EADS

EADS INNOVATION WORKS

Systems Engineering

Virtual Verification of System Designs against System Requirements - A Method Proposal

Wladimir Schamai (EADS Innovation Works, Germany)
Philipp Helle (EADS Innovation Works, UK)
Peter Fritzson (Linköping University, Sweden)
Chris Paredis (Georgia Institute of Technology, USA)

Table of Contents

- Motivation
- Scope
- Case Study Example
- vVDR Method
- Conclusion

EADS

Motivation

- System requirements are (still) written in natural language
  - Natural language is understood by everyone
  - Formal methods are overwhelming or overdone
  - Formal methods are not widely used in industry
  - For certification, authorities demand requirements to be written in natural language
  - ...
- How to verify system design against textual requirements by means of simulation?
  - Textual requirements cannot be processed by computers: How to formalize requirements so that they can be processed and evaluated during system simulations in order to detect errors or inconsistencies?

Scope – where to start?

Stakeholder Requirements,
Regulations, Design Constraints, etc.

Input
textual
requirements

Requirements Analysis
and Negotiation

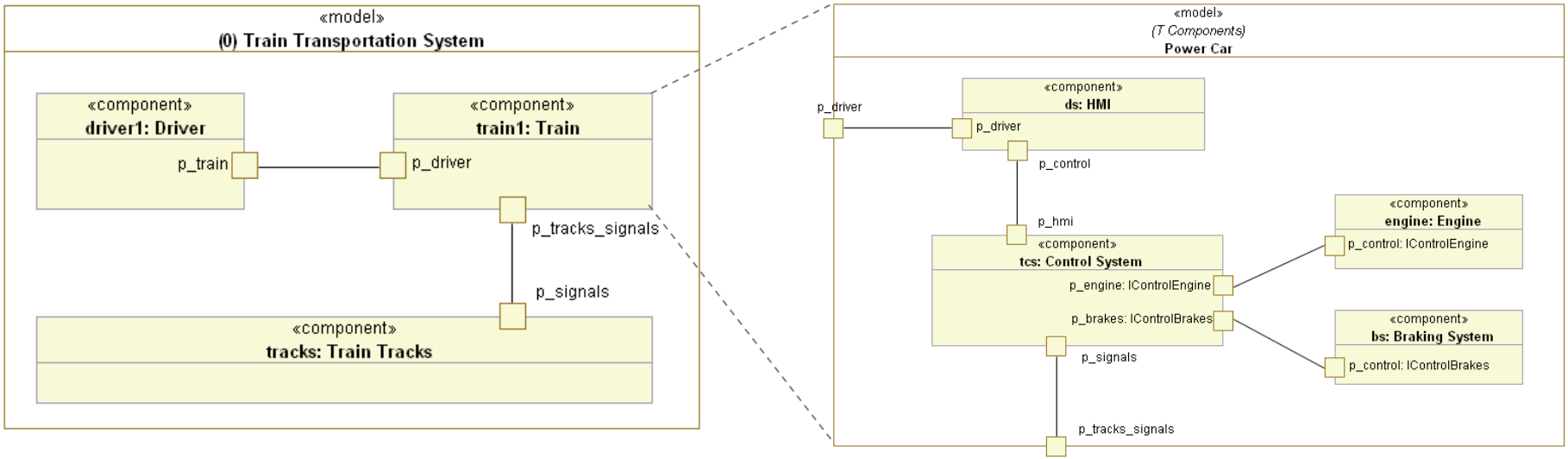
Agreed
textual
requirements

Out of scope: Writing good requirements,
requirements analysis and negotiation.

Example: Automated Train Protection System

Most rail systems have some form of train protection system that use track-side signals to indicate potentially dangerous situations to the driver. The simplest train protection systems consist of signals with two states: green to continue along the track and red to apply the brake to stop the train. More sophisticated systems include detailed information such as speed profiles for each section of the track.

Accidents still occur using a train protection system when a driver fails to notice or respond correctly to a signal. To reduce the risk of these accidents, Automated Train Protection (ATP) systems are used that automate the train's response to the track-side signals by sensing each signal and monitoring the driver's reaction. If the driver fails to act appropriately, the ATP system takes control of the train and responds as required.



Example: Natural Language (Textual) System Requirements

Requirements Management Tool (e.g. IBM Rational DOORS)

Textual System Requirement Examples

ID: xyz
Text: If at any time the controller calculates a "caution" signal, it shall, within 0.5 seconds, enable the alarm in the driver cabin.

ID: xyz
Text: If the alarm in the driver cabin has been activated due to a "caution" signal and the train speed is not decreasing by at least 0.5 m/s^2 within 2 seconds after activation of alarm, then the controller shall within 0.5 seconds activate the automatic braking.

