Seamlessly Integrating Software & Hardware Modelling for Large-Scale Systems

Toby Myers, Peter Fritzson and R. Geoff Dromey

BEHAVIOR ENGINEERING

MODELICA

Griffith UNIVERSITY

Linköping University

pelab

Overview

- The Software-Hardware Integration Problem
- A Brief Introduction to Behavior Engineering
- Integrating Modelica & BE Models
- Case Study: An Automated Train Protection System

Overview

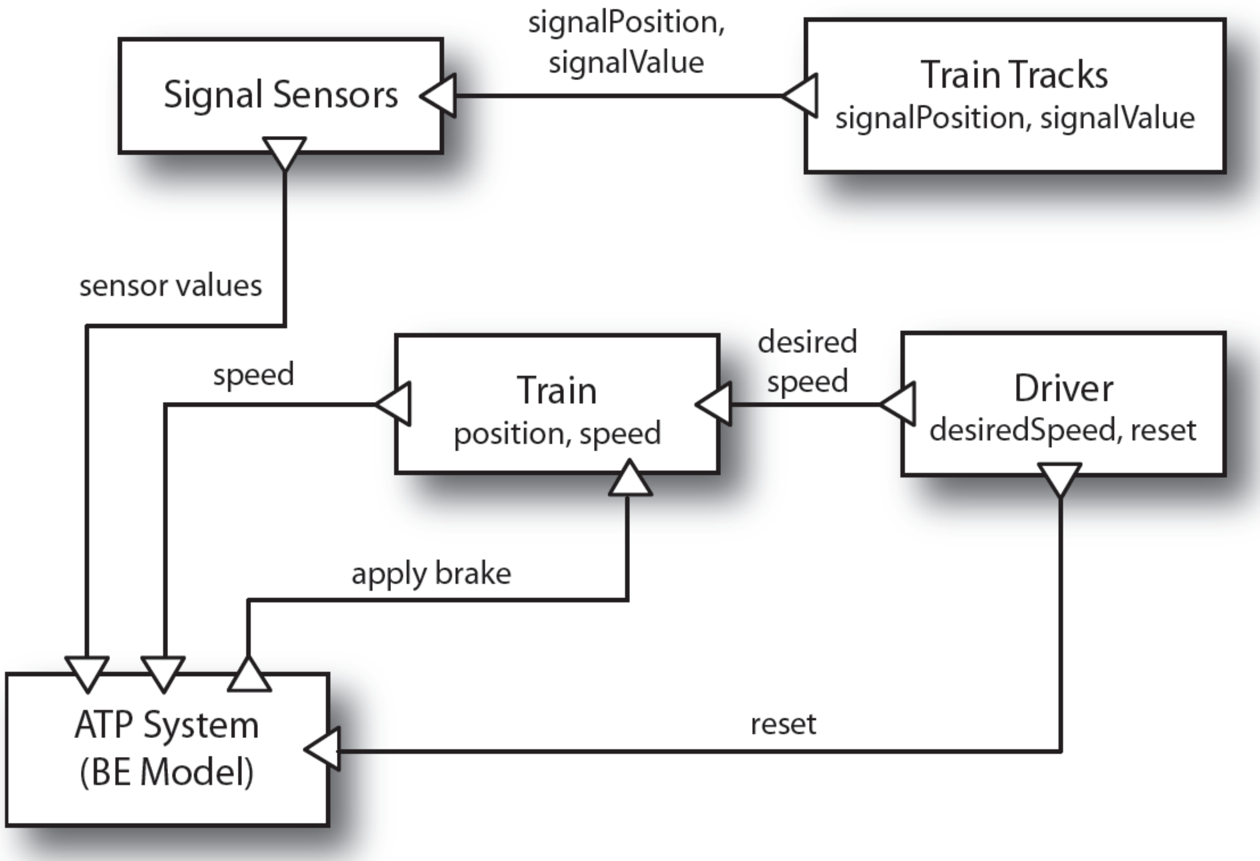
- **The Software-Hardware Integration Problem**
- A Brief Introduction to Behavior Engineering
- Integrating Modelica & BE Models
- Case Study: An Automated Train Protection System

The Software-Hardware Integration Problem

- At the **early stages** of system development, many **decisions** must be made about how the **system will be realised** as a combination of Software and Hardware
- **Requirements** of the system at these early stages **lack quantified and temporal information** so it is hard to make an informed decision
- **Changing** the partioning of software / hardware or how they interact later in development can be **time-consuming and costly**
- There is a potential for **errors** and **incompatibility** to be introduced as **software/hardware** specifications are created **independently**

Example: Model of Automated Train Protection System

An ATP System monitors train position and speed, and may apply brakes if the driver does not react in time



