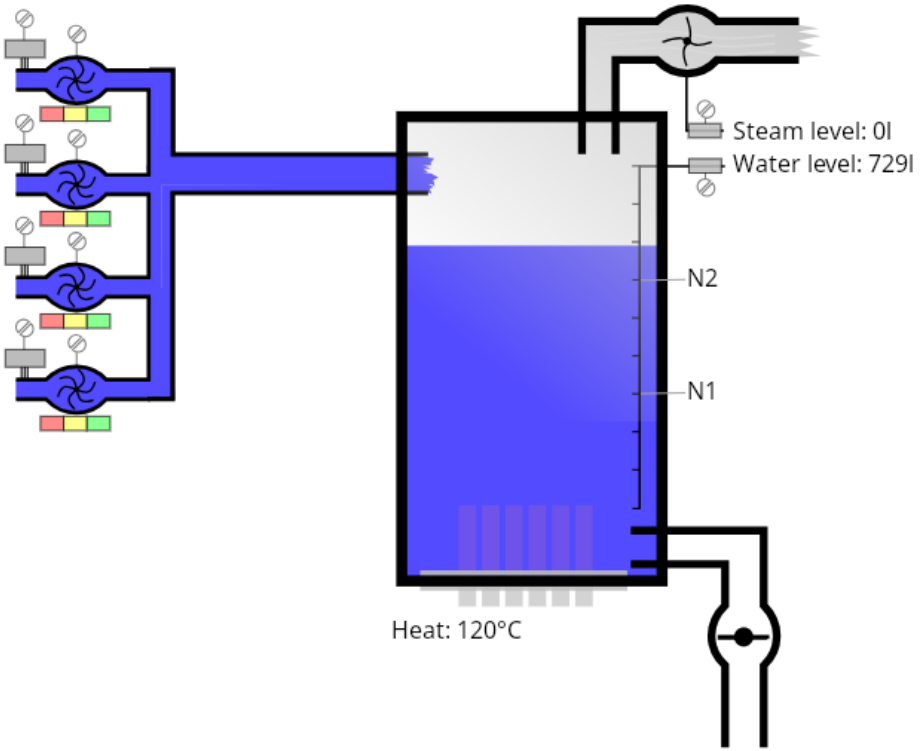
1 Introduction

This text constitutes an informal specification of a program which serves to control the level of water in a steam-boiler. It is important that the program works correctly because the quantity of water present when the steam-boiler is working has to be neither too low nor to high; otherwise the steam-boiler or the turbine sitting in front of it might be seriously affected.

The proposed specification is derived from an original text that has been written by LtCol. J.C. Bauer for the Institute for Risk Research of the University of Waterloo, Ontario, Canada. The original text has been submitted as a competition problem to be solved by the participants of the International Software Safety Symposium organized by the Institute for Risk Research. It

J.R.Abrial, JR. (1996).
Steam-boiler control specification problem
In: Formal Methods for Industrial Applications

Basic Setup



- 4x Pumps
  + throughput monitoring
- Valve
- Steam sensor
- Water sensor

Springer LNCS Special of 1996

Jean-Raymond Abrial Egon Börger
Hans Langmaack (Eds.)

Formal Methods for Industrial Applications

Specifying and Programming the Steam Boiler Control

Springer

Table of Contents

Verification using Temporal Logic

SPIN

WL: $\Box((NORMAL \vee DEGRADED) \Rightarrow ((WaterLevel > N1) \wedge (WaterLevel < N2)))$

IM : $\Box((INITIALIZE) \Rightarrow \Diamond((WaterLevel > N1) \wedge (WaterLevel < N2)))$

PF : $Fail[n] \Rightarrow ((Unused[n]) \; W(Repaired[n]))$ n : pump indice

NM : $\Box(NORMAL \Rightarrow NoFail)$

PR : $\Box((NoFail[n] \wedge RcvOrder[n]) \Rightarrow \Diamond(ExecOrder[n] \vee Fail[n]))$

PL : $\Box((\neg EMERGENCY \wedge \neg INIT) \Rightarrow (WaterLevel \neq PreviousWaterLevel))$

SF: $\Box((SteamFail \wedge \neg WaterFailure) \Rightarrow (DEGRADED))$

WF: $\Box(WaterFail \Rightarrow (RESCUE))$

Duval, G., Cattel, T. (1996). ***Specifying and verifying the Steam Boiler Problem with SPIN.*** In: Formal Methods for Industrial Applications

TRIO

pumpDiagnosis:

$$pb \leftrightarrow \left(\begin{array}{c} UpToNow(pb) \wedge \neg pr \vee \\ expectedOpen \wedge ps(closed) \vee \\ \neg expectedOpen \wedge ps(open) \vee \\ \left(\begin{array}{c} UpToNow(\neg pcb) \vee \\ pcr \end{array} \right) \wedge \left(\begin{array}{c} ps(closed) \wedge pcs(open) \vee \\ ps(open) \wedge pcs(open) \end{array} \right) \end{array} \right)$$

pumpControlDiagnosis:

$$pcb \leftrightarrow \left(\begin{array}{c} UpToNow(pcb) \wedge \neg pcr \vee \\ expectedOpen \wedge pcs(closed) \vee \\ \neg expectedOpen \wedge pcs(open) \vee \\ \left(\begin{array}{c} UpToNow(\neg pb) \vee \\ pr \end{array} \right) \wedge \left(\begin{array}{c} ps(closed) \wedge pcs(open) \vee \\ ps(open) \wedge pcs(open) \end{array} \right) \end{array} \right)$$

A. Gargantini, A. Morzenti (1996). ***TRIO specification of a steam boiler controller***. In: Formal Methods for Industrial Applications.

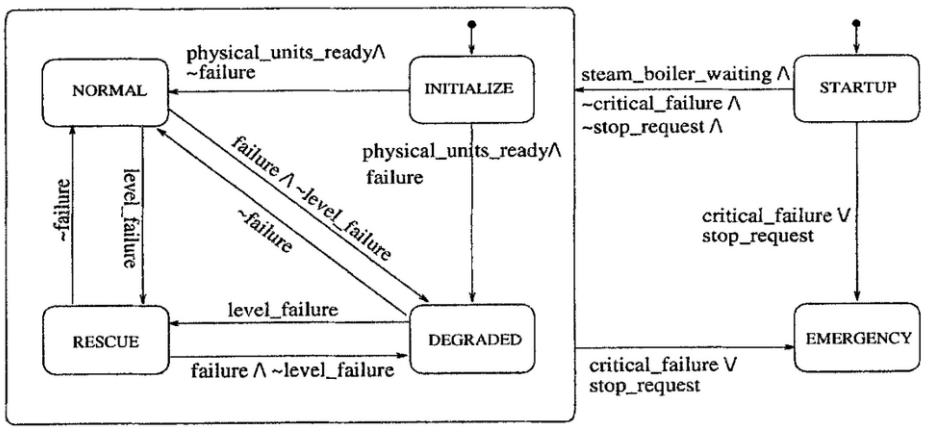
Modeling and Verification in Lustre

Verification

- **P1**: The mode is in {startup, initialize,normal,degraded, rescue, emergency}
- **P2**: Once the mode is emergency it is forever.
- **P3**: In normal mode no device is signalled to be in failure

```
1. p3 = implies(
2.          op_mode=normal ,
3.          level_defect=ok and
4.          steam_defect=ok and
5.          AND(N_pump,pump_defect=ok) and
6.          AND(N_pump,pump_control_defect=ok) and
7.          not transmission_failure(pump_state));
```

Modes of Operation



Cattel, T., Duval, G. (1996). ***The Steam Boiler problem in Lustre.***
In: Formal Methods for Industrial Applications.

