UNITED

Engineering Knowledge Transfer Units to Increase Student´s Employability and Regional Development

Module 1: Electric Powertrain

FOR EDUCATIONAL PURPOSE ONLY

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
598710-EPP-1-2018-1-AT-EPPKA2-CBHE-JP

Components and design criteria of electrified powertrains

Co-funded by the
Erasmus+ Programme
of the European Union

Agenda

- **Classification of electrified powertrains**
  - Classification based on the topology
  - Classification based on the size of electric powertrains
  - Classification based on the position of the secondary energy converter
- **System Level Modeling**
  - Modelling for energy analysis
  - HEV Components modelling
- **System Requirements**
  - Performance requirements
  - Definition of the electric motor specifications based on performance requirements

Co-funded by the Erasmus+ Programme of the European Union

Slot 1: Classification of electrified powertrains

- Classification based on the topology
- Classification based on the size of electric powertrains
- Classification based on the position of the secondary energy converter

Co-funded by the Erasmus+ Programme of the European Union

Introduction

- Possible benefits from the Vehicle Powertrain Electrification:

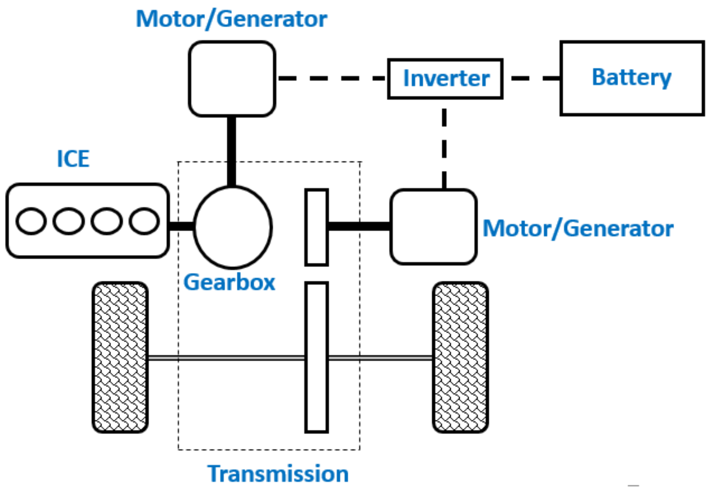
- Engine Stop&Start to avoid engine idling;
- Engine downsizing;
  - The maximum power requirements of the vehicle is fulfilled by electric motor
  - Engine works in it's high efficiency region
- Optimize the power distribution between the prime movers;
  - Electric traction at low speeds, Engine is switched off
  - Electric boost, Engine works in it's high efficiency region
- Regenerative braking;
- Reduced clutch losses.

Co-funded by the
Erasmus+ Programme
of the European Union

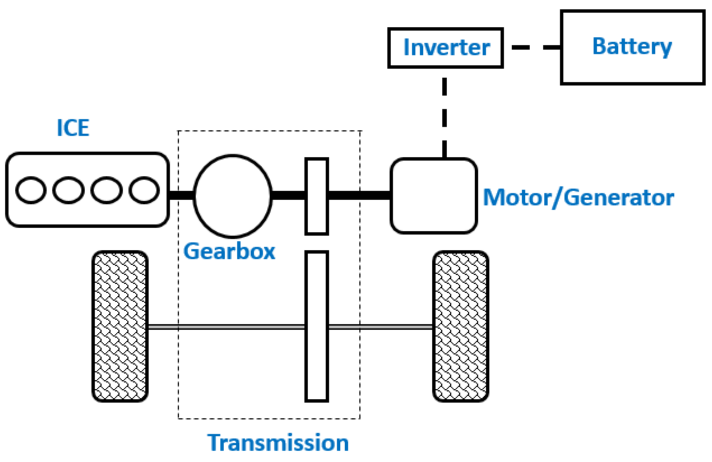
Classification based on the topology

- Hybrid electric vehicles use powertrain composed of two energy sources: Engine and electric motor.
- *Driving modes:*
  - Electric traction
  - Pure ICE
  - Hybrid
  - Regenerative braking
  - Charging

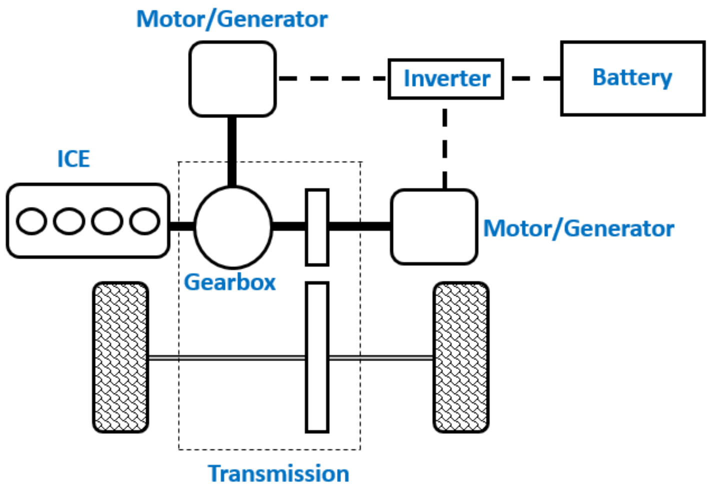
SERIES



PARALLEL



POWER SPLIT

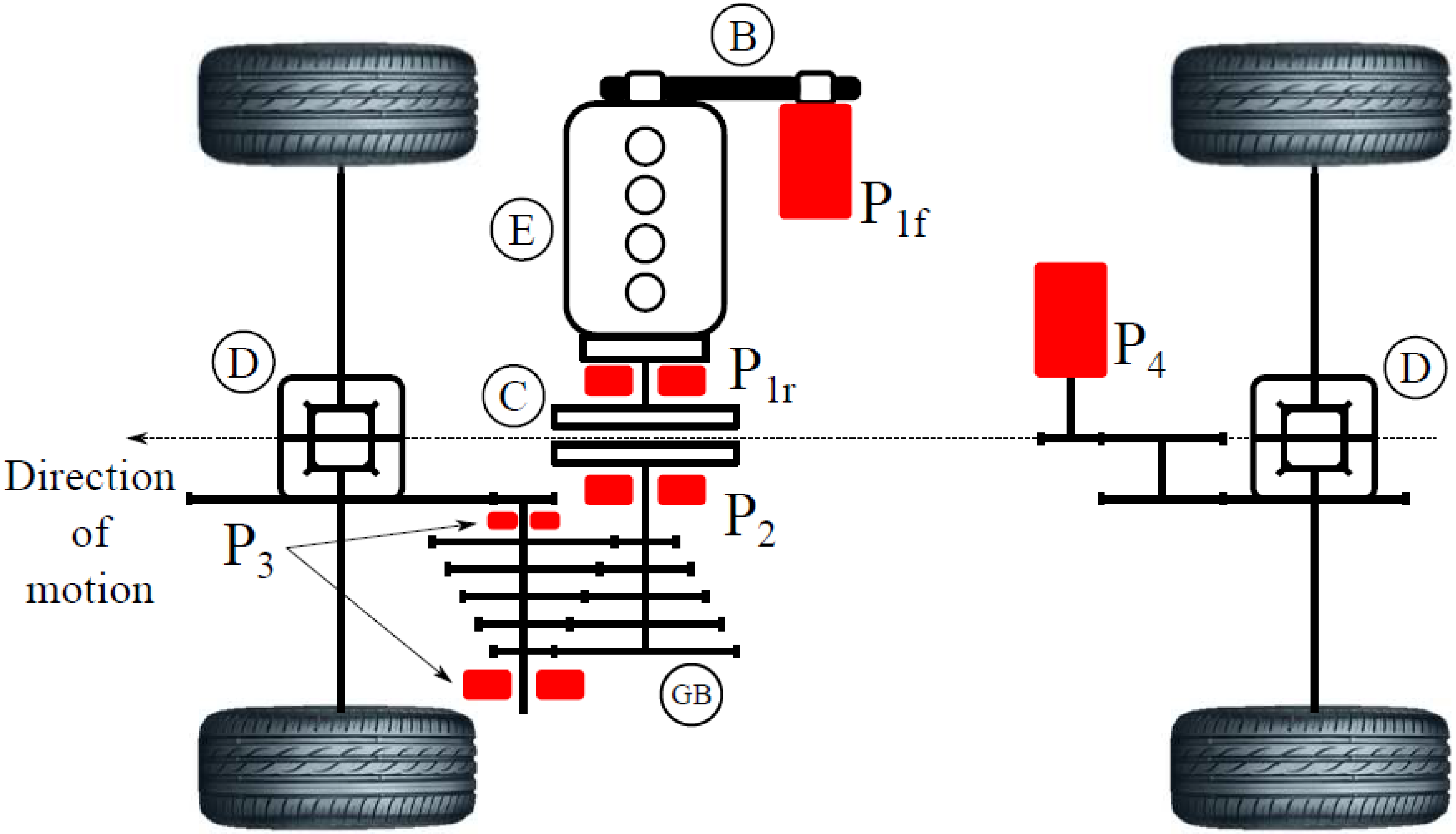


Classification based on the size of the electric powertrain

	Stop/Start	Regenerative braking	Electric assist	Electric boost	Electric traction	Electric range
Conventional vehicle	■					
Micro Hybrid, 12 V (4-6 kW)	■	◧				
Mild Hybrid, <48 V (6-12 kW)	■	■	■	■	◧	
Full Hybrid, >48 V	■	■	■	■	■	
Plug-in Hybrid	■	■	■	■	■	■
Electric Vehicle		■	■	■	■	■

Co-funded by the
Erasmus+ Programme
of the European Union

Classification based on the position of the secondary energy converter

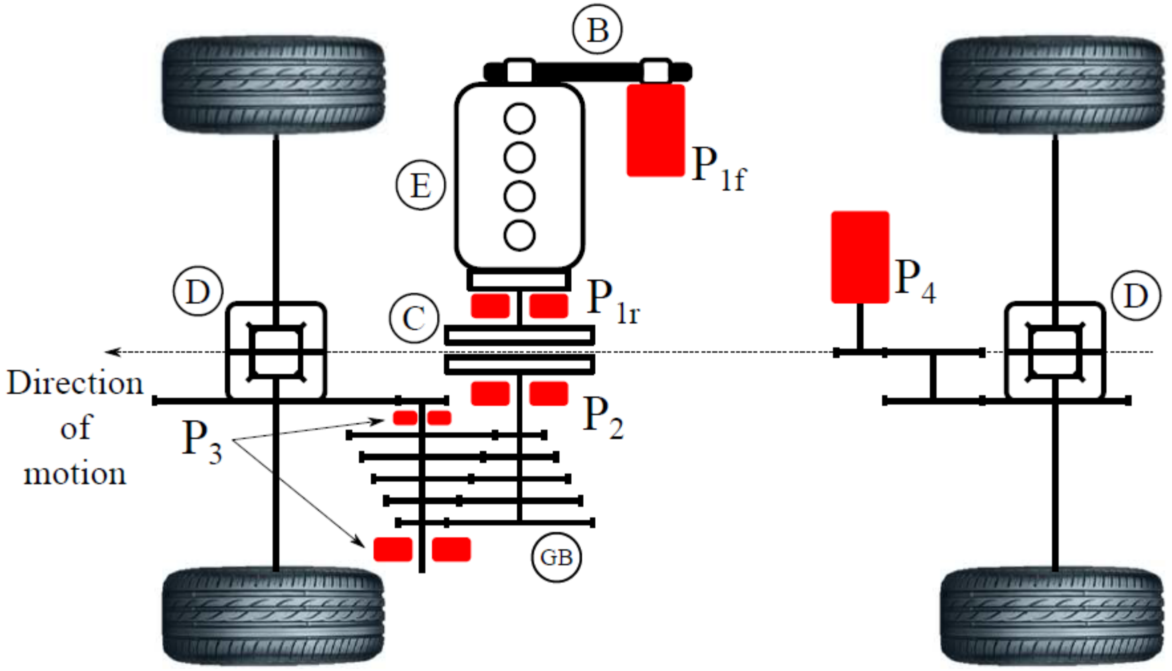


E = Engine;
B = Belt;
C = Clutch;
GB = Gearbox;
D = Differential

Co-funded by the Erasmus+ Programme of the European Union

Source: Castellazzi, Schaeffler

Classification based on the position of the secondary energy converter: *P1f*

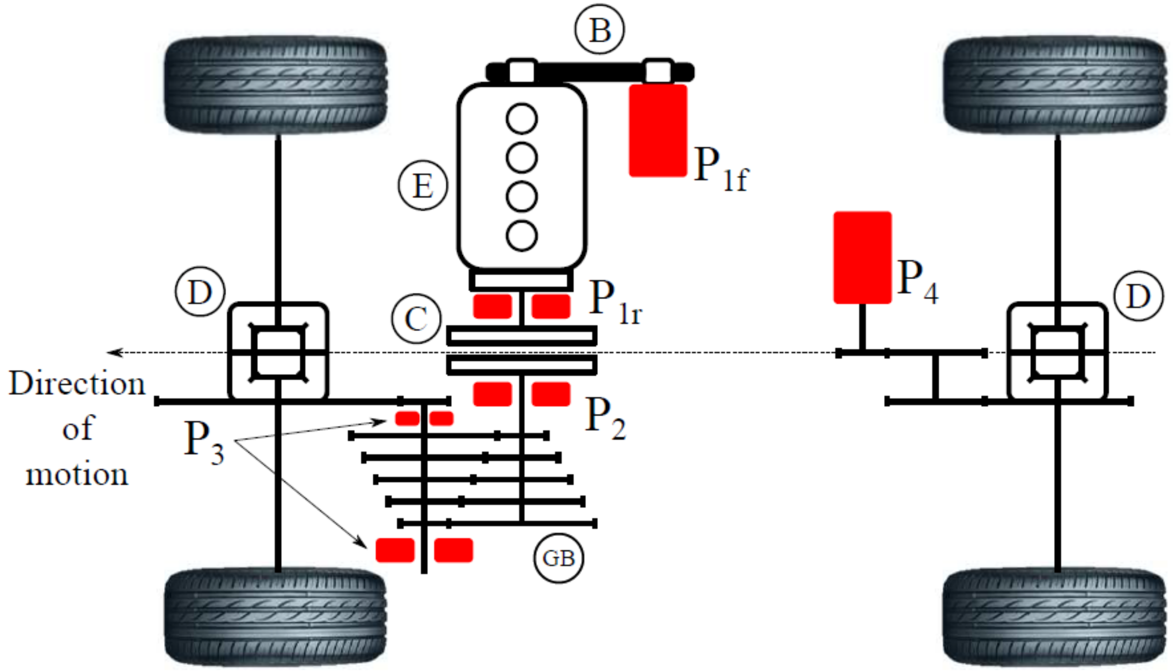


- Standard electric generator mounted on the FEAD is substituted with a powerful electric starter/generator;
- Retrofitting to existing powertrain is relatively less complex;
- Architecture called BSA or BSG;
- Low traction/regenerative braking efficiency due to the large amount of dissipative components between wheels and electric motor;
- Connection between ICE and electric motor through the belt forces the former to be designed for high speeds (15 – 18 krpm);
- Limited torque;
- However, FEAD must be redesigned.

Source: Castellazzi, Schaeffler

Co-funded by the Erasmus+ Programme of the European Union

Classification based on the position of the secondary energy converter: *P1r*

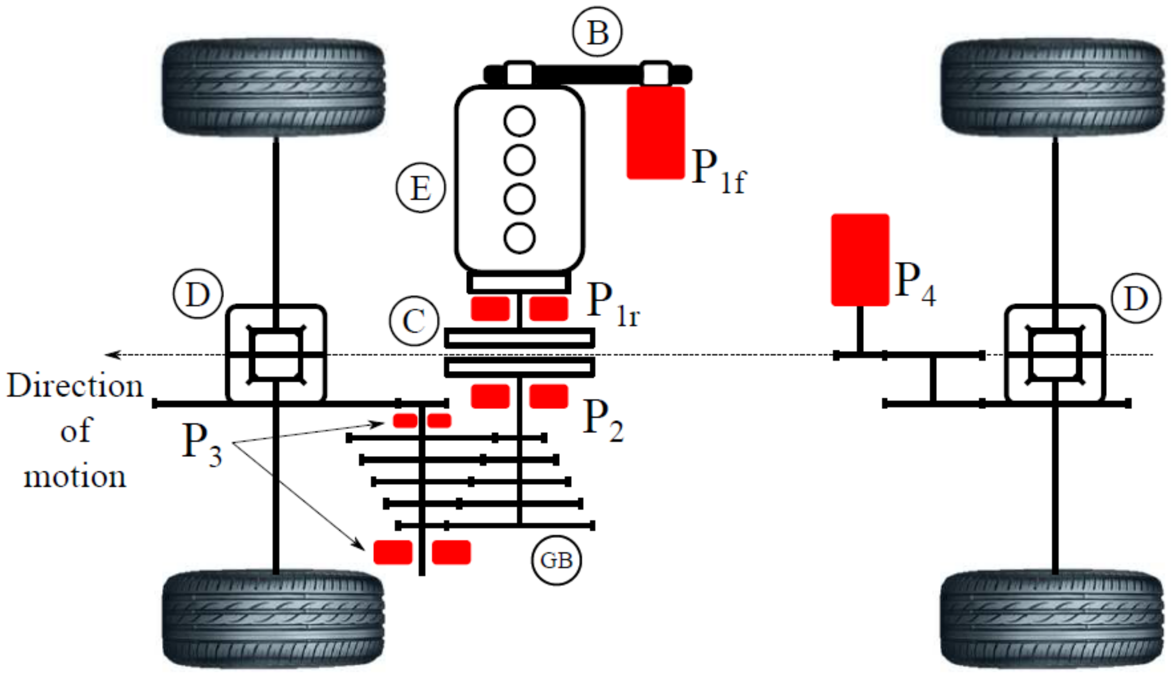


- Electric motor mounted on the output of the engine crankshaft;
- More efficient regenerative braking;
- Engine assistance and active damper (for engine torque oscillations attenuation);
- Possibility of replacing the flywheel
- Small installation space requires high torque density and axial length, leading to high cost of the components

Co-funded by the
Erasmus+ Programme
of the European Union

Source: Castellazzi, Schaeffler

Classification based on the position of the secondary energy converter: *P2*

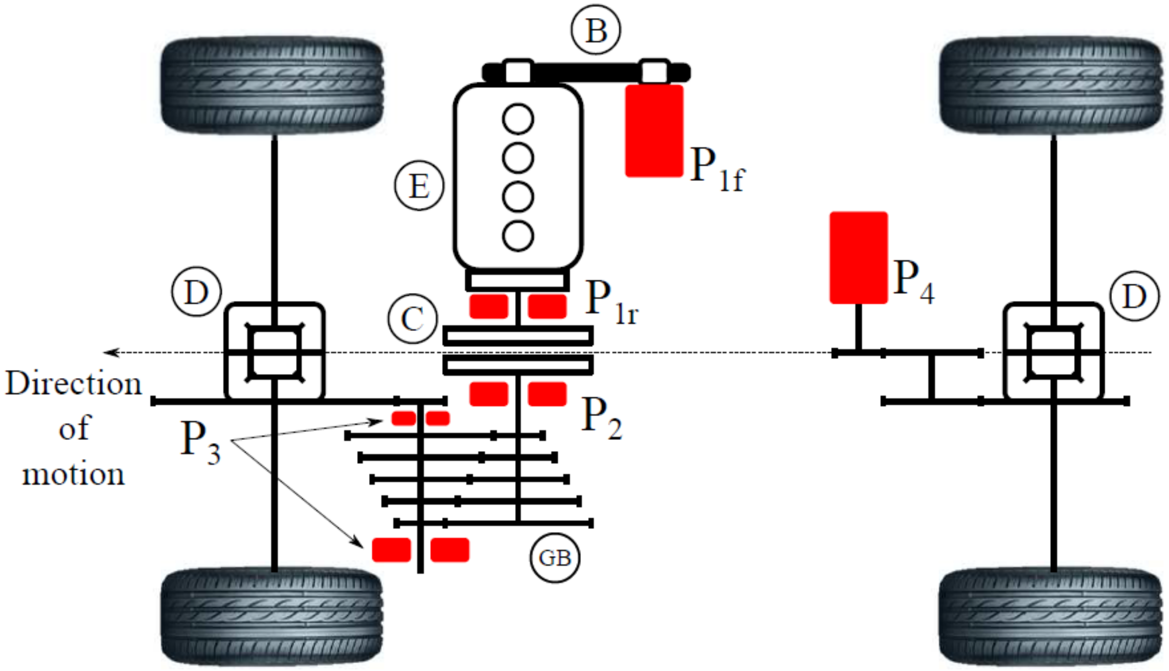


- Engine and motor can be disengaged by clutch, hence, engine drag with its inertia and pumping losses can be excluded;
- Higher efficiency of traction and regenerative braking;
- Can be integrated to the shaft or side mounted (using belt or gear drive)
- High cost of implementation;
- Axial length requirement is less critical.