The Software-Hardware Integration Problem
Starting from System Requirements

Requirement	Description
R1	The ATP system is located on board the train. It involves a central controller and five boundary subsystems that manage the sensors, speedometer, brakes, alarm and a reset mechanism.
R2	The sensors are attached to the side of the train and detect information on the approach to track-side signals, i.e. they detect what the signal is displaying to the train driver.
R3	In order to reduce the effects of component failure three sensors are used. Each sensor generates a value in the range 0 to 3, where 0, 1 and 2 denote the danger, caution, and proceed signals respectively. The fourth sensor value, i.e. 3, is generated if an undefined signal is detected, e.g. may correspond to noise between the signal and the sensor.
R4	The sensor value returned to the ATP controller is calculated as the majority of the three sensor readings. If there does not exist a majority then an undefined value is returned to the ATP controller.
R5	If a proceed signal is returned to the ATP controller then no action is taken with respect to the train's brakes.
R6	If a caution signal is returned to the ATP controller then the alarm is enabled within the driver's cab. Furthermore, once the alarm has been enabled, if the speed of the train is not observed to be decreasing then the ATP controller activates the train's braking system.
R7	In the case of a danger signal being returned to the ATP controller, the braking system is immediately activated and the alarm is enabled. Once enabled, the alarm is disabled if a proceed signal is subsequently returned to the ATP controller.
R8	Note that if the braking system is activated then the ATP controller ignores all sensor input until the system has been reset.
R9	If enabled, the reset mechanism deactivates the train's brakes and disables the alarm.

Table 1. Requirements of the ATP system

The Software-Hardware Integration Problem

Interaction with Sensors …

Requirement	Description
R1	The ATP system is located on board the train. It involves a central controller and five boundary subsystems that manage the sensors, speedometer, brakes, alarm and a reset mechanism.
R2	The sensors are attached to the side of the train and detect information on the approach to track-side signals, i.e. they detect what the signal is displaying to the train driver.
R3	In order to reduce the effects of component failure three sensors are used. Each sensor generates a value in the range 0 to 3, where 0, 1 and 2 denote the danger, caution, and proceed signals respectively. The fourth sensor value, i.e. 3, is generated if an undefined signal is detected, e.g. may correspond to noise between the signal and the sensor.
R4	The sensor value returned to the ATP controller is calculated as the majority of the three sensor readings. If there does not exist a majority then an undefined value is returned to the ATP controller.
R5	If a proceed signal is returned to the ATP controller then no action is taken with respect to the train's brakes.
R6	If a caution signal is returned to the ATP controller then the alarm is enabled within the driver's cab. Furthermore, once the alarm has been enabled, if the speed of the train is not observed to be decreasing then the ATP controller activates the train's braking system.
R7	In the case of a danger signal being returned to the ATP controller, the braking system is immediately activated and the alarm is enabled. Once enabled, the alarm is disabled if a proceed signal is subsequently returned to the ATP controller.
R8	Note that if the braking system is activated then the ATP controller ignores all sensor input until the system has been reset.
R9	If enabled, the reset mechanism deactivates the train's brakes and disables the alarm.

Table 1. Requirements of the ATP system

How often does this need to be checked?

Decreasing by how much?

The Software-Hardware Integration Problem

Interaction with Actuators …

Requirement	Description
R1	The ATP system is located on board the train. It involves a central controller and five boundary subsystems that manage the sensors, speedometer, brakes, alarm and a reset mechanism.
R2	The sensors are attached to the side of the train and detect information on the approach to track-side signals, i.e. they detect what the signal is displaying to the train driver.
R3	In order to reduce the effects of component failure three sensors are used. Each sensor generates a value in the range 0 to 3, where 0, 1 and 2 denote the danger, caution, and proceed signals respectively. The fourth sensor value, i.e. 3, is generated if an undefined signal is detected, e.g. may correspond to noise between the signal and the sensor.
R4	The sensor value returned to the ATP controller is calculated as the majority of the three sensor readings. If there does not exist a majority then an undefined value is returned to the ATP controller.
R5	If a proceed signal is returned to the ATP controller then no action is taken with respect to the train's brakes.
R6	If a caution signal is returned to the ATP controller then the alarm is enabled within the driver's cab. Furthermore, once the alarm has been enabled, if the speed of the train is not observed to be decreasing then the ATP controller activates the train's braking system.
R7	In the case of a danger signal being returned to the ATP controller, the braking system is immediately activated and the alarm is enabled. Once enabled, the alarm is disabled if a proceed signal is subsequently returned to the ATP controller.
R8	Note that if the braking system is activated then the ATP controller ignores all sensor input until the system has been reset.
R9	If enabled, the reset mechanism deactivates the train's brakes and disables the alarm.

Table 1. Requirements of the ATP system

What response time is realistically acceptable?