SCCharts

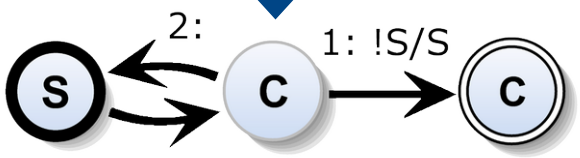
A Statecharts Dialect with Sequentially Constructive Semantics

ELK
Eclipse Layout Kernel

Extensive SCCharts Tooling

KIELER
The Key to Efficient Modeling

- Automatic Layout
- Transient Views
- Interactive Browsing
- Smart Zoom



- Simulation
- Code Synthesis
- Model Checking
- Visualization

R. von Hanxleden, B. Duderstadt, C. Motika, S. Smyth, M. Mendler, J. Aguado, S Mercer, O. O'Brien.
SCCharts: Sequentially Constructive Statecharts for Safety-Critical Applications. PLDI '14, 2014.

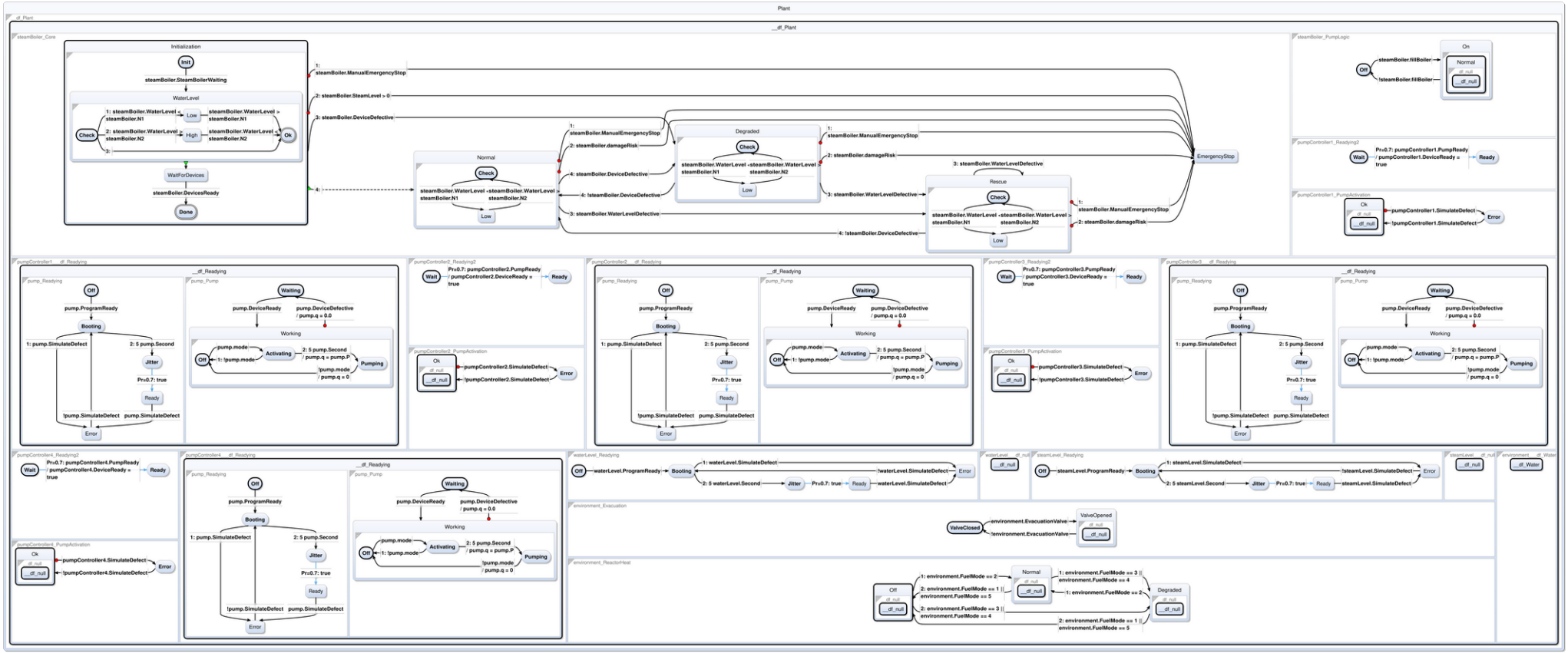
The 1st Generation

Simulation and Tool Evaluation

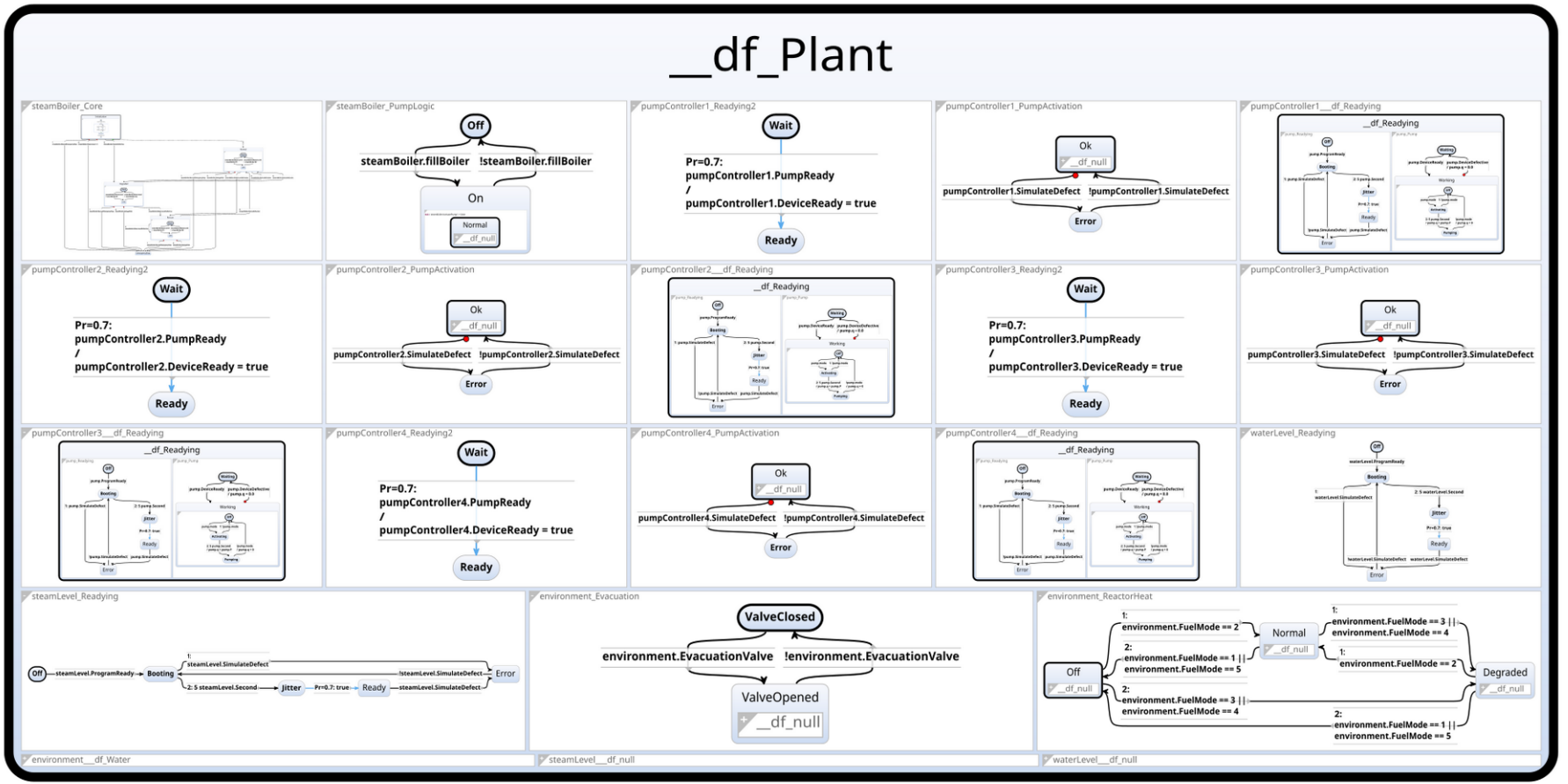
from 2019
by Steven Smyth

S. Smyth, S. Domrös, R. von Hanxleden. *A Case-Study on Manual Verification of State-based Source Code Generated by KIELER SCCharts.* Kiel University, Department of Computer Science, TR 1905, 2019.