Co-funded by the
Erasmus+ Programme
of the European Union

Source: Castellazzi, Schaeffler

Classification based on the position of the secondary energy converter: *P3*



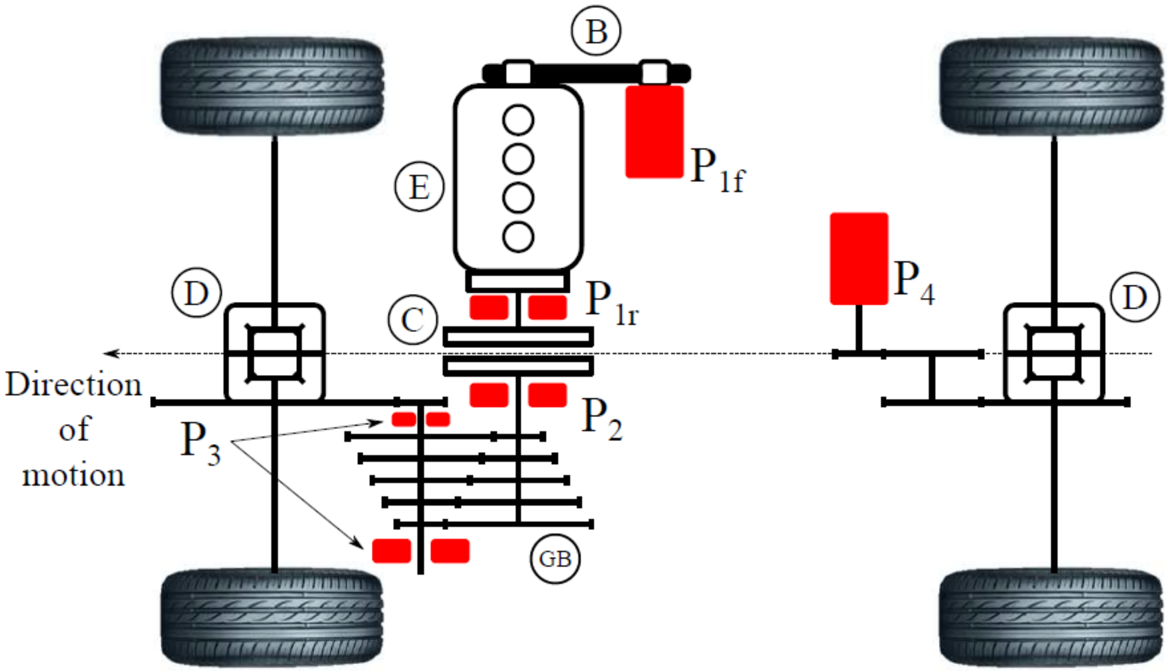
P3.

- Parallel through the road;
- Electric motor can be placed either at the input or at the secondary gearbox shaft output;
- High regenerative braking efficiency;
- Electric motor volume is constrained;
- Requires complete redesign of the overall gearbox;
- Idle charge is not possible.

Co-funded by the Erasmus+ Programme of the European Union

Source: Castellazzi, Schaeffler

Classification based on the position of the secondary energy converter: *P4*

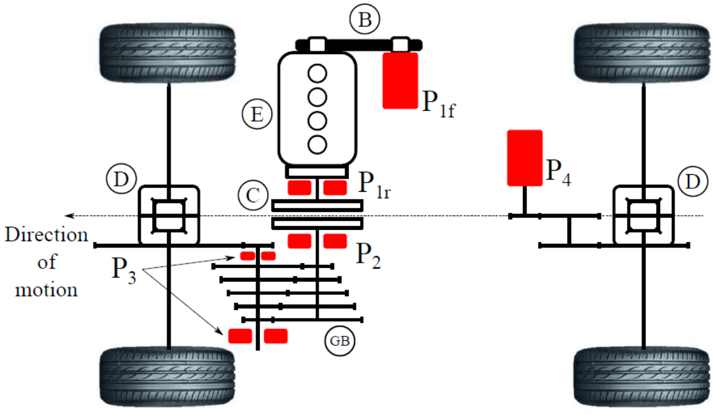


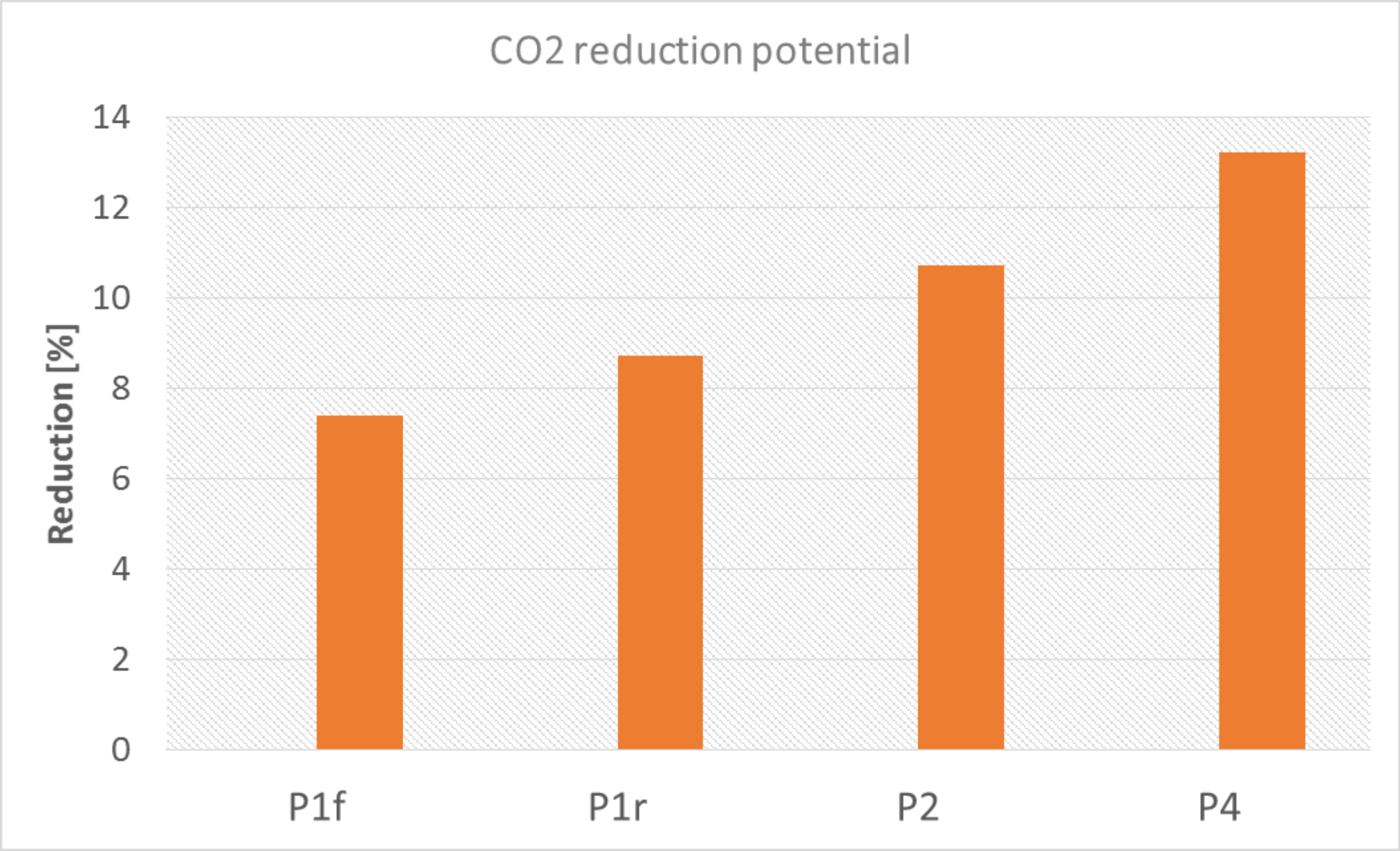
- Electric motor placed on the vehicle rear axle;
- Motor torque flows through additional gearbox and differential to the rear wheels;
- Retrofit solution;
- Regenerative braking efficiency is maximized;
- High potential for torque split with the engine;
- High potential for full electric driving;
- Redesign of the rear axle is required;
- Engine cranking with electric motor is not possible;
- Idle charge of the battery is not possible.

Co-funded by the Erasmus+ Programme of the European Union

Source: Castellazzi, Schaeffler

Comparison of different architectures





Source: Continental

Co-funded by the Erasmus+ Programme of the European Union

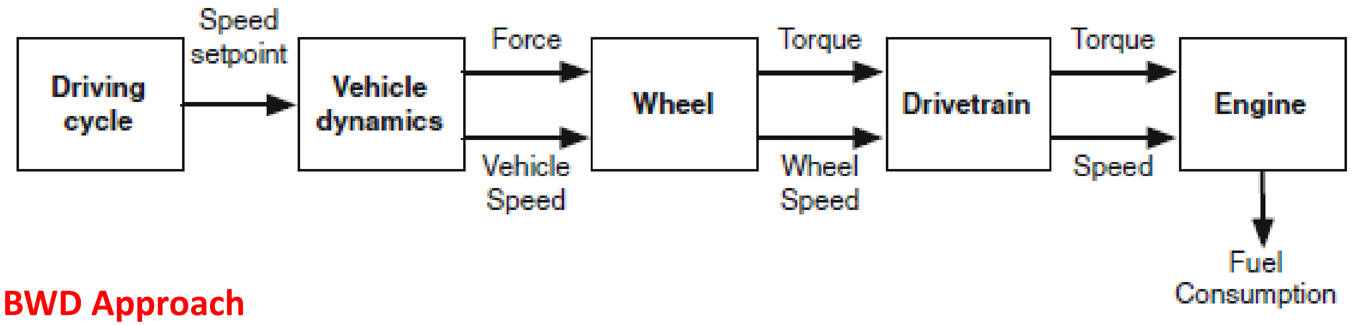
Slot 2: System Level Modeling

- Modelling for energy analysis
- HEV Components modelling

Co-funded by the
Erasmus+ Programme
of the European Union

Modeling for Energy Analysis

- Forward and Backward Modeling Approaches



BWD Approach

Source: Onori

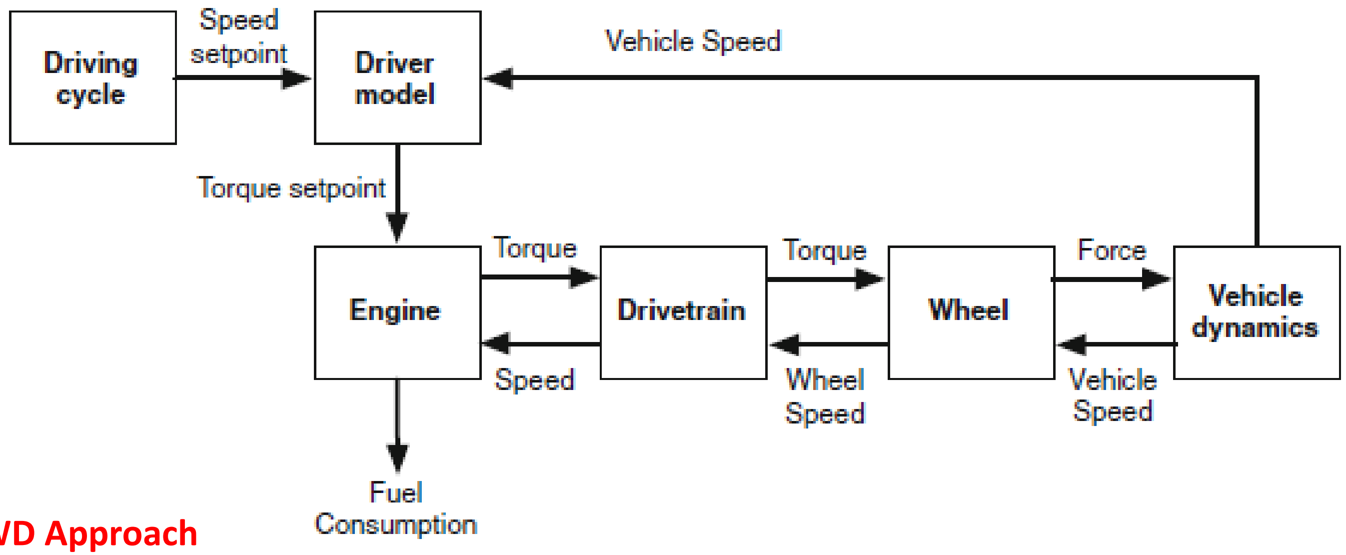
- Desired speed is used to compute accelerations, power at the ground level, forces and torques;
- Driver model is not necessary;
- Torque/speed characteristics of the different powertrain components are considered in order to determine the engine operating conditions and its fuel consumptions;
- Powertrain limitations are not considered;
- Actual speed is exactly the same as speed set point;
- Preliminary analysis of different EMS.

Co-funded by the Erasmus+ Programme of the European Union

Source: Onori S, Serrao L., Rizzoni G. – Hybrid Electric Vehicles – Energy management strategies, Springer 2016

Modeling for Energy Analysis

- Forward and Backward Modeling Approaches



FWD Approach

Source: Onori

- Physical causality is held;
- Driving cycle speed is compared with the actual vehicle speed;
- Driver model (e.g. PID controller) generates braking and throttle commands;
- Commands are sent to the supervisor block responsible to generate the actuators set points.
- Deviation from speed set point;
- Powertrain limitations are taken into consideration;
- Useful to develop online control strategy;
- Drivability model can be included.

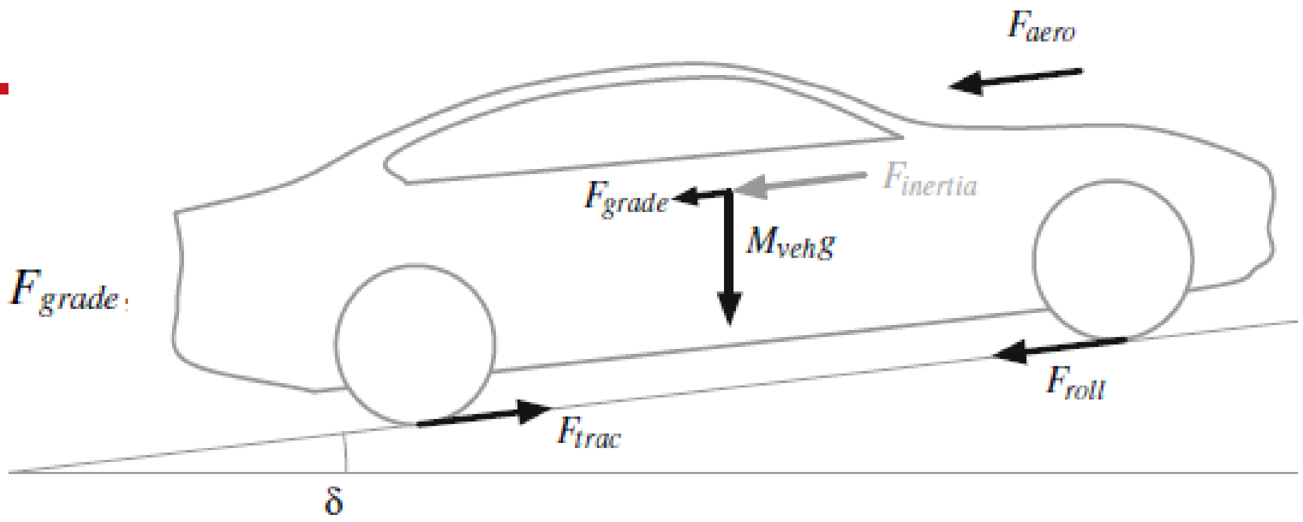
Co-funded by the Erasmus+ Programme of the European Union

Source: Onori S, Serrao L., Rizzoni G. – Hybrid Electric Vehicles – Energy management strategies, Springer 2016

Modeling for Energy Analysis

- Vehicle Energy Balance

$$M_{veh}\frac{dv_{veh}}{dt} = F_{inertia} = F_{trac} - F_{roll} - F_{aero} - F_{grade},$$



$$F_{aero} = \frac{1}{2}\rho_{air}A_f C_d v_{veh}^2,$$

$$F_{roll} = c_{roll}(v_{veh}, p_{tire}, \ldots)M_{veh}g\cos\delta,$$

$$F_{grade} = M_{veh}g\sin\delta.$$

F_{tract} = traction force = $F_{pwt} - F_{brake}$
F_{roll} = rolling resistance forces
F_{aero} = aero resistance forces
F_{grade} = road slope resistance forces

M_{veh} = equivalent vehicle mass
v_{veh} = actual vehicle speed

Co-funded by the Erasmus+ Programme of the European Union

Source: Onori S, Serrao L., Rizzoni G. – Hybrid Electric Vehicles – Energy management strategies, Springer 2016

Modeling for Energy Analysis

- Vehicle Energy Balance

$$F_{trac} = F_{pwt} - F_{brake} = F_{inertia} + F_{grade} + F_{roll} + F_{aero}.$$