The Software-Hardware Integration Problem

Software / Hardware Partitioning …

Requirement	Description
R1	The ATP system is located on board the train. It involves a central controller and five boundary subsystems that manage the sensors, speedometer, brakes, alarm and a reset mechanism.
R2	The sensors are attached to the side of the train and detect information on the approach to track-side signals, i.e. they detect what the signal is displaying to the train driver.
R3	In order to reduce the effects of component failure three sensors are used. Each sensor generates a value in the range 0 to 3, where 0, 1 and 2 denote the danger, caution, and proceed signals respectively. The fourth sensor value, i.e. 3, is generated if an undefined signal is detected, e.g. may correspond to noise between the signal and the sensor.
R4	The sensor value returned to the ATP controller is calculated as the majority of the three sensor readings. If there does not exist a majority then an undefined value is returned to the ATP controller.
R5	If a proceed signal is returned to the ATP controller then no action is taken with respect to the train's brakes.
R6	If a caution signal is returned to the ATP controller then the alarm is enabled within the driver's cab. Furthermore, once the alarm has been enabled, if the speed of the train is not observed to be decreasing then the ATP controller activates the train's braking system.
R7	In the case of a danger signal being returned to the ATP controller, the braking system is immediately activated and the alarm is enabled. Once enabled, the alarm is disabled if a proceed signal is subsequently returned to the ATP controller.
R8	Note that if the braking system is activated then the ATP controller ignores all sensor input until the system has been reset.
R9	If enabled, the reset mechanism deactivates the train's brakes and disables the alarm.

Table 1. Requirements of the ATP system

Perform in Software or Hardware?

The Software-Hardware Integration Problem

The Environment in which the system will exist …

Requirement	Description
R1	The ATP system is located on board the train. It involves a central controller and five boundary subsystems that manage the sensors, speedometer, brakes, alarm and a reset mechanism.
R2	The sensors are attached to the side of the train and detect information on the approach to track-side signals, i.e. they detect what the signal is displaying to the train driver.
R3	In order to reduce the effects of component failure three sensors are used. Each sensor generates a value in the range 0 to 3, where 0, 1 and 2 denote the danger, caution, and proceed signals respectively. The fourth sensor value, i.e. 3, is generated if an undefined signal is detected, e.g. may correspond to noise between the signal and the sensor.
R4	The sensor value returned to the ATP controller is calculated as the majority of the three sensor readings. If there does not exist a majority then an undefined value is returned to the ATP controller.
R5	If a proceed signal is returned to the ATP controller then no action is taken with respect to the train's brakes.
R6	If a caution signal is returned to the ATP controller then the alarm is enabled within the driver's cab. Furthermore, once the alarm has been enabled, if the speed of the train is not observed to be decreasing then the ATP controller activates the train's braking system.
R7	In the case of a danger signal being returned to the ATP controller, the braking system is immediately activated and the alarm is enabled. Once enabled, the alarm is disabled if a proceed signal is subsequently returned to the ATP controller.
R8	Note that if the braking system is activated then the ATP controller ignores all sensor input until the system has been reset.
R9	If enabled, the reset mechanism deactivates the train's brakes and disables the alarm.

Table 1. Requirements of the ATP system

How far apart are the signals?

What are the characteristics of the train?

Will it be deployed on many different types of trains?

Overview

- The Software-Hardware Integration Problem
- **A Brief Introduction to Behavior Engineering**
- Integrating Modelica & BE Models
- Case Study: An Automated Train Protection System

A Brief Introduction to Behavior Engineering

Behavior Engineering for Requirements Analysis

- 5 Large-scale industry projects
  - In Defence, Transportation, Banking and Finance
  - Between 800-1250 requirements
- All previously reviewed with respective organisations internal review processes
- Defect detection rate approximately 2 to 3 times that of traditional ad-hoc, checklist-based, and scenario-based reading techniques reported in Porter, 1998.

Requirements Evaluation Using Behavior Trees

Findings from Industry

Daniel Powell

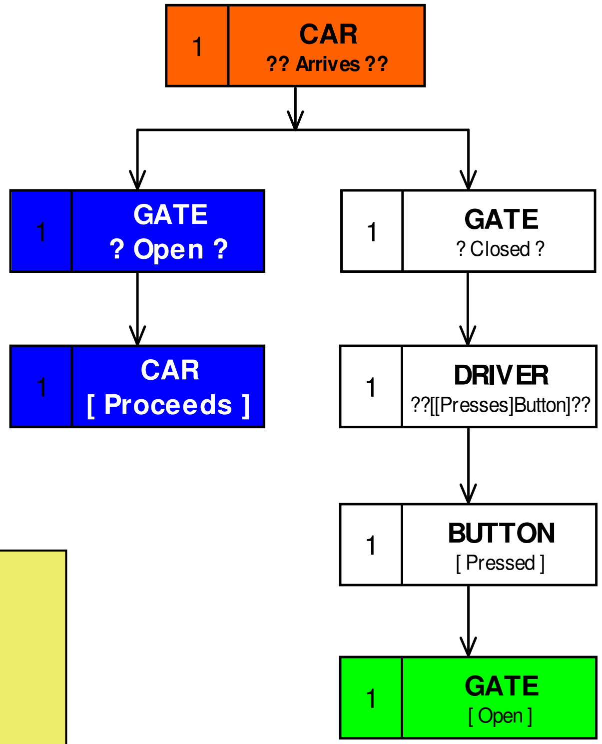
http://aswec07.cs.latrobe.edu.au/5.zip

Formalization - Requirements Translation

Functional Requirement

When a car is arrives,
if the gate is open the car proceeds,
otherwise if the gate is closed, when
the driver presses the button
it causes the gate to open