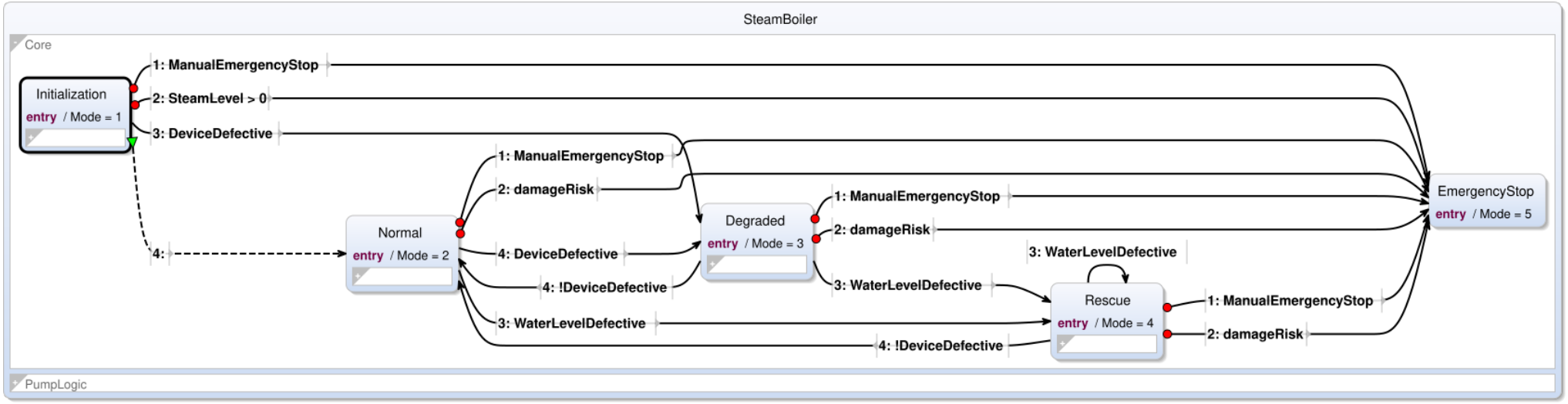
The Full Model



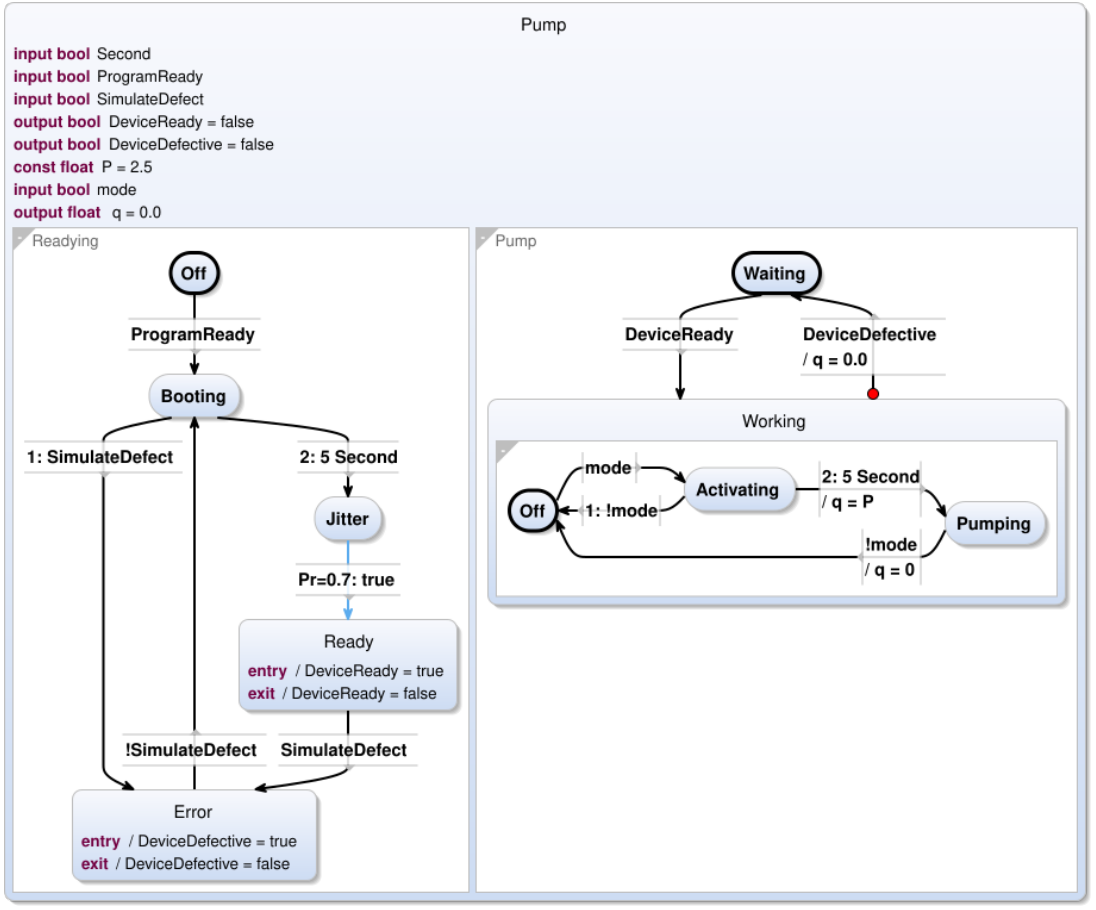
The Full Model with Top Down Layout



Modes of Operation

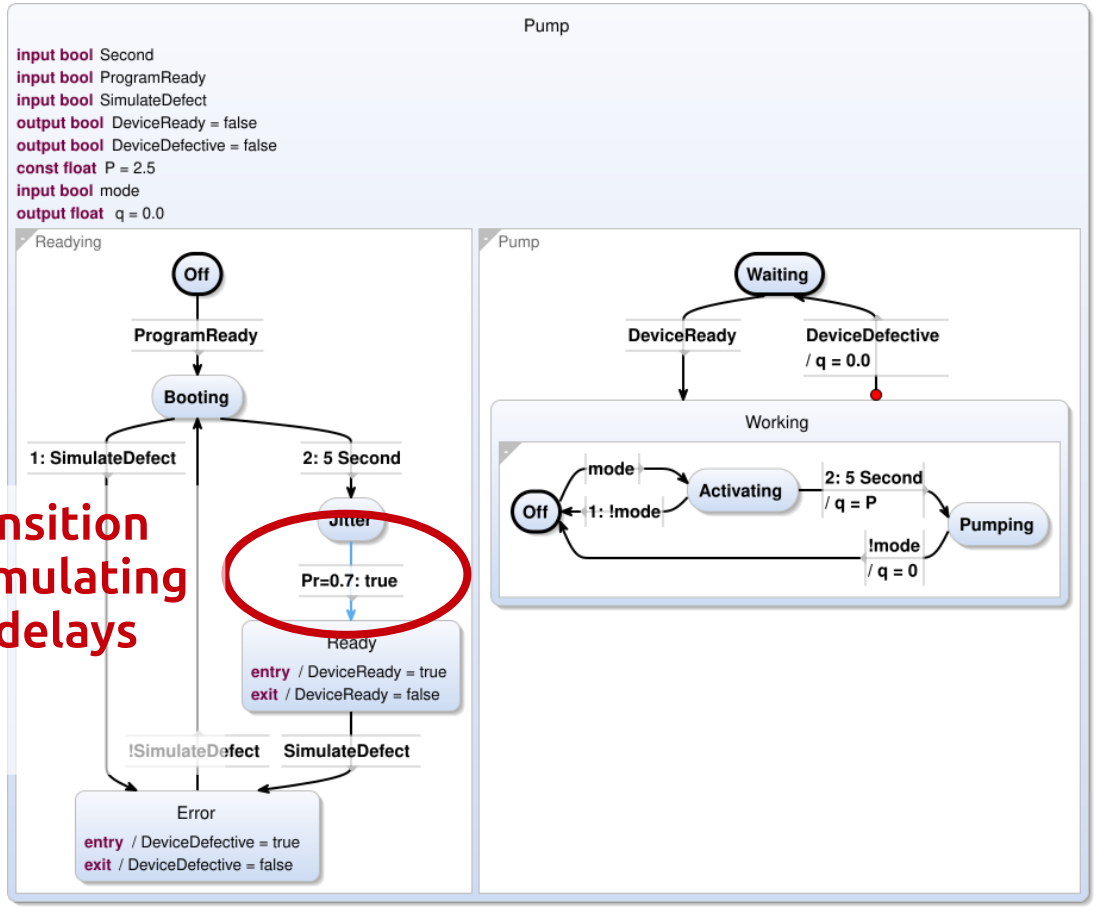


A Single Pump

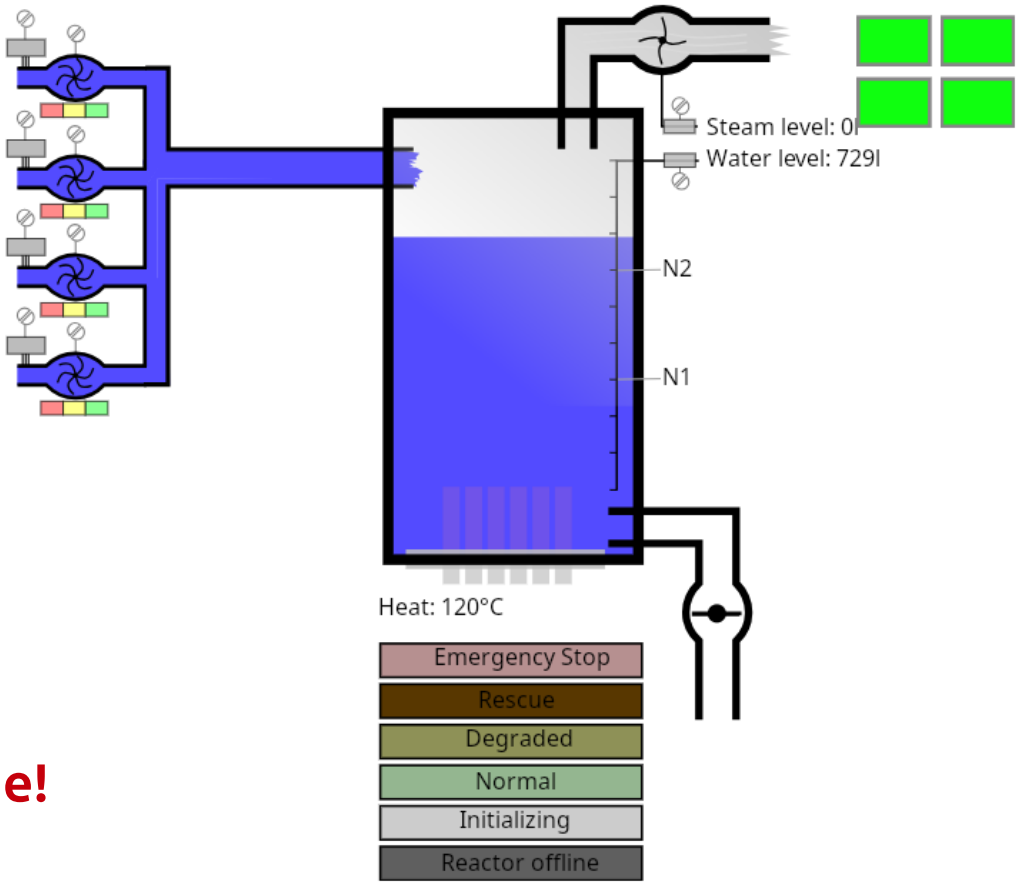


A Single Pump

Probabilistic transition triggering for simulating boot and repair delays

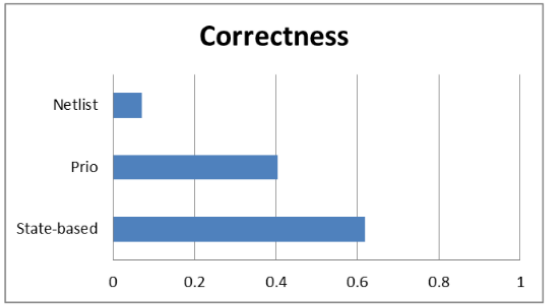


Interactive Simulation

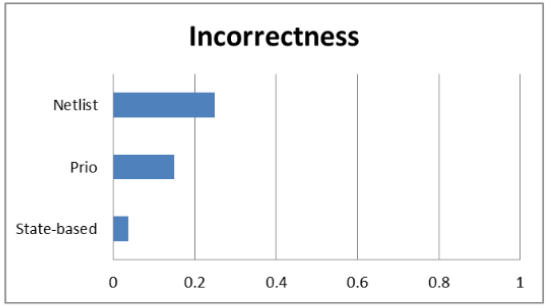


It’s demo time!

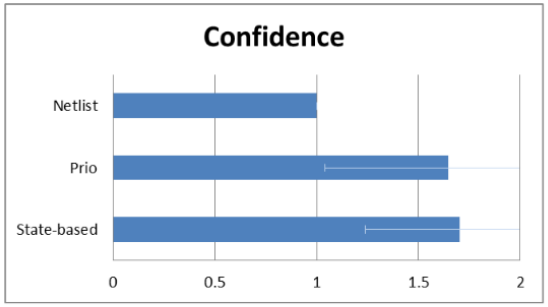
Case Study Code Generation



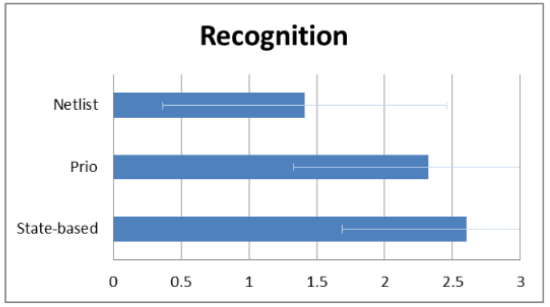