Multiplying by the speed:

$$P_{trac} = P_{inertia} + P_{grade} + P_{roll} + P_{aero}.$$

Integrating over the time:

$$E_{trac} = \int_{t_0}^{t_f} P_{trac} dt = E_{kin} + E_{pot} + E_{roll} + E_{aero},$$

$$E_{kin} = \int_{t_0}^{t_f} P_{inertia} dt = M_{veh} \int_{t_0}^{t_f} v_{veh}(t)\dot{v}_{veh}(t)dt;$$

$$E_{pot} = \int_{t_0}^{t_f} P_{grade} dt = M_{veh} g \int_{t_0}^{t_f} v_{veh}(t) \sin\delta(t)dt;$$

$$E_{roll} = \int_{t_0}^{t_f} P_{roll} dt = M_{veh} g \int_{t_0}^{t_f} c_{roll} v_{veh}(t) \cos\delta(t)dt;$$

$$E_{aero} = \int_{t_0}^{t_f} P_{aero} dt = \frac{1}{2}\rho_{air} A_f C_d \int_{t_0}^{t_f} v_{veh}(t)^3 dt.$$

Separating

Traction phase ($P_{tract} > 0$, superscript +)

Deceleration phase($P_{tract} < 0$, superscript -)

$$E_{kin}^{-} = -E_{kin}^{+}$$

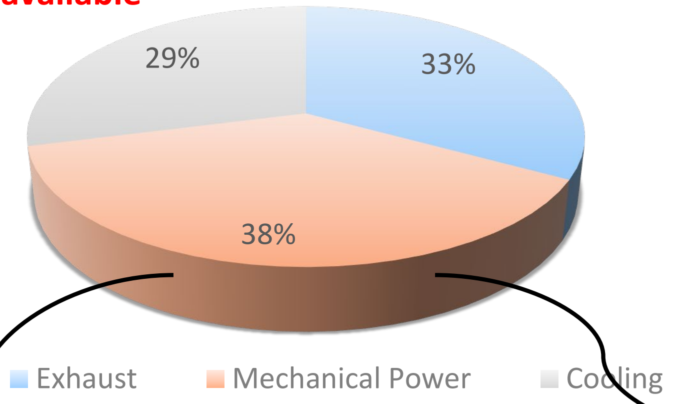
$$E_{pwt}^{+} = E_{roll}^{+} + E_{aero}^{+} + E_{pot}^{+} + E_{kin}^{+},$$

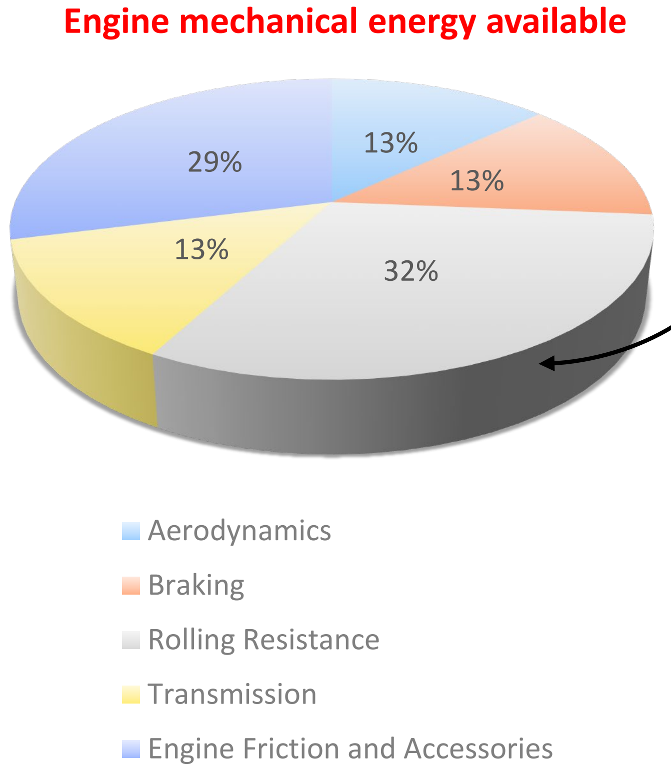
$$E_{regen,pot} = E_{kin}^{+} + E_{pot}^{+} - E_{roll}^{-} - E_{aero}^{-} - E_{pot}^{-}$$

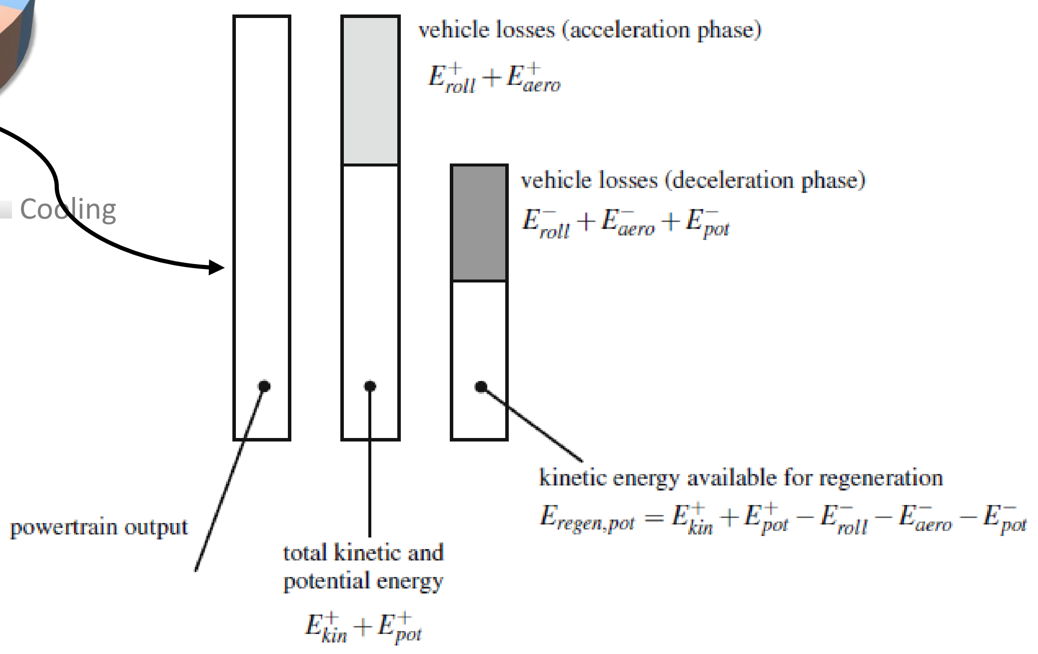
Source: Onori S, Serrao L., Rizzoni G. – Hybrid Electric Vehicles – Energy management strategies, Springer 2016

System Level Modeling

- Vehicle Energy Balance





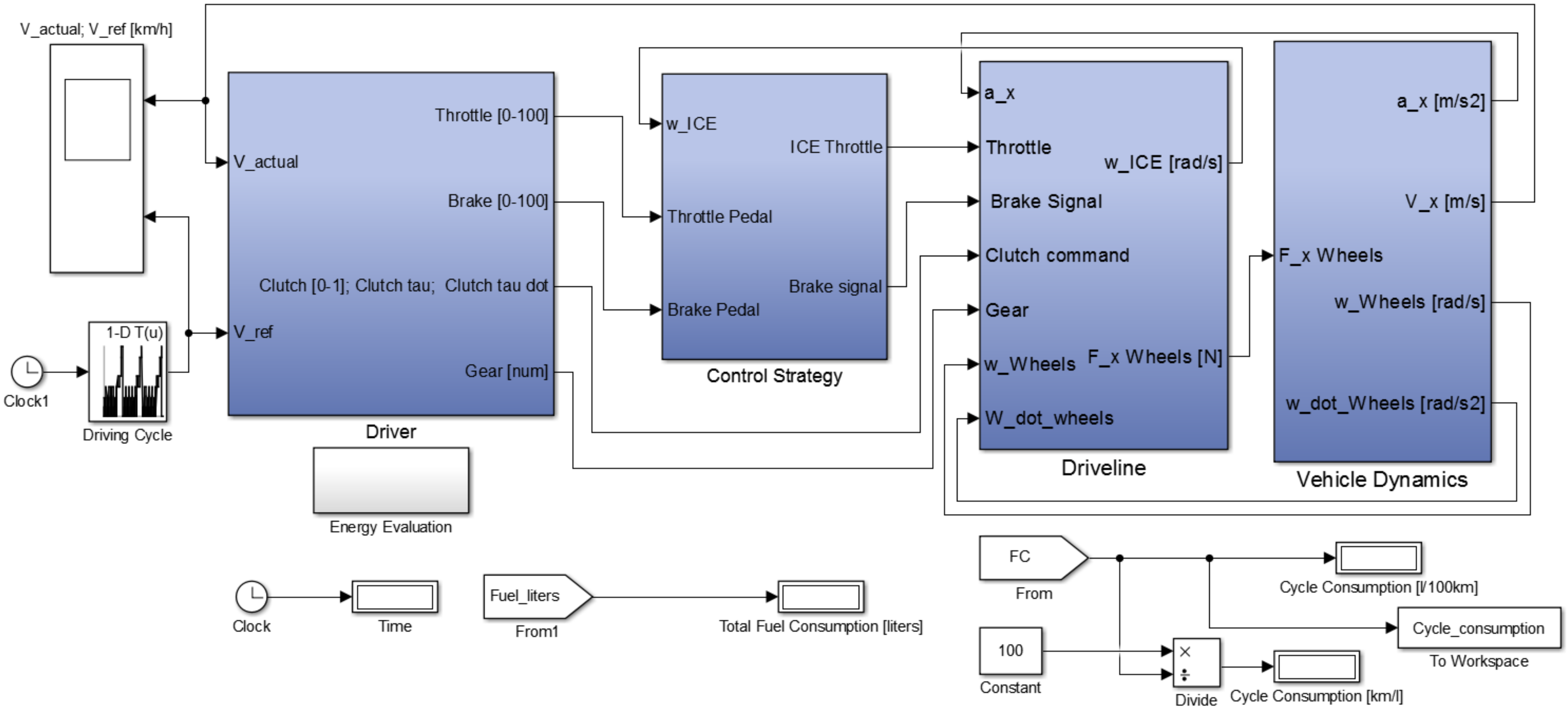


Co-funded by the Erasmus+ Programme of the European Union

Source: Onori S, Serrao L., Rizzoni G. – Hybrid Electric Vehicles – Energy management strategies, Springer 2016

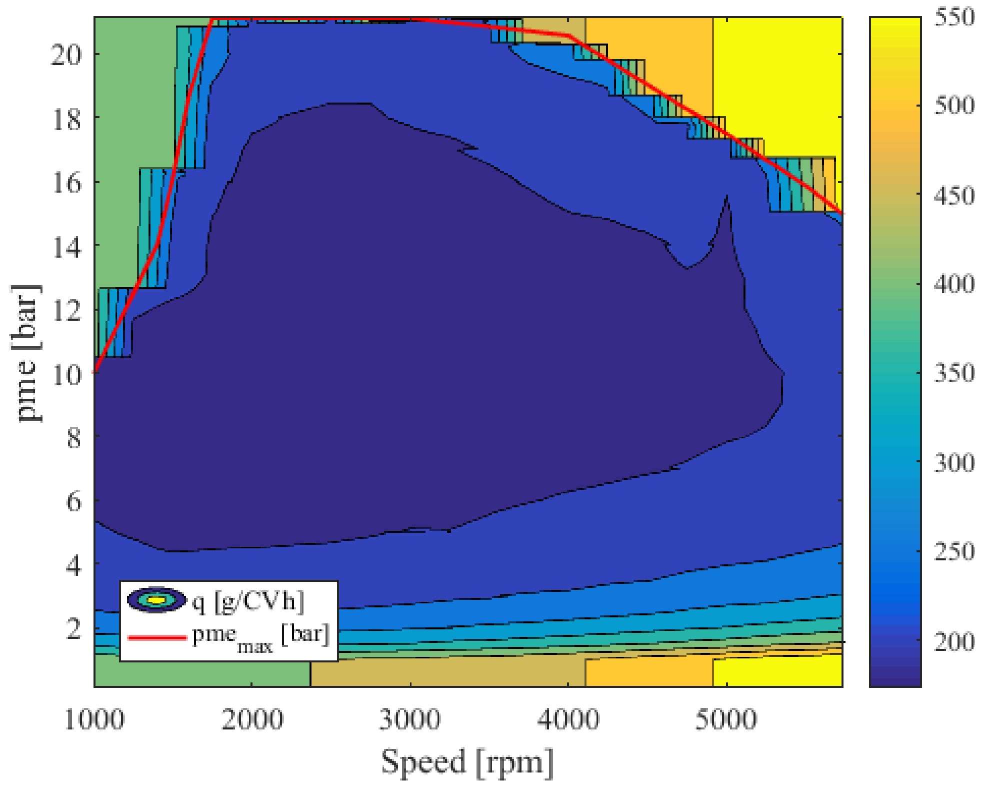
Hybrid Powertrain Model

Model for P1f – P4 Analysis



Components Modeling: Internal Combustion Engine

Torque



- Static map of torque and fuel consumption;
- ICE throttle partializes the maximum available ICE torque:

$$T_{ice} = \alpha_{ice} T_{ice,max}(\omega_{ice})$$

- ICE over-running torque is considered as a constant percentage (γ_{or}) of the maximum torque:

$$T_{ice}^{or} = \gamma_{or} T_{ice,max}(\omega_{ice})$$

- Considering crankshaft inertia and, if present in FEAD, the accessories, the output torque becomes:

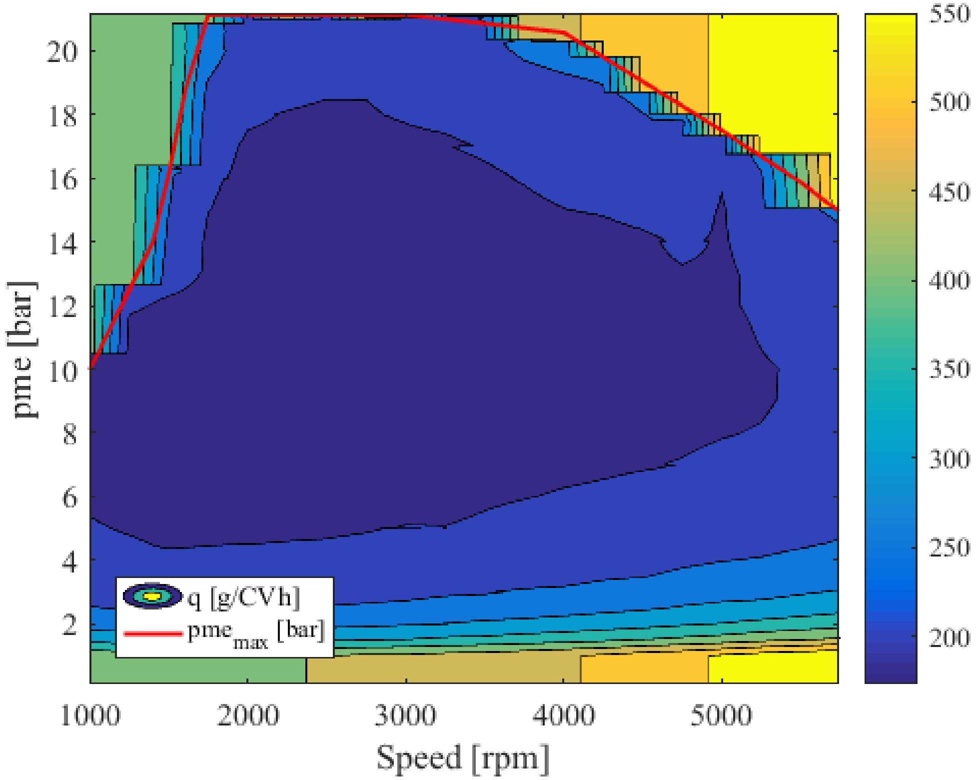
$$T_{ice}^{out} = T_{ice} - T_{ice}^{or} - J_{ice}\dot{\omega}_{ice} - T_{acc}$$

- Accessories can be modeled with their speed/torque static maps or by dynamic model.

