EMBEDDED SOFTWARE LABORATORY

RWTH AACHEN UNIVERSITY

Supporting evolving requirements in CPS by abstraction layers in the architecture

Stefan Kowalewski, Andre Stollenwerk

April 11th 2011, ACPS, Chicago, IL

Outline

- → Motivation: Lifelong evolving constraints
- Abstraction Layers in rapid control prototyping
- Model based generated code in medical engineering

Lifetime Adaption

- Fastened development process of embedded systems
  - Smart phones
  - Automotive assist systems
  - Biomedical engineering
  - Rapid prototyping (Fabbing @ home)
- Result in continuous adaption to variations of constraints
- Requirement changes during runtime may throw back to falling branch of V-model
- Changes can also result of usage of agile methods
- Moving boundary between design, development and operation

Design Constraints

- Small adoptions during design process enable fast adoptions during life-time

- Impossible aspects to predict during design time
  - All possible interactions
  - Use cases of a system’s life

→ Lifelong evolving requirements

Approach: introduction of abstraction layers

- Well defined interfaces between different modules
- Increased interoperability
- Lower effort in maintaining and lifetime development
- Automated data management
- Predictable code modules

Outline

- Motivation: Lifelong evolving constraints

→ Abstraction Layers in rapid control prototyping

- Model based generated code in medical engineering

Requirements

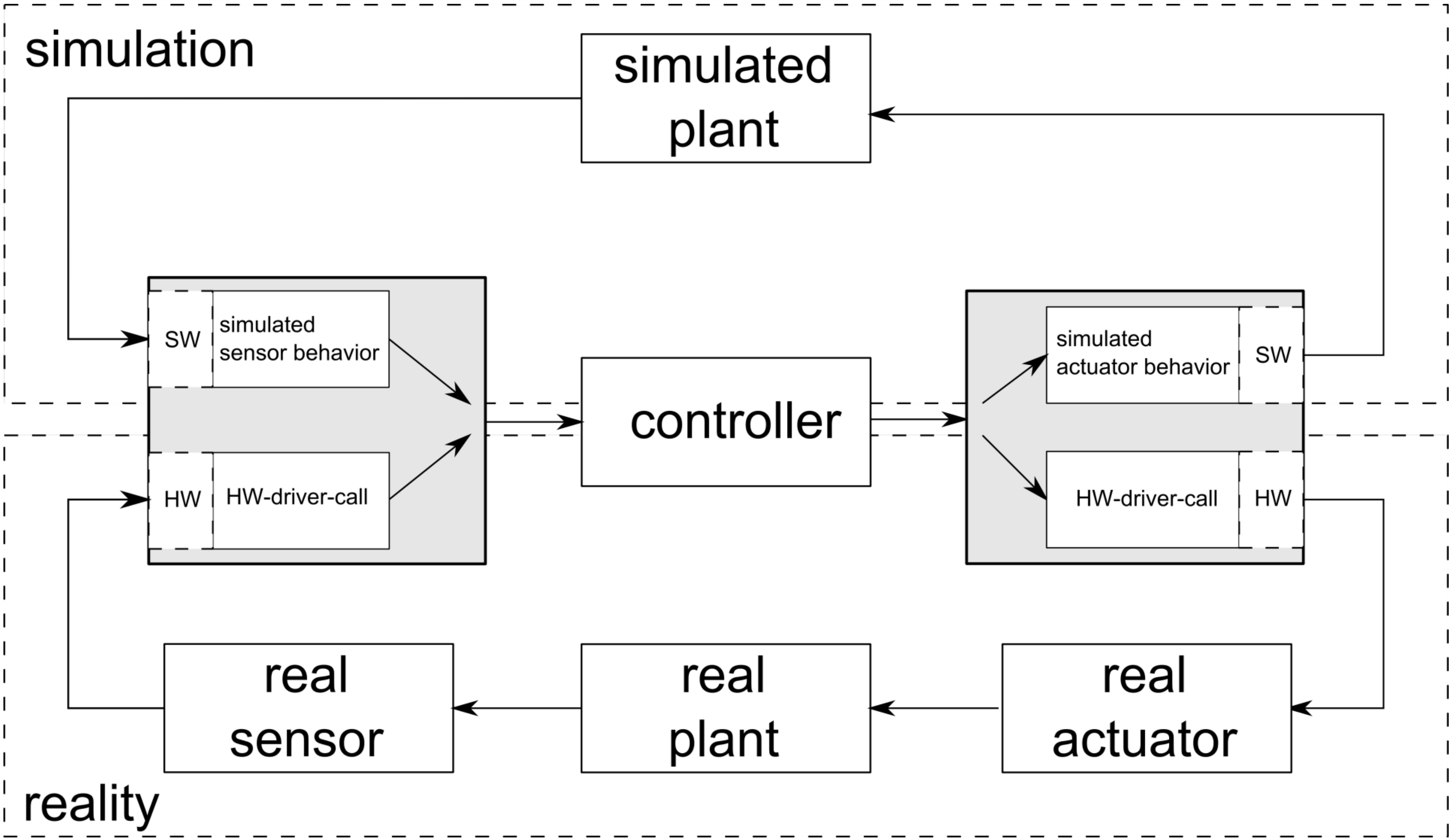
Software Engineering:

- reusability
- support of different modeling environments
- maintainability
- configurable sensors