(a) Correct answers



(b) Incorrect answers despite being confident



(c) Confidence rating



(d) Recognition of model elements

- Study with 42 students
- Finding structural errors in generated code
- Evaluation of 3 approaches
  - Netlist
  - Priority
  - State-based

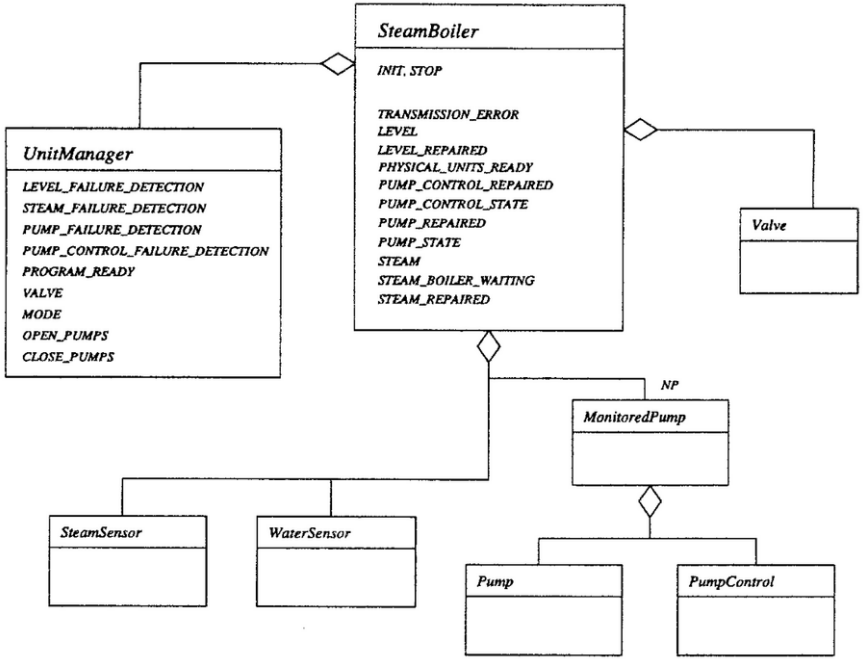
The 2nd Generation

Object Orientation

from 2022
by Alexander Schulz-Rosengarten

A. Schulz-Rosengarten. *Language Design for Reactive Systems — On Modal Models, Time, and Object Orientation in Lingua Franca and SCCharts.* Dissertation 2024.