Co-funded by the Erasmus+ Programme of the European Union

Components Modeling: Internal Combustion Engine

Fuel consumption, CO_2



- Static map of torque and fuel consumption;
- Fuel consumption is computed using the map;
- Amount of fuel consumed is computed integrating fuel consumption rate;
- CO_2 is computed based on "Technical Guidelines for the preparation of applications for the approval of innovative technologies pursuant to Regulation (EC) No 443/2009 of the European Parliament and of the Council"
- Conversion of fuel consumption to CO_2 emission

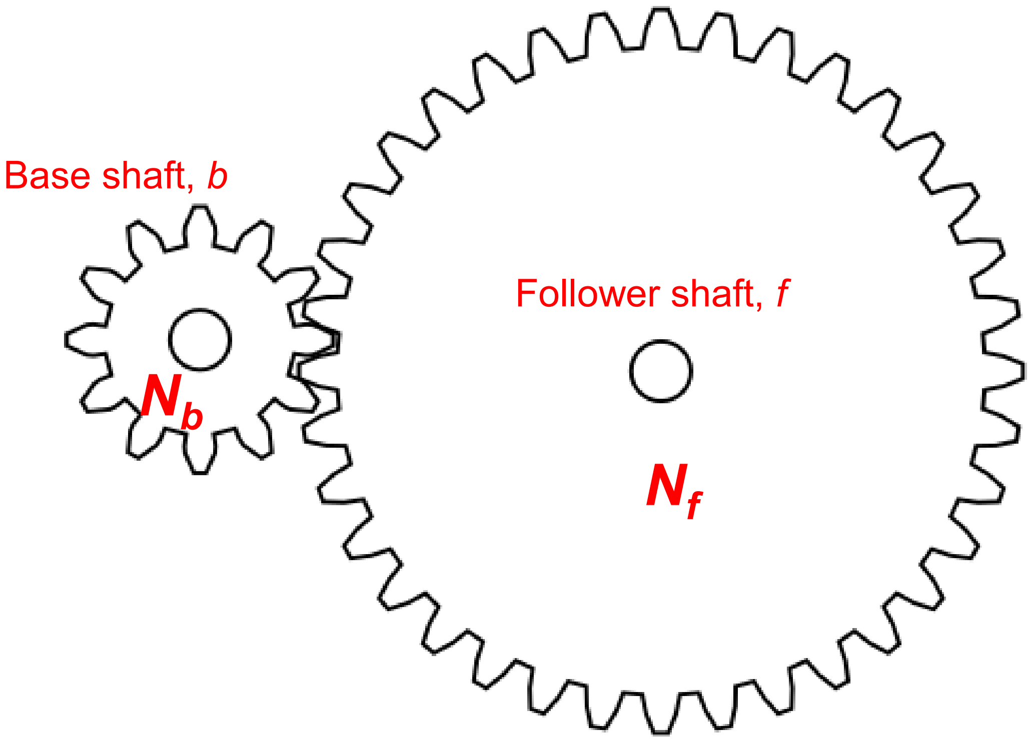
Type of fuel	Conversion factor (l / 100 km) → (g CO_2 / km) [100 g / l]
Petrol	23.3 (= 2330 g CO_2 / l)
Diesel	26.4 (= 2640 g CO_2 / l)

Source: International Energy Agency

Co-funded by the Erasmus+ Programme of the European Union

Components Modeling

Gearbox



- Gearbox is modeled as transmission ratio with efficiency;
- Shaft flexibility is not modeled (dynamic effects are neglected);
- Given the transmission ratio as:

$$g_{fb} = \frac{N_b}{N_f}$$

- Output speed and torque are:

$$\begin{cases} \omega_f = g_{fb}\omega_b, \\ T_f = \dfrac{1}{g_{fb}} T_b. \end{cases}$$

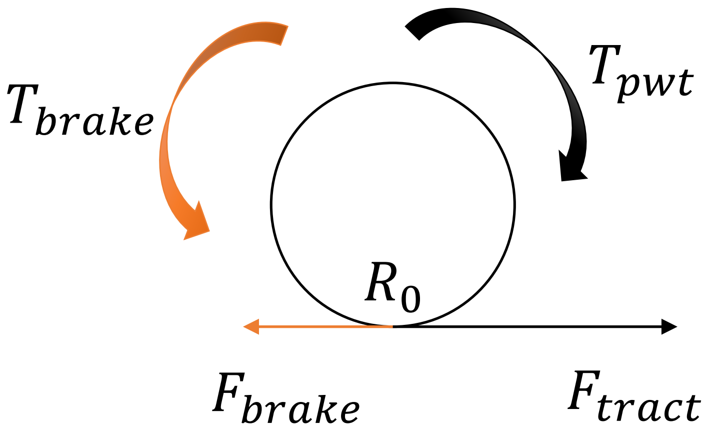
- The power loss in transmission:

$$P_{loss} = \begin{cases} \omega_b T_b (1 - \eta_{fb}) & \text{if } P_b = T_b \omega_b \geq 0 \\ \omega_f T_f (1 - \eta_{fb}) & \text{if } P_b = T_b \omega_b < 0 \end{cases}$$

Co-funded by the Erasmus+ Programme of the European Union

Components Modeling

Wheels, Brakes and Tires



- Tires are modeled as rigid bodies in perfect rolling;
- Longitudinal slips (Pacejka's Magic Formula) are not accounted due to the low accelerations/decelerations perceived by the vehicle following the driving cycle;
- Traction force is:

$$F_{tract} = \frac{1}{R_0}\left(T_{pwt} - T_{brake}\right)$$

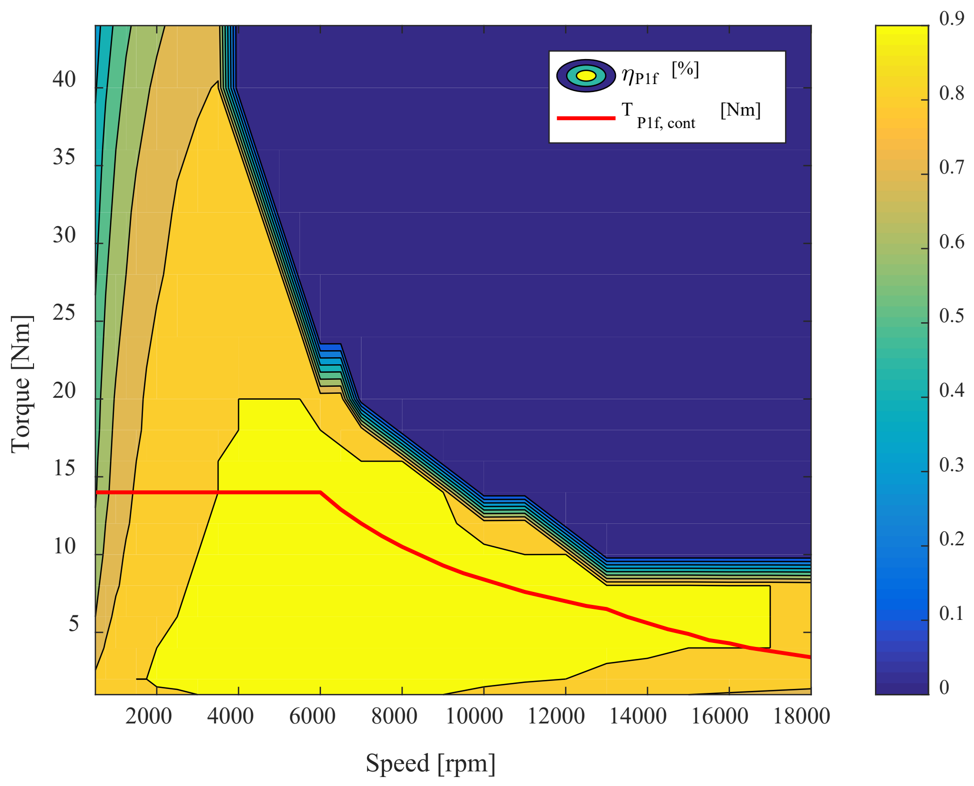
- Brakes can be modeled as simple components producing a braking torque or by more complex model of hydraulic braking system including pressures and disc brakes geometry;
- Both traction and braking forces should be limited with maximum and minimum tires capabilities considering road/tire friction coefficient μ and load transfers:

$$-\mu F_{z,min} \leq F_{tract} \leq \mu F_{z,min}$$

Co-funded by the Erasmus+ Programme of the European Union

Components Modeling

Electric Motors



Example of P1f electric motor map

- Static maps with efficiency;
- The efficiency is:

$$\eta = \frac{T\omega}{V_{dc} I_{dc}}$$

- Efficiency is mirrored for generator mode;
- The computation of the dc current absorbed or generated by the motor is straightforward but important for being an input for the battery;
- In motor (traction) mode the power loss is:

$$P_{loss} = T\omega\left(\frac{1}{\eta} - 1\right)$$

- In generator (braking) mode:

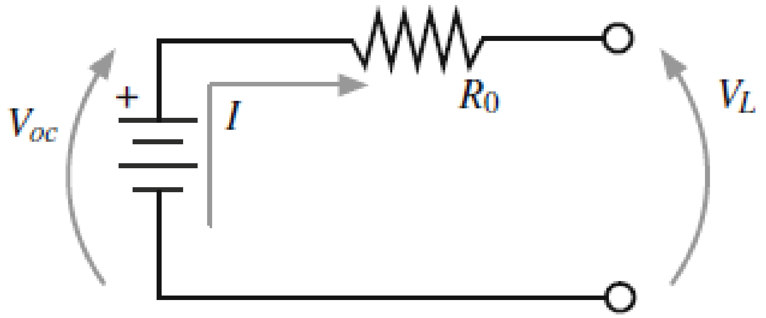
$$P_{loss} = T\omega(1 - \eta)$$

- Electric motor shaft inertia is included;
- Electric motors dynamic modeling will be further discussed.

Co-funded by the
Erasmus+ Programme
of the European Union

Components Modeling

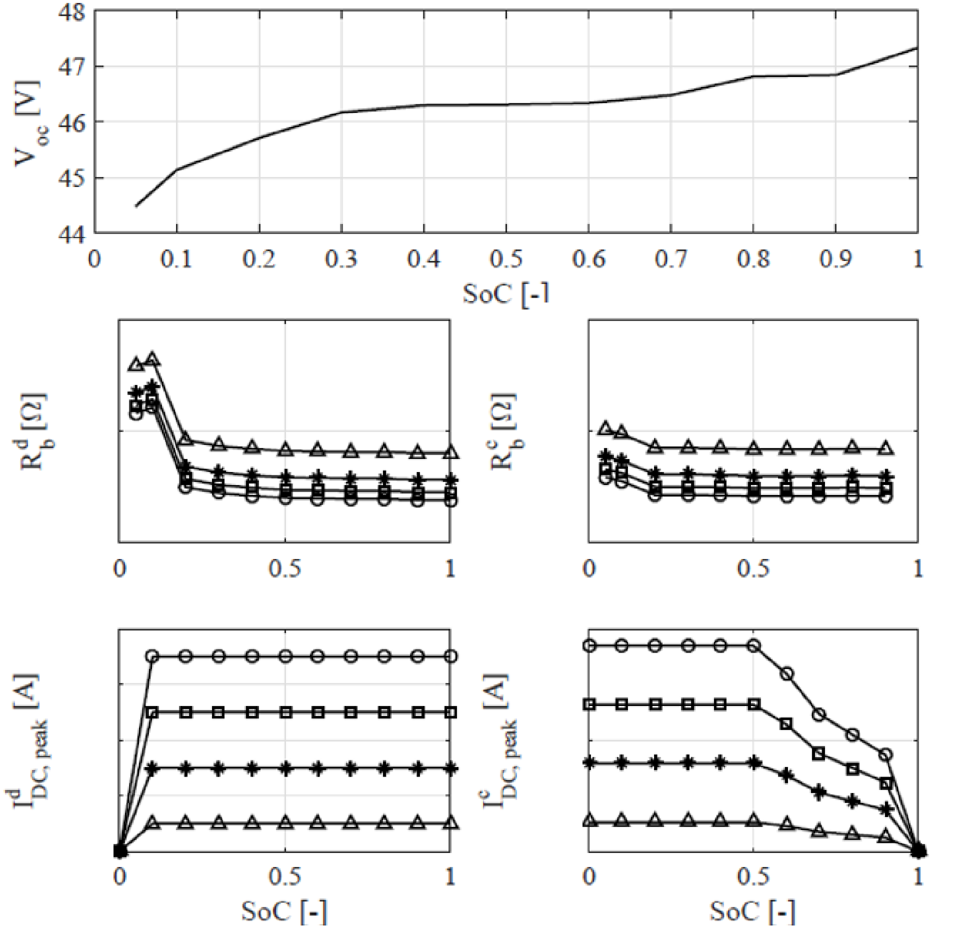
Electric Battery



- Different dynamic models are available;
- For system level analysis a simple zero order model is sufficient for highlighting the main effects without considering voltage and current dynamics;
- The zero order model considers a voltage generator in series with a resistance;
- Open circuit voltage, resistance and maximum admissible current are mapped as a function of the battery capacity and of the main battery parameter, the state of charge:

$$SoC = SoC_0 - \frac{\int I_{dc}dt}{Q_{nom}}$$

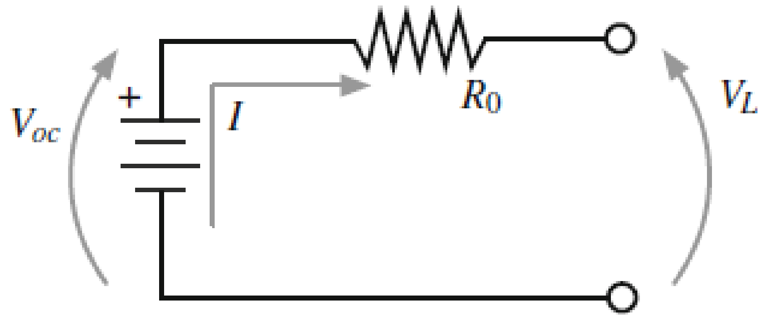
Triangle marked lines: Q_{nom} = 4.4 Ah, asterisk marked lines: Q_{nom} = 13.2 Ah, square marked lines: Q_{nom} = 22 Ah, circle marked lines: Q_{nom} = 30.8 Ah.



Co-funded by the Erasmus+ Programme of the European Union

Components Modeling

Electric Battery



- The load voltage is:

$$V_l = V_{oc} - R_{pack} I_{dc}$$

- The power that the battery should provide or is able to store is the load power plus the resistances:

$$P_b = V_{oc} I_{dc} = V_l I_{dc} + R_{pack} I_{dc}^2$$

- Considering the cell voltage, the open circuit voltage of the pack is:

$$V_{oc} = N_s V_{oc,cell}$$

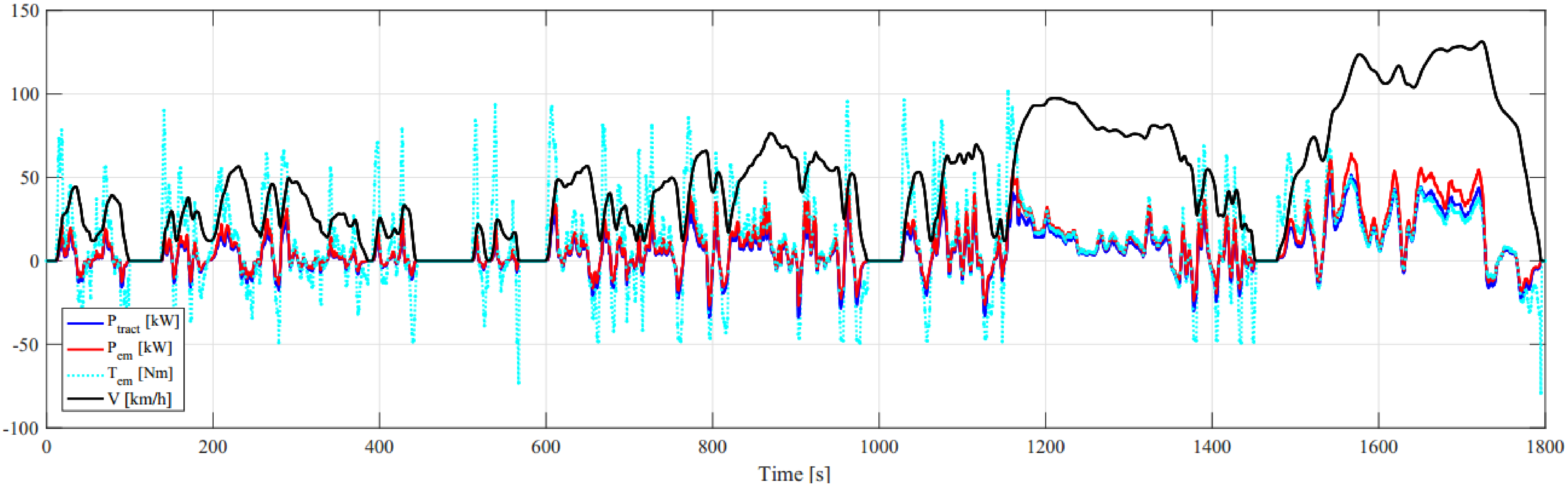
- Where Ns is the number of cells connected in series;
- Considering the cell resistance, the pack resistance is:

$$R_{pack} = \frac{N_s}{N_p} R_{cell}$$

- Where Np is the number of cells connected in parallel.

Components modelling

Results of simulation



Co-funded by the Erasmus+ Programme of the European Union

Slot 3: System Requirements

- Performance requirements
- Definition of the electric motor specifications based on performance requirements

Co-funded by the
Erasmus+ Programme
of the European Union

System Requirements

Procedure of defining the electric motor specifications involve:

- Computation of the required power, torque and speed to fulfill specific performance requirements;

- The analysis are focused on P4 parallel hybrid configuration;
  - The electric machine in this case:
    - Drives the vehicle in pure electric mode;
    - Assists the internal combustion engine in parallel mode during high load requirements;
    - Works as a generator to charge the battery (while driving);
    - Recovers the kinetic energy available during the vehicle braking phases.

Co-funded by the Erasmus+ Programme of the European Union

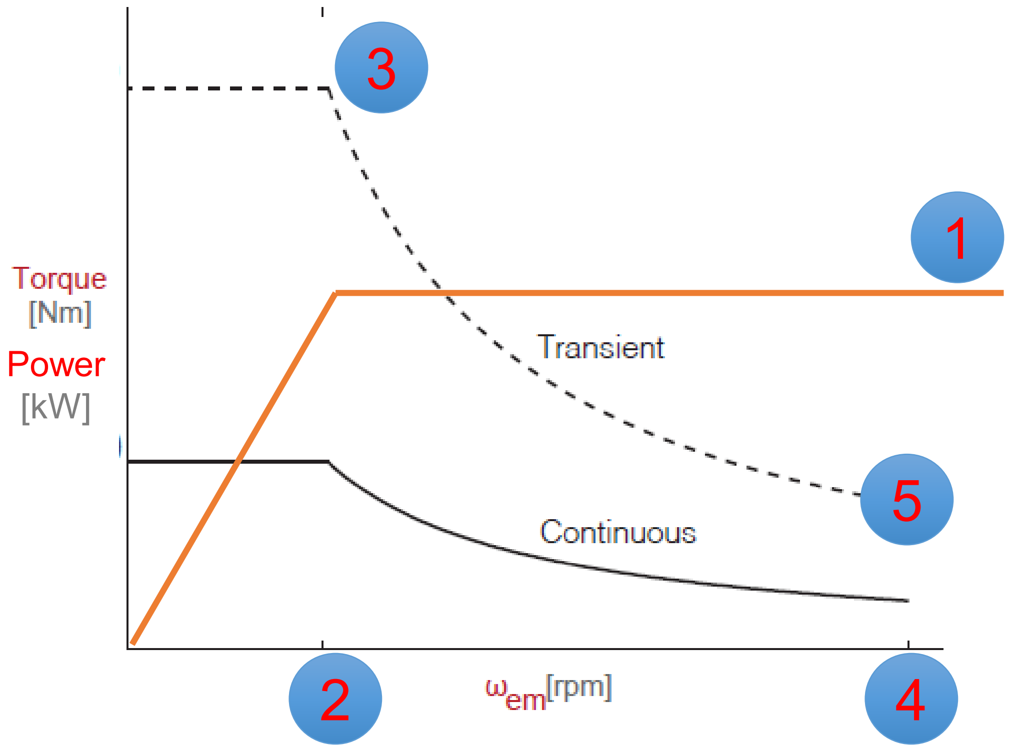
Performance Requirements

- **Hill start:** maximum ground slope that the vehicle must be able to overcome in transient conditions. The ground slope is set to 15% and the acceleration to 1 km/h/s.
- **Max speed:** the vehicle must be able to travel in pure electric mode at a speed which can be considered the maximum WLTP speed.
- **Maximum acceleration:** the acceleration time 0 - 50 km/h must be less than 10 s

Co-funded by the Erasmus+ Programme of the European Union

System Requirements

- Electric Motor Specifications

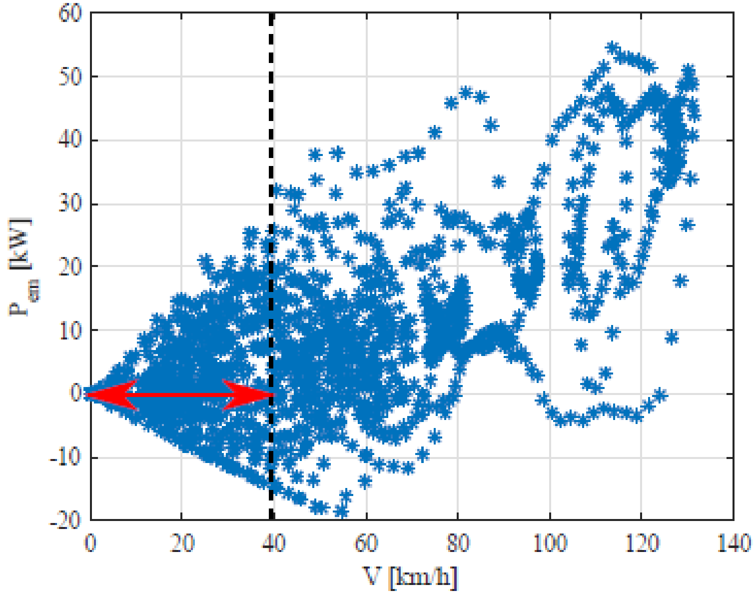


1 – Peak Power
2 – Base Speed
3 – Peak Torque
4 – Max Speed
5 – Torque@Max Speed

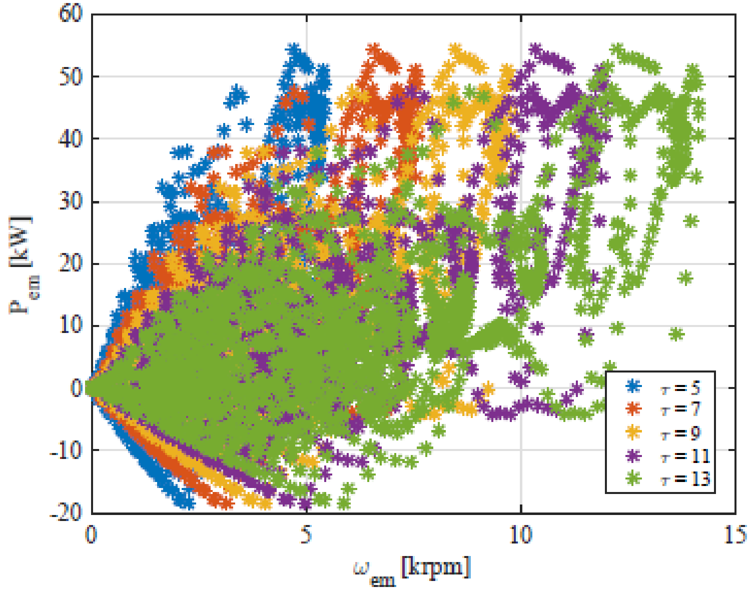
Co-funded by the
Erasmus+ Programme
of the European Union

System Requirements

- Electric Motor Specifications
  - 1. Peak Power
    - Peak Power is obtained considering the working power points on a driving cycle;
    - A maximum electric traction speed is considered as the maximum vehicle speed that the vehicle can be driven in pure electric mode;
    - It should be compliant with power available in regeneration mode.