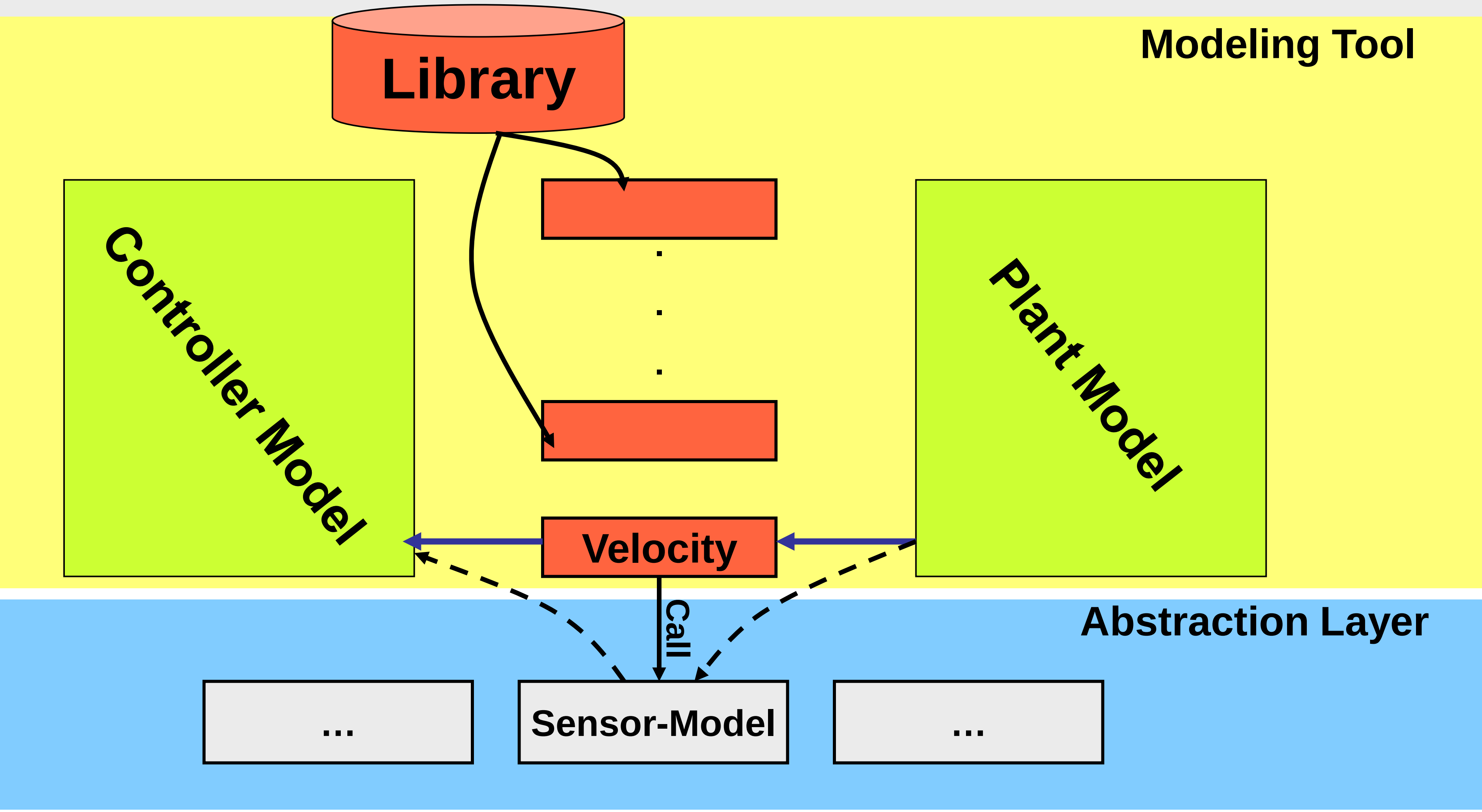
Control Engineering:

- modeling environment e.g. Matlab / Simulink
- ability to simulate
- ability to change sensors and actuators without reimplementing the model