Behavior Tree



Formalization
- clarification and preservation of intent
- strict use of original vocabulary
- removes ambiguity, aliases, etc
- aids stakeholder validation, understanding
- approaches repeatability

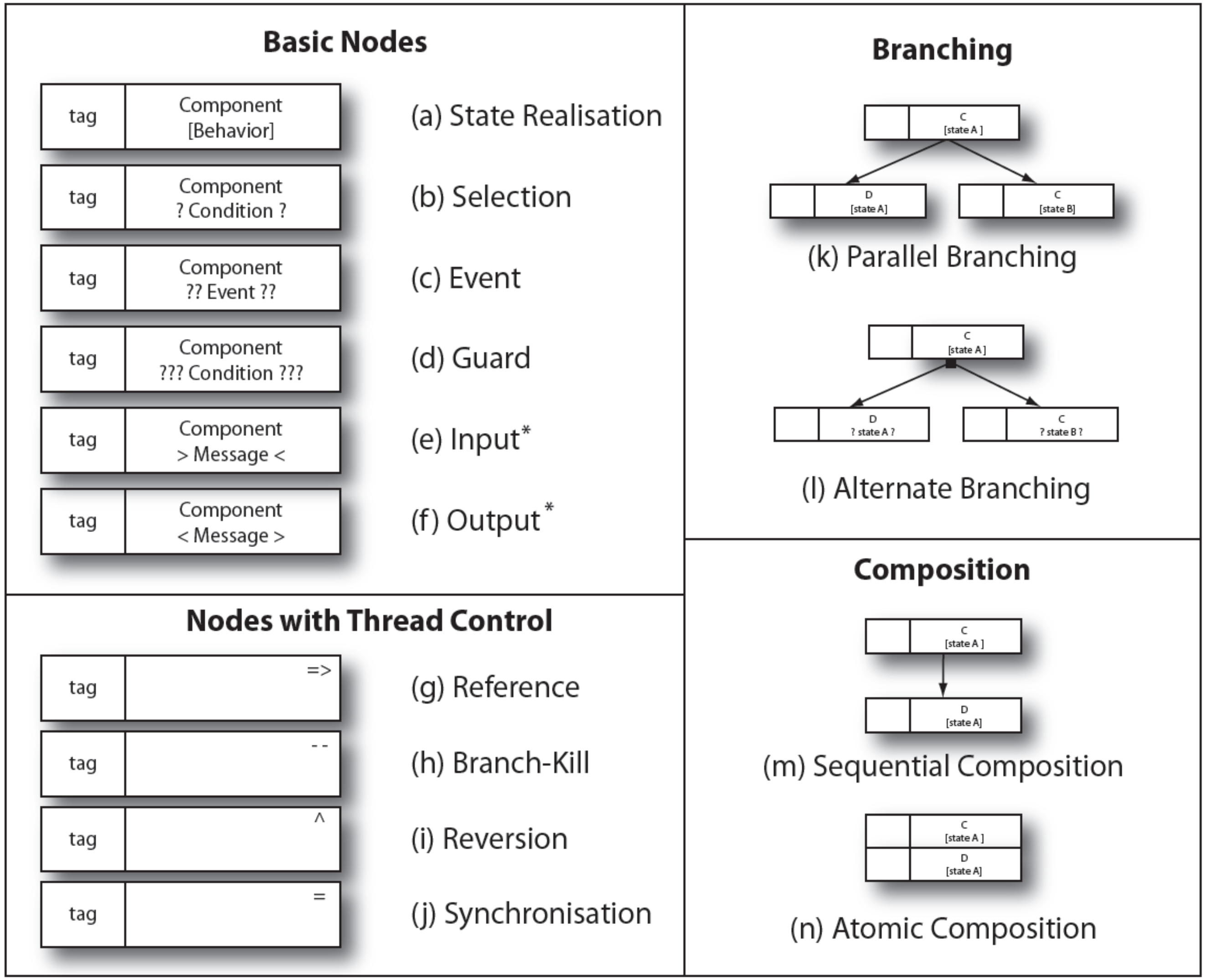
A Brief Introduction to Behavior Engineering

- Behavior Engineering (BE) acronyms …

Behavior Modeling Process (BMP)	Behavior Modeling Language (BML)	
	Behavior Trees (BT)	**Composition Trees (CT)**
Requirements Translation	Requirement Behavior Trees (RBTs)	Requirement Composition Tree (RCT)
Requirements Integration	Integrated Behavior Tree (IBT)	Integrated Composition Tree (ICT)
System Specification	Model Behavior Tree (MBT)	Model Composition Tree (MCT)
System Design	Design Behavior Tree (DBT)	Design Composition Tree (DCT)

