Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Model-Based Systems Engineering

Master Thesis Proposal

Open System Simulation and Test with SysML, Modelica and Eclipse

Parham Vasaiely 16. December 2010

Table of Contents

Table of Contents

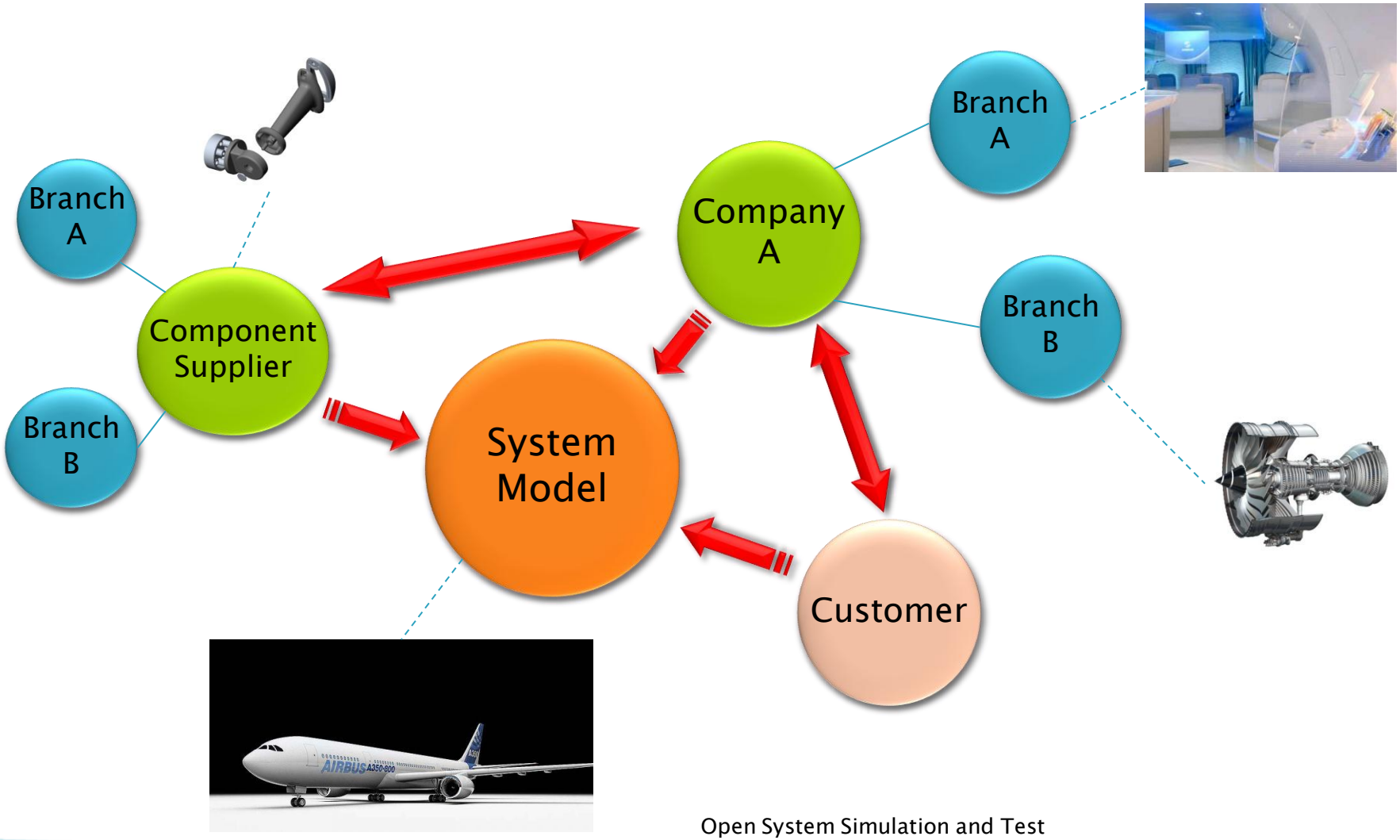
Introduction and Motivation

Systems Engineering

- Problem: Increasing complexity of technical systems
  - Increased degree of automation
  - Increased number of involved technologies
  - SW and HW development is strongly coupled
  - Component suppliers are increasingly involved into the design process
  - Validation of the functionality long before the first prototype is built
- Motivation:
  - Simplify and Speed systems development
  - Cut time to market
  - Improve Quality
- Solutions:
  - Model- Based Systems Engineering
  - Simulation and Test

