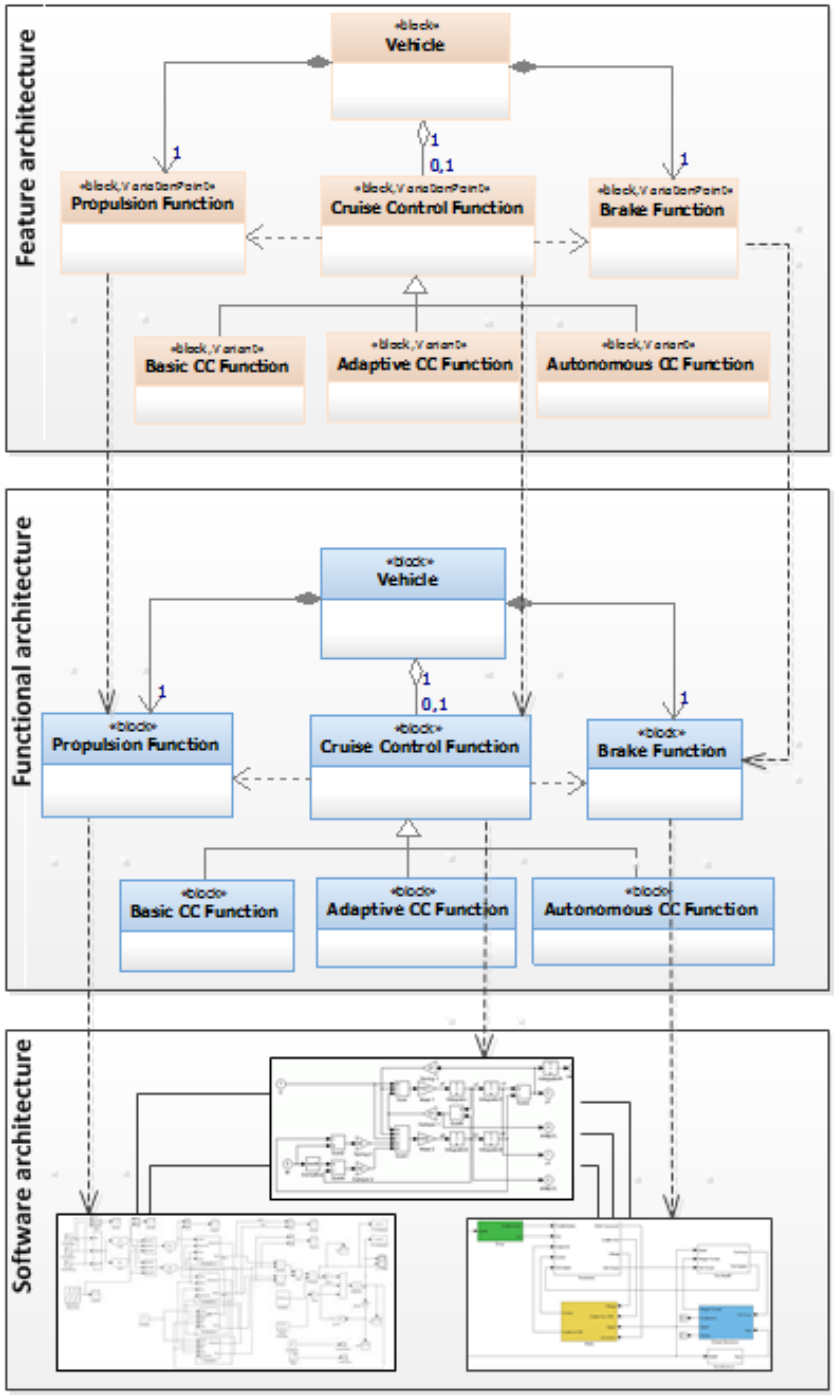
# Introduction

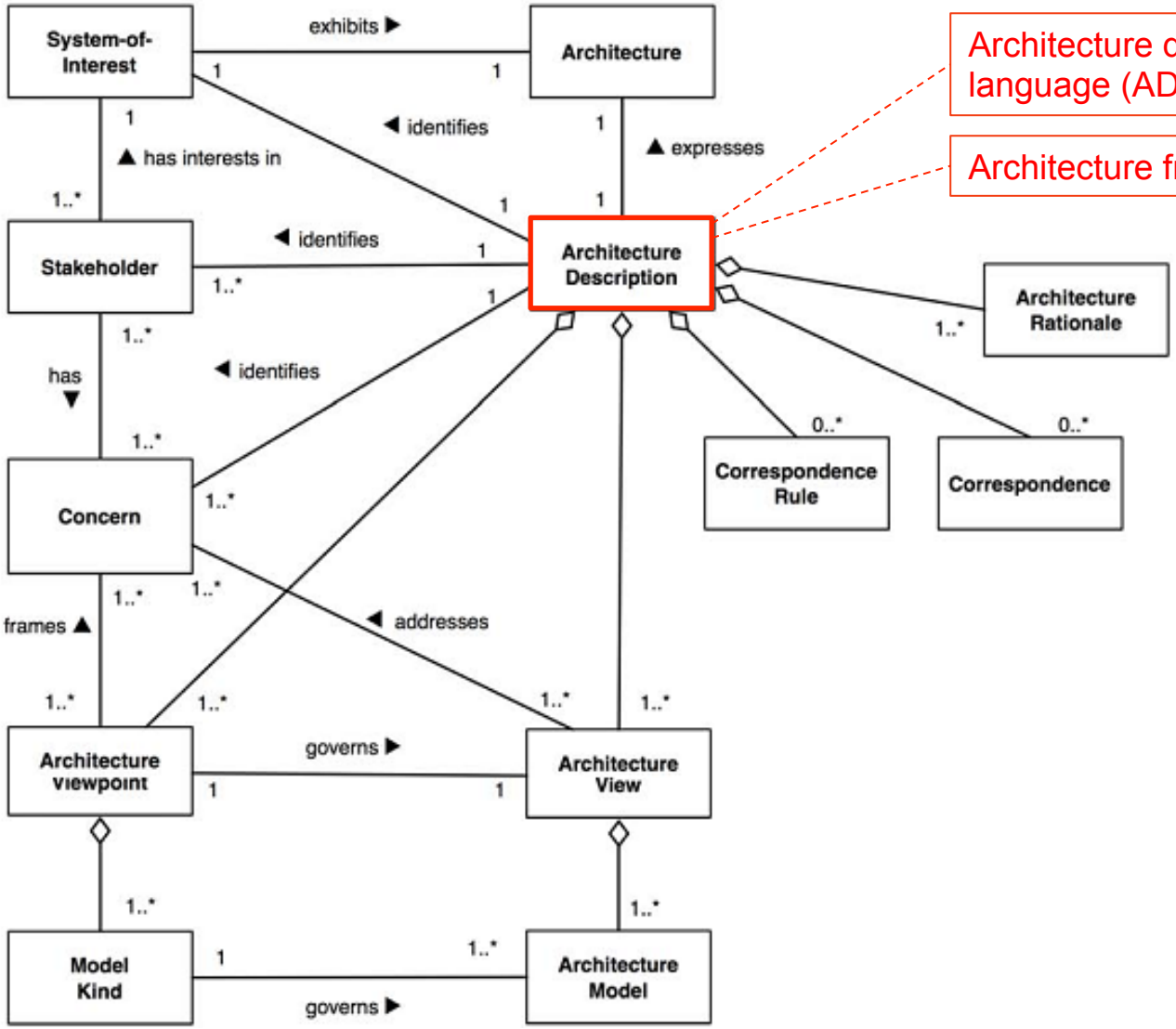
**More and more complex electronics and software:**