Object Orientation in Past Steam Boilers



R. Büssow and M. Weber (1996) **A steam-boiler control specification with Statecharts and Z**. In: Formal Methods for Industrial Applications, Specifying and Programming the Steam Boiler Control.

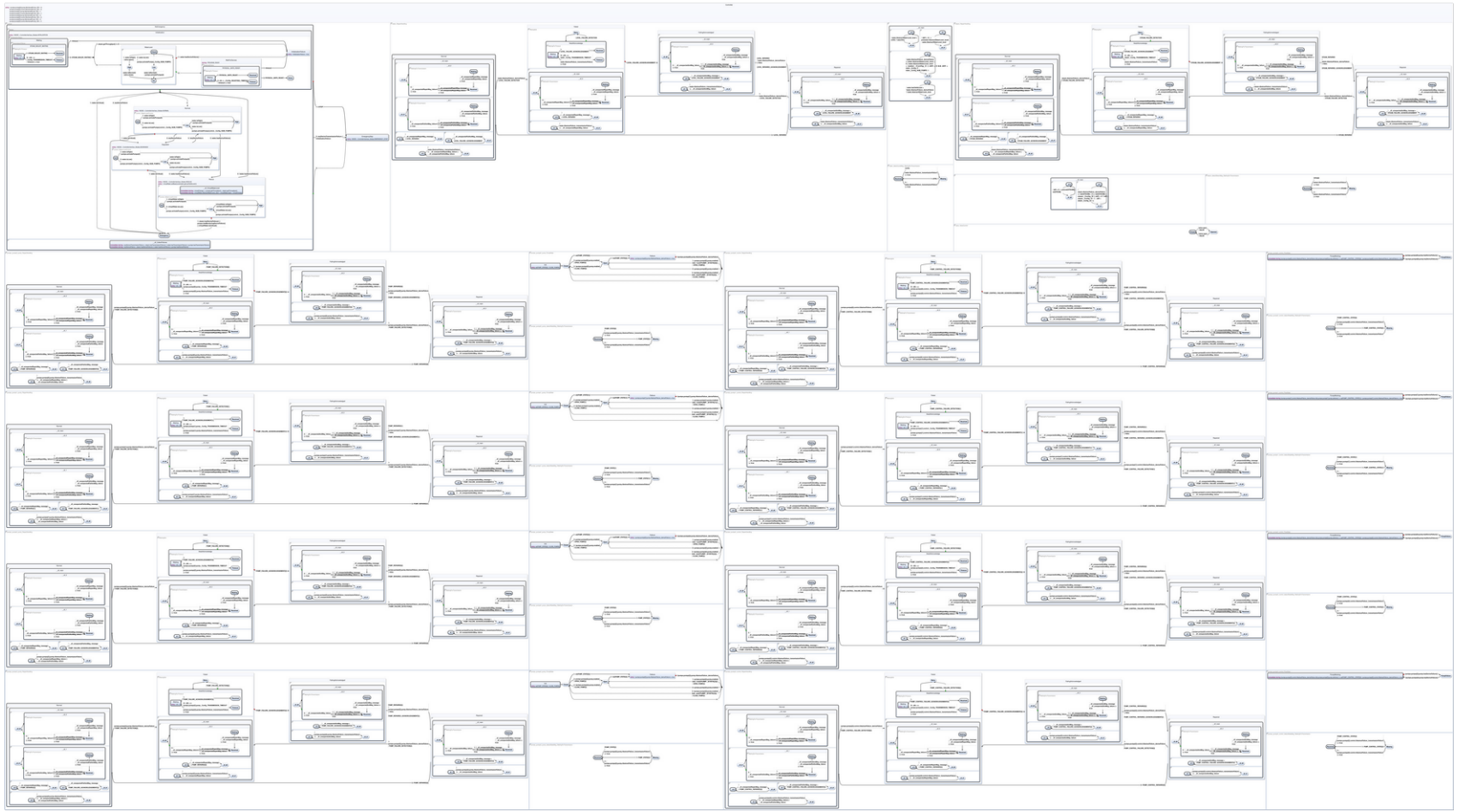
sorts	*cstate*, *boilingstate*, *not_boilingstate*	
subsorts	*not_boilingstate*, *boilingstate* $\leq$ *cstate*	
ops	*startup*, *ch_w_s*, *StBR*, *PrReady*, *PUReady* : $\rightarrow$ *not_boilingstate*	
	normal, *degraded*, *rescue*, *emergency* : $\rightarrow$ *boilingstate*	
vars	*boiling*, *newmode* : *boilingstate*, *not_boiling* : *not_boilingstate*	
class	*controller*	
atts	*timer* : *Timer*	– "rings" at time $n\Delta t$.
	state : *cstate*	– states of the controller object.
	stop_v : *nat*	– number of *stop* messages received in row.
	stoprec : *bool*	– *stop* message received in current round?
initially	*timer* := *timer*(Δt), *state* := *startup*, *stop_v* := 0, *stoprec* := *false*.	

Table 1. *The class controller.*

P.C. Ölveczky,P. Kosiuczenko, M. Wirsing. (1996). **An object-oriented algebraic steam-boiler control specification**. In: Formal Methods for Industrial Applications, Specifying and Programming the Steam Boiler Control.

P. Carreira, and C. Miguel. **Automatically verifying an object-oriented specification of the steam-boiler system.** Proceedings of the 5th International ERCIM Workshop on Formal Methods for Industrial Critical Systems (FMICS'2000).