ID: xyz
Text: If at any time the controller calculates a "danger" signal it shall within 0.5 seconds activate the braking system and enable the alarm in the driver cabin.

vVDR Method Steps

1. Select Requirements to be Verified by Means of Simulation
2. Formalize Textual Requirements
3. Select or Create Design Model To Be Verified Against Requirements
4. Create Test Models, Instantiate Requirements and Design Models
5. Link Requirement Properties to Design Model Properties
6. Simulate and Observe Requirement Violations
7. Analyze Simulation Results

Roles and Tasks

Requirements Analyst

-Select requirements to be verified using simulations
-Formalize textual requirements

System Designer

-Support Requirements Analyst in selecting requirements
-Create system design models to be verified against requirements
-Support System Tester in deciding which requirements are to be verified using which test cases
-Support System Tester in linking requirement properties to design model properties

---

System Tester

-Create test modes including test cases and decide which requirement are to be verified using which test cases
-Link requirements to design models
-Simulate and create a Simulation Summary Report

1) Select Requirements To Be Evaluated by Means of Simulation

Agreed textual requirements

Requirements Analyst
in collaboration with System Designer

1. Read a requirement
2. Decide if this requirement shall be evaluated using a simulation model (involve the System Designer)
3. Is this requirement complete, unambiguous and testable by using a simulation model?
4. If yes: Mark this requirement as selected

Selected textual requirements

1) Select Requirements to Be Evaluated by Means of Simulation

- Example of selected requirements:
  - "If at any time the controller calculates a "caution" signal, it shall, within 0.5 seconds, enable the alarm in the driver cabin."
  - "If the alarm in the driver cabin has been activated due to a "caution" signal and the train speed is not decreasing by at least 0.5 m/s^2 within 2 seconds after activation of alarm, then the controller shall within 0.5 seconds activate the automatic braking."
  - "If at any time the controller calculates a "danger" signal it shall within 0.5 seconds activate the braking system and enable the alarm in the driver cabin."

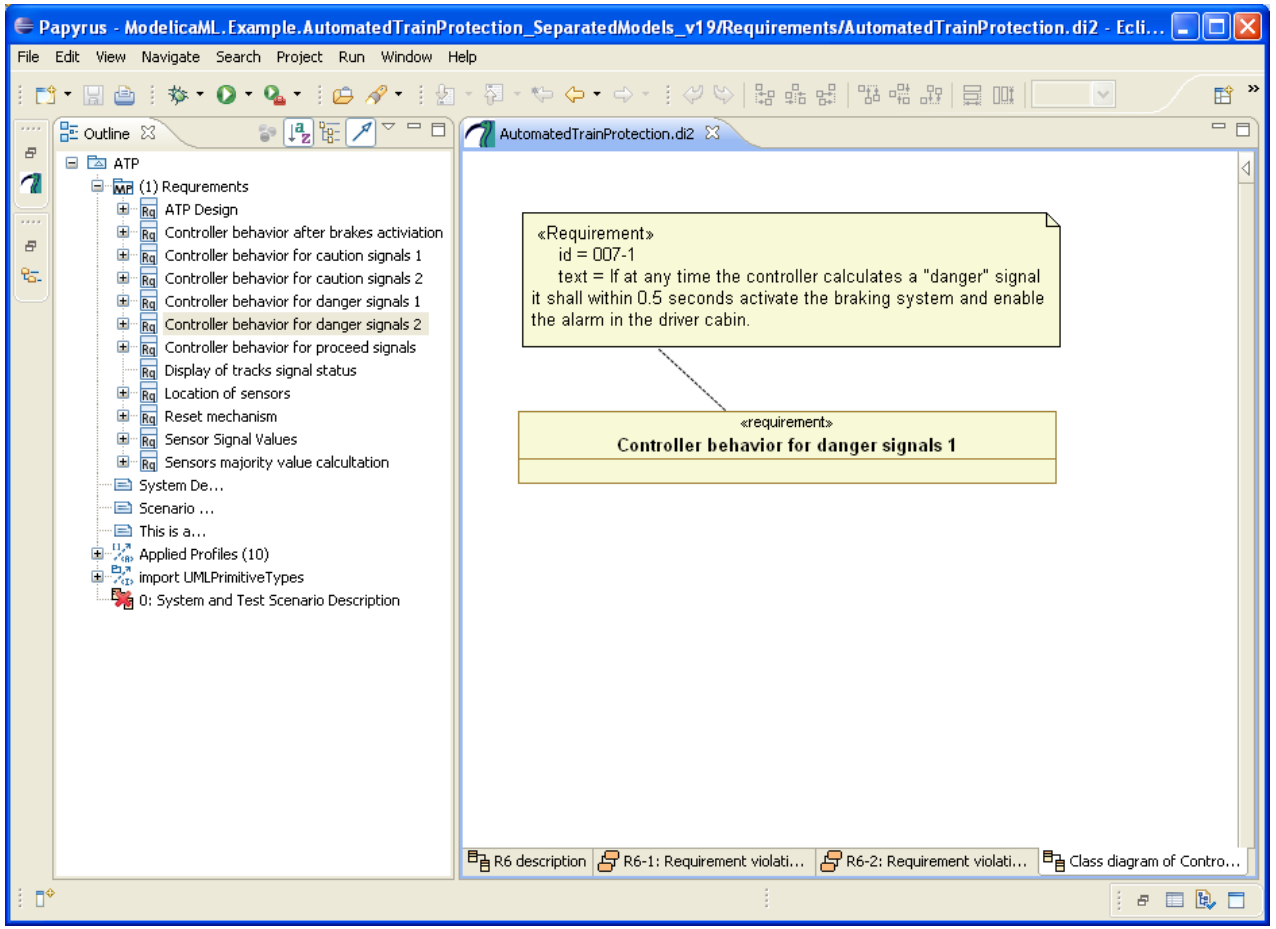
1) Select Requirements to Be Evaluated by Means of Simulation

- Example of requirements that are not selected:
  - "The sensors shall be attached to the side of the train and read information from approaching track-side signals, i.e. they detect what the signal is signaling to the train driver."
    - Why not?
      - We do not plan to create a model that will contain all information required to detect whether the sensors are attached to the side of the train. "Simulation" may not be best suited means to verify this requirement. "Inspection" of the design may be more appropriate.
  - "The ATP system shall consist of a central controller and five boundary subsystems that manage the sensors, speedometer, brakes, alarm and a reset mechanism."
    - Why not?
      - This is a design constraint to be taken into account. "Inspection" of the design will be sufficient to verify this requirement.