- **To enable innovation**
- **To decrease costs**
- **To fulfill legal needs (e.g. CO2 emission) etc.**

***How to ensure consistency among different views?***



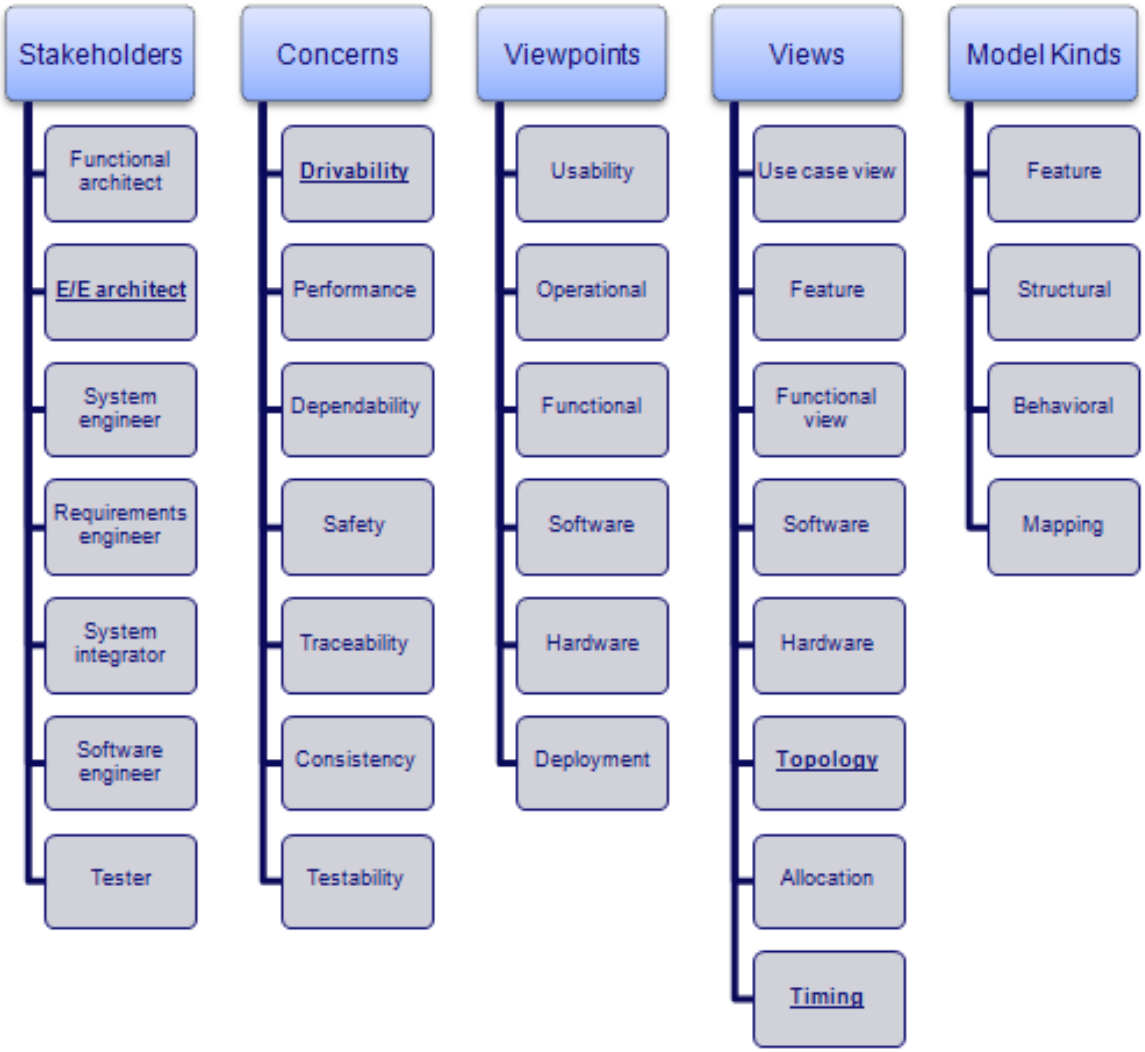
# ISO/IEC/IEEE 42010 Conceptual Model of an Architecture Description