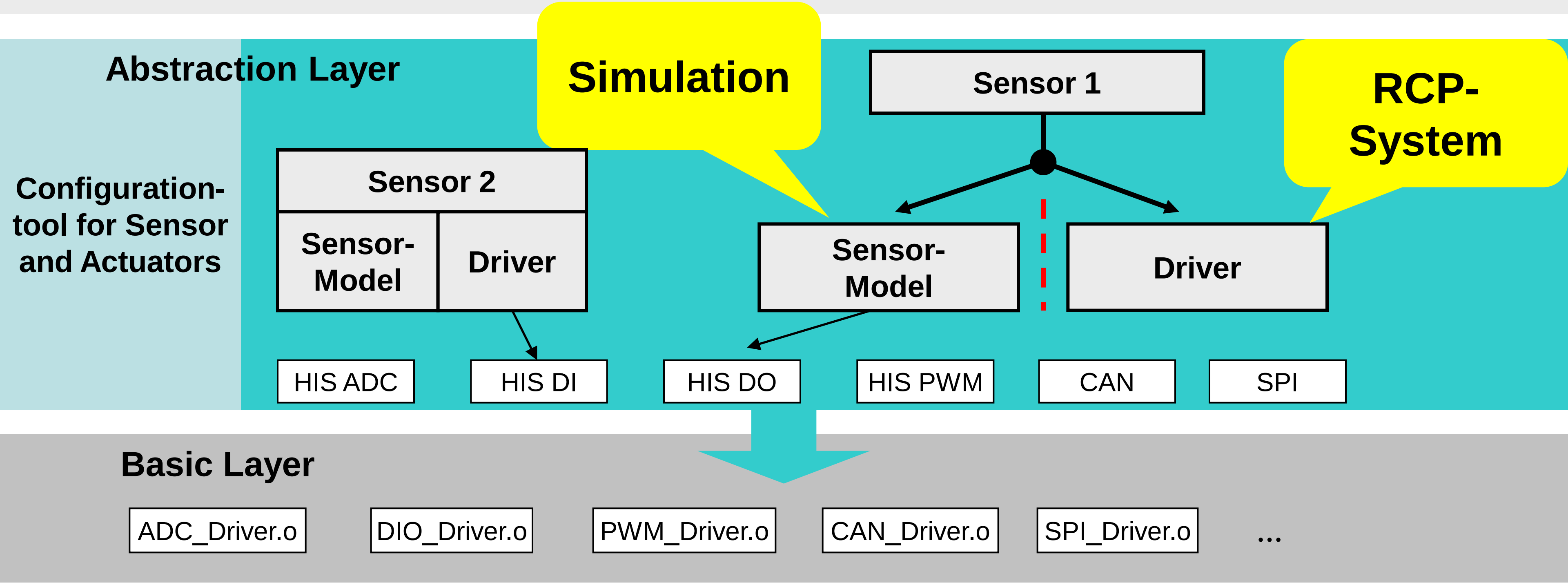
Simulation and code generation



Architecture 1/2



Architecture 2/2

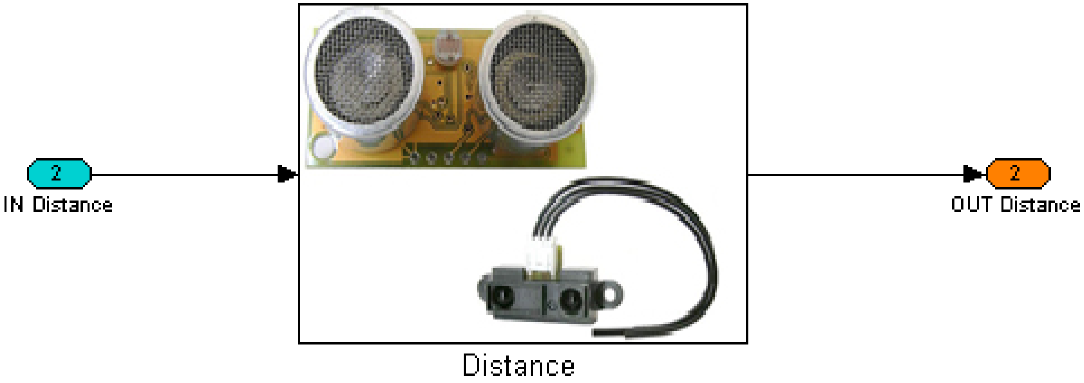


Features:

- managed variability sensors / actuators
- change of Modeling Environment
- easy switch form simulation to RCP-System

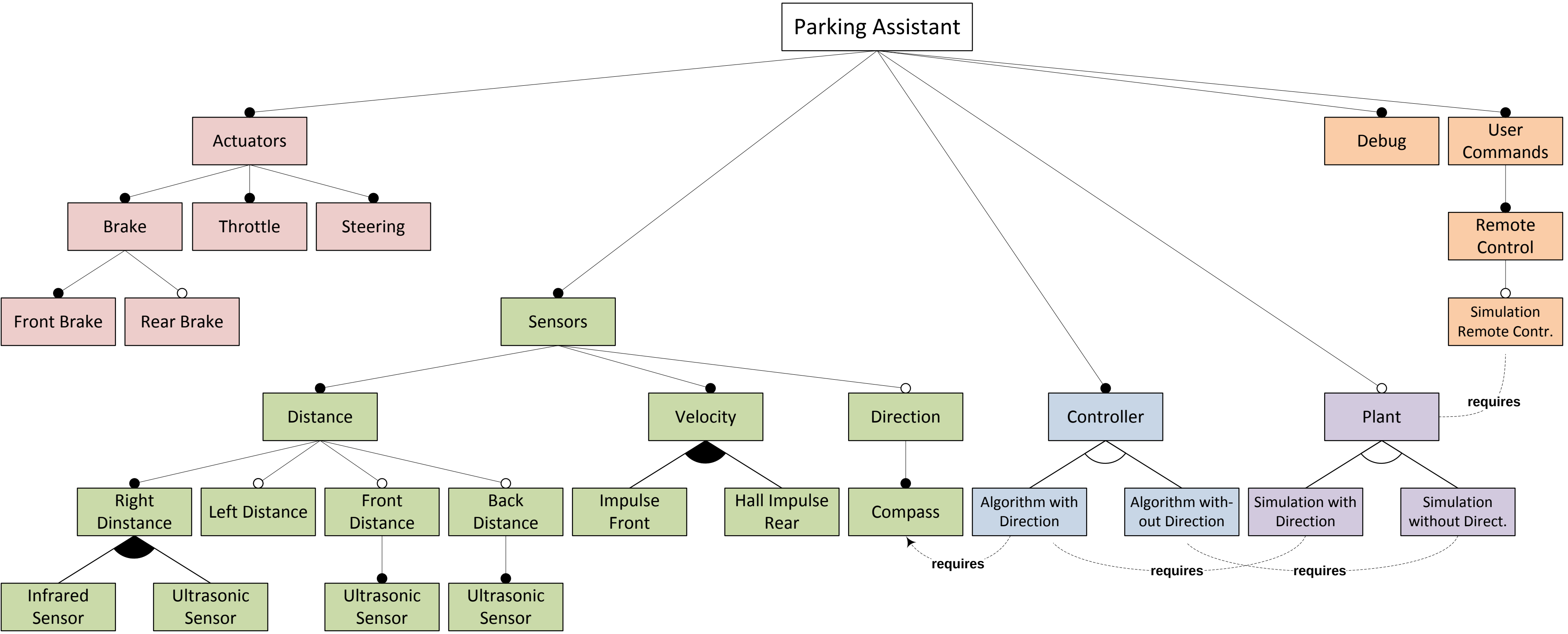
Variability

- sensors or actuators:

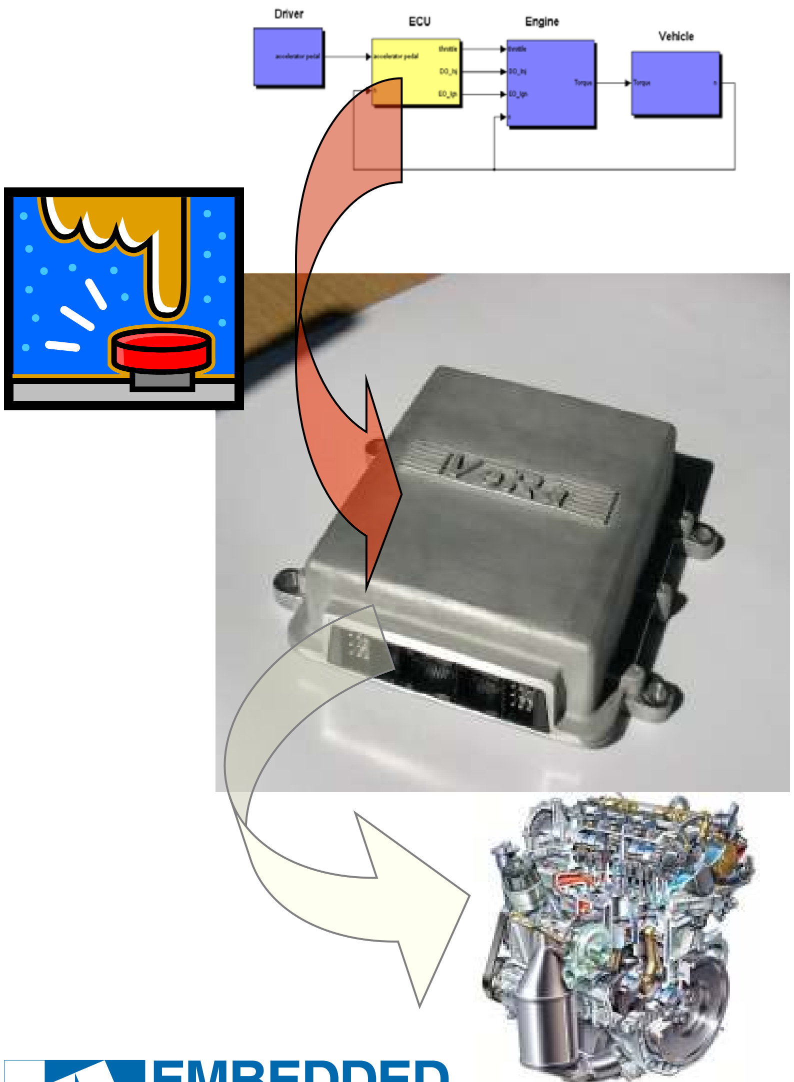


- model
- documentation

Feature Tree – Parking Assistant



Development of a Rapid-Control-Prototyping-System

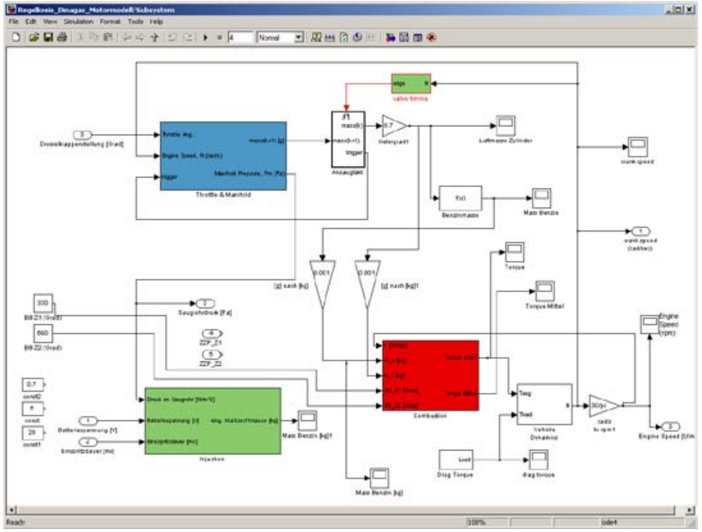


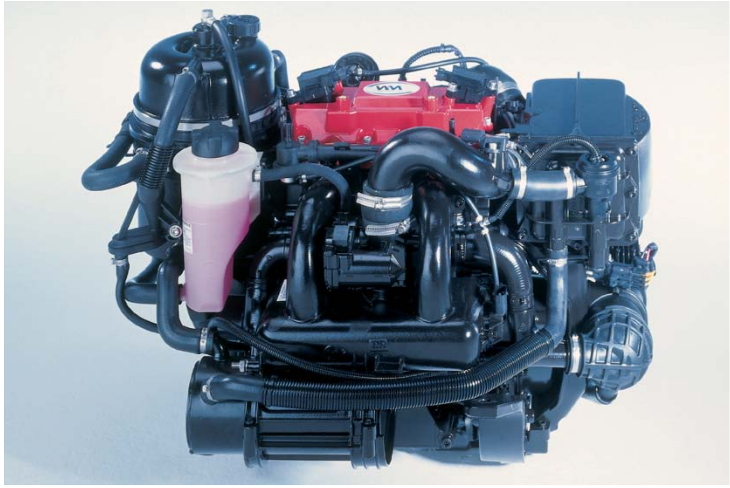
Aspects:

- **consistent modelbased Development**
- **systematic design of the Hardware- and Softwaresystem**
- **enable early simulation**
- **support the developer configuration sensors and actuators**
- **functional and nonfunctional requirements of a small company**

Evaluation: Engine Control Unit

- integration of all Sensors and Actuators





- evaluation with a real engine

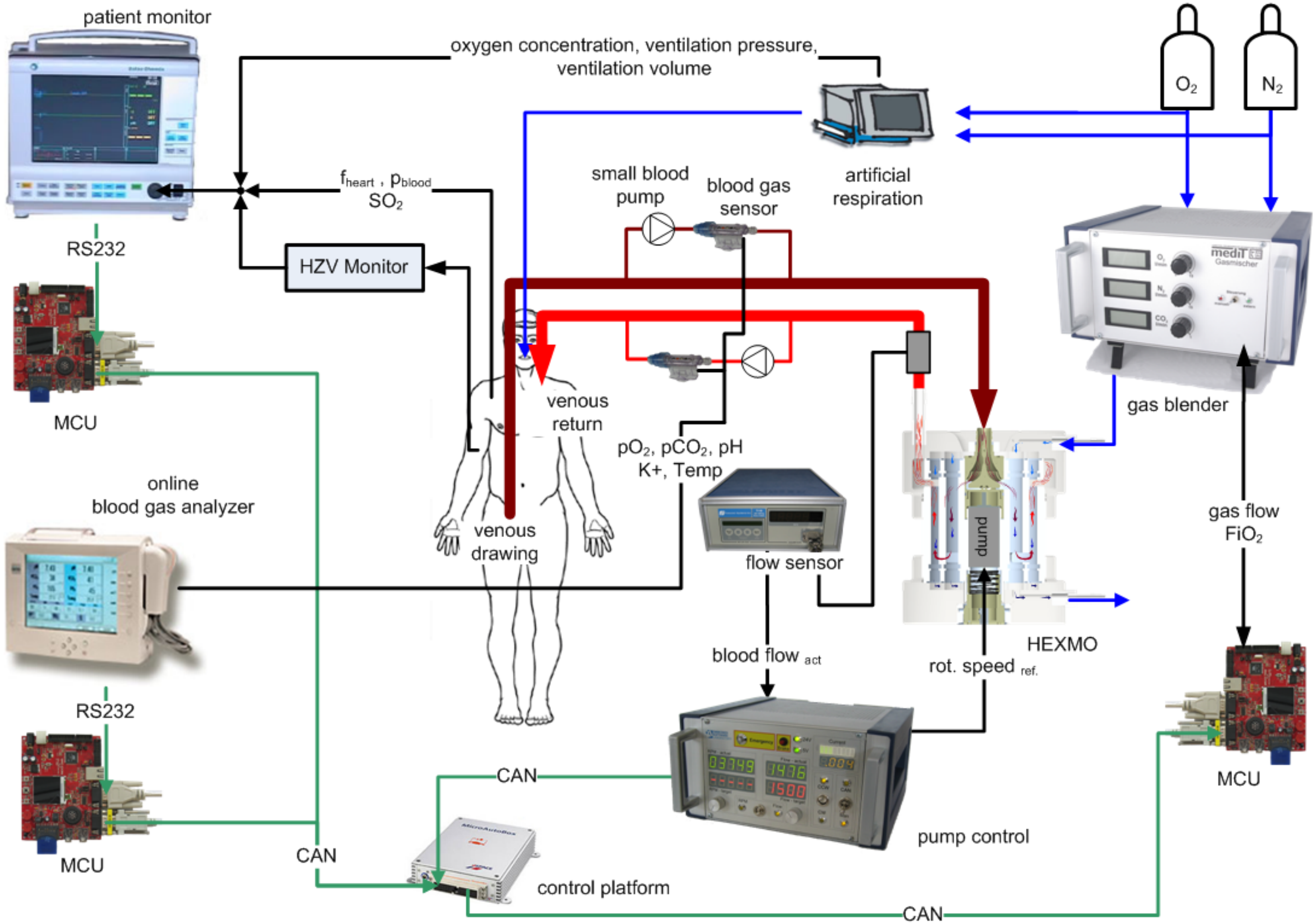
Outline

- Motivation: Lifelong evolving constraints

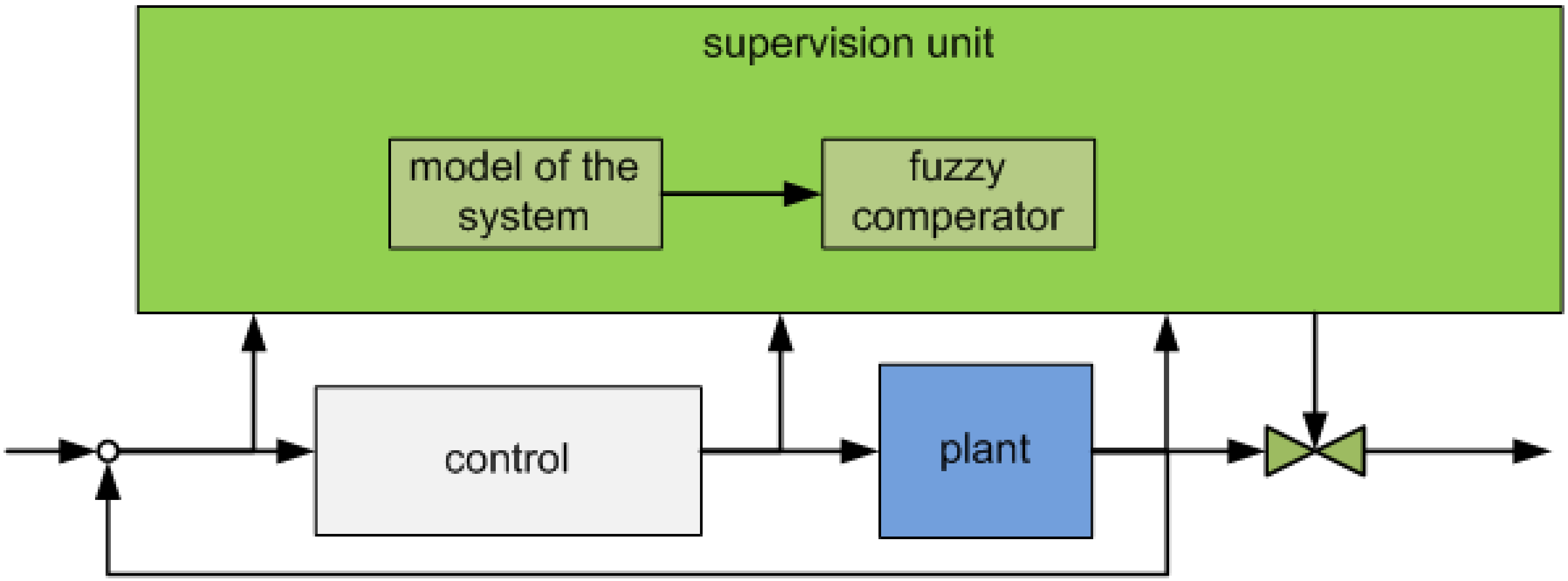
- Abstraction Layers in rapid control prototyping