A Brief Introduction to Behavior Engineering

Summary of the Behavior Tree Notation



A Brief Introduction to Behavior Engineering

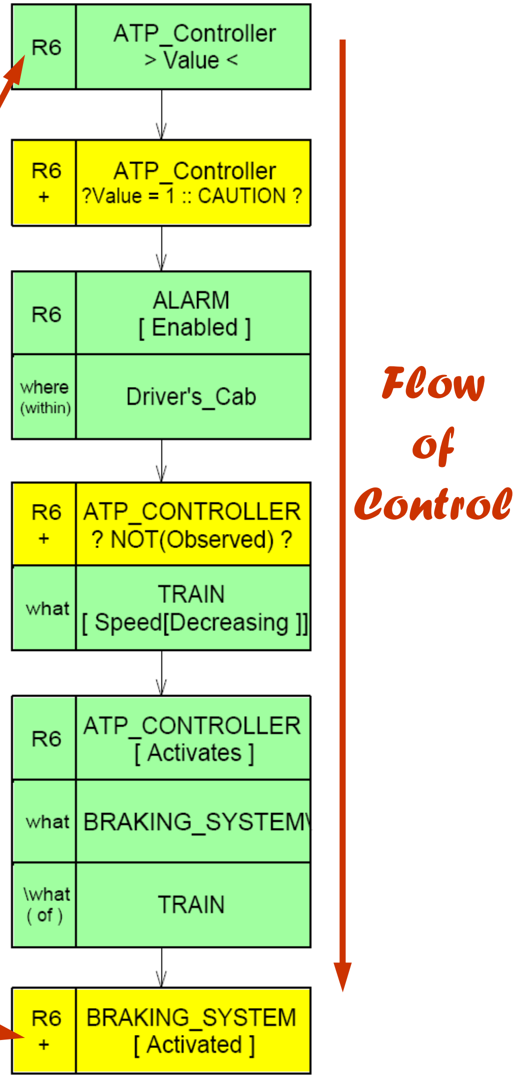
How to translate from a Requirement in Natural Language to an RBT

R6. If a caution signal is returned to the ATP controller then the alarm is enabled within the driver's cab. Furthermore, once the alarm has been enabled, if the speed of the train is not observed to be decreasing then the ATP controller activates the train's braking system.

The Tag traces these Behavior Tree nodes back to Requirement 6.

A '+' and a yellow color denote the behavior is implied by the requirements

Red color denotes behavior is missing in the requirements



A Brief Introduction to Behavior Engineering

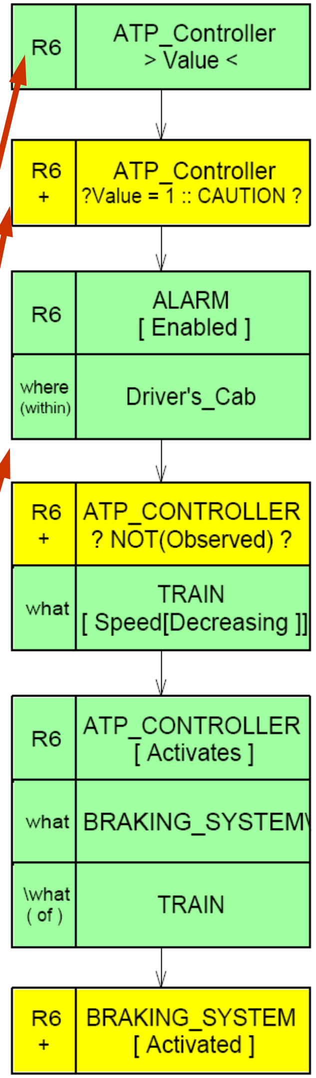
How to translate from a Requirement in Natural Language to an RBT

R6. If a caution signal is returned to the ATP controller then the alarm is enabled within the driver's cab. Furthermore, once the alarm has been enabled, if the speed of the train is not observed to be decreasing then the ATP controller activates the train's braking system.

ATP Controller receives a value from another component

Check if the value is a caution signal

If it is, enable the Alarm. To maintain the intent of the original requirement, use a relation to show the Alarm is enabled in the Driver's Cab.



A Brief Introduction to Behavior Engineering

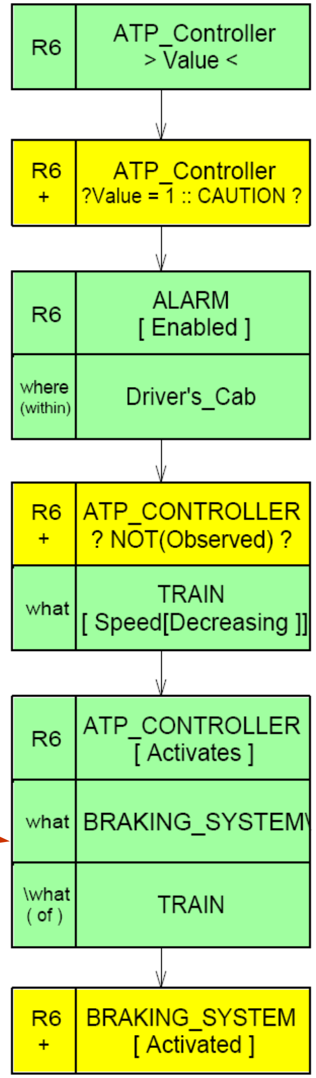
How to translate from a Requirement in Natural Language to an RBT

R6. If a caution signal is returned to the ATP controller then the alarm is enabled within the driver's cab. Furthermore, once the alarm has been enabled, if the speed of the train is not observed to be decreasing then the ATP controller activates the train's braking system.