(a) P_{em} vs. V in a WLTP cycle.



(b) P_{em} vs. ω_{em} in a WLTP cycle

Co-funded
Erasmus+ Prog
of the Europea

System Requirements

- Electric Motor Specifications
  - 2. Base speed
    - The electric motor base speed is the speed at which the power reaches its maximum value. It is computed based on the previously imposed maximum electric traction vehicle speed.

$$\omega_{\mathrm{em,base}} = V_{\mathrm{v,max,PE}}/R_{\mathrm{r}}\tau_{\mathrm{em}}$$

    - The effect of the transmission ratio between motor and wheels should be understood based also on further analysis related to the peak torque.

τ_{em} [−]	10	11	12	13
$\omega_{\mathrm{em,base}}$ [rpm]	3537	3890	4244	4598

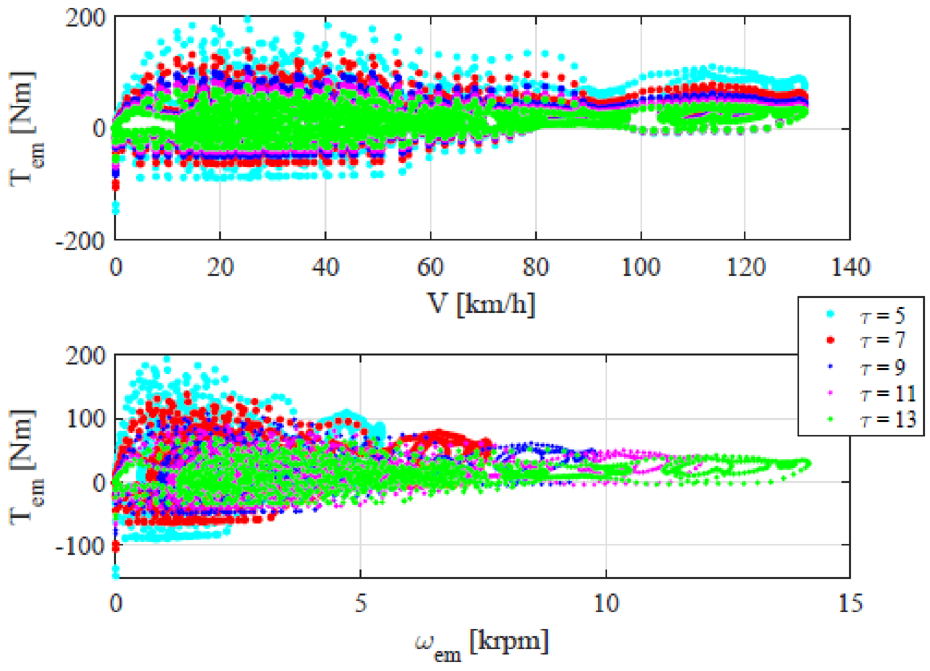
Considering VmaxPE = 40 km/h

Co-funded by the Erasmus+ Programme of the European Union

System Requirements

- Electric Motor Specifications
  - 3. Peak Torque
    - Peak torque is obtained considering the vehicle travelling along the driving cycle;
    - A sensitivity regarding the motor to wheels transmission ratio should be investigated;

- The design can be driven to a high speed – low torque motor or to a high torque – low speed motor
- In the first case the electromagnetic design should be properly performed to allow the motor to be driven towards high speeds
- In the second case the motor volume should be properly optimized since this kind of design could lead to high mass motors.



Co-funded by the Erasmus+ Programme of the European Union

FOR EDUCATIONAL PURPOSE ONLY

System Requirements

- Electric Motor Specifications
  - 3. Peak Torque
    - Another specification for the peak torque regards a functionality independent from the driving cycle following;
    - The vehicle should actually be able to overcome a ramp in electric mode;
    - 15% grade, 1- 2 km/h/s

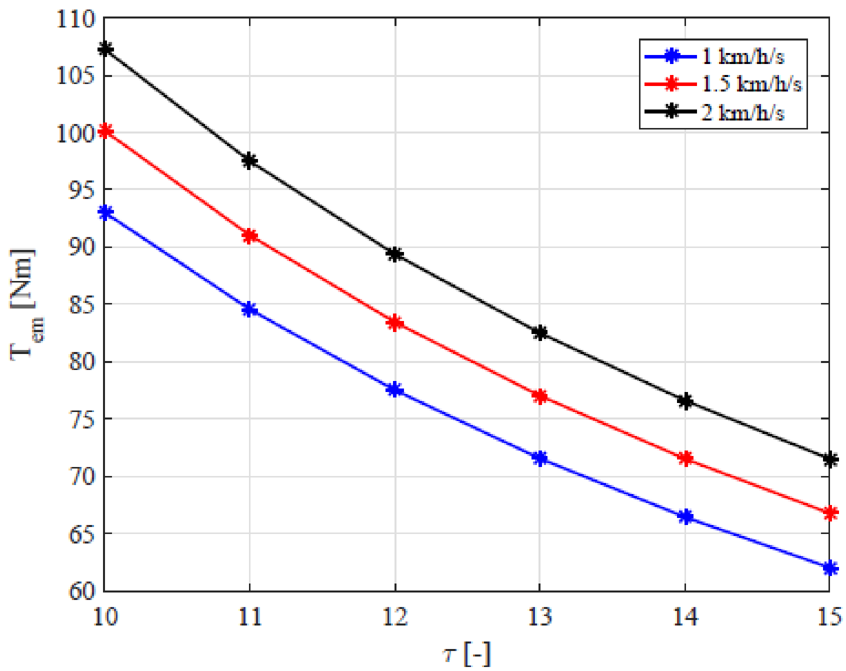
Road grade

$$Mg\,(f_0 \cos\alpha + \sin\alpha)\,V = AV$$

$$T_{em} = \left(\frac{J_{em}\eta_t \tau^2}{R_r^2} + \frac{J_w}{R_r^2} + M \right) \frac{R_r}{\tau \eta_t} a_x + \frac{AR_r}{\tau \eta_t}$$

Equivalent Mass

Acceleration Imposed



Co-funded by the Erasmus+ Programme of the European Union

System Requirements

- Electric Motor Specifications
  - 4. Max Speed
    - Maximum speed of the electric motor should be selected based on the connection between the electric motor and the wheels;
    - If the motor is connected to the wheels by friction or dog clutches, upon certain speeds the motor can be uncoupled with the vehicle. Its maximum speed is therefore function of the torque/transmission ratio selection;
    - If the motor cannot be uncoupled with the vehicle, its maximum speed should be equal to the maximum vehicle speed. As an example, it can be considered the maximum vehicle speed during the driving cycle.

Co-funded by the Erasmus+ Programme of the European Union

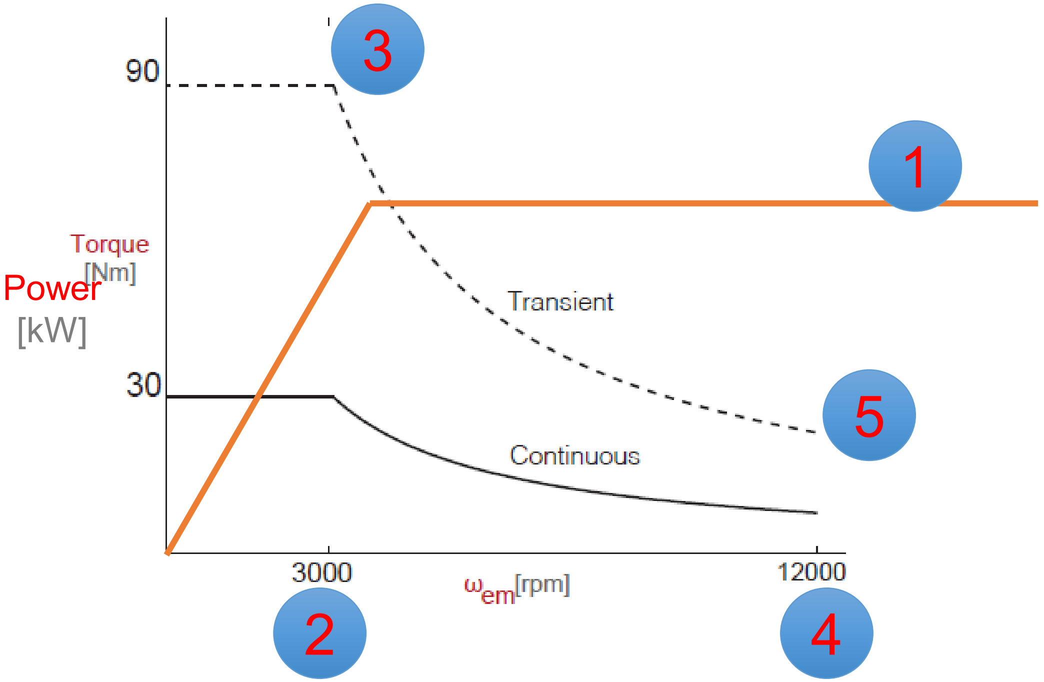
System Requirements

- Electric Motor Specifications
  - 5. Torque@Max Speed
    - There can be the possibility to drive the vehicle in pure electric mode up to the maximum speed in the driving cycle.
    - To this end, since at the maximum speed the power is at its maximum, the torque should be sufficiently high to overcome rolling and aerodynamic resistances:

$$T_{em} = \left[Mg\left(f_0 + KV^2\right) + \frac{1}{2}\rho C_x A V^2 \right] \frac{R_r}{\tau}$$

Co-funded by the Erasmus+ Programme of the European Union

Electric Motor Torque and Power requirements



1 – Peak Power.
2 – Base Speed.
3 – Peak Torque
4 – Max Speed.
5 – Torque@Max Speed.

ω_b	3000 rpm
V_{DC}	48 V
$T_{\max}$	90 Nm
T_{cont}	30 Nm
ω_{max}	12000 rpm
$T@\omega_{max}$	≈15 Nm
P_{max}	30 kW
P_{cont}	10 kW
$i_{DC,\max}$	Depends on battery
$i_{DC,cont}$	Depends on battery

Co-funded by the Erasmus+ Programme of the European Union

Energy management, supply and storage systems

Co-funded by the
Erasmus+ Programme
of the European Union

Agenda

- Energy Management System
- Torque Split Strategy
- Influence of electric powertrain limitations

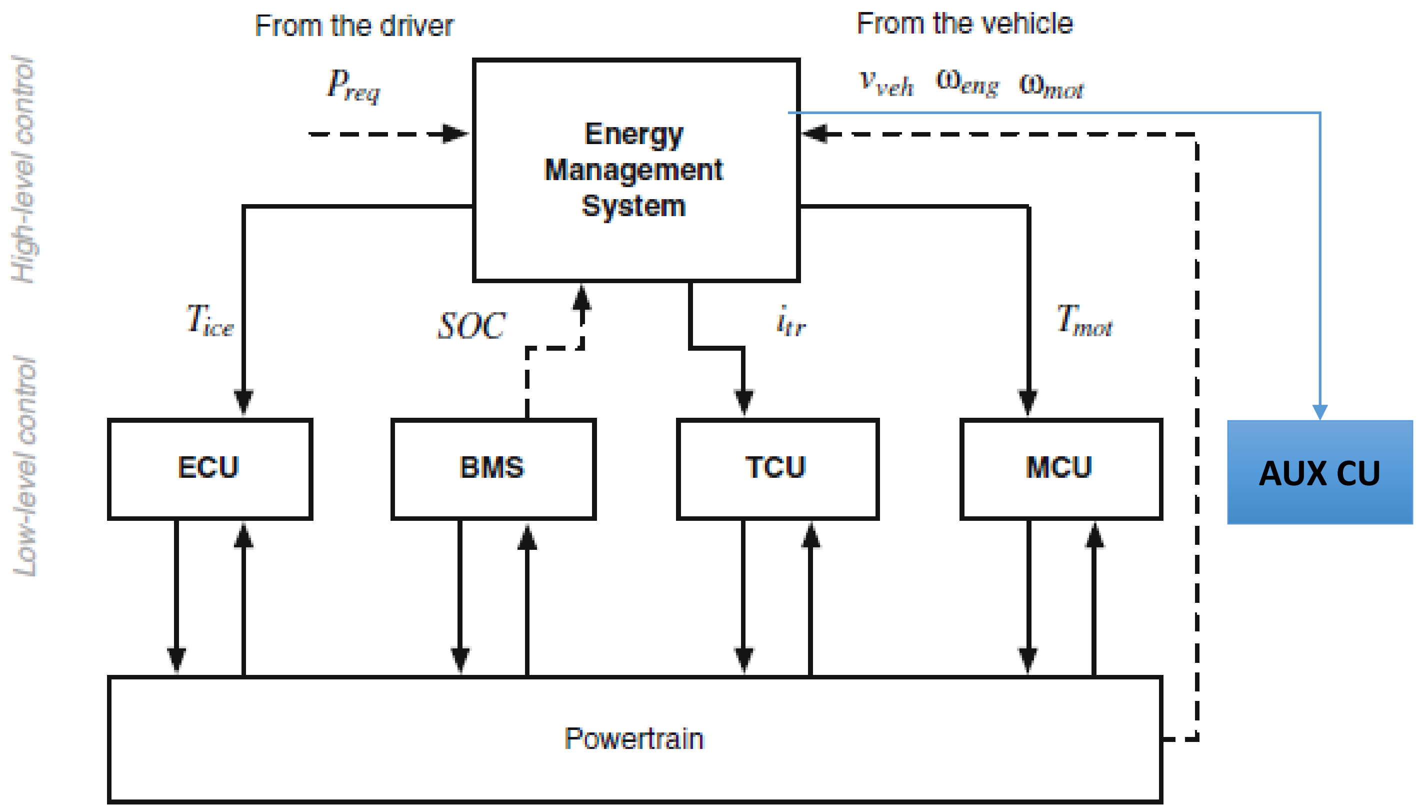
Co-funded by the
Erasmus+ Programme
of the European Union

Energy management and control strategies

- Energy Management System
  - Supervisory controller of each hybrid architecture
  - Responsible for the optimization of the energy flow on the vehicle with the goal of maintain the battery state of charge within certain thresholds;
  - Layer between the driver commands and the low level powertrain components controllers (vehicle management unit, inverters etc.)

Co-funded by the Erasmus+ Programme of the European Union

Energy Management System



Source: Onori

ECU Engine Control Unit

BMS Battery Management System

TCU Transmission Control Unit

MCU Motor Control Unit

Co-funded by the Erasmus+ Programme of the European Union

Energy management and control strategies

Energy Management System is responsible for power split between the ICE and electric motor

Rule based strategy

- Rules are set to control on each time instant
- Deep knowledge of the process is required
- Heuristic approache (not optimal, suffciently effective)

Optimization based strategy

- Set points are obatained by minimizing the cost function
- Driving path should be known prior to optimization.