→ Model based generated code in medical engineering

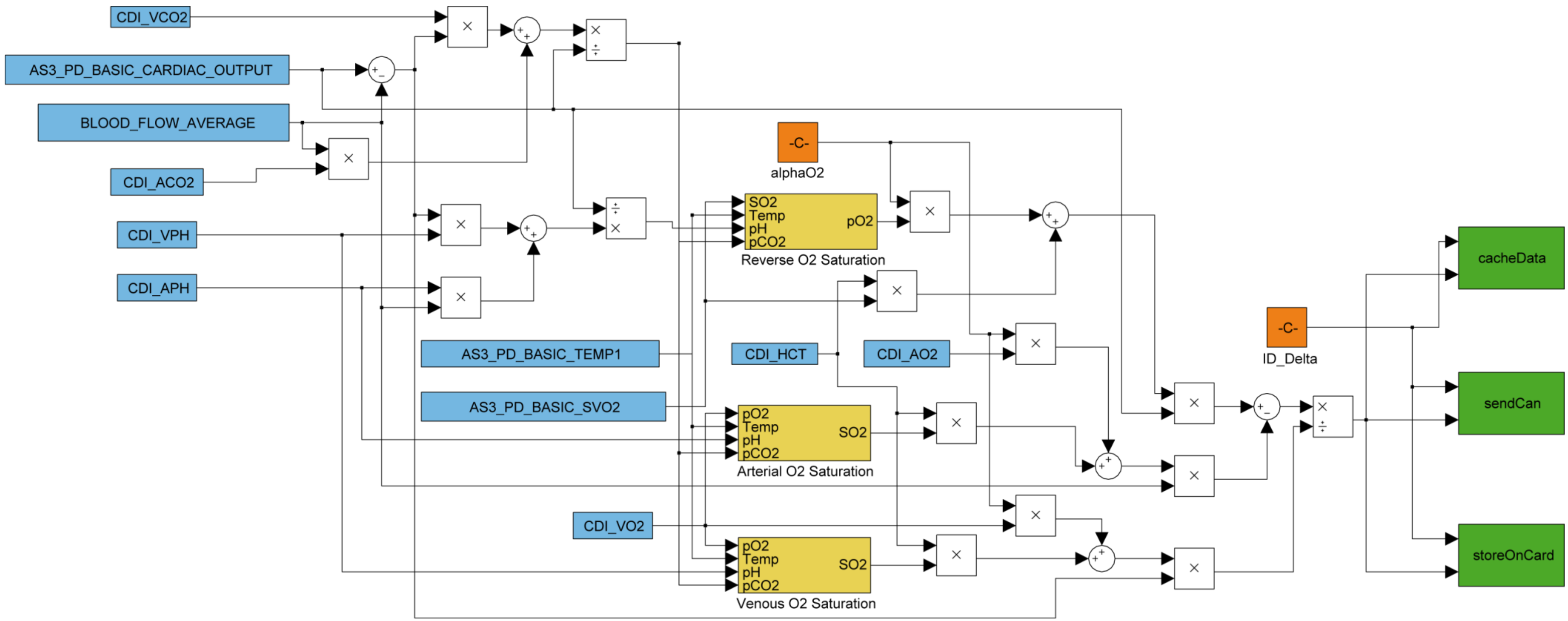
System Setup of the SmartECLA Project



Model Based Safety Measures



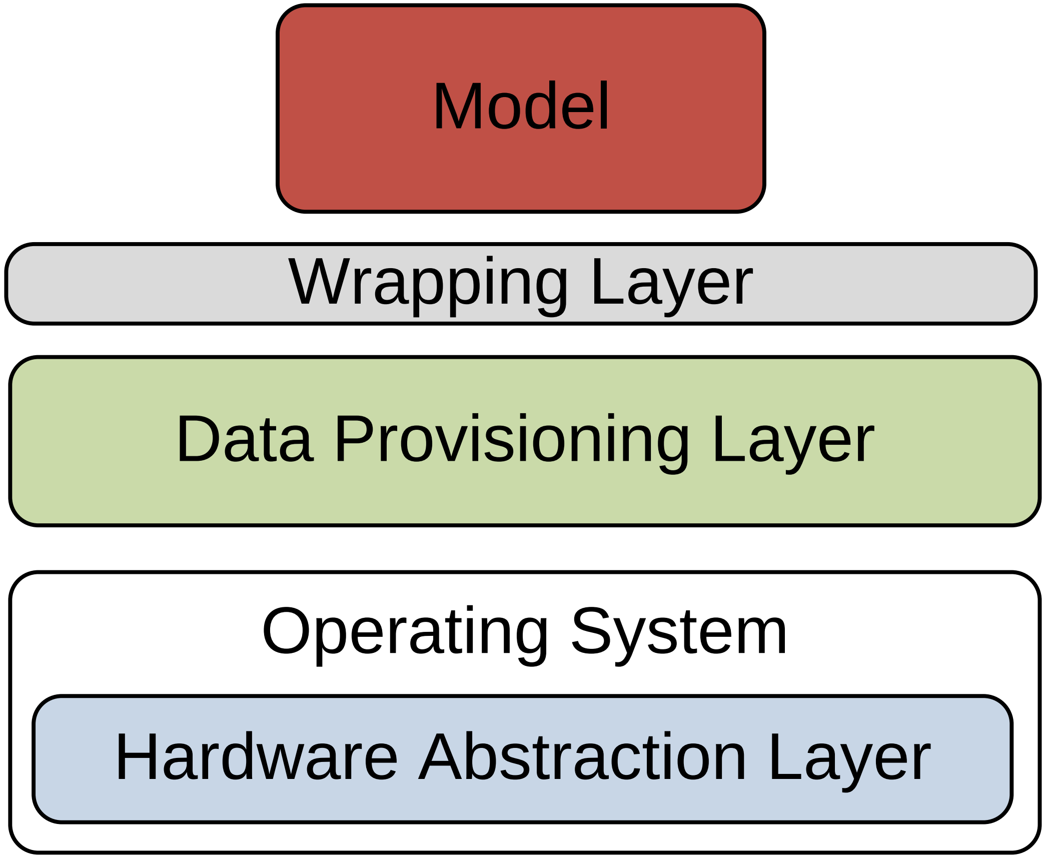
Model Based Safety Measures - Example



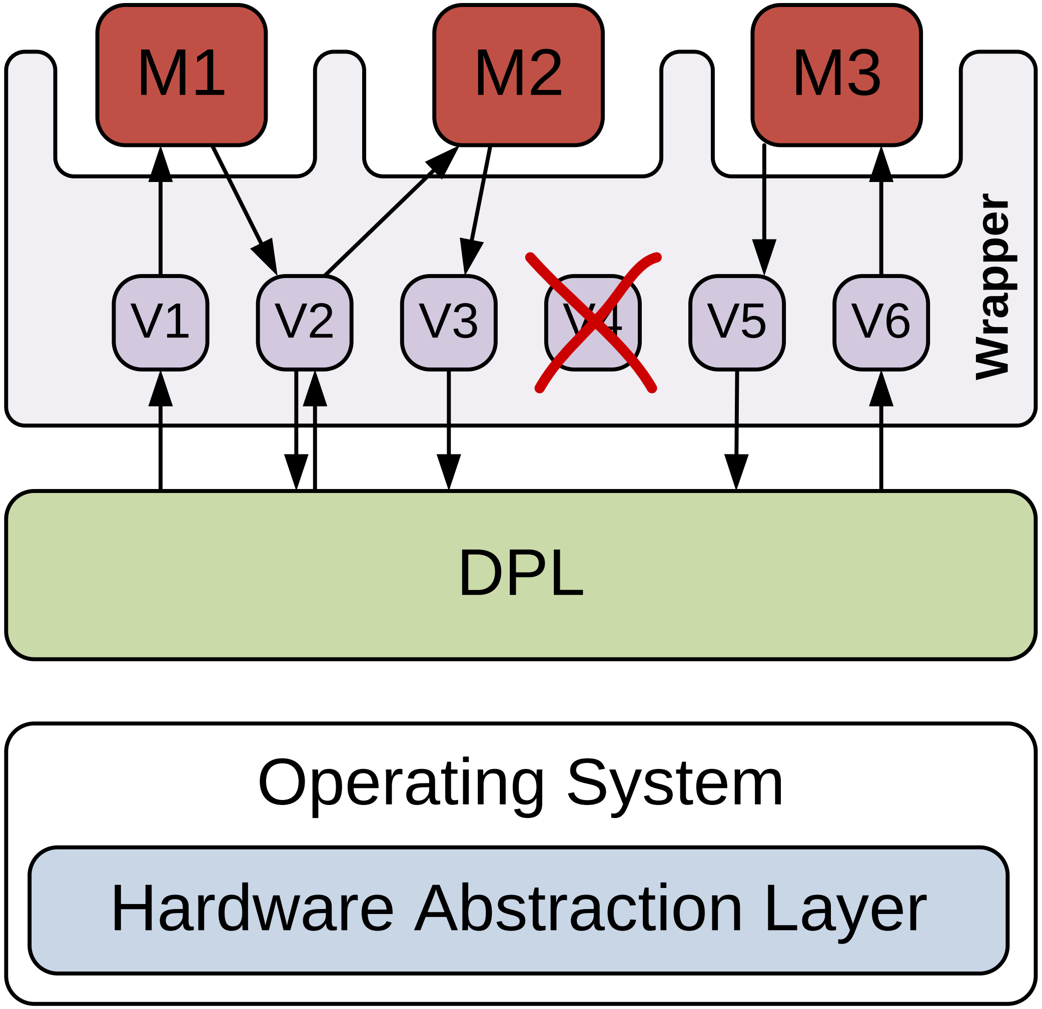
EMBEDDED SOFTWARE LABORATORY

RWTH AACHEN UNIVERSITY

Software System Architecture



Wrapping Layer in Detail



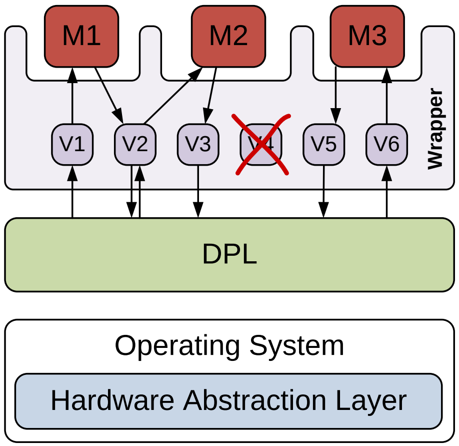
Resulting SW architecture