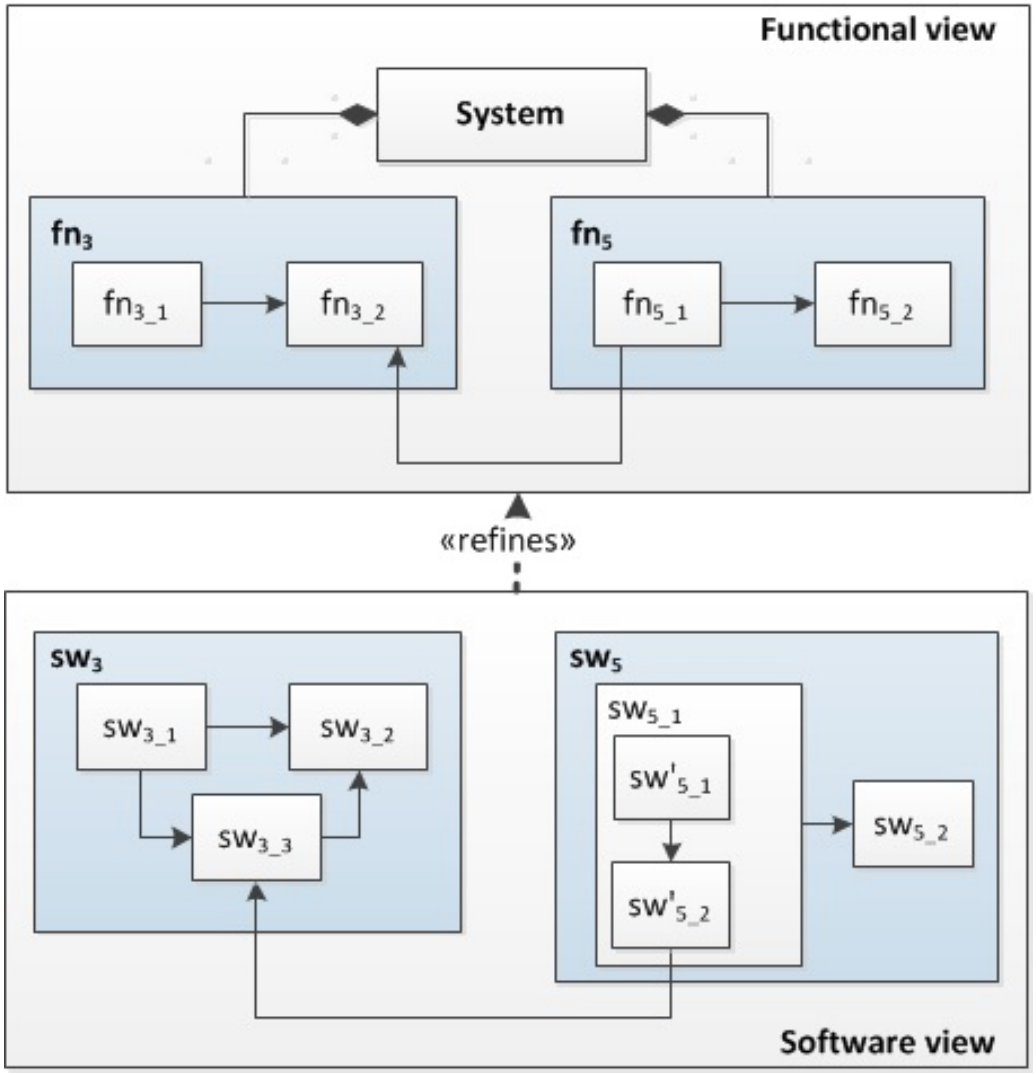

Architecture description language (ADL)

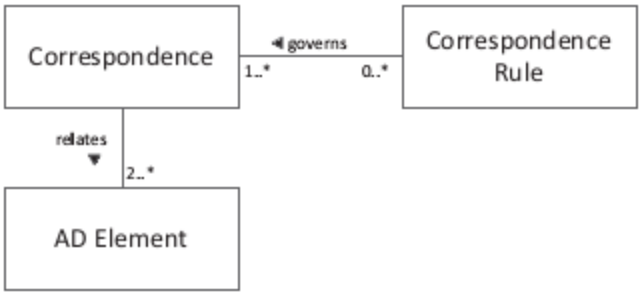
Architecture framework

# Architecture Framework for Automotive Systems (AFAS)



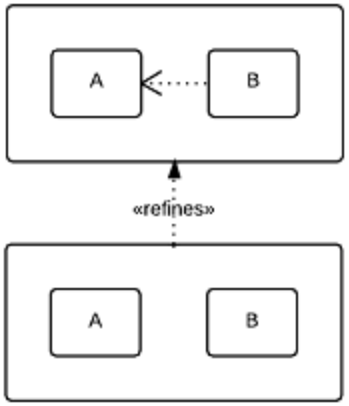
# Architectural correspondence



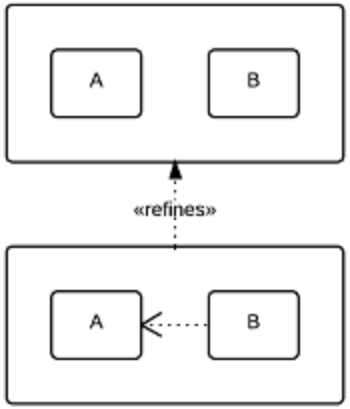




| *SW(S) refines FN(S)* See rule: R1 | |
|---|---|
| $sw_3$ | $fn_3$ |
| $sw_5$ | $fn_5$ |

# Correspondence Rules




(a) Absence

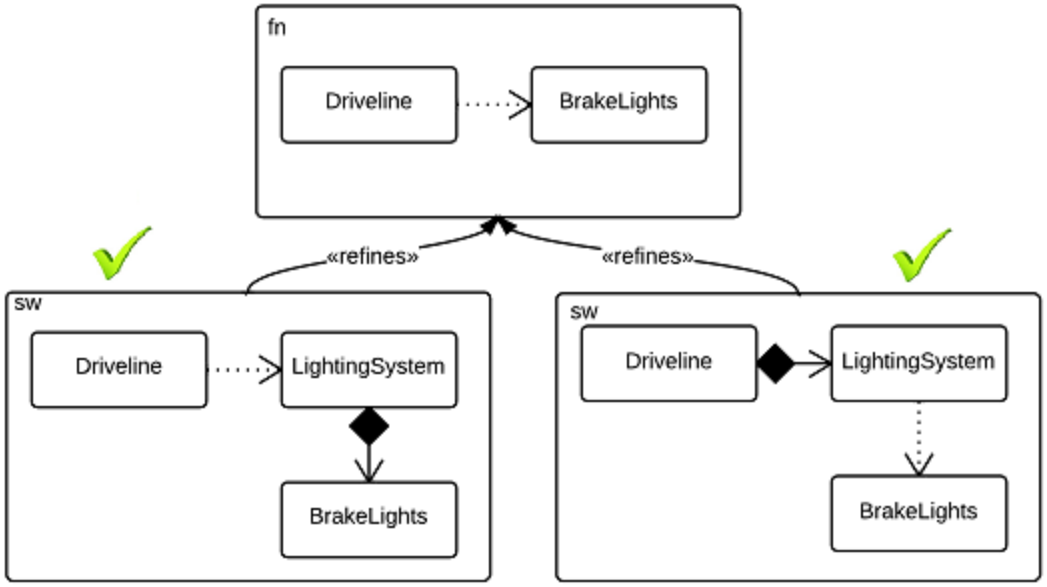



(b) Divergence

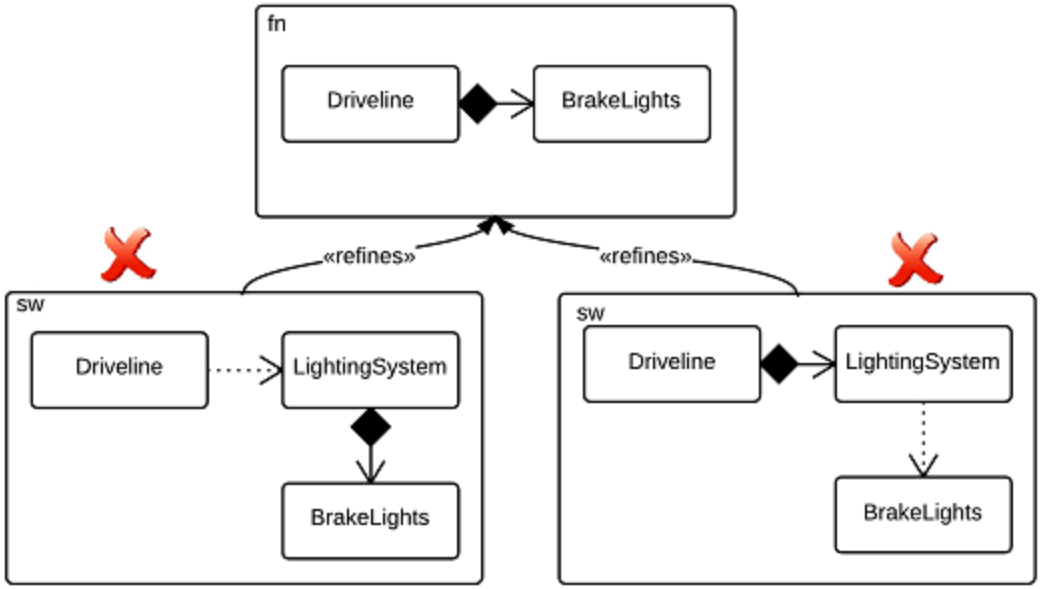
$$absence_{rel_i}(A,B) \Leftrightarrow rel_i^{+}(A,B) \wedge \neg \widetilde{rel_i}^{\uparrow}(A,B)$$