2) Formalize Textual Requirements

Selected textual requirements

Requirements Analyst



2) Formalize Textual Requirements

Textual requirement example:

"If at any time the controller calculates a "caution" signal, it shall, within 0.5 seconds, enable the alarm in the driver cabin."

Requirements Analyst

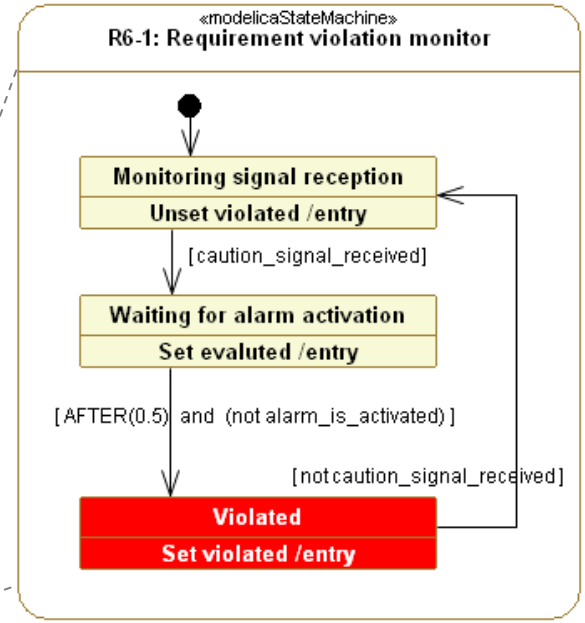
1. Identify measurable properties addressed in the requirement statement
2. Formalize properties and define requirement violation monitor

Formalized requirement:

«Requirement»
id = 006-1
text = If at any time the controller calculates a "caution" signal, it shall, within 0.5 seconds, enable the alarm in the driver cabin.

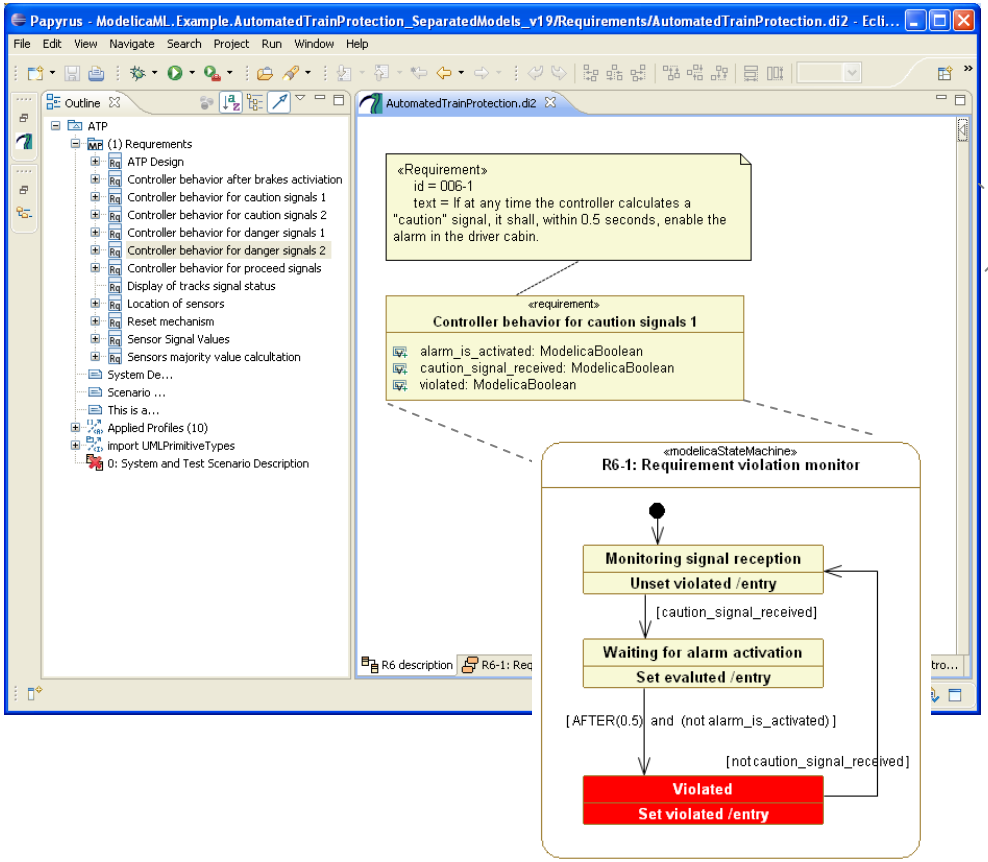
«requirement»
Controller behavior for caution signals 1

- alarm_is_activated: ModelicaBoolean
- caution_signal_received: ModelicaBoolean
- violated: ModelicaBoolean
- evaluated: ModelicaBoolean