Project Description

Model-based Systems Engineering



Project Description

Kind of Systems

- Technical Systems
  - Aerospace, Automotive, ...
- Dynamic
  - Time depended, Behaviour
- Reactive Systems
  - It continuously interacts with its environment

Project Goals

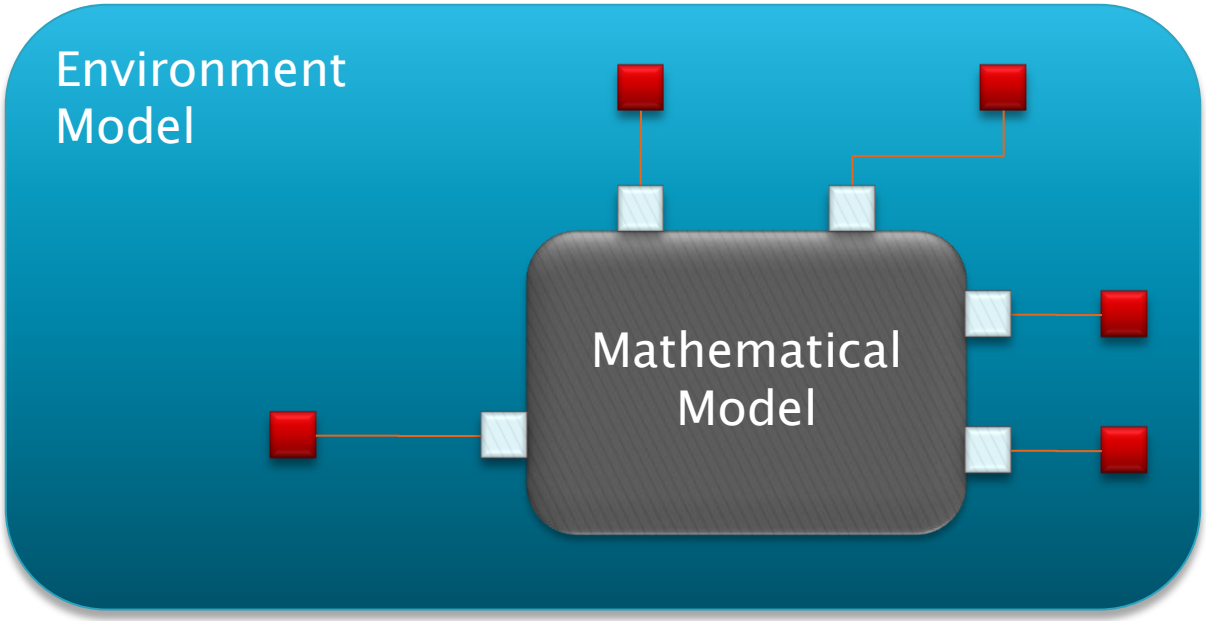
Open System Simulation and Test

- Open Technologies
  - Open Reusable Solutions
- Standardized Model Representation
  - SysML and Modelica
- Simulation
  - Experiment
  - Support Understanding
  - Communication
- Test
  - Searching for a failure caused by fails
  - Improve Quality
  - Verification and Validation

Project Goals

Model Simulation and Test

- Mathematical Model
  - System behaviour expressed in mathematical form
- Model in the Loop (MiL)



Variables (Input & Output)