$$divergence_{rel_i}(A,B) \Leftrightarrow \neg rel_i^{+}(A,B) \wedge \widetilde{rel_i}^{\uparrow}(A,B)$$

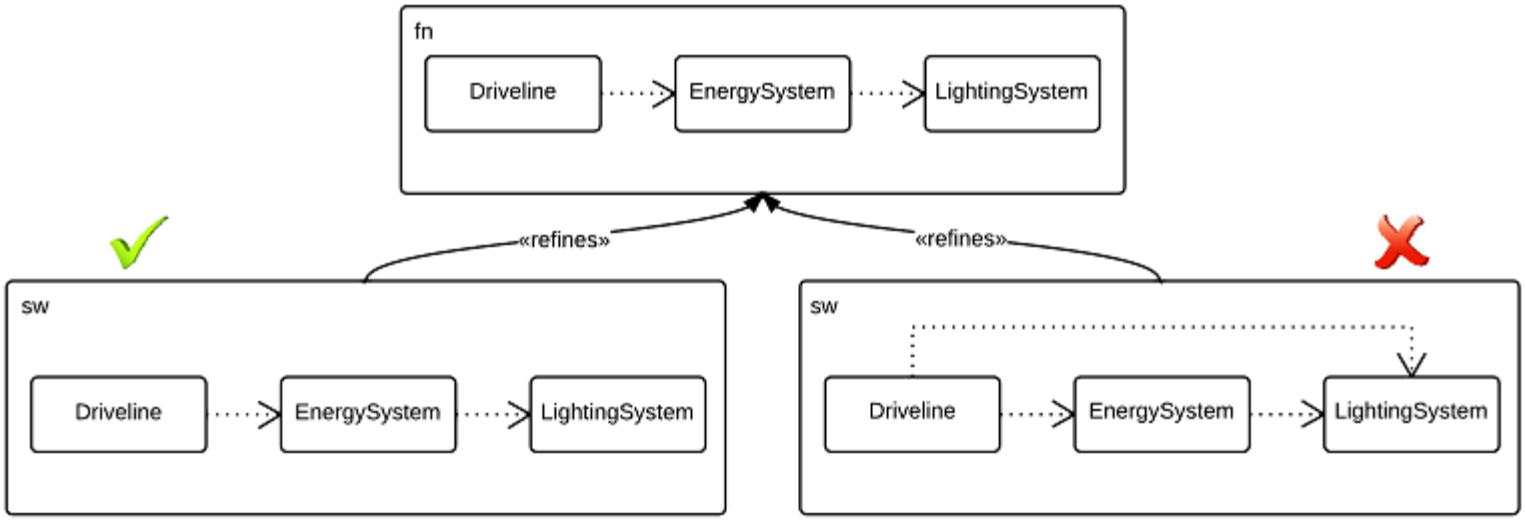
# Consistency Semantics




Semantic differences between dependency and composition refinements

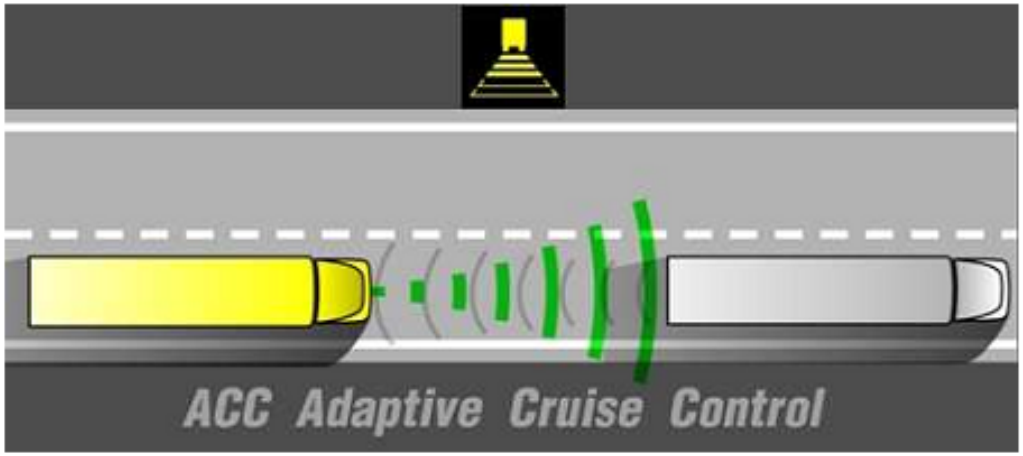