Co-funded by the Erasmus+ Programme of the European Union

Energy Management System – Torque Split Strategy

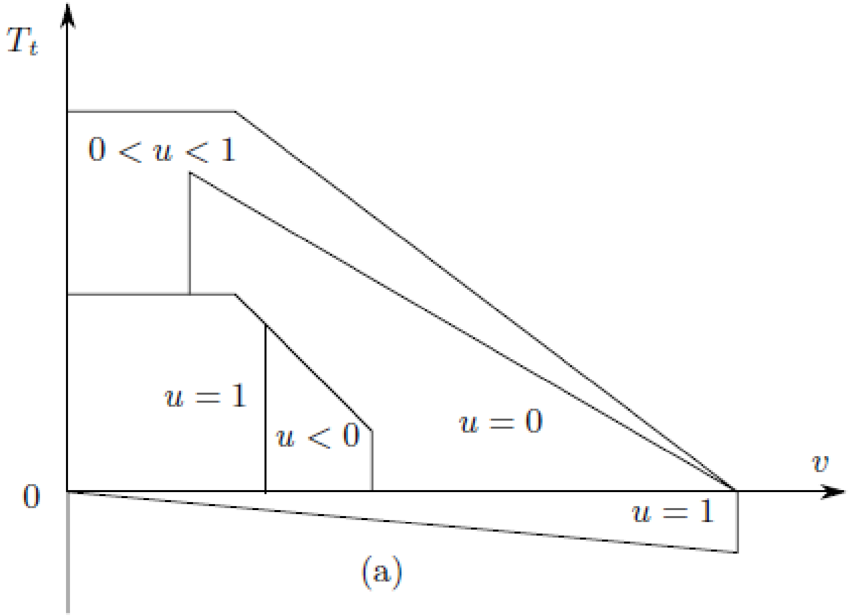
Rule – based EMS, example

“Thermostat”

Engine ON and charges battery when SOCmin
Engine OFF when battery SOCmax
Engine works in Optimal Operating line

“Electric assist”

IF SOC is close to lower limit
  Engine torque is increased to charge the battery
  No electric traction at low speeds
ELSE
  Engine works only in favorable zones
  Electric traction at low speeds
  High torque demands are accomplished by Engine



(a)

- Tt = required torque at ground level;
- V = vehicle speed;
- u = torque split ratio:

$$u = \frac{T_{em,ground}}{T_t}$$

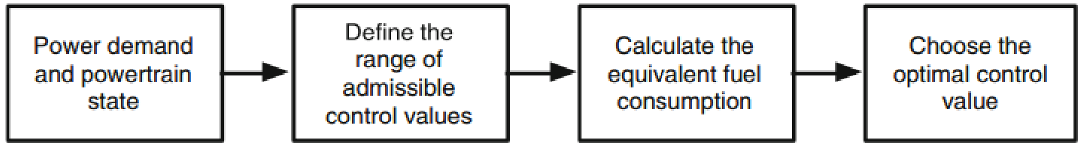
- u = 1 pure electric;
- u = 0 pure thermal;
- 0 < u < 1 power split;

Co-funded by the Erasmus+ Programme of the European Union

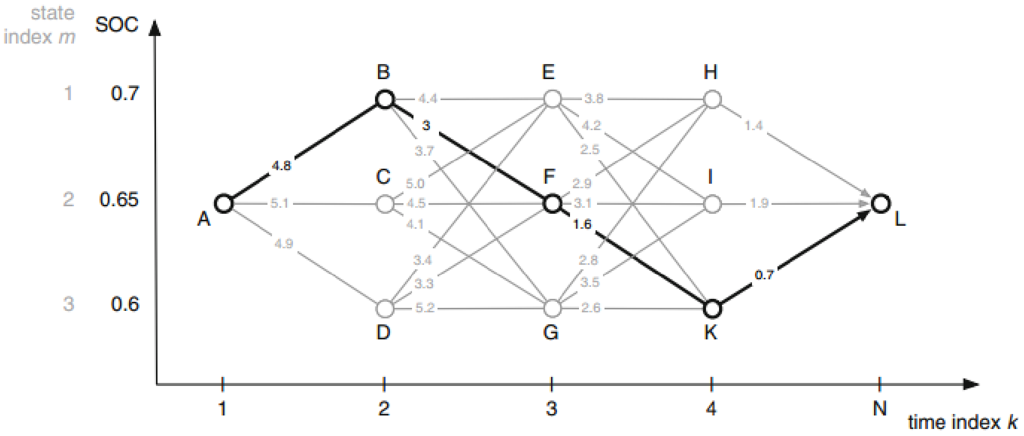
Source: Onori

Energy Management System – Torque Split Strategy

Equivalent Consumption Minimization Strategy



Dynamic programming

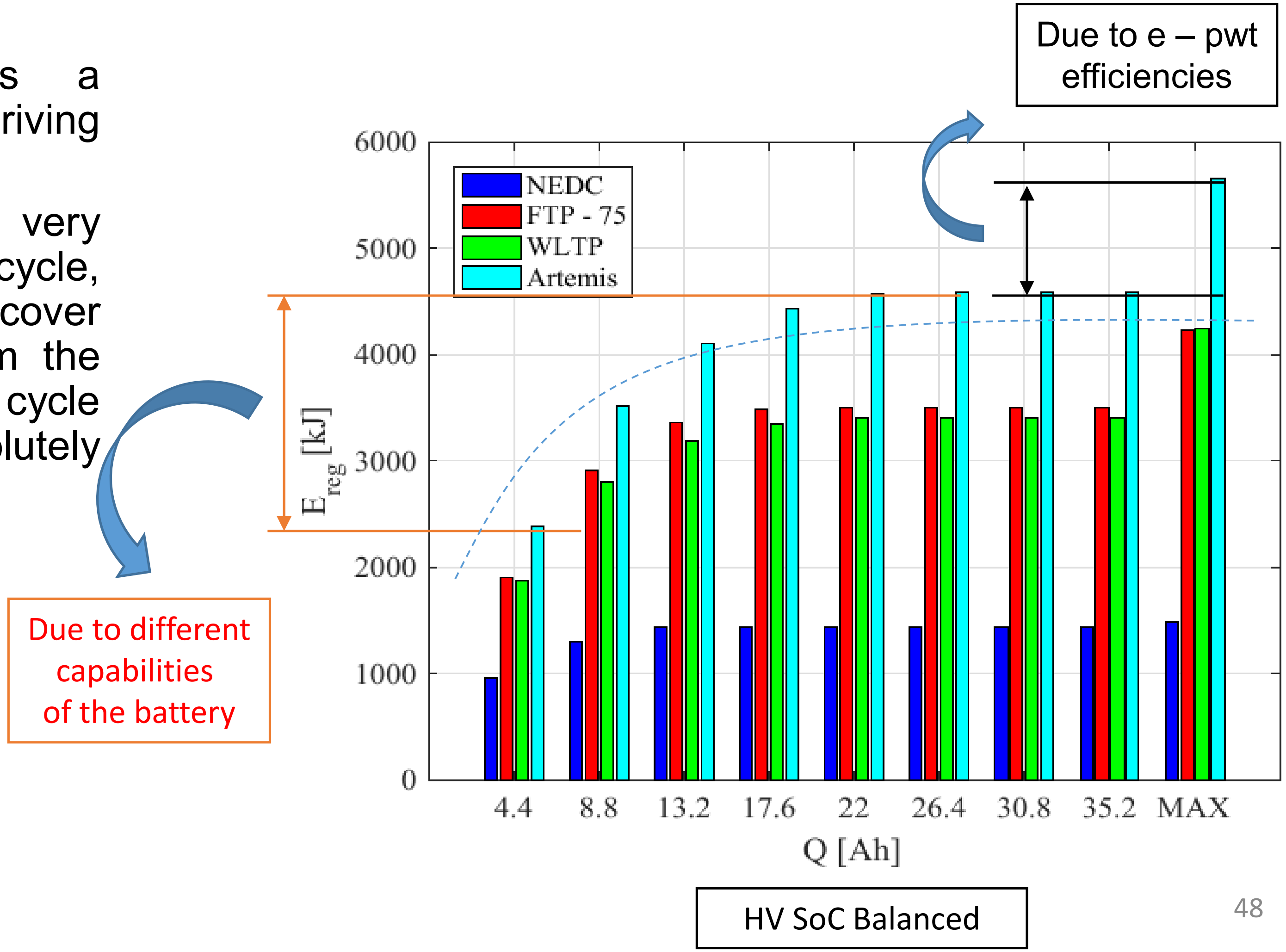


Source: Onori

Co-funded by the Erasmus+ Programme of the European Union

Influence of e – PWT limitations: Battery Pack Sizing

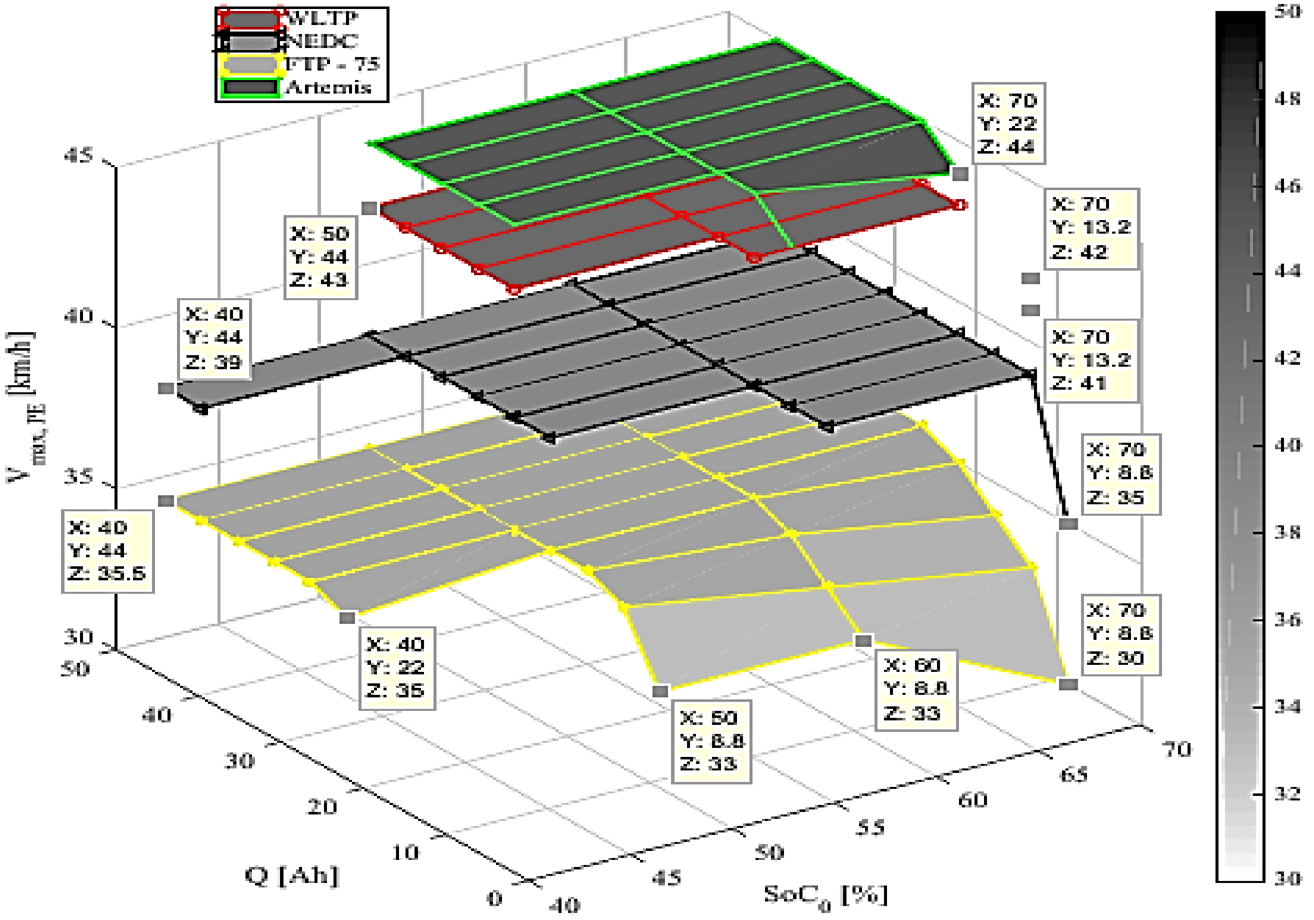
- The recovered energy presents a saturation for all the considered driving cycles;
- The choice of the cycle is still very important: considering an NEDC cycle, even a small battery pack is able to recover almost all the available energy from the cycle itself. Moving to a heavier cycle (WLTP) such battery is absolutely undersized.



Co-funded by the Erasmus+ Programme of the European Union

Influence of e – PWT limitations: Battery Pack Sizing

Influence of maximum electric traction vehicle speed



Co-funded by the Erasmus+ Programme of the European Union

References

1. Onori S, Serrao L, Rizzoni G. Hybrid electric vehicles: energy management strategies. Berlin Heidelberg: Springer; 2016.

2. Guzzella L, Sciarretta A. Vehicle propulsion systems. Springer-Verlag Berlin Heidelberg; 2007 June

3. Genta G. Motor vehicle dynamics: modeling and simulation. World Scientific; 1997.

4. Castellazzi L. Mild Hybrid Electric Vehicles: Powertrain Optimization for Energy Consumption, Driveability and Vehicle Dynamics Enhancements.

5. International Energy Agency. Key trends in CO2 emissions. In: CO2 emissions from fuel combustion. IEA, 2015

6. Schaeffler Symposium 2018, Mobility for Tomorrow, https://www.schaeffler.com/remotemedien/media/_shared_media/08_media_library/01_publications/schaeffler_2/symposia_1/downloads_11/schaeffler_kolloquium_2018_en.pdf

Co-funded by the Erasmus+ Programme of the European Union

UNITED

Engineering Knowledge Transfer Units to Increase Student´s Employability and Regional Development

https://www.facebook.com/unitederasmus/

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
598710-EPP-1-2018-1-AT-EPPKA2-CBHE-JP

Engineering Knowledge Transfer Units to Increase Student´s Employability and Regional Development

Introduction to automotive development

Day 2 – Slot 1

Christian Granrath, M.Sc.

FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
598710-EPP-1-2018-1-AT-EPPKA2-CBHE-JP

Agenda

- Introduction and motivation
- Definition of technical terms
- Conventional development approaches
- Agile development approaches
- Challenges of agile engineering
- References

Co-funded by the
Erasmus+ Programme
of the European Union

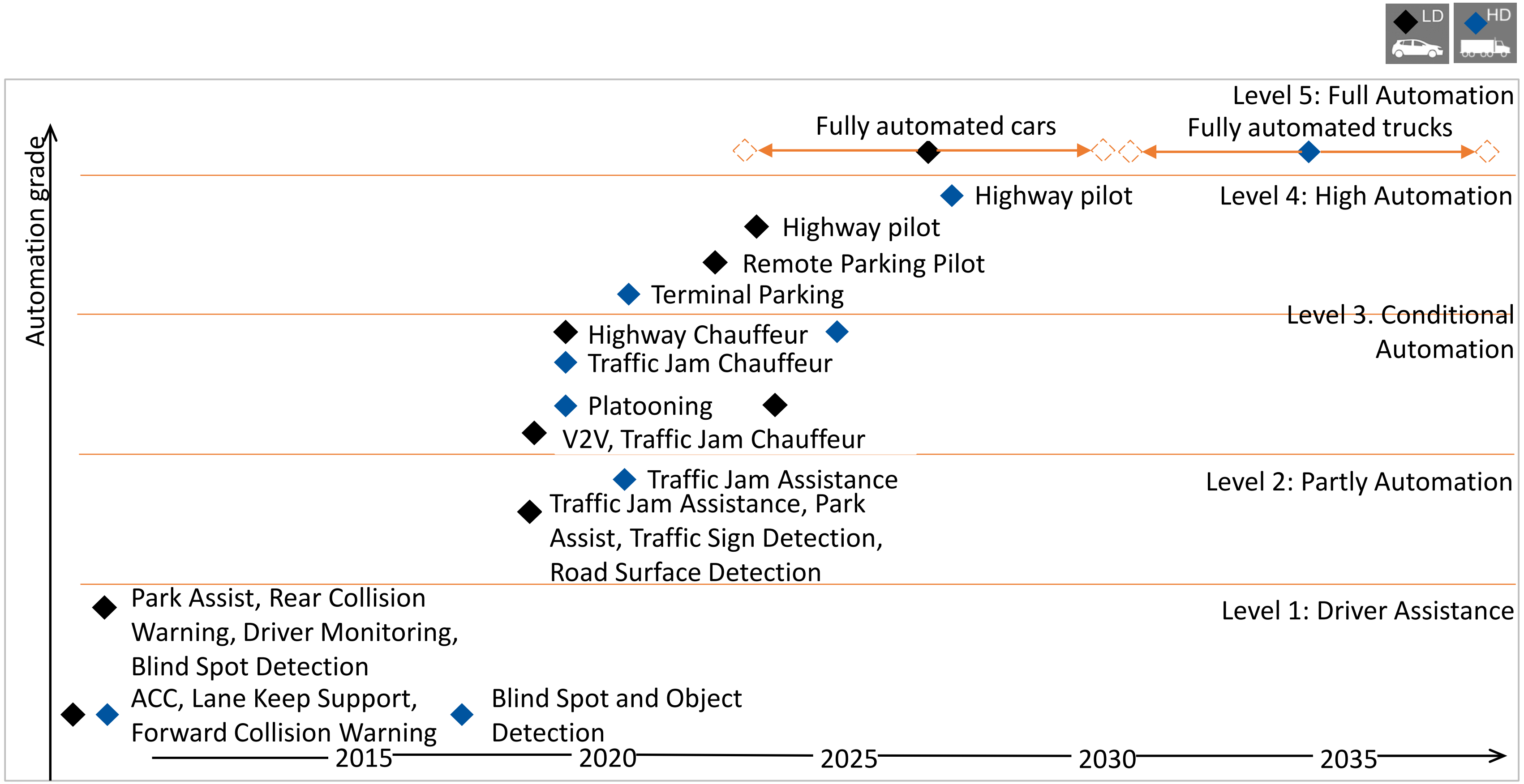
Learning objectives

- Motivation
  - What are the current trends within the automotive sector?
  - Why are these trends highly relevant for the entire development?
  - How are the current trends and the development approach influencing the organization structure?