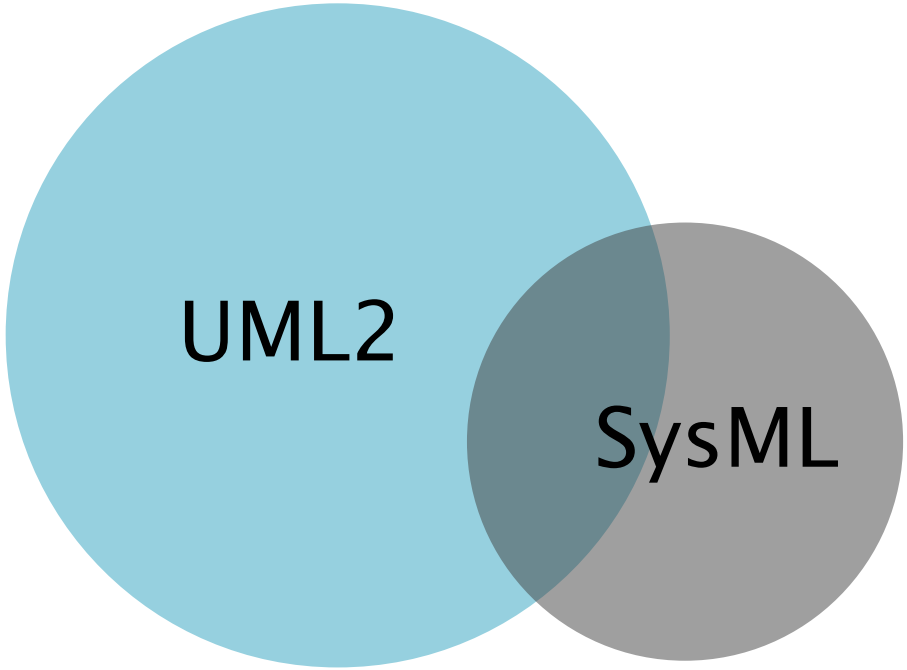
Table of Contents

Technologies

The Systems Modelling Language

- OMG UML2 Profil for Systems Engineering
- SysML Version 1.2 (using UML 2.3) (June 16, 2010)

- Diagrams
  - Structure
  - Behaviour
  - Requirements

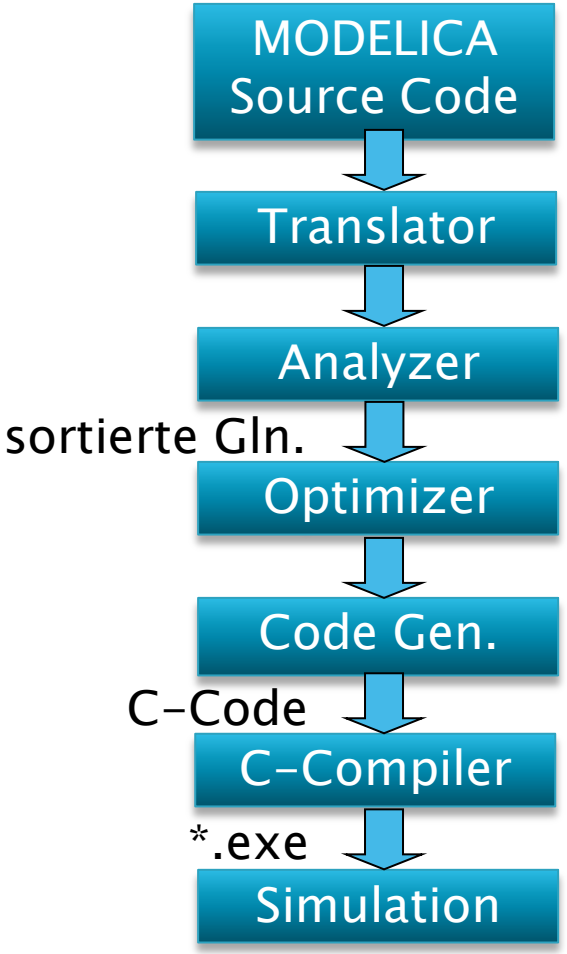


Technologies
Modelica

- Next Generation Modeling- and Simulation Language
- Equation based instead of assignment statements
  - Acausal Physical Modeling
  - Deklarative
- Object Oriented
  - Class Model (+handle complexity)
  - Component based (+reuse)