It is implied the ATP Controller must observe whether the Train's speed is decreasing.

If the Train isn't decreasing in speed, the ATP Controller activates the Braking System of the Train.

.. Which results in the Braking System being Activated



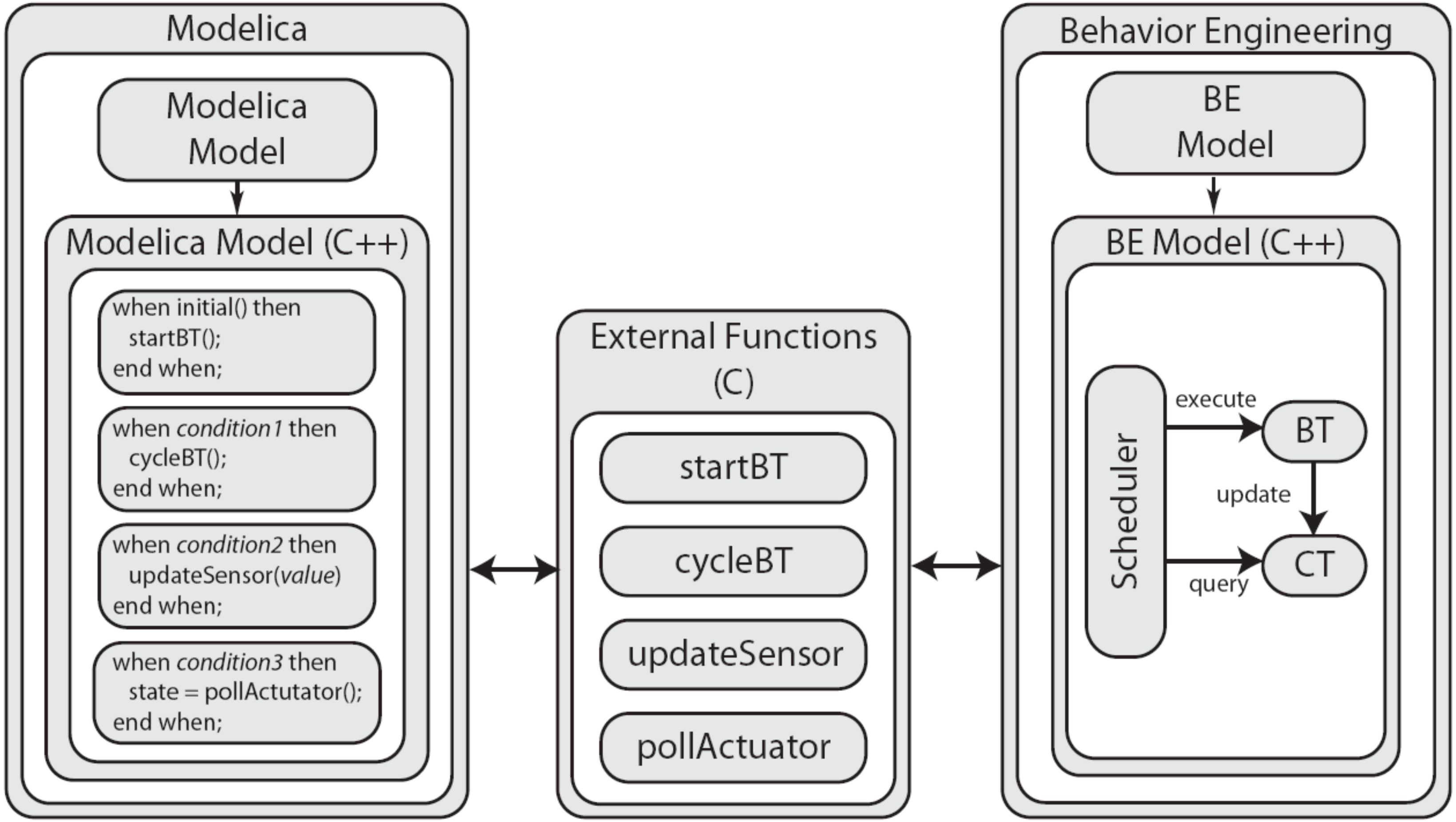
Overview

- The Software-Hardware Integration Problem
- A Brief Introduction to Behavior Engineering
- **Integrating Modelica & BE Models**
- Case Study: An Automated Train Protection System

The Software-Hardware Integration Problem

- Integration of Modelica and BE models occurs after the models are compiled into C/C++ source files.
- Uses Modelica external functions mapped to C source code which link to the ‘C++’ implementation of the BE model.
- The Modelica model is responsible for managing all interactions with the BE model.
  - When to execute the BE Model
  - When to send Sensor Information
  - When to receive Actuator Information