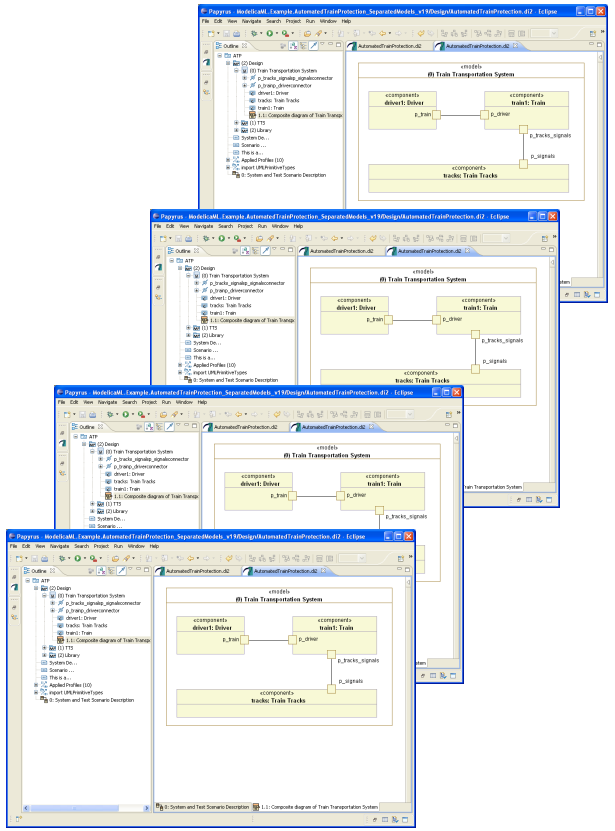
3) Select or Create Design Model To Be Verified Against Requirements

System Requirements



System Designer

System Design Alternatives



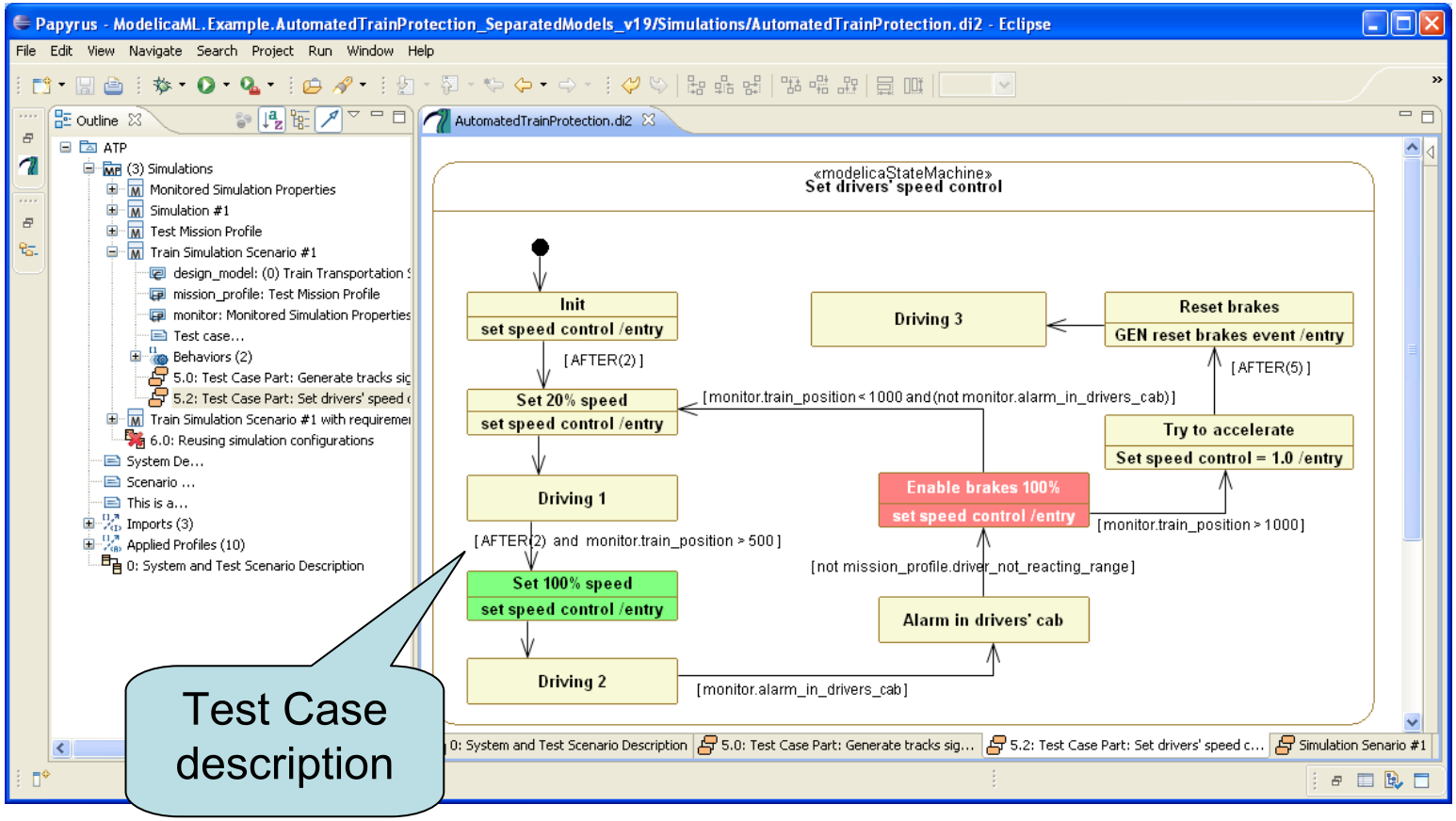
4) Create Test Models, Instantiate Models

System Tester
in collaboration with
System Designer

- Define test models that will contain a test case, requirements models and a design model to be verified against requirements
  - This model will contain the information required for reproducing tests results
- Define test cases for evaluating requirements
  - One test case can be used for evaluating one or more requirements
- Create additional models if necessary
  - For example, models that simulate the environment of the system, models that stimulate the system, models that monitor specific values, etc.