Technologies
OpenModelica

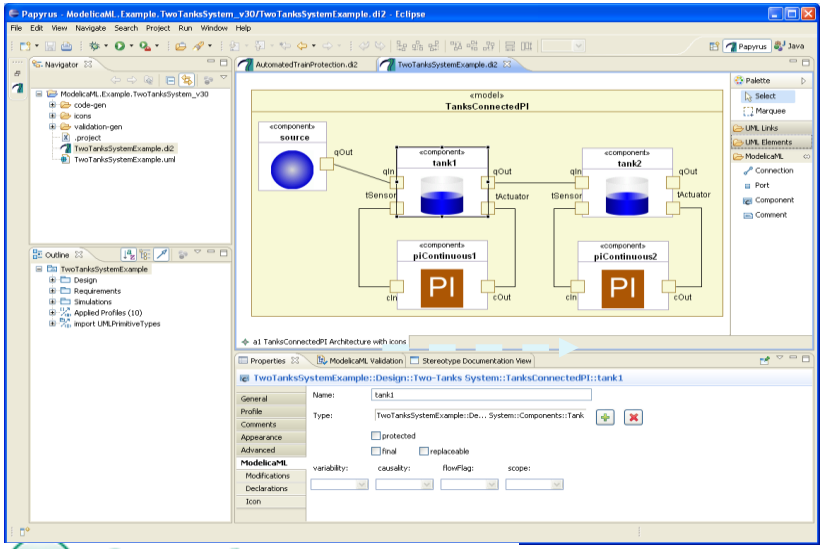
- Open Source Modelica Tool
  - Compiler for Modelica Code
  - Solver for Simulation
    - DASSL, Euler, Rungekutter
- Non-Interactive Simulation
- Interactive Simulation



Technologies
Eclipse & Papyrus

- Eclipse Plug-In Papyrus
  - Papyrus UML as Modeling Environment

Papyrus UML
SysML Profil



Acceleo

Model to Text Transformation

Methods

Modeling: The System Model [1]

- Mathematical Model
  - Behaviour
- Model in the Loop
  - Communication with its Environment using its Inputs and Outputs
- Deterministic Model
  - Same Behaviour (Outputs) using same Inputs

Methods

Modeling: Using SysML [2]

Requirements

OMG SYSTEMS MODELING LANGUAGE

MODELICA

Environment Model

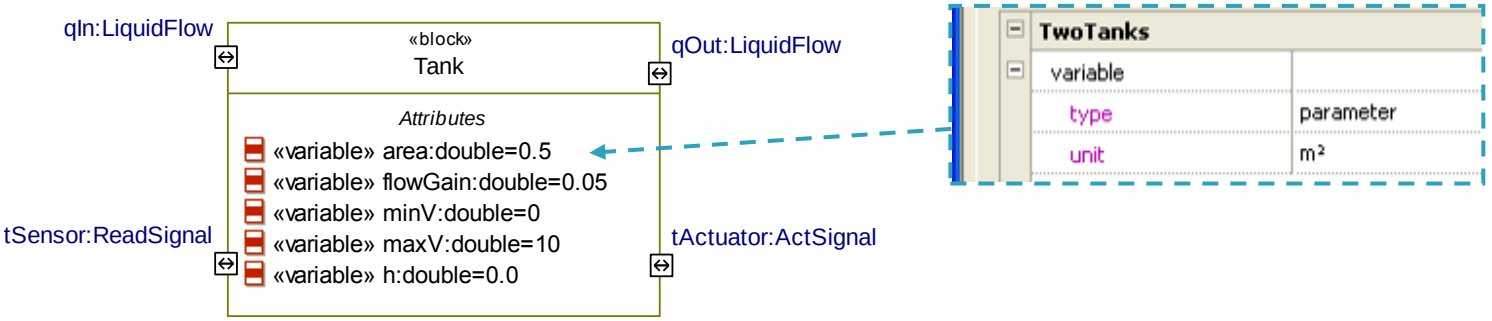
Mathematical Model

Using Structure Diagrams

- Block Definition Diagram
- Internal Block- Diagram
- Parametric Diagrams

Methods

Modeling: SysML to Modelica [3]