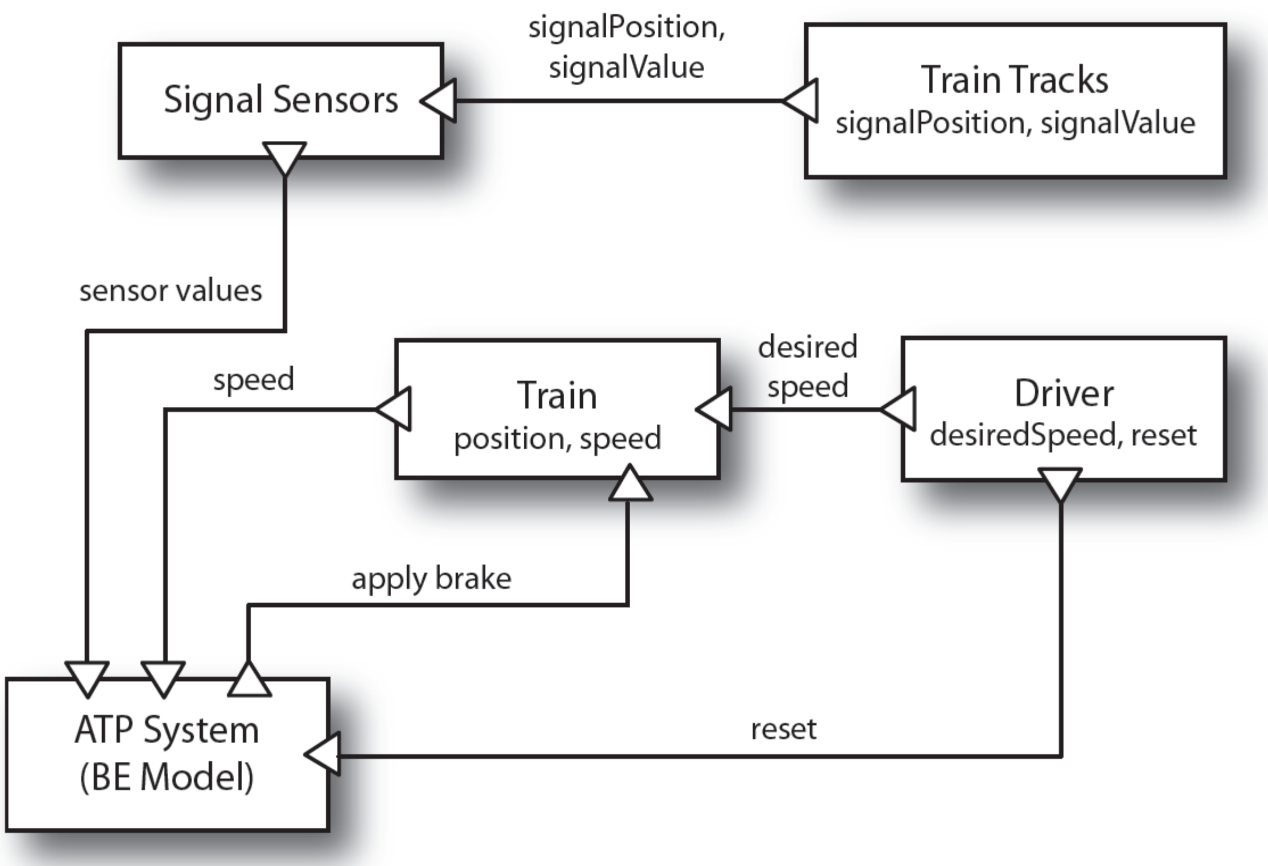
Integrating Modelica & BE Models



Overview

- The Software-Hardware Integration Problem
- A Brief Introduction to Behavior Engineering
- Integrating Modelica & BE Models
- **Case Study: An Automated Train Protection System**

Modelica Model of the ATP System (graphical view)