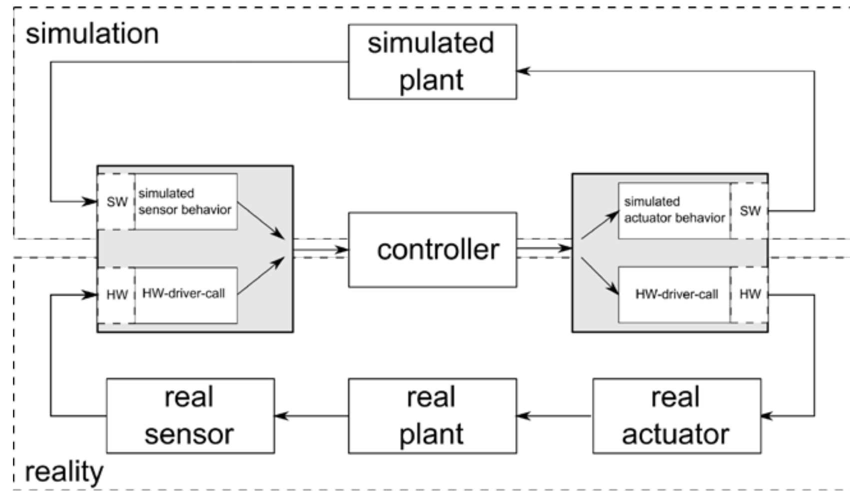
- Integration of model based generated code to existing code framework
- Model changes do not cause changes in surrounding SW framework
- Dynamic adaptations take place at compile time
- Lean and static data management
- Fully predictable memory consumption and runtime behavior of the data management

Conclusion

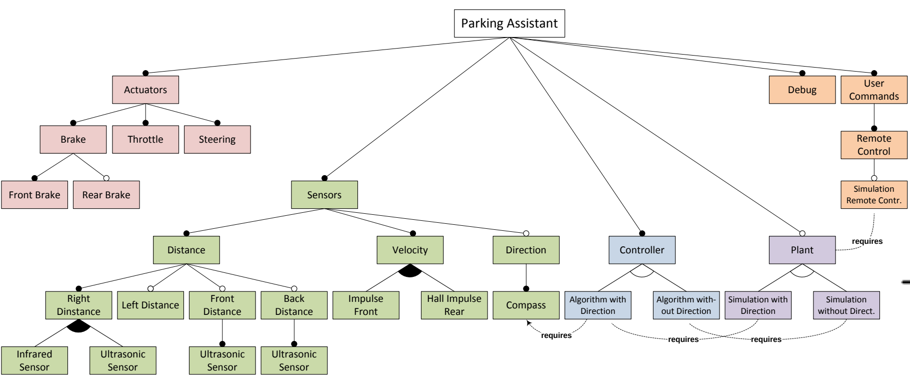
- Fuzzy boundary between development and operation