<<block>>Tank → Tank.mo (Modelica)

```
within TwoTanks;
block Tank
    ReadSignal tSensor;
    ActSignal tActuator;
    LiquidFlow qIn;
    LiquidFlow qOut;
    parameter Real area (unit = "m2") = 0.5;
    parameter Real flowGain (unit = "m2/s") = 0.05;
    Real h (start = 0.0, unit = "m");
end Tank;
```

See Bachelor Thesis:
Interactive Simulation of SysML Models using Modelica

Methods

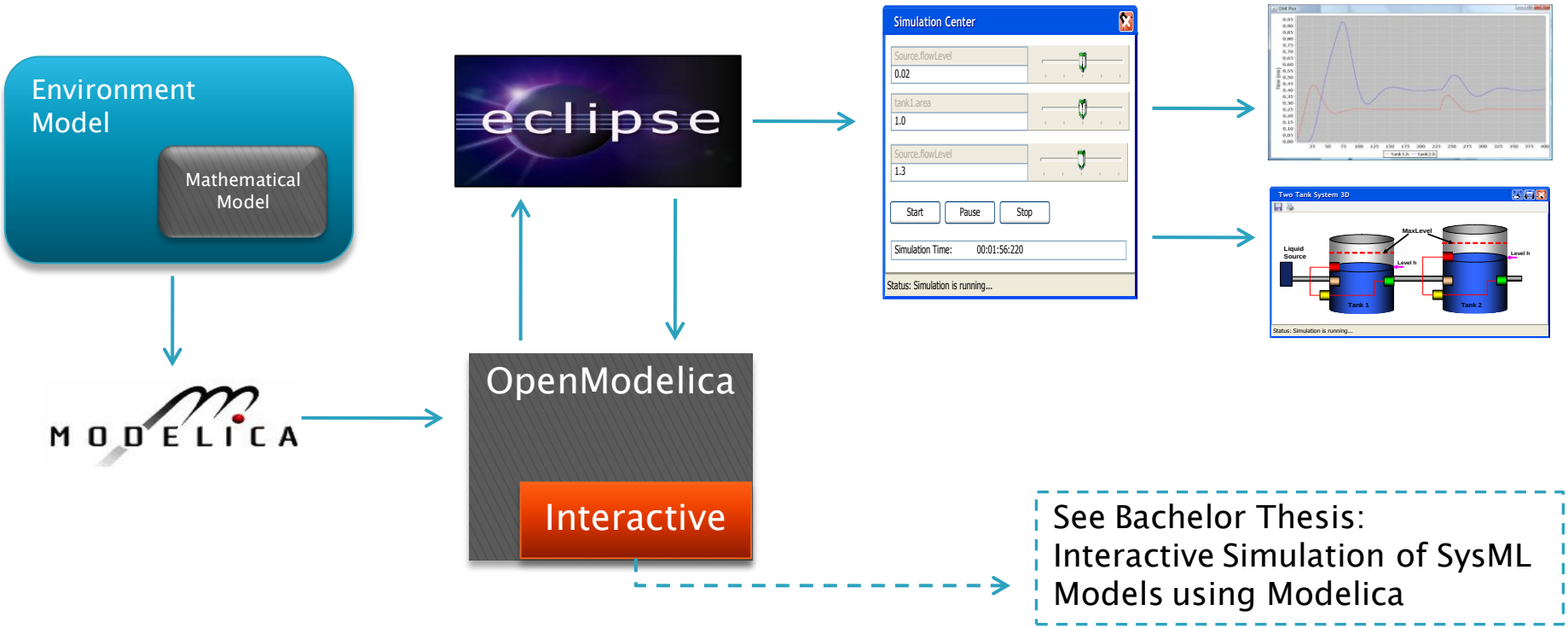
System Simulation [1]

- Interactive
  - Real-time, User Interaction
- Non-Interactive
  - As fast as computer power allows, event, condition
- Dangers of Simulation
  - Falling in love with a Model
  - Forcing Reality into the Constraints of a Model
  - Forgetting the model's level of accuracy

Methods

System Simulation [2]

- Interactive & Non-Interactive Simulation
  - Support Understanding, Experiment & Communication



Methods

Test [1]