The Full Model

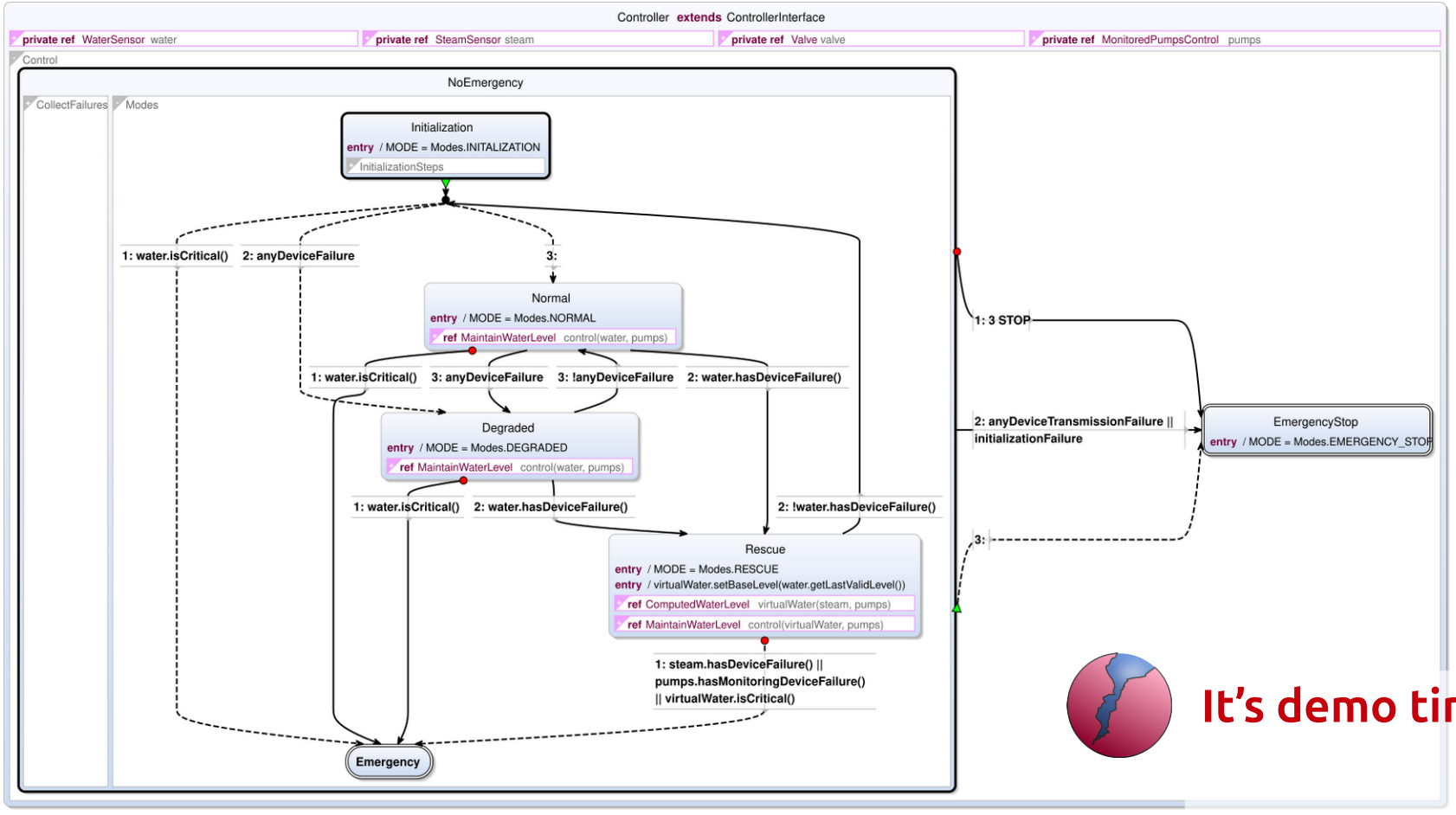


Structure is Everything

Object-orientation offers:

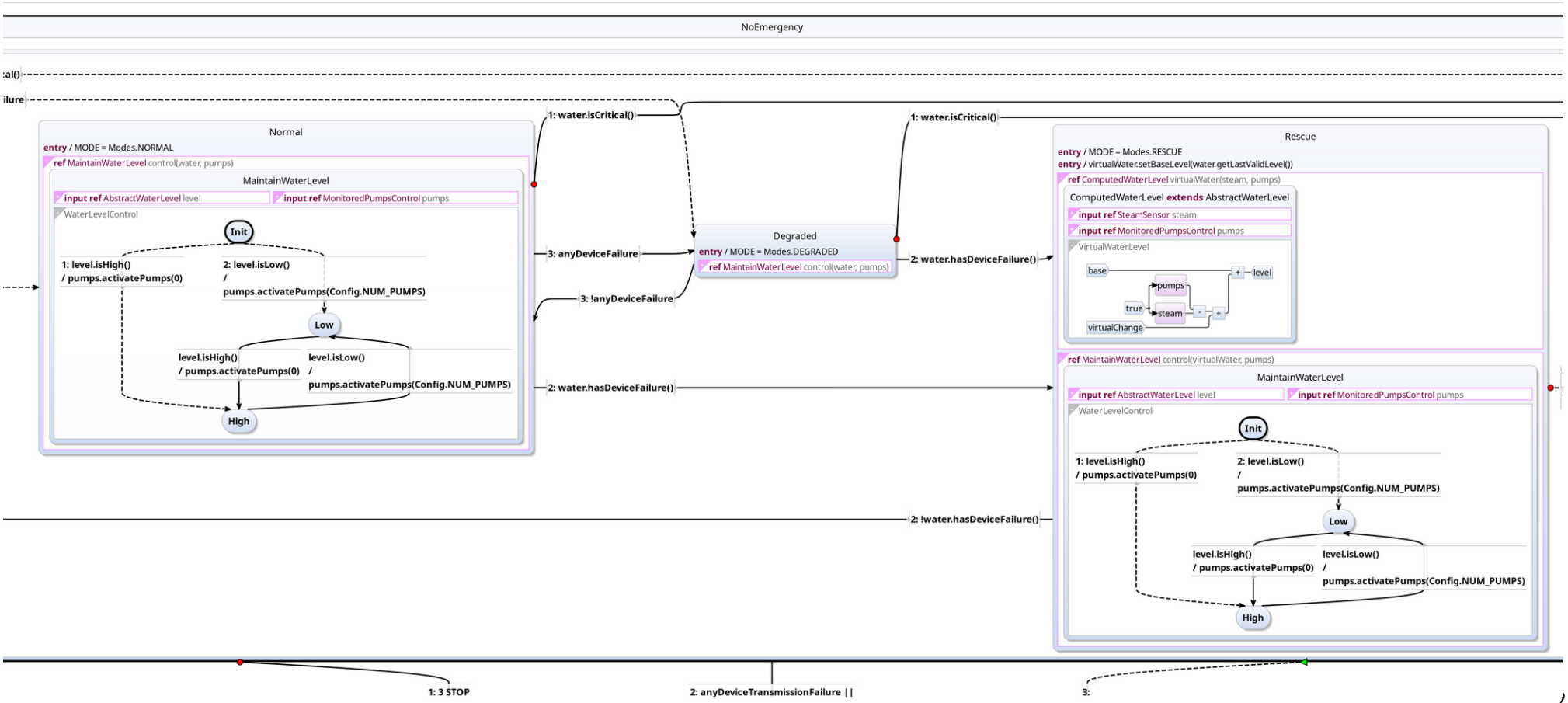
- Object-based composition
- Expressing commonalities via inheritance
- Adjustability via subtyping
- Modeling pragmatics of SCChart enable UML-like documentation