4) Create Test Models, Instantiate Models

System Tester
in collaboration with
System Designer



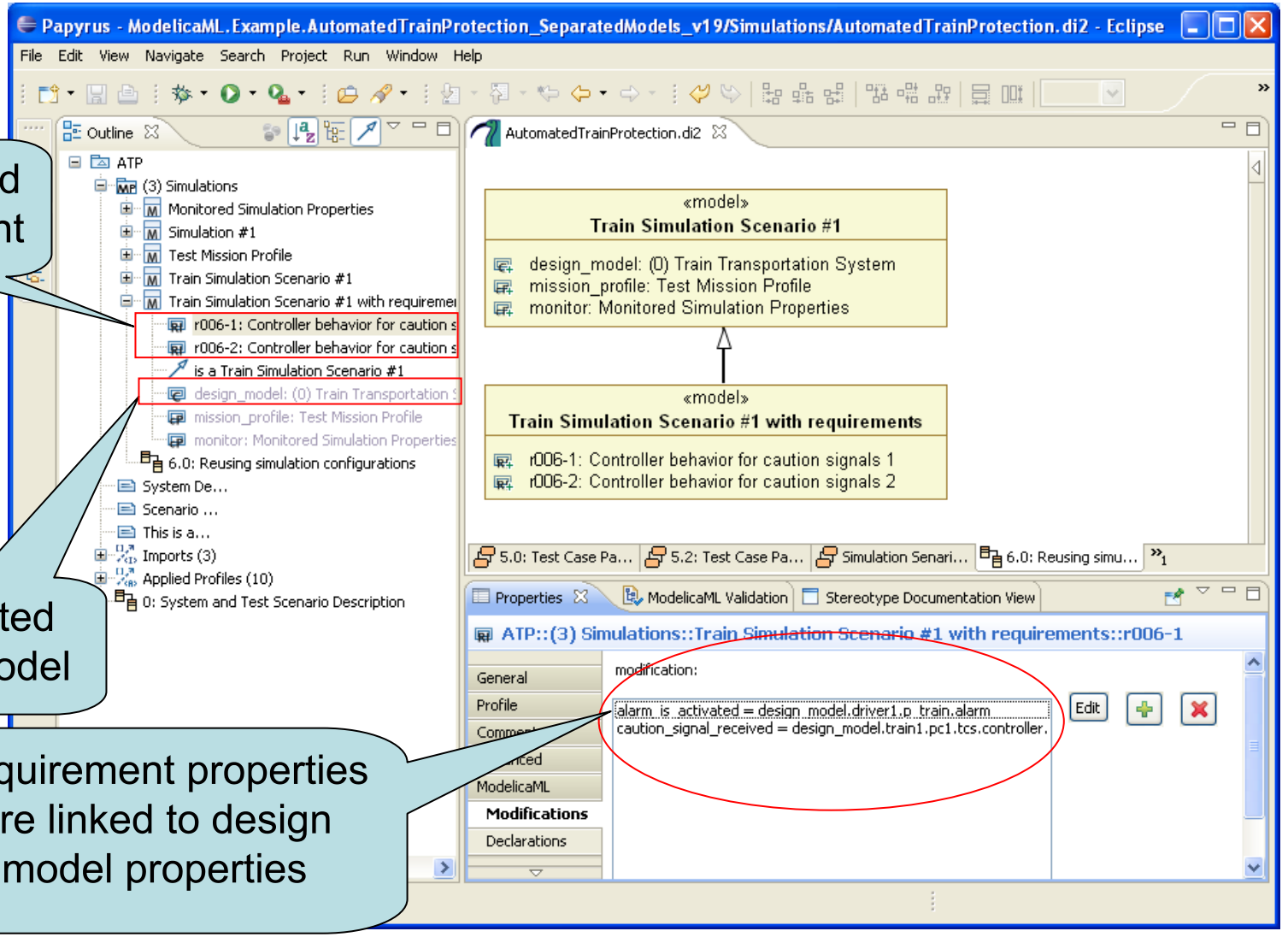
5) Link Requirement Properties to Design Model Properties