- Functional Specification
  - Formal Requirements
- Dynamic Testing
  - Model Execution (Simulation)
- Black-Box Testing
  - Input & Output Behaviour
- Model-Based Testing
  - Test Cases Derived from a Model
  - Automatic Test Generation

Methods

Test: Automatic Test Generation [2]

- Specification Language with Formal Semantics
  - Papyrus UML as Modeling Environment

Requirements

Each tank has a continuous proportional-integral (PI) controller connected to it, which regulates the level of liquid contained in the tanks to a reference level. While the liquid source fills the first tank with liquid the PI continuous controller regulates the outflow from the tank depending on its actual level…

Non-Formal Specification

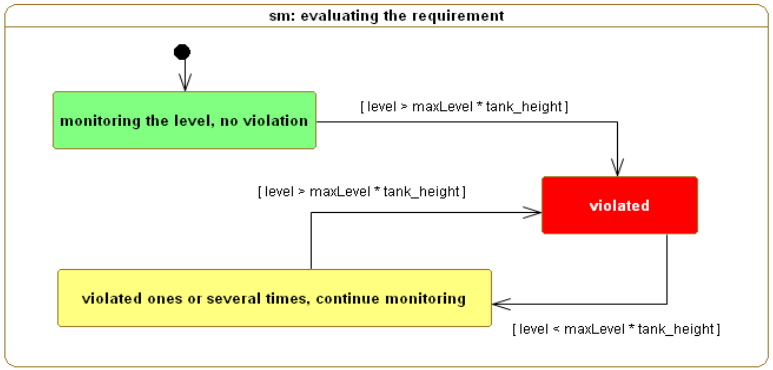
Using SysML Requirements Diagram

Max_level_Tank_A
Id: 001
Description: The level of a tank should not rise over 10m, to avoid a Overflow of liquid.

Non-Formal Specification

Using SysML Behaviour Diagrams

- State Diagram
- Activity Diagram

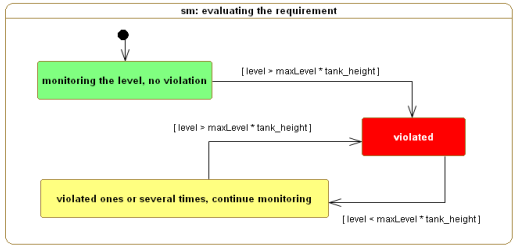


Formal Specification

Methods

Test: Automatic Test Generation [3]

Formal Specification



Environment Model

Mathematical Model

Algorithm for Test Derivation

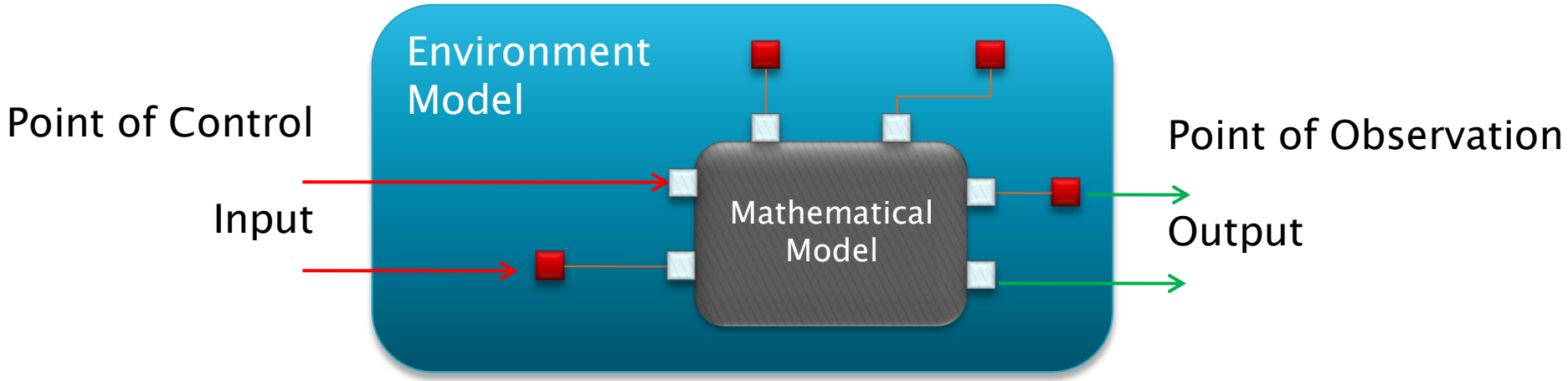
Test Cases
Test Data

Using Behaviour Diagrams

- State Diagram
- Activity Diagram
- Sequence Diagram

Methods

Test: Black-Box [4]