Structure is Everything

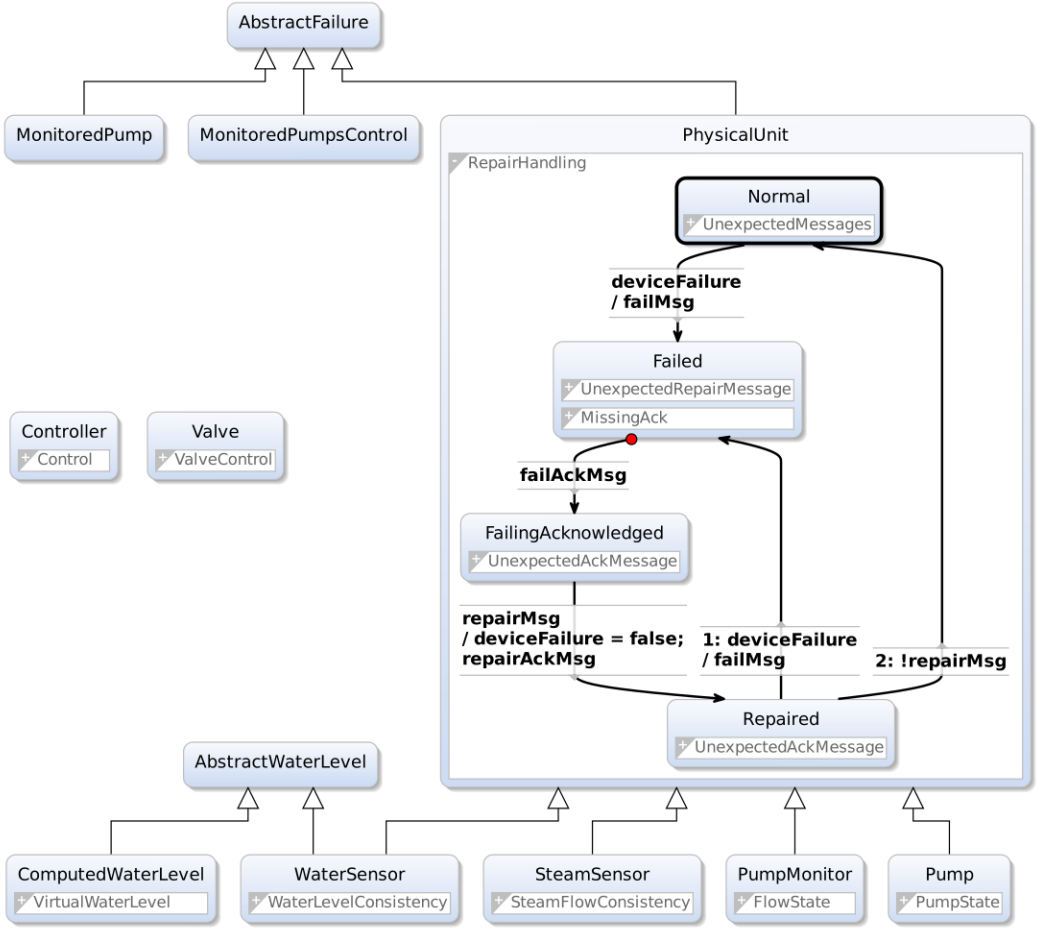


It’s demo time!