Co-funded by the Erasmus+ Programme of the European Union

Level of automation will increase the next 20 years

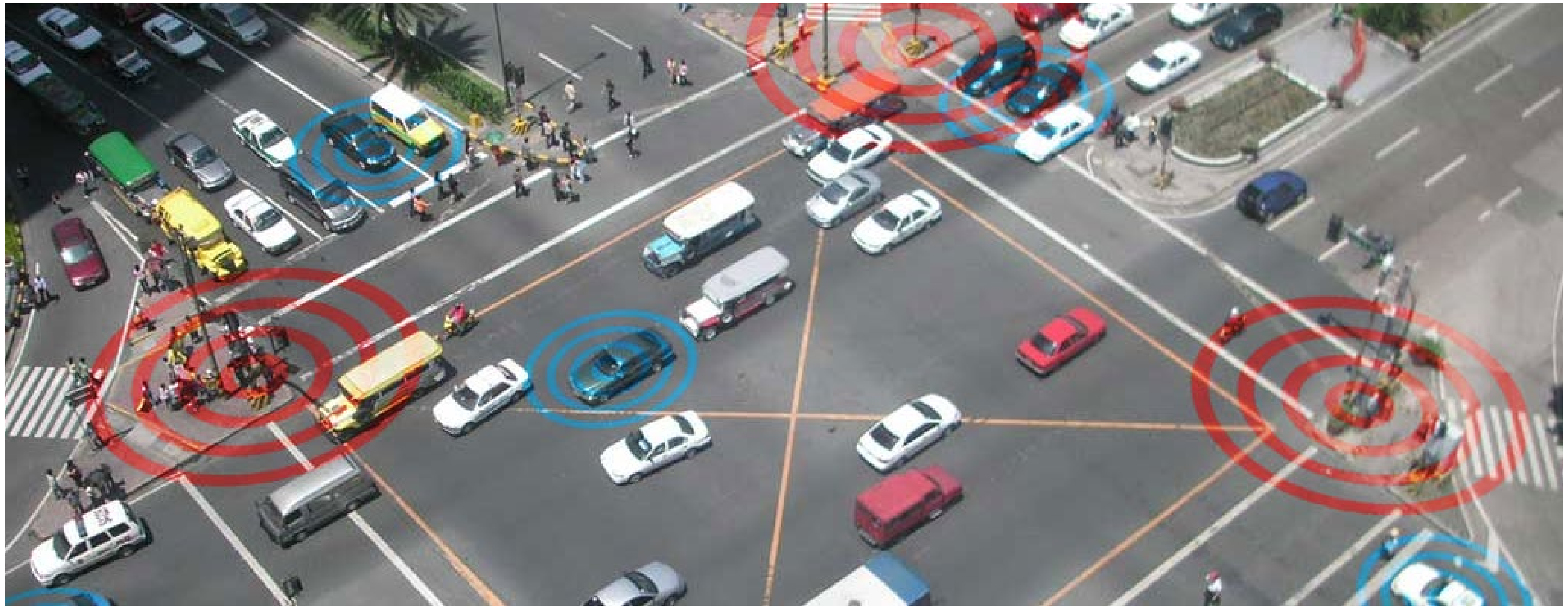
ESTIMATIONS FOR THE GRADE OF DRIVING AUTOMATION UNTIL 2035



Source: Prognose FEV, 05/2019 [16]

Co-funded by the Erasmus+ Programme of the European Union

5GAA (5G Automotive Association) Mission statement



Develop, test and promote **communications solutions**, initiate their standardization and accelerate their **commercial availability and global market penetration** to address **society's connected mobility and road safety needs** with applications such as autonomous driving, ubiquitous access to services and integration into smart city and **intelligent transportation**

Co-funded by the Erasmus+ Programme of the European Union

Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [22]
Image: Mike Gonzalez (TheCoffee) (https://commons.wikimedia.org/wiki/File:Makati_intersection.jpg), „Makati intersection", Crop and color changes by FH Aachen, https://creativecommons.org/licenses/by-sa/3.0/legalcode

Several players will coordinate the processes that enable car data monetization

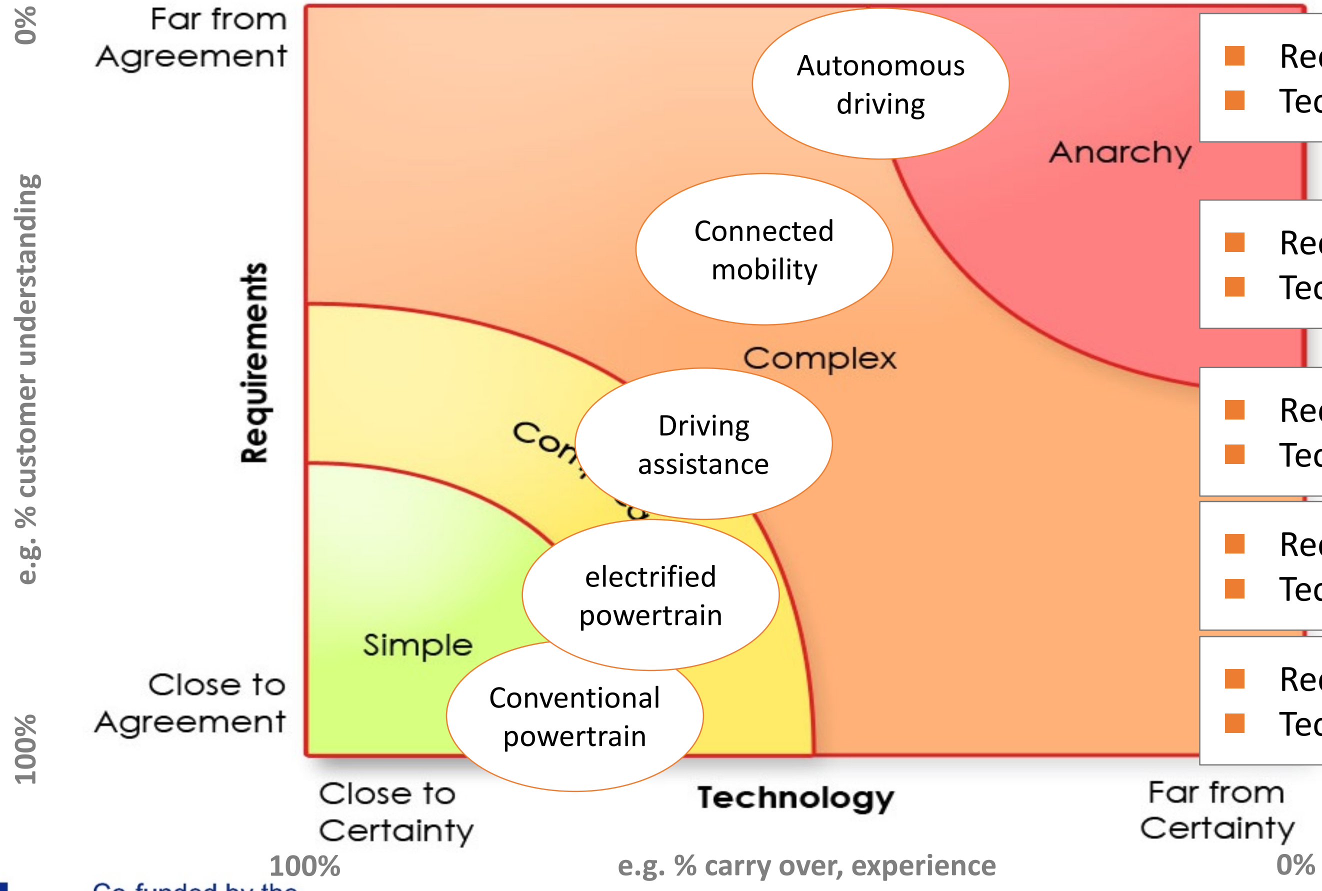
SERVICES BEYOND THE PRODUCT "VEHICLE" AND NETWORKED DRIVING

Source: McKinsey, "Monetizing car data" [1]
Image: Creative Commons Zero (https://visualcocaine.org/photo/525/city-night-view-from-above)

Co-funded by the Erasmus+ Programme of the European Union

The world is getting Volatile, Uncertain, Complex, Ambiguous (“VUCA”)

ANALYSIS AUTOMOTIVE INDUSTRY – VUCA EXTEND DEPENDS ON APPLICATION AREA



- Requirements: infrastructure? maps? connectivity?
- Technology: vehicle-centered? Connected?

- Requirements: CV2X standard? Business models?
- Technology: 802.11p vs. 4/5G, public/private cloud

- Requirements: safety strategy? maturity?
- Technology: sensor diversity, verifying AI

- Requirements: charging infrastructure
- Technology: battery technology, comp. integration

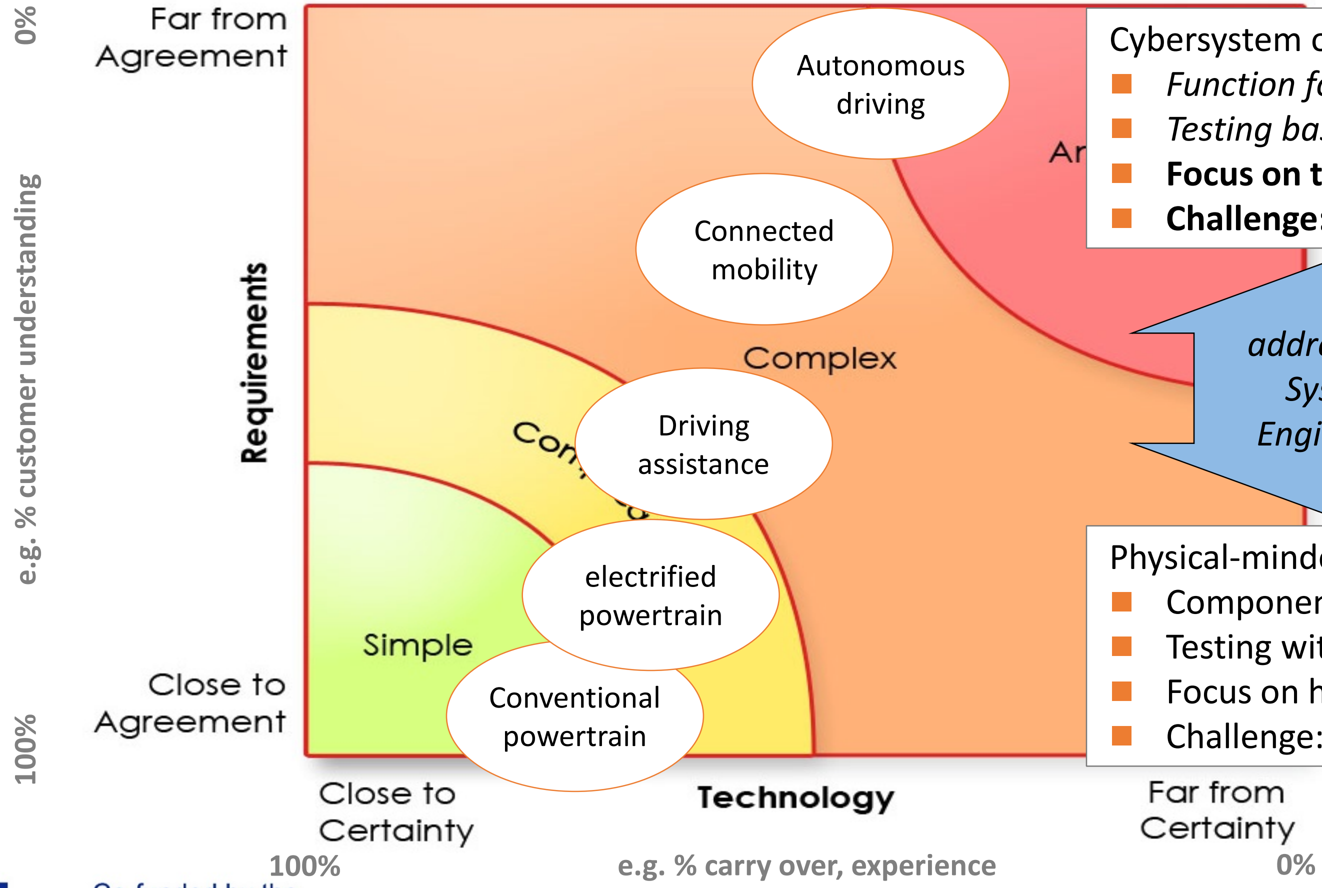
- Requirements: carry over, long experience
- Technology: emission reduction how?

Co-funded by the Erasmus+ Programme of the European Union

Source: https://www.scrum-tips.com/ [19]
Source: S. Kriebel, Lecture “Software for Combustion Engines”, RWTH Aachen University, 2019 [23]

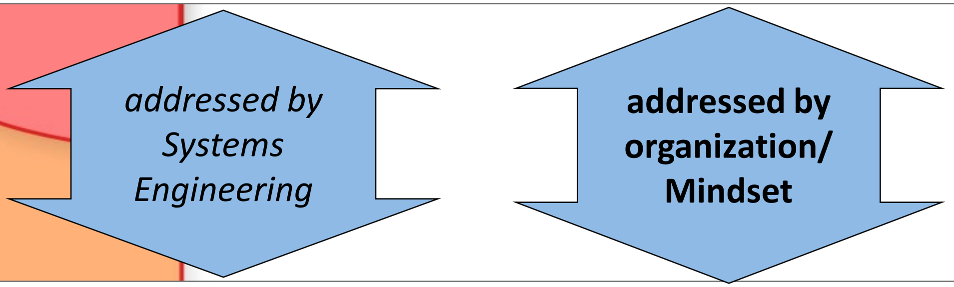
The world is getting Volatile, Uncertain, Complex, Ambiguous (“VUCA”)

FINDING THE RIGHT ORGANIZATION APPROACH IS CRUCIAL



Cybersystem organization

- *Function focus: system as function provider*
- *Testing based on user acceptance and requirements*
- **Focus on teams and quick change, agile is mindset**
- **Challenge: empowerment,** ***crossfunctionality*****, change**



Physical-minded organization

- Component focus: engine, transmission, battery …
- Testing with simulation and drive
- Focus on hierarchy, decision makers, agile is process
- Challenge: overcome heroism, stop micromanagement

Co-funded by the Erasmus+ Programme of the European Union

Source: https://www.scrum-tips.com/ [19]
Source: S. Kriebel, Lecture “Software for Combustion Engines”, RWTH Aachen University, 2019 [23]

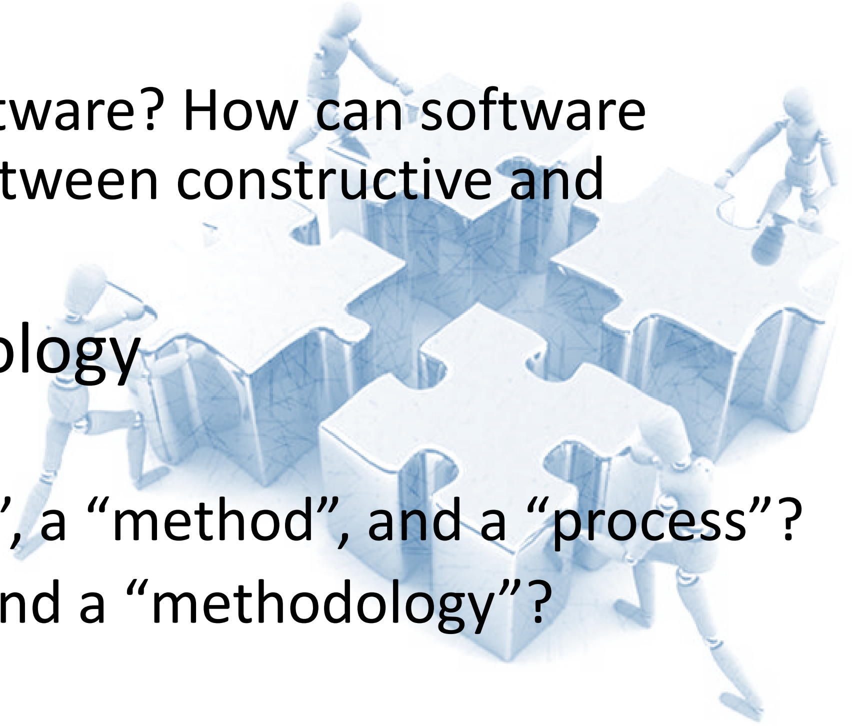
Agenda

- Introduction and motivation
- Definition of technical terms
- Conventional development approaches
- Agile development approaches
- Challenges of agile engineering
- References

Co-funded by the
Erasmus+ Programme
of the European Union

Learning objectives

- Software quality
  - What does this term mean (definition)? How is software quality defined by ISO 25010 (quality characteristics)?
  - Why is the assurance of quality crucial for software? How can software quality be ensured? What is the difference between constructive and analytical measures?
- Process, procedure, method, and methodology
  - What does the term “process” mean?
  - What is the difference between a “procedure”, a “method”, and a “process”?
  - What is the difference between a “method” and a “methodology”?



Co-funded by the Erasmus+ Programme of the European Union

Image: Jasmin Hamadeh (http://www.jasmin-hamadeh.de/wp-content/uploads/2013/03/team_puzzle_clipart.jpg), https://creativecommons.org/licenses/by-nd/3.0/deed.de

How is quality defined?

AS DEFINED BY EN ISO 8402 : 1995:

"Totality of characteristics

- of an entity (*for example a product or a process*)
- that bear on its ability to satisfy stated and implied needs."

Quality of product

Quality of process

Subject to requirements

Co-funded by the Erasmus+ Programme of the European Union

Source: EN ISO 8402 : 1995, DIN Deutsches Institut für Normung e.V., Beuth Verlag GmbH, 10772 Berlin [17]

There is more than only functional requirements

THE PRODUCT QUALITY MODEL OF ISO 25010

Software Quality

Functional Suitability	Performance efficiency	Compatibility	Usability	Reliability	Security	Maintainability	Portability
Functional appropriateness	Time-behaviour	Co-existence	Appropriateness recognizability	Maturity	Confidentiality	Modularity	Adaptability
Functional correctness	Resource utilization	Interoperability	Learnability	Availability	Integrity	Reusability	Installability
Functional completeness	Capacity		Operability	Fault tolerance	Non-repudiation	Analyzability	Replaceability
			User error protection	Recoverability	Accountability	Modifiability	
			User interface aesthetics		Authenticity	Testability	
			Accessibility				

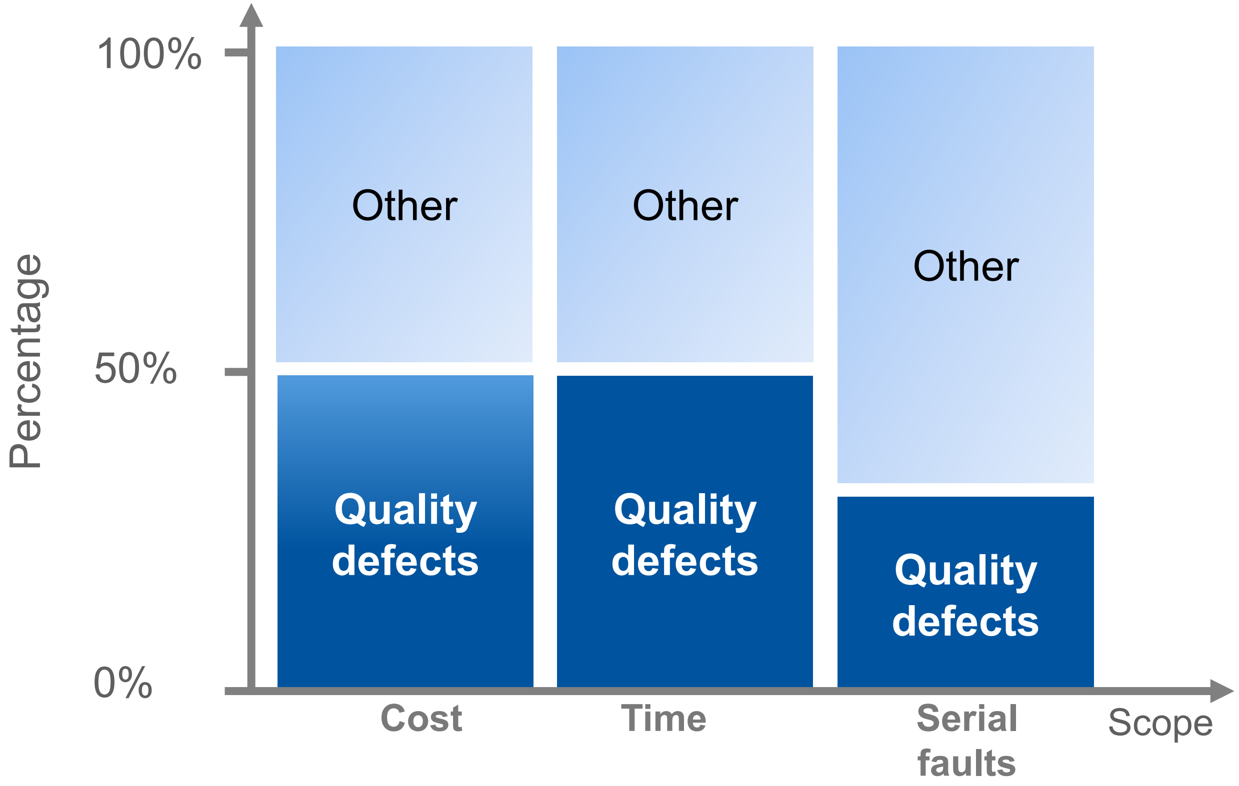
Source: International Standardization Organization (ISO) [18]
Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [22]

Co-funded by the Erasmus+ Programme of the European Union

Troubleshooting takes time and money

NEED FOR SOFTWARE QUALITY

- **Quality defects cause effort**
  - Troubleshooting raises effort
  - Damage of up to 50%
- **Additional efforts through market conditions**
  - Growing complexity
  - Increasing number of product variants / software projects
  - New technologies with a lack of experience



Source: Dumke, Ebert (2007) "Software Measurement" [4]

Co-funded by the Erasmus+ Programme of the European Union

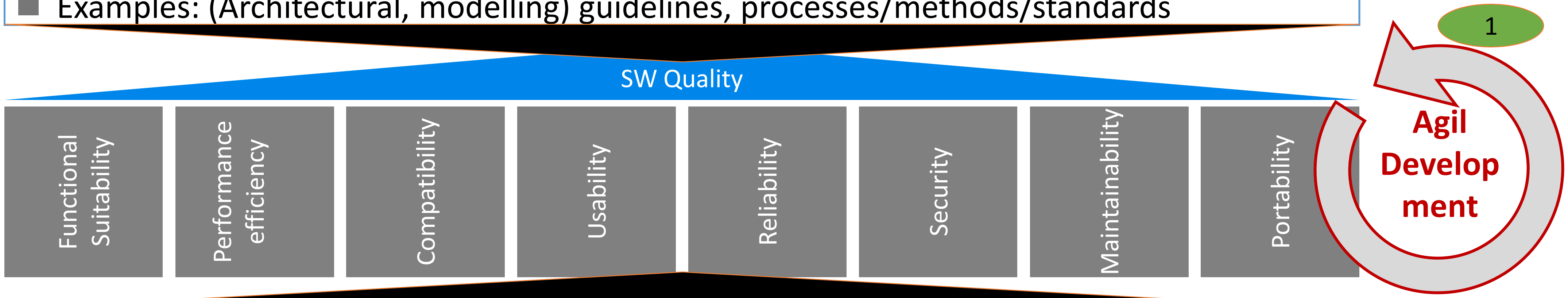
How can we ensure software quality?