Instantiated requirement

System Tester
in collaboration with
System Designer

Instantiated design model

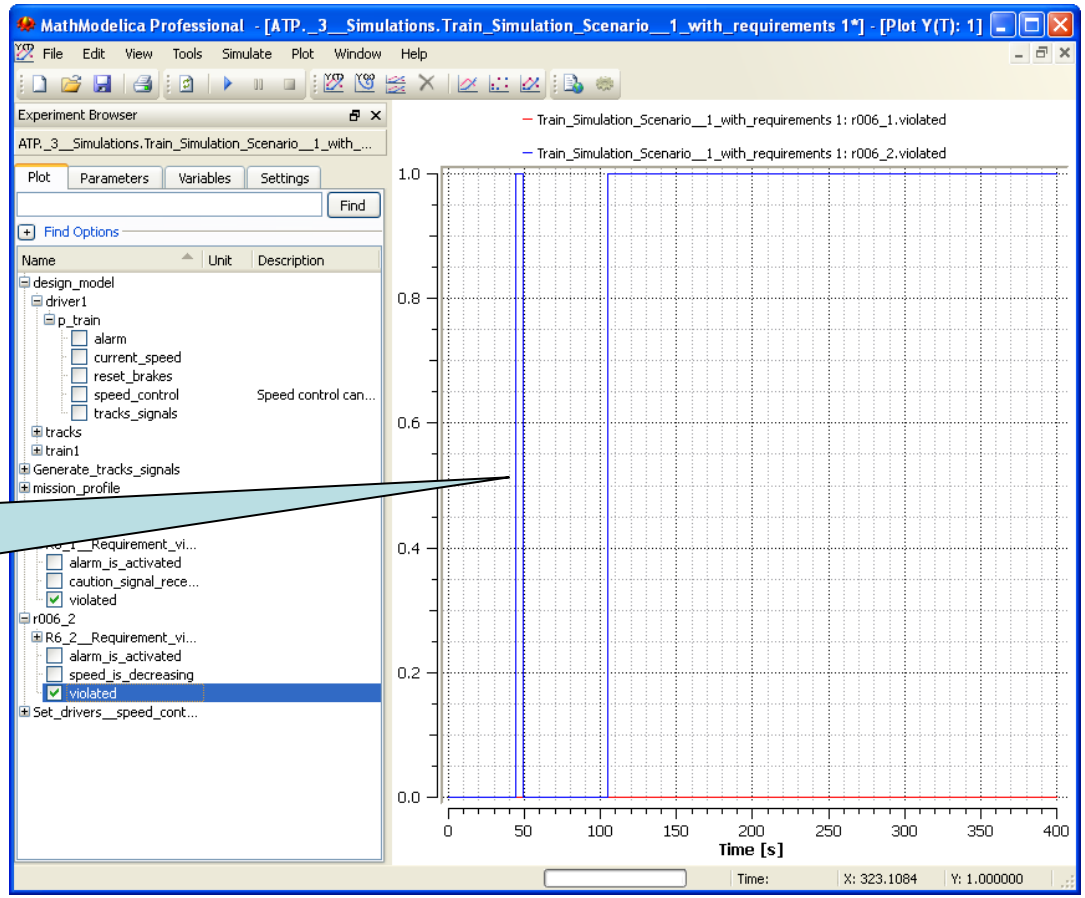
Requirement properties are linked to design model properties



6) Simulate and Observe Requirement Violations

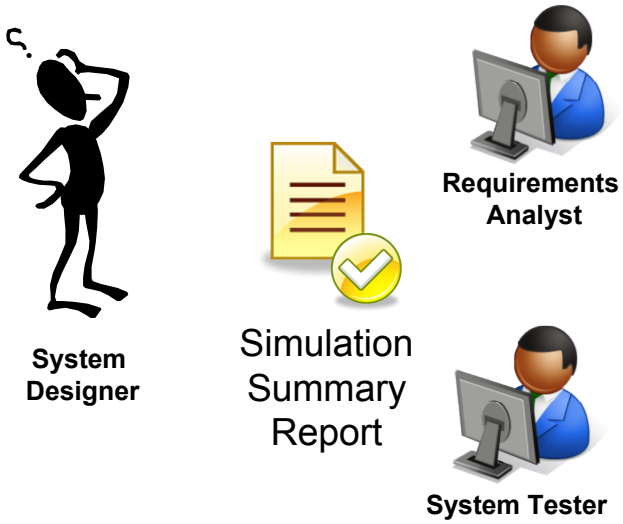
System Tester

Requirement violation indication



7) Analyze Simulation Results

- Create a Simulation Summary Report including:
  - For each test model include the simulation configuration:
    - Which design model, test cases and requirements were included
    - Requirements violations, if any.
    - This configuration allows the reproducing of test results
  - The reports can be used as reference for product verification



Conclusion

- vVDR is a method for the verification of design against requirements by using simulations
- The method applicability depends on
  - Design simulation models that are planned to be created
  - Quality (testability, completeness and correctness) of requirements to be verified
- Formalization and modeling activities are performed by different roles according to their competencies
- The separation of requirements, designs and test cases
  - Enables reuse and combination of requirements in different test cases for different design alternatives
  - Enables a automated re-evaluation of requirements along the system design evolution

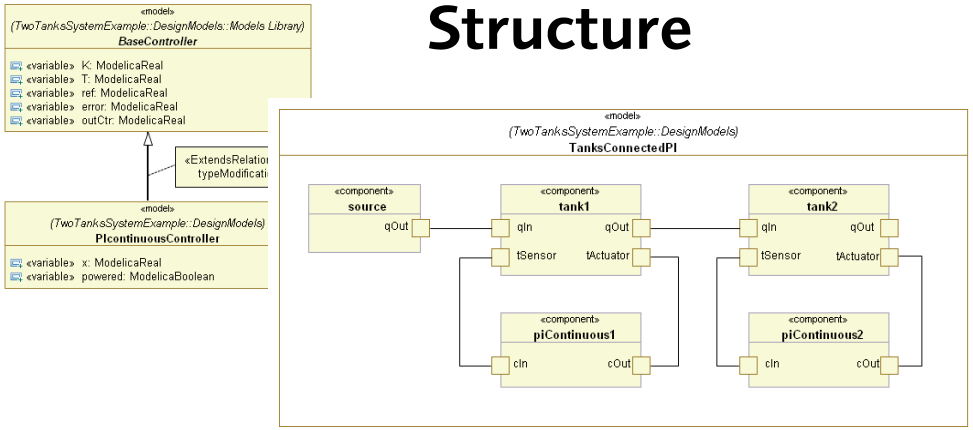
Thank you for your attention!