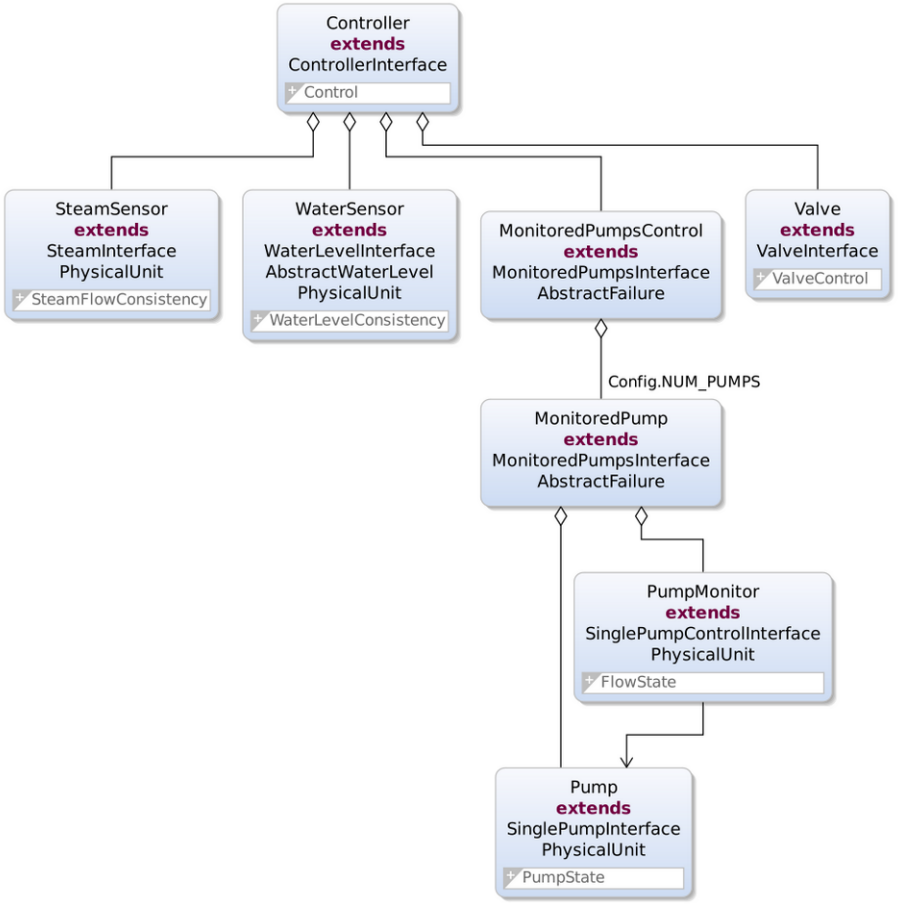
Demo Recap: Subtyping



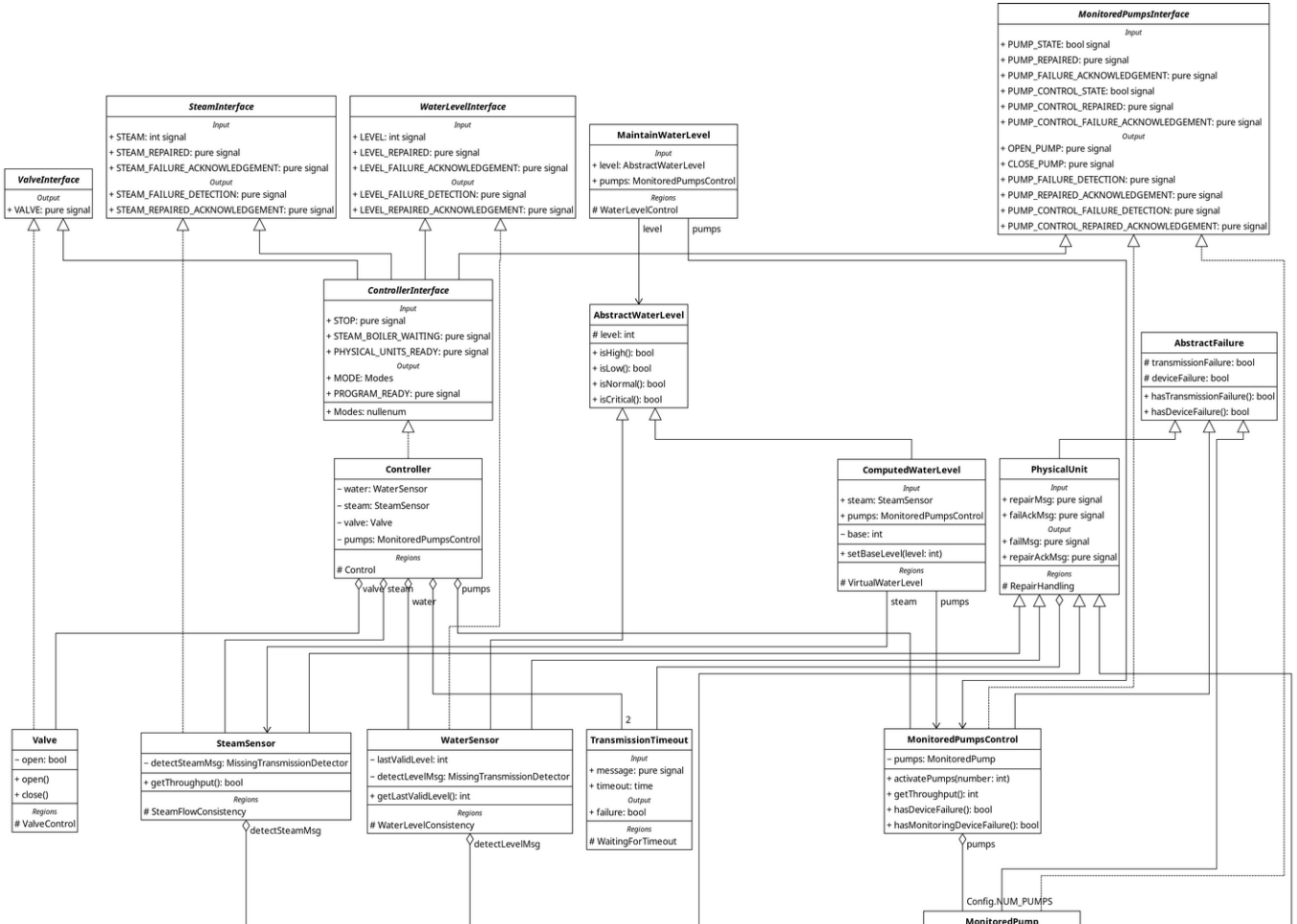
Demo Recap: Inheritance



Demo Recap: Object Composition



Demo Recap: UML Documentation



The 3rd Generation

Verification and Risk Analysis

from 2024
by Tokessa Hamann and Jette Petzold

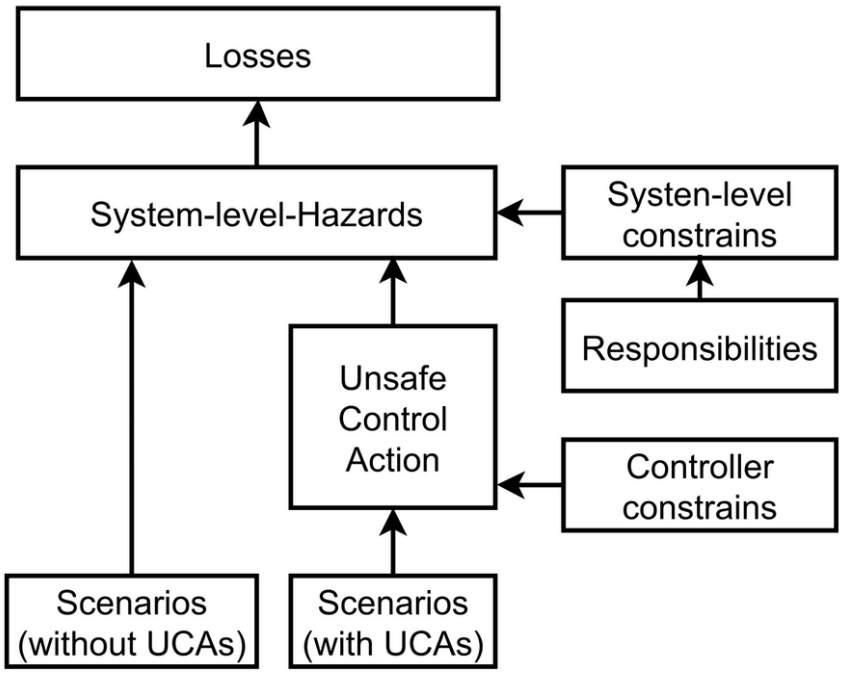
Tokessa Hamann, *Safety Analysis of the Steam Boiler in SCCharts*, Bachelor's Thesis, Kiel University, Department of Computer Science, 2024.

Risk Analysis

- Analyzes the bigger picture
- Helps to avoid or mitigate risks
- Manual but structured process
- Helps to identify plans for component failures
- Many techniques

System-Theoretic Process Analysis (STPA)

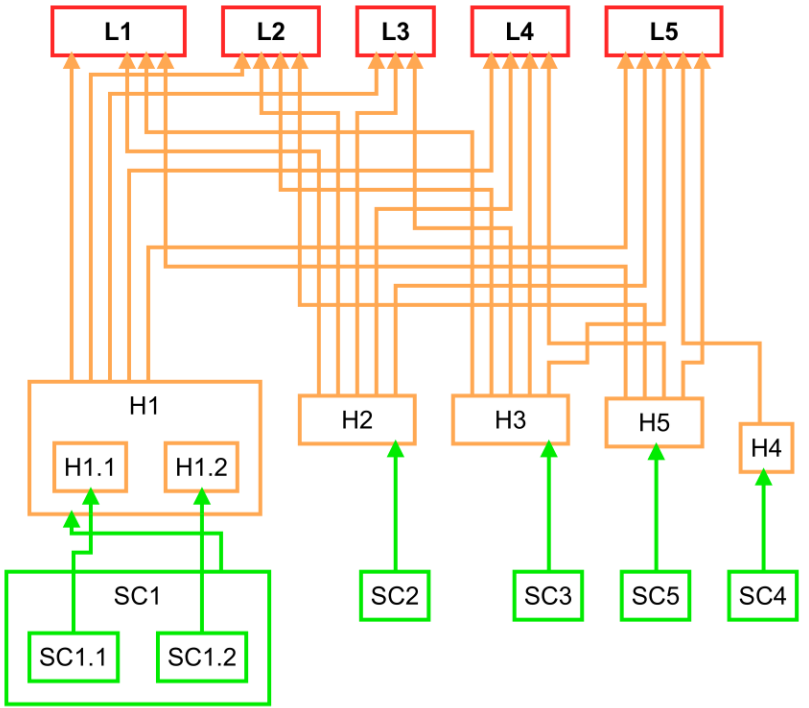
- Capable of identifying unsafe interaction between components
- Usable in early design stages
- Still identifies component failures



STPA for the Steam Boiler in PASTA

```
22  Hazards
23  H1 "Steam-boiler outside safe water levels" [L1, L2, L3, L4, L5] {
24      H1.1 "Water level is too low"
25      H1.2 "Water level is too high"
26  }
27  H2 "The heat is outside safe levels" [L1, L2, L3, L4, L5]
28  H3 "System integrity is lost" [L1, L2, L3, L4, L5]
29  H4 "Inadequate steam production" [L5]
30  H5 "Wrong operation mode" [L1, L2, L4, L5]
```

It’s demo time!



Verification

Benefits:

- Automatic generation of LTL formulas
- LTL formulas based on UCAs ensures good coverage

Current limitations:

- Only simplified models (derived from OO variant)
- Only NuXmv support
- Only single components

Verifying the Valve Behavior

```
@LTL "G ((progInit && X(mode==0 && waterLevel>=7 && !valve
    && !fail && !progReady)) -> X(VALVE))", "UCA70"
```

UCA70 = *"If water level is greater than N2 and valve closed, the valve command is issued"*
UCA71 = *"If water level is normal and valve open, the valve command is issued"*
UCA72 = *"If water level is below normal and valve open, the valve command is" issued"*
UCA73-76 = *"If mode is not Initialization, no valve command is issued"*
UCA77 = *"If water level is normal and valve closed, no valve command is issued"*
UCA78 = *"If water level is below normal and valve closed, no valve command is issued"*

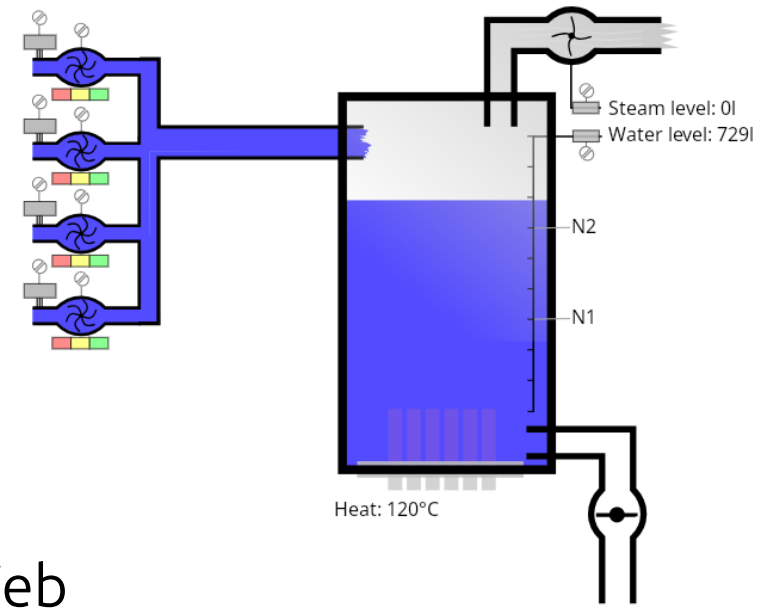
Properties related to valve correct state, for instance:

- **P6**: Valve commands are issued only in initialize mode.
- **P7**: In initialize mode if water level is greater than N2 the valve is open.

Cattel, T., Duval, G. (1996). ***The Steam Boiler problem in Lustre.***
In: Formal Methods for Industrial Applications.

Wrap up

- Steam Boiler is still a good benchmark for
  - Modeling capabilities
  - Tooling
- SCCharts with different emphases
  - Interactive simulation
  - Object-oriented design
  - Risk analysis



KIELER in the Web
github.com/kieler