- Improvements during development process
  - Efficient switching between simulation and real environment
  - Exchangeability of sensors and actuators
  - Exchangeability of development tools
- Tendency to static and thus predictable code
- Layered SW architecture enables SW partitioning

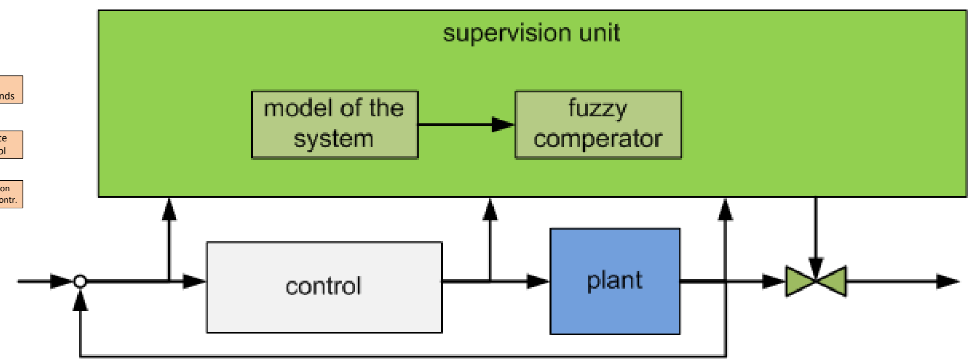
Thanks for your attention !





Questions, Comments, Suggestions… ?





EMBEDDED SOFTWARE LABORATORY

RWTH AACHEN UNIVERSITY