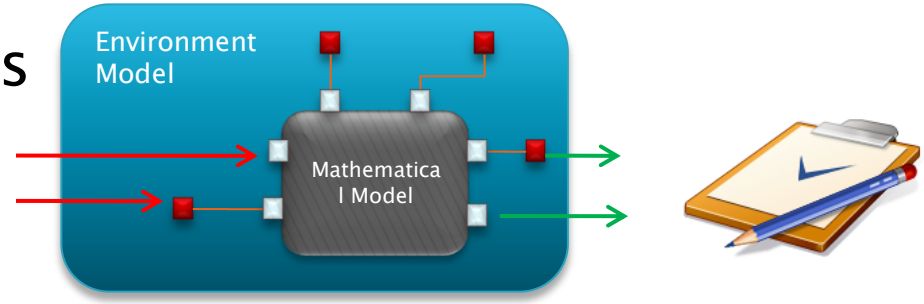


Variables (Input & Output)

Methods

Test: Test Execution [5]

- Automatically
  - Test Tool forces the Tests
  - Test Script
  - Unit Tests



- Interactive System Test
  - User controls the system interactively
  - Undefined Usage
  - In Real-time

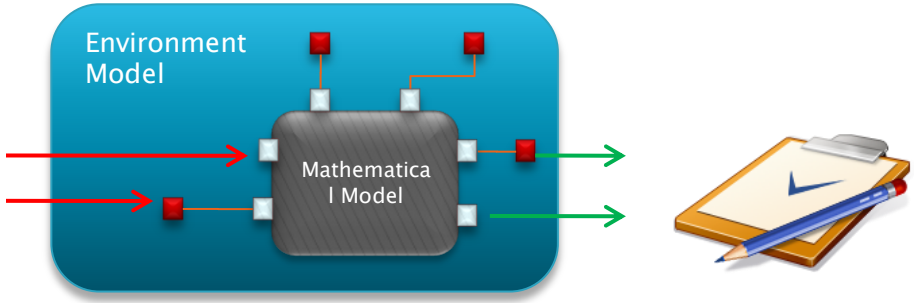


Table of Contents

Risks

- SysML
  - Modelica Profile for UML
  - Test Representation in SysML
- OpenModelica
  - Interactive Simulation Performance
- Automatic Generation of Tests from SysML
  - Generation from UML is still an research topic

Table of Contents

Conclusion and Outlook

- Simulation and Test
  - Support Model-based Systems Engineering
  - Improve System Quality
  - Cut time to market
- Open Reusable Solutions
  - SysML, Modelica, Eclipse
- Next Step: Automatic Test Generation

References

- Sanford Friedenthal, Alan Moore and Rick Steiner, Practical Guide to SysML, 2008
- Fritzson Peter. Principles of Object-Oriented Modeling and Simulation with Modelica 2.1, 2004
- Modelica Association. Modelica: A Unified Object-Oriented Language for Physical Systems Modeling: Language Specification Version 3.0, Sept 2007. www.modelica.org
- OMG. OMG Systems Modeling Language (OMG SysML™), Version 1.1, November 2008.
- Acceleo, Eclipse Plug-In. www.acceleo.org/pages/home/en
- Papyrus UML, www.papyrusuml.org
- Baker, P., Dai, Z.R., Grabowski, J., Schieferdecker, I., Williams, C, Model-Driven Testing, 2008
- Christof Enert, Systematically Requirements Engineering and Management, 2008
- Johannes Link, Software Tests with JUnit, 2008
- Harel D. and Politi M., Modeling Reactive Systems with Statecharts, 1998

Thank you for your attention!

Parham Vasaiely
Parham.Vasaiely@haw-hamburg.de