# Consistency Semantics



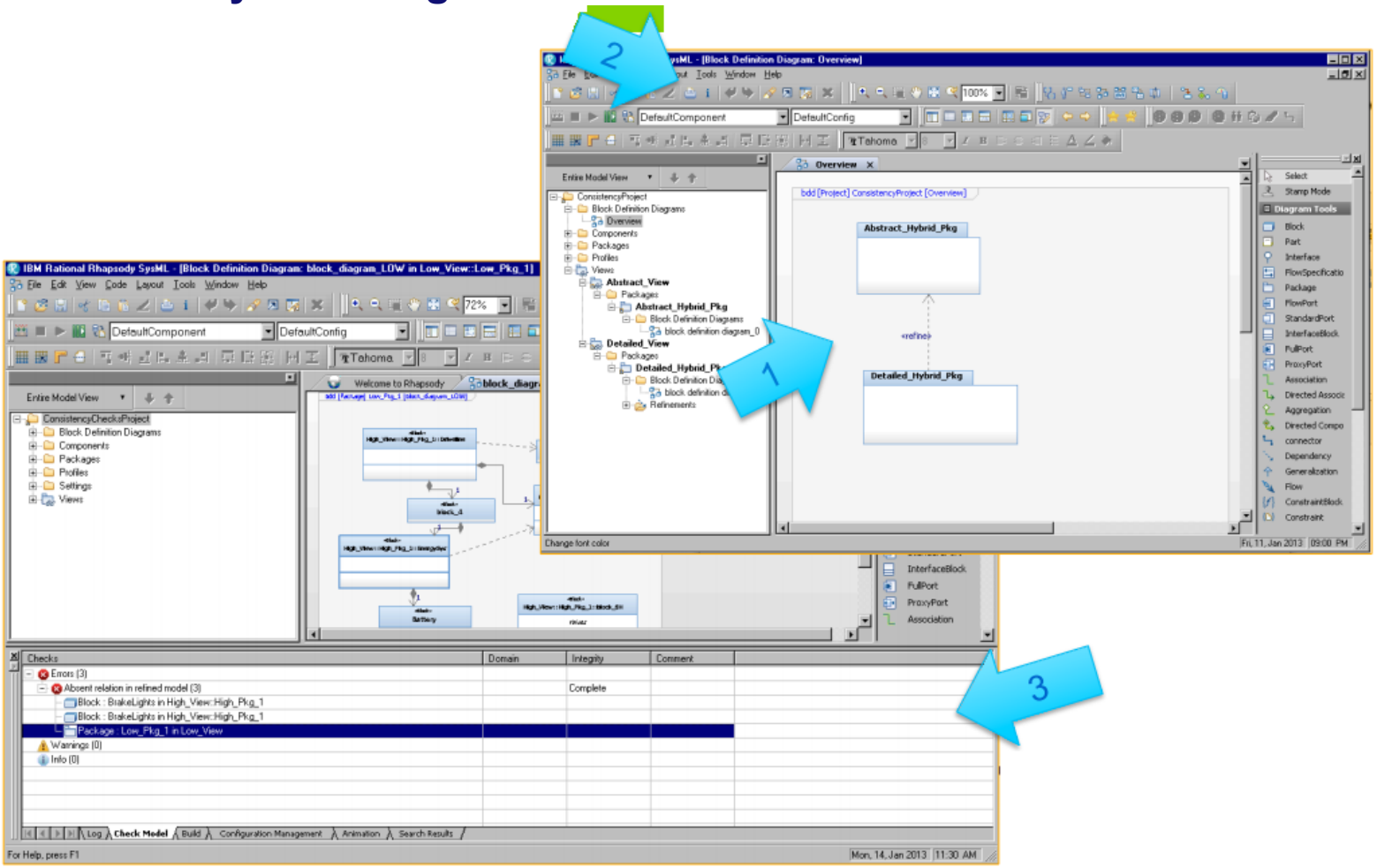
# Evaluation

- **Adaptive Cruise Control**

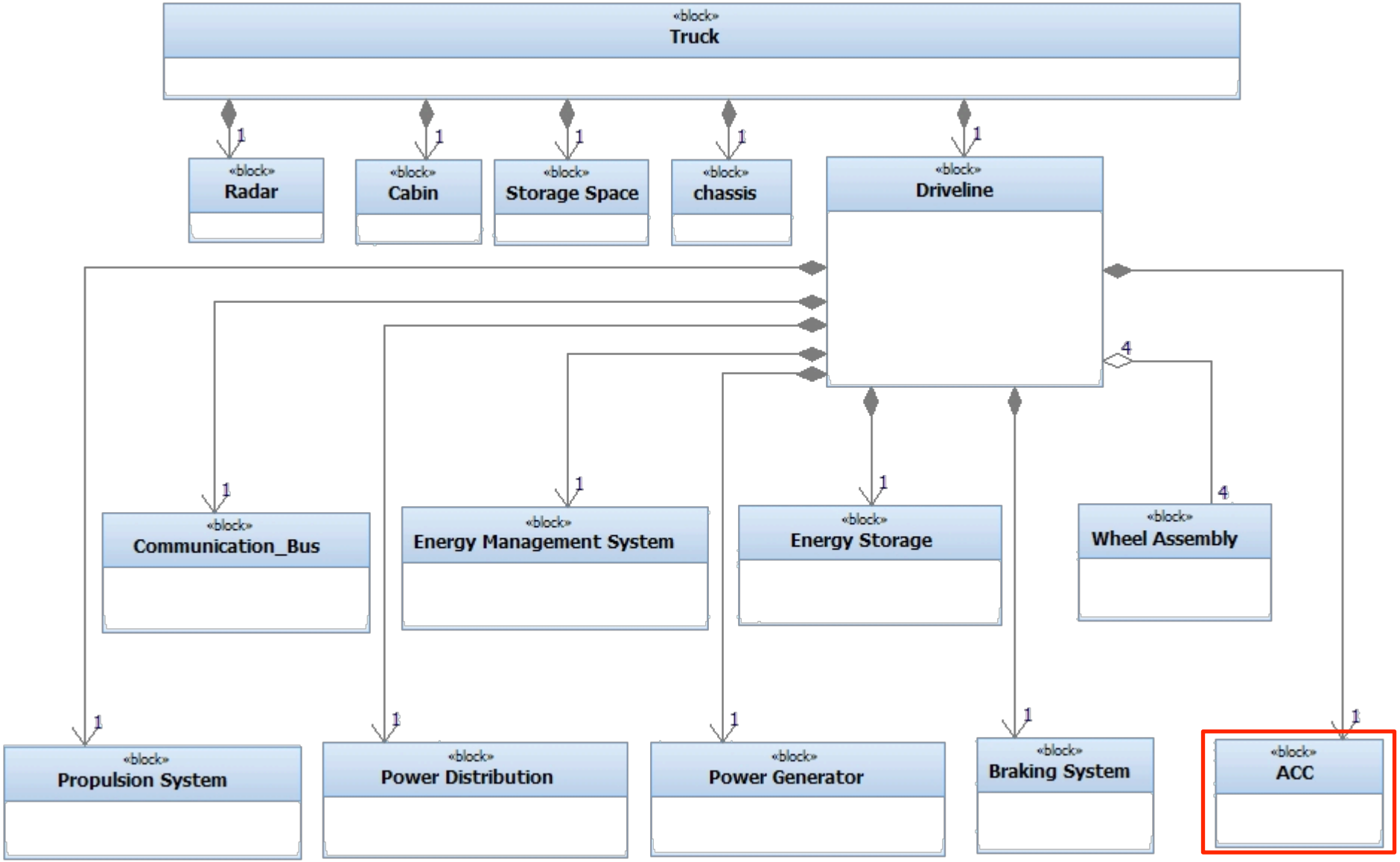


# Consistency checking tool

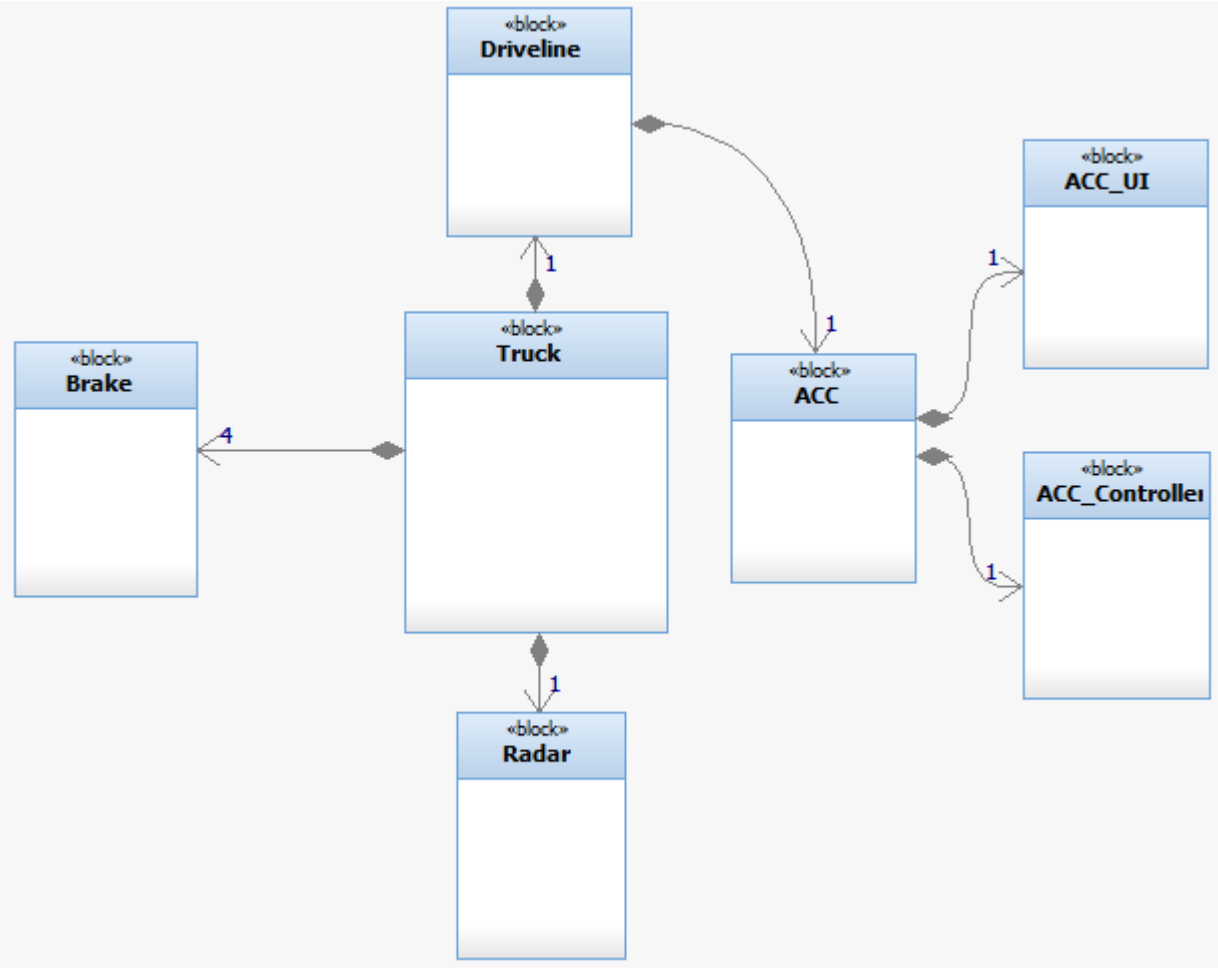
## A consistency-checking for automotive architectural models:



# Evaluation on Adaptive Cruise Control

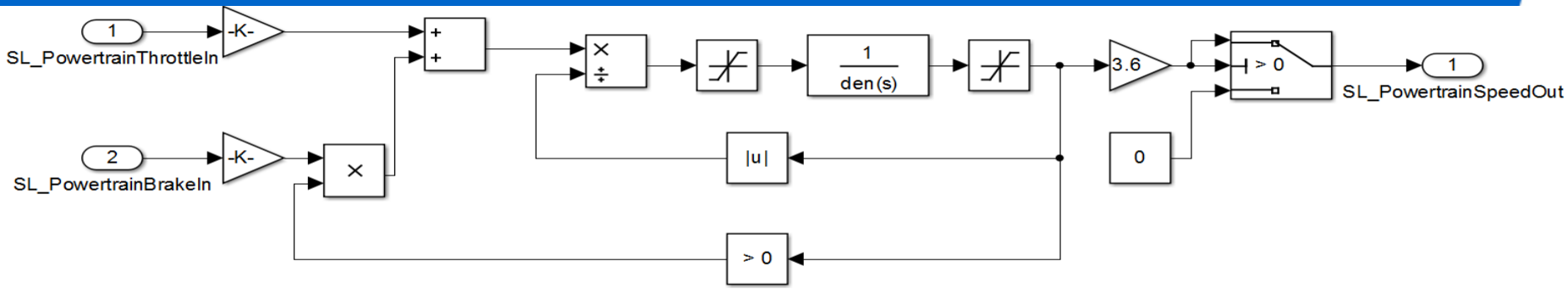


# ACC system model

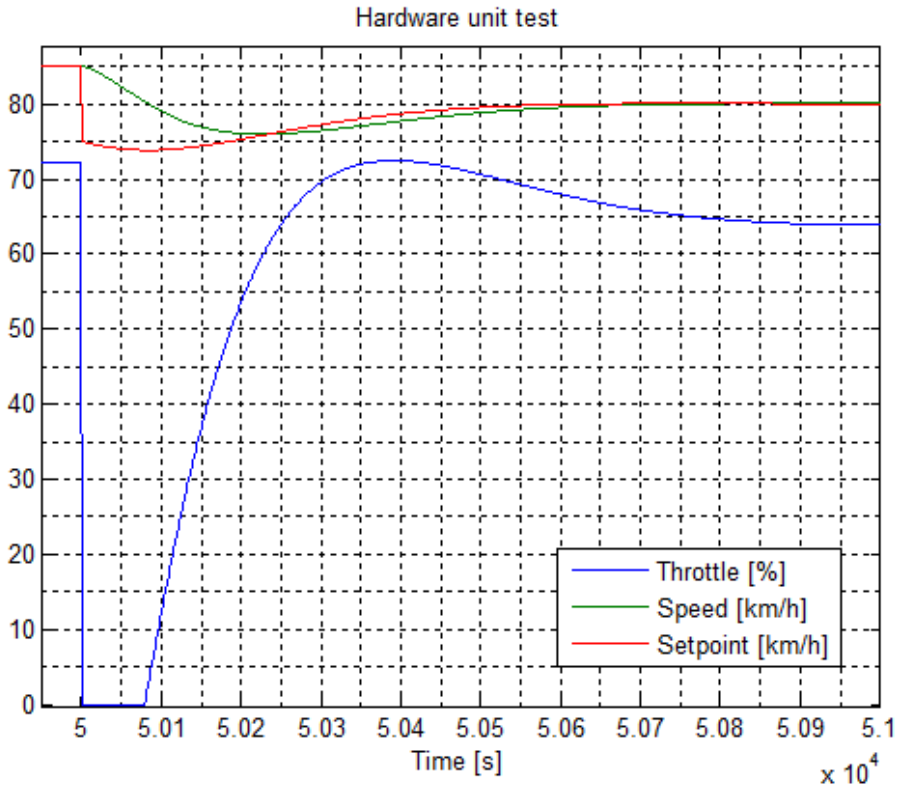



- ACC is split into:
  - ACC_UI (SW)
  - ACC_Controller (HW)
- SW-model is made in IBM Rhapsody
- HW-model is made in Matlab Simulink