Case Study: An Automated Train Protection System Modelica Textual Model

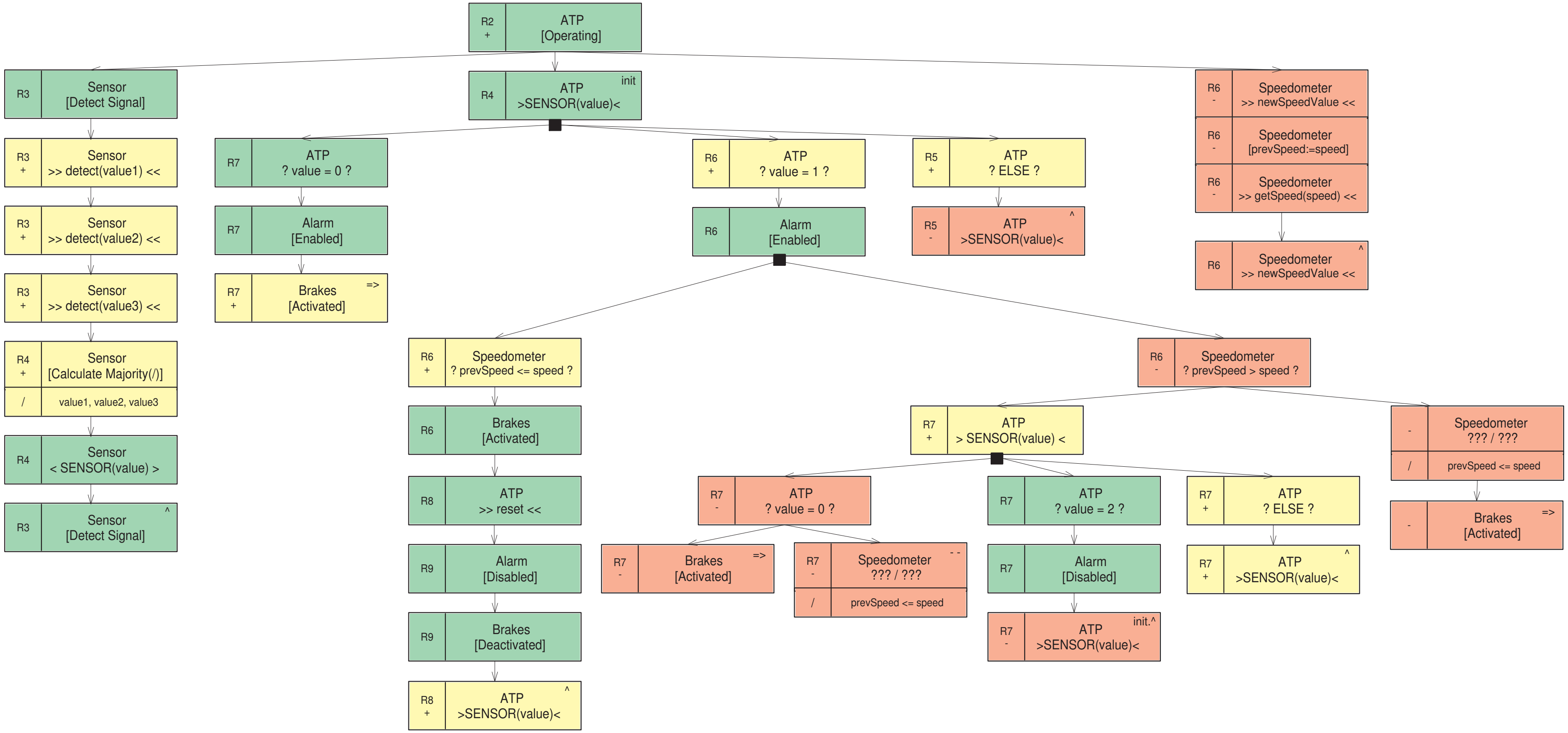
```
// External Functions included here

model Track
  discrete Integer currentSignalValue "Value of Last Signal
displayed to Driver/ATP System";
  parameter Real[:] signalPosition "Positions of Signals on
   the Track";
parameter Integer[:] signalValue "Values of Signals on
   the Track";
equation
  // Determine current signal value
end Track;

model Train
  Real s, v, m, maxSpeed, maxBrakeForce,
    maxAccelerationPower, maxAccelerationForce;
  parameter Real accPowerEff = 0.80 "Engine Efficiency in %";
equation
  maxAccelerationPower/accPowerEff  =
   maxAccelerationForce*v;
end Train;

record Driver
  Real desiredAcceleration;
  parameter Real[:] desiredSpeed;
  parameter Real[:] position;
end Driver;
```

```
model Main
// Define track, train, driver parameters
parameter Real[10] sensor1 = {0,0,1,2,0,0,2,2,0,0} "Sensor1
value at signalPosition";
Real sensor1Reading "Current Sensor1 reading";
// Similar for Sensor 2 & 3
Real fa, fd, doBrake(start=0), minAccelerationForce,
desiredAccelerationForce;
discrete Boolean clock1, clock2, ...;
// Define clock frequencies
equation
when initial() then startBT(0); end when;
when clock1 then cycleBT(0); end when;
when clock2 then doBrake = if (train1.v >= 0) then
getBrake(0) else 0;
// if driver reset's ATP send message
// if signal changes send new sensor values
fa = if doBrake>0 then 0
elseif // ensure not over maximum Acceleration force
else desiredAccelerationForce;
fd = if doBrake>0 then train1.maxBrakeForce else 0;
a = (fa-fd)/train1.m;
der(v) = a;
der(track1.s) = train1.v;
// if train passing signal then update sensors
// determine driver's desired acceleration (a = (desiredSpeed -
train1.v)/ (2*distance))
end Main;
```

Case Study: An Automated Train Protection System

BE Model of the ATP System

(yellow: implied from requirements, red: missing)



ERROR: invalidrestore
OFFENDING COMMAND: restore

STACK:

-savelevel-
-savelevel-