EADS Innovation Works
Systems Engineering Team

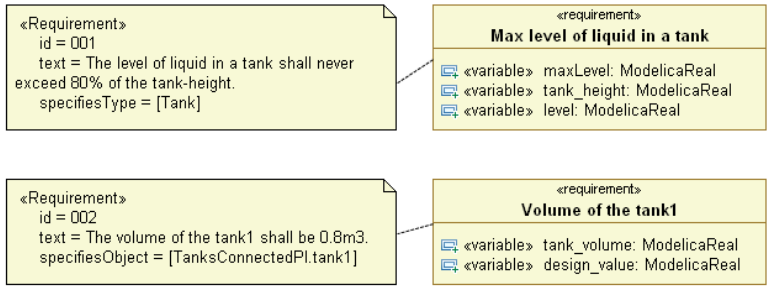
Wladimir Schamai
Wladimir.Schamai@eads.net

Introduction: ModelicaML Graphical Notation

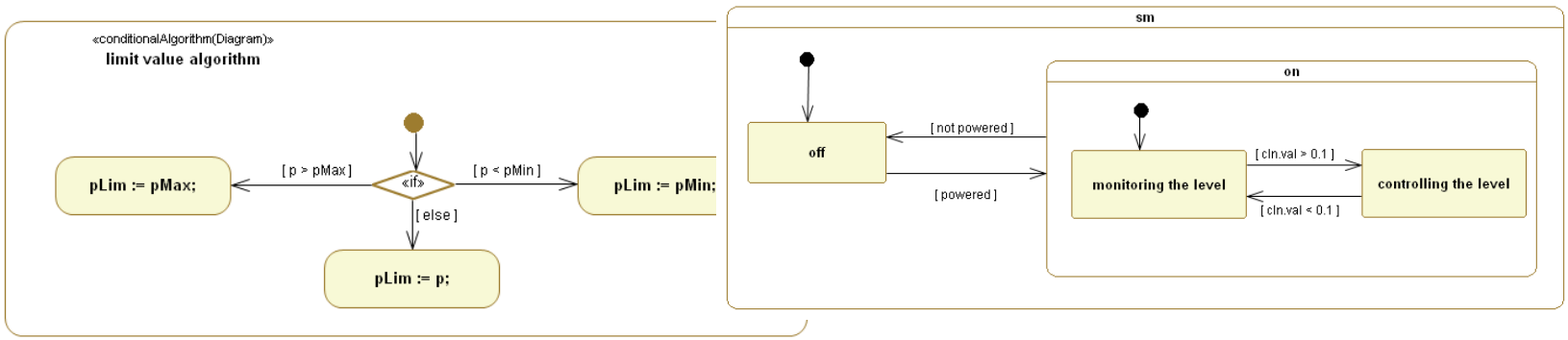
Structure



Requirements

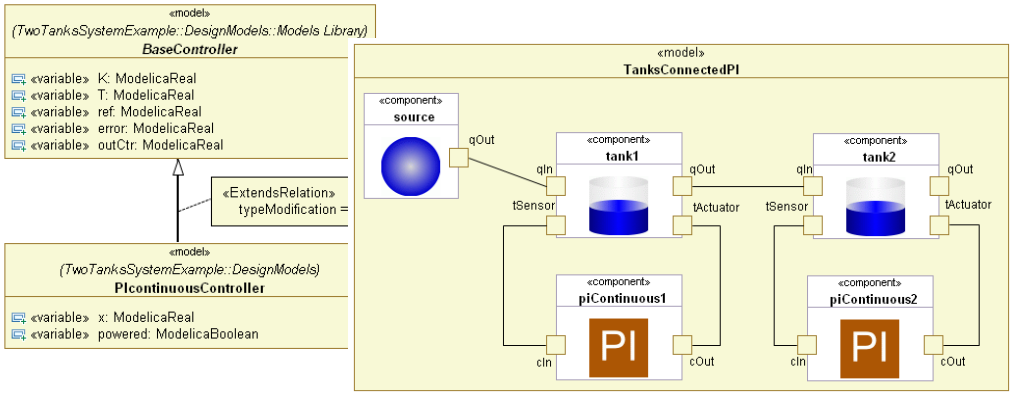


Behavior

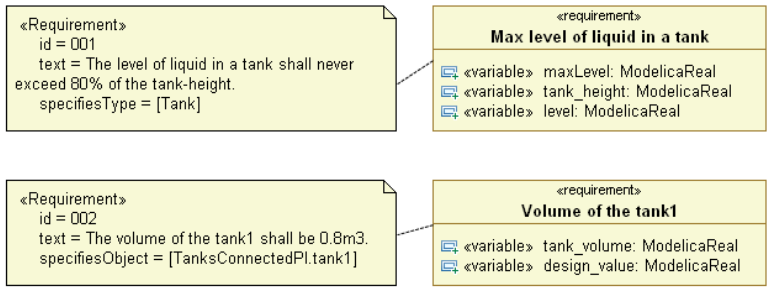


Introduction: ModelicaML Graphical Notation

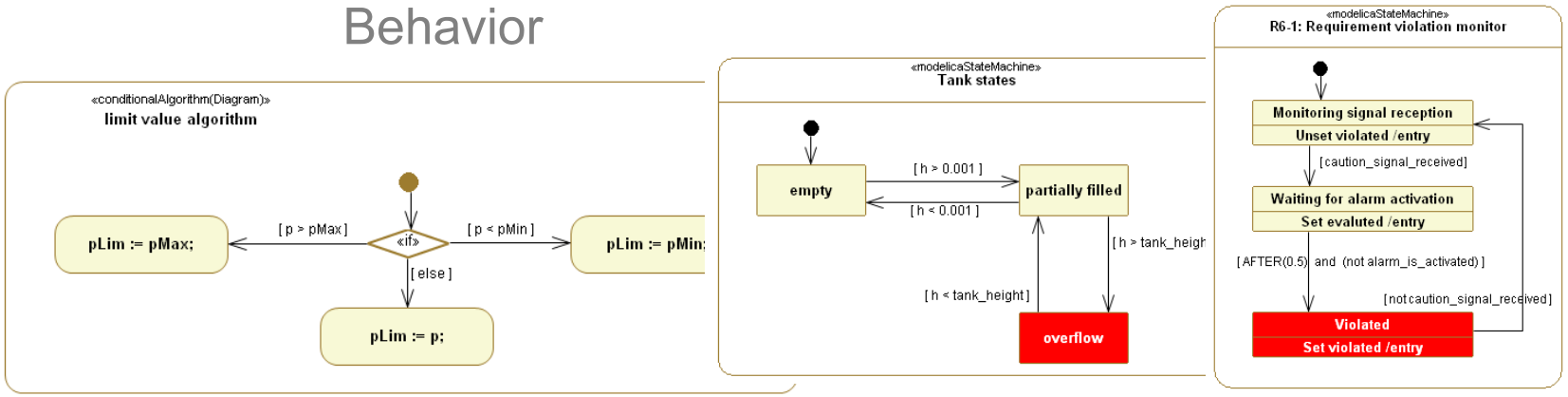
Structure



Requirements

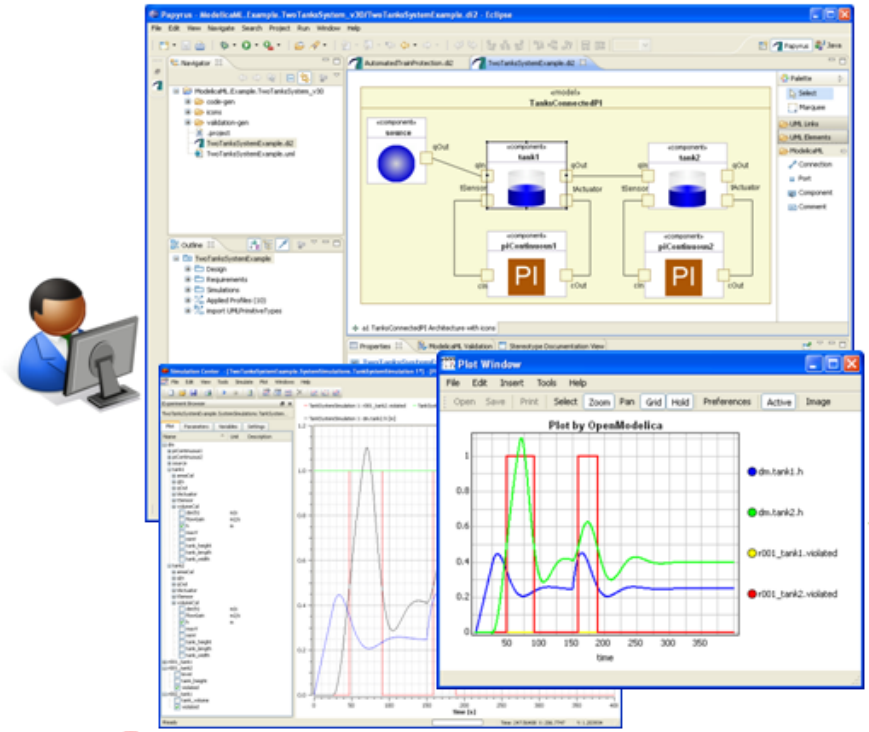


Behavior



Introduction: ModelicaML Concept

1 System Modeling with ModelicaML



2 Modelica Code Generation

```
within TwoTanks;

function limitValue

  input Real pMin;
  input Real pMax;
  output Real pLim;
  input Real p;

  algorithm
  // code generated from the Activity "algorithm (diagram)"
  // Activity.name: "algorithm (diagram)"
    if p < pMin then
      pLim := pMin; // OpaqueAction.name: "pLim := pMin;"
    elseif p > pMax then
```

```
within DesignModels;

model TanksConnectedPI

  TwoTanksSystemExample.DesignModels.LiquidSource source(flowLevel = 0.02);
  TwoTanksSystemExample.DesignModels.Tank tank1;
  TwoTanksSystemExample.DesignModels.Tank tank2(tank_length = 1);
  TwoTanksSystemExample.DesignModels.PIcontinuousController piContinuous1(ref = 0.25);
  TwoTanksSystemExample.DesignModels.PIcontinuousController piContinuous2(ref = 0.4);

  equation
    connect(source.qOut, tank1.qIn);
    connect(piContinuous1.cIn, tank1.tSensor);
    connect(piContinuous1.cOut, tank1.tActuator);
    connect(tank1.qOut, tank2.qIn);
    connect(piContinuous2.cIn, tank2.tSensor);
    connect(piContinuous2.cOut, tank2.tActuator);

end TanksConnectedPI;
```

3 System Simulation with Modelica Tools