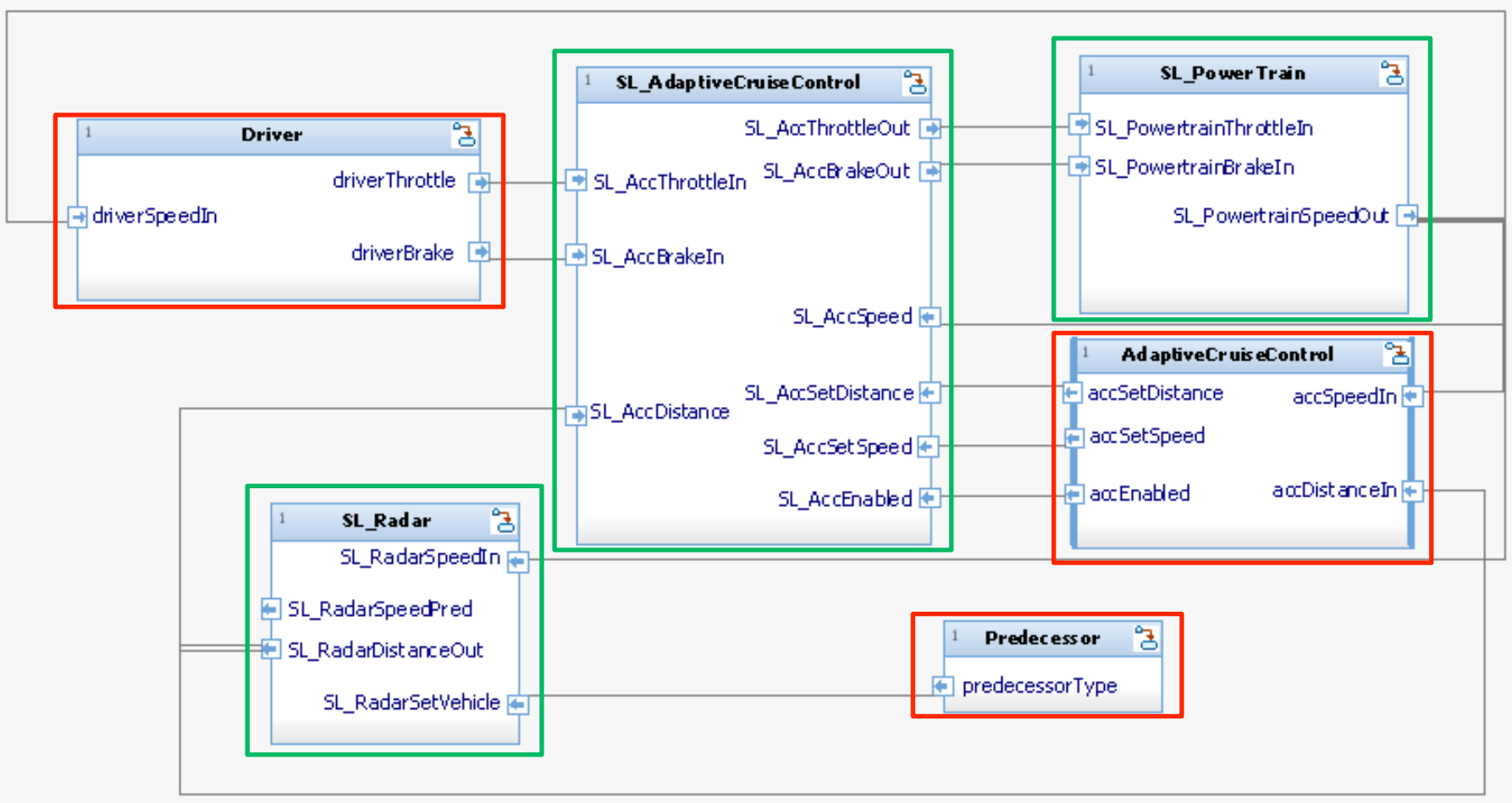
# ACC hardware models




- 3 Hardware models
  - Radar model
  - Powertrain model
  - ACC model
- Hardware models are unit tested in Matlab

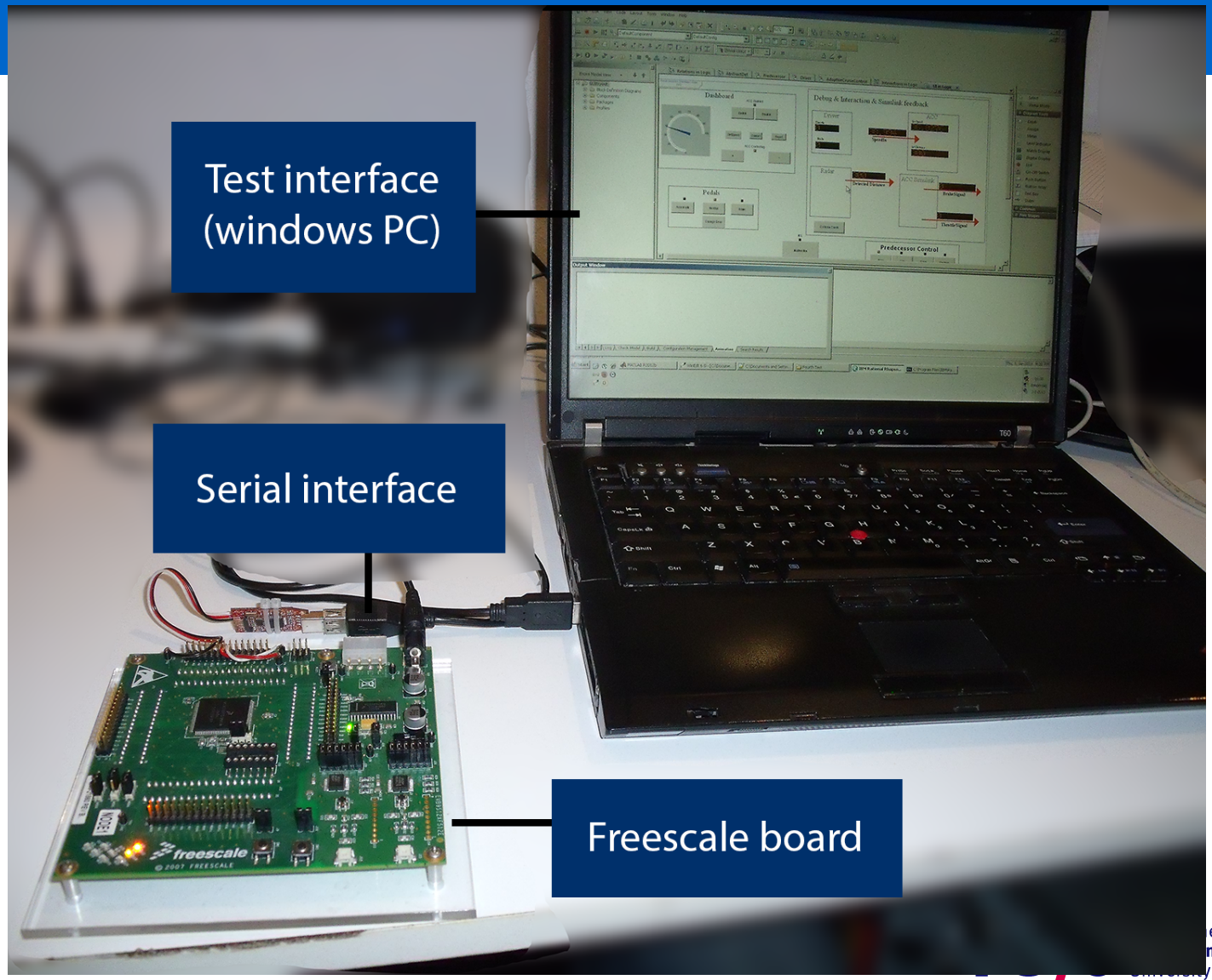


# System integration




- **Software**
- **Hardware**

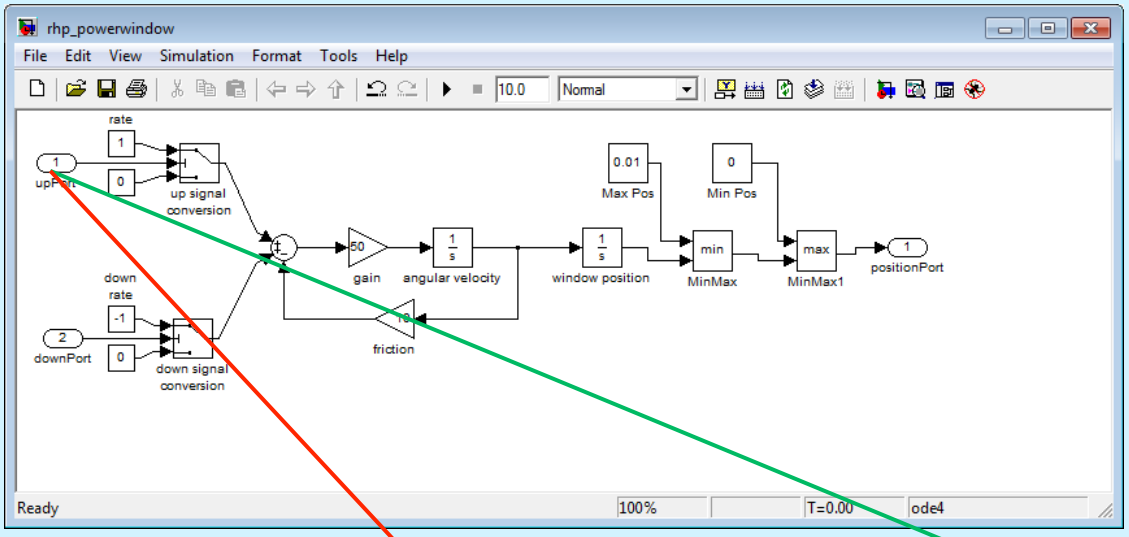
# Evaluation


Test interface
(windows PC)
Serial interface
Freescale board

# Conclusion and Future work

- Automotive Architectural Views defined
- Automotive architecture consistency rules defined IBM Rhapsody tool plugin developed
- Need to formalize other correspondence rules
- Need to support consistency between automotive ADLs (e.g. between SysML and MATLAB/Simulink, Stateflow)

# SysML/Simulink Model Sync

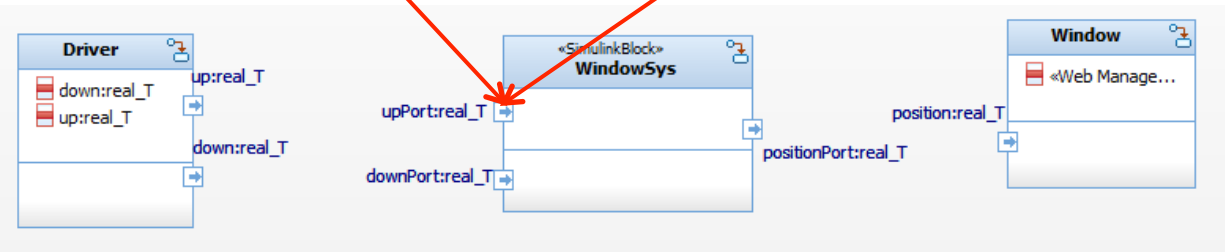



**Old generated code snippet**

```
/* Switch: '<Root>/up signal conversion' incorporates:
 *  Constant: '<Root>/c0'
 *  Constant: '<Root>/up rate'
 *  Inport: '<Root>/up'
 */
if (rhp_powerwindow_U.up >= rhp_powerwindow_P.upsignalconversion_Threshold)
{
  rtb_upsignalconversion = rhp_powerwindow_P.uprate_Value;
} else {
  rtb_upsignalconversion = rhp_powerwindow_P.c0_Value;
}
```

**New generated code snippet**

```
/* Switch: '<Root>/up signal conversion' incorporates:
 *  Constant: '<Root>/c0'
 *  Constant: '<Root>/up rate'
 *  Inport: '<Root>/upPort'
 */
if (rhp_powerwindow_U.upPort >= rhp_powerwindow_P.upsignalconversion_Threshold)
{
  u = rhp_powerwindow_P.uprate_Value;
} else {
  u = rhp_powerwindow_P.c0_Value;
}
```




- **Input/Output ports of a SimulinkBlock is updated after importing both updated Simulink model and the newly generated code (only updated Simulink MDL file is not enough).**


**Contact for comments and collaboration:**

**Tel:** +31(0)402475052
**Email:** y.dajsuren@tue.nl
**Address:**
Eindhoven University of Technology
5612 AZ Eindhoven, The Netherlands