SOFTWARE DEVELOPMENT = REALIZATION OF CONSTRUCTIVE & ANALYTIC MEASURES

Constructive measures

- Quality assurance a priori (in advance)
- Examples: (Architectural, modelling) guidelines, processes/methods/standards

1, 2



Analytic measures

4

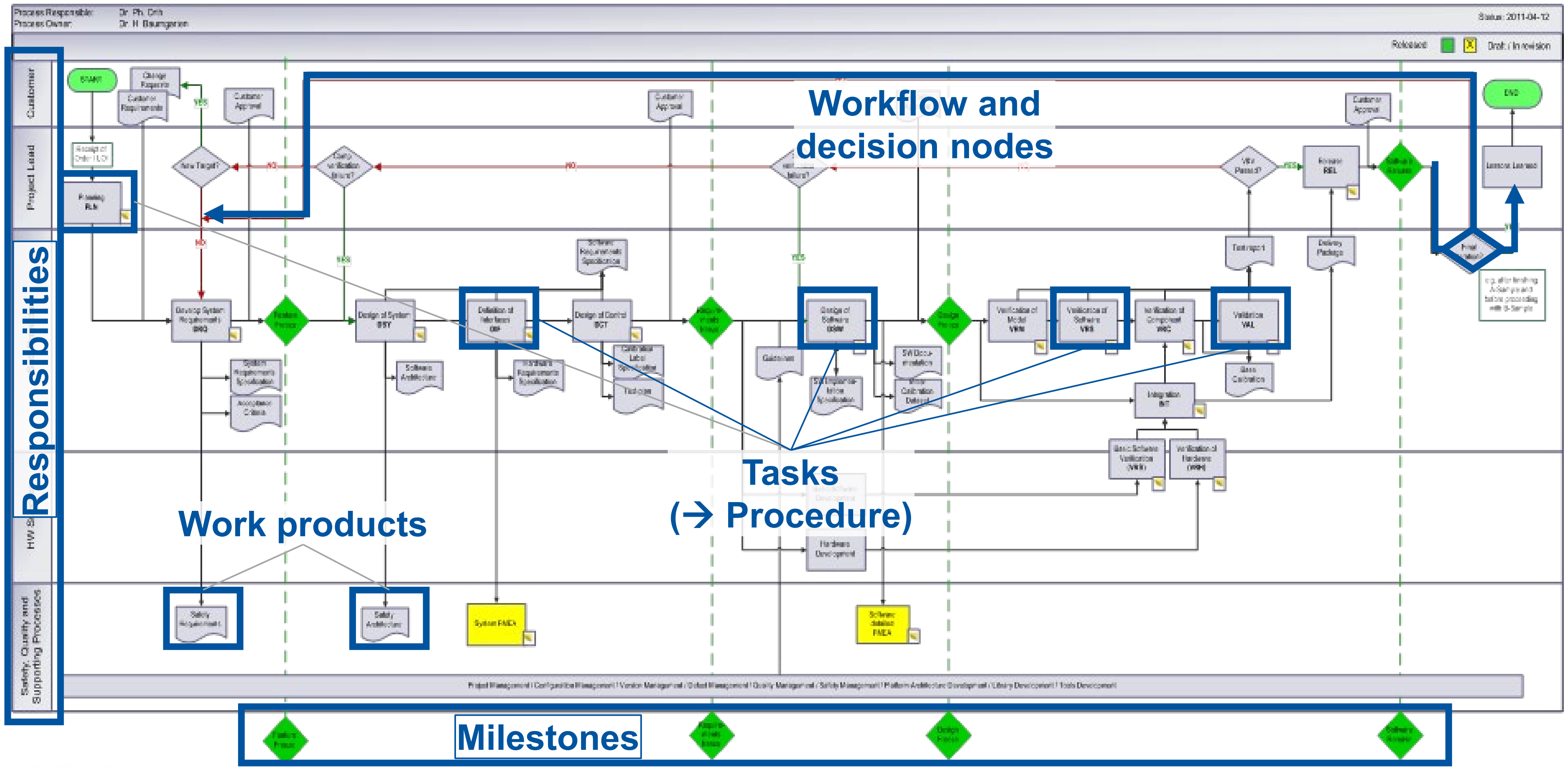
- Quality assurance a posteriori (measurement, improvement)
- Examples: Screening processes (reviews, guideline examinations, testing, process reviews)

Mapping to slots

Co-funded by the Erasmus+ Programme of the European Union

Source: J. Richenhagen, Lecture “Software for Combustion Engines”, RWTH Aachen University, 2019 [22]

Development process flow chart
Examples and key attributes

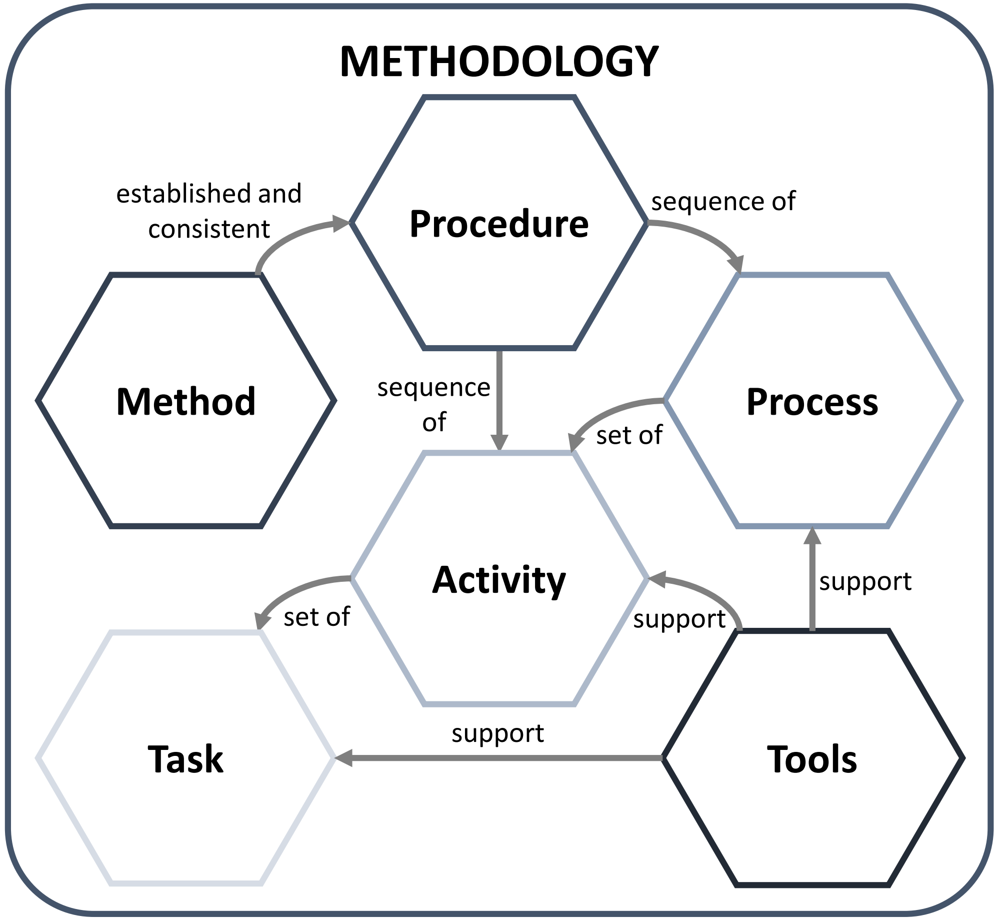


Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [22]

Common understanding of technical terms in interdisciplinary teams is mandatory

DEFINITION OF TECHNICAL TERMS

- **Methodology**: Collection of related processes, methods, procedures, tools and rules used to support a specific discipline
- **Method**: Systematic approach to accomplishing projects through detailed and logical plans based on a specific way of thinking
- **Procedure**: Established method of accomplishing a consistent performance or result
- **Process**: Set of interrelated activities that is concerned with transformation of input to output, including definition of roles, responsibilities, milestones, work products, duration, resources and not concerning how the required output is obtained
- **Activity**: Set of distinct scheduled tasks that consume time and resource and that are assigned to perform the realization of necessary outcomes
- **Task**: Element of an activity intended to contribute to the achievement of one or more outcomes

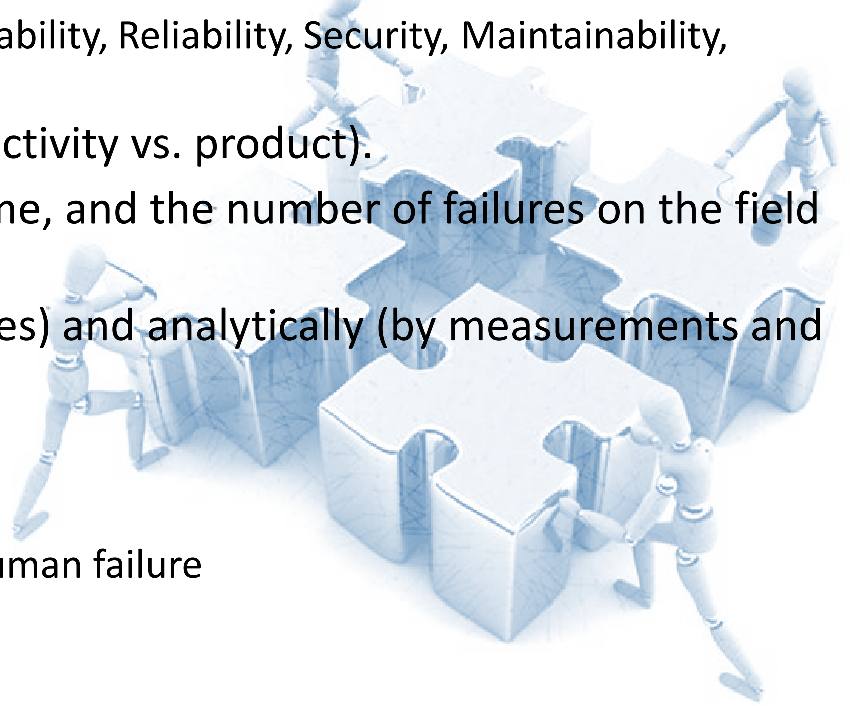


Source: International Organization for Standardization, Institute of Electrical and Electronics Engineers [26]

Co-funded by the Erasmus+ Programme of the European Union

Learning objectives and first conclusion

- Software quality
  - Quality is not an absolute figure, but rather describes the fulfillment of requirements.
  - These “requirements” are defined as quality characteristics e.g. in 8 groups as per ISO 25010:
    - Functional Suitability, Performance Efficiency, Compatibility, Usability, Reliability, Security, Maintainability, Portability
  - There is a distinction between process and product quality (activity vs. product).
  - Quality defects increase development costs, development time, and the number of failures on the field (mass production) → costs
  - Quality can be ensured constructively (by preventive measures) and analytically (by measurements and improvement).
  - Both approaches are necessary:
    - Constructive: Prevention of errors
    - Analytical: Detection of unavoidable errors, e.g. due to risks, human failure



Co-funded by the Erasmus+ Programme of the European Union

Agenda

- Introduction and motivation
- Definition of technical terms
- Conventional development approaches
- Agile development approaches
- Challenges of agile engineering
- References

Co-funded by the
Erasmus+ Programme
of the European Union

Learning objectives

- Procedure models
  - What software procedure models are available?
  - What are the steps these models are divided into?
    What are objectives and work products of the single working steps?
  - How can the V-Model be adapted to match current development needs?

Co-funded by the Erasmus+ Programme of the European Union

Why use procedure models?

UNITED

STRUCTURING ORGANIZATIONS USING DIVISION OF LABOR

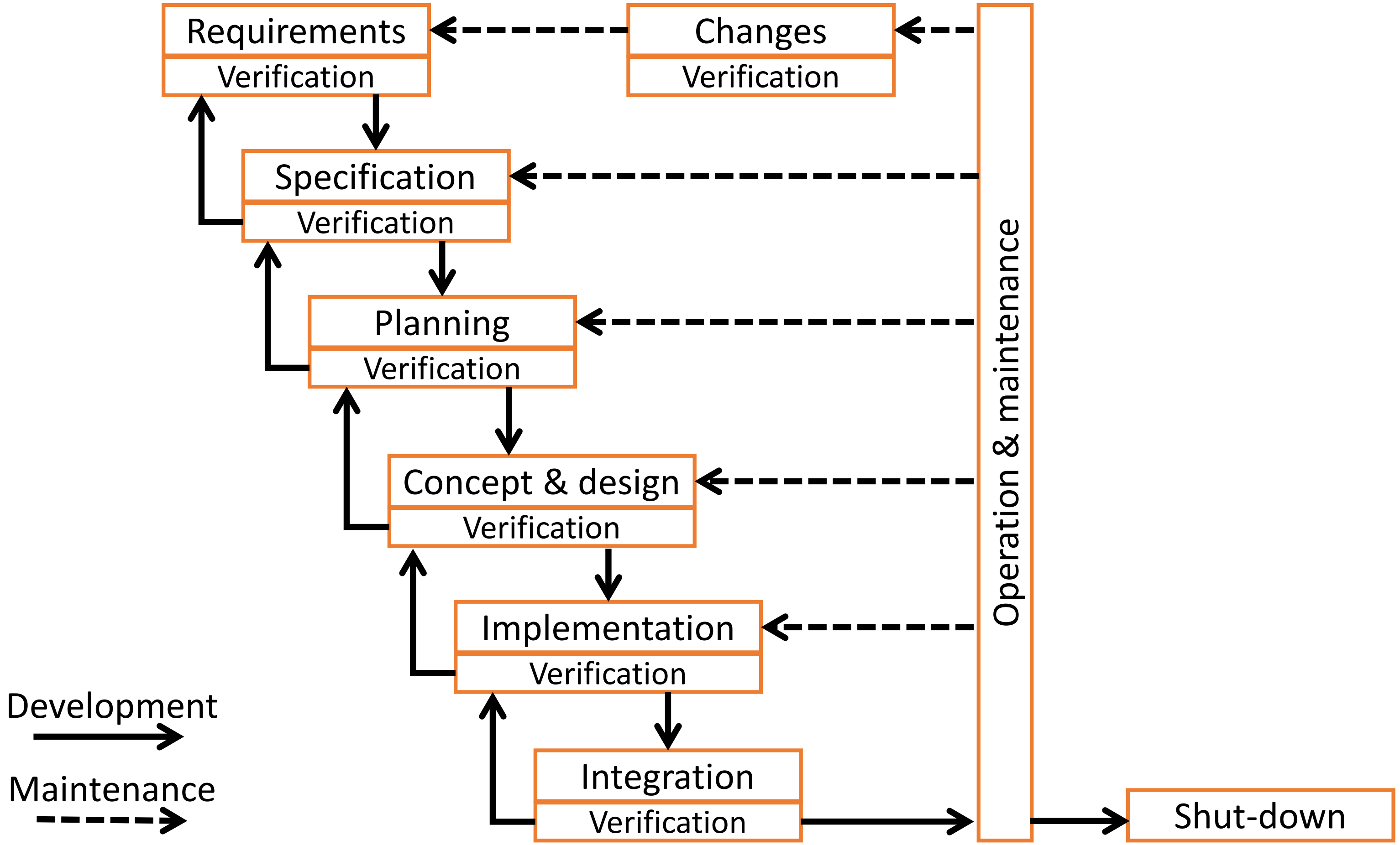
PROCEDURE MODELS

- define a specific sequence of steps
- specify a project strategy
  - chronological sequence of products to be created
  - necessary level of quality at a certain time
- assist the developer to find his way in the project
- arrange what products of a certain stage of completion shall be available in a defined quality
- **help to plan and monitor the software project within the scope of project management**

Co-funded by the Erasmus+ Programme of the European Union

Source: A. Schatten, M. Demolsky, D. Winkler, S. Biffl , E. Gostischa-Franta, T. Östreicher; Best Practice Software-Engineering [7]

Examples of procedure models
Waterfall model



Source: SOMMERVILLE, IAN: “Software Engineering” [8]
Source: J. Richenhagen, Lecture “Software for Combustion Engines”, RWTH Aachen University, 2019 [22]

Co-funded by the Erasmus+ Programme of the European Union

Evaluation of procedure models
Waterfall model

+

- good predictability
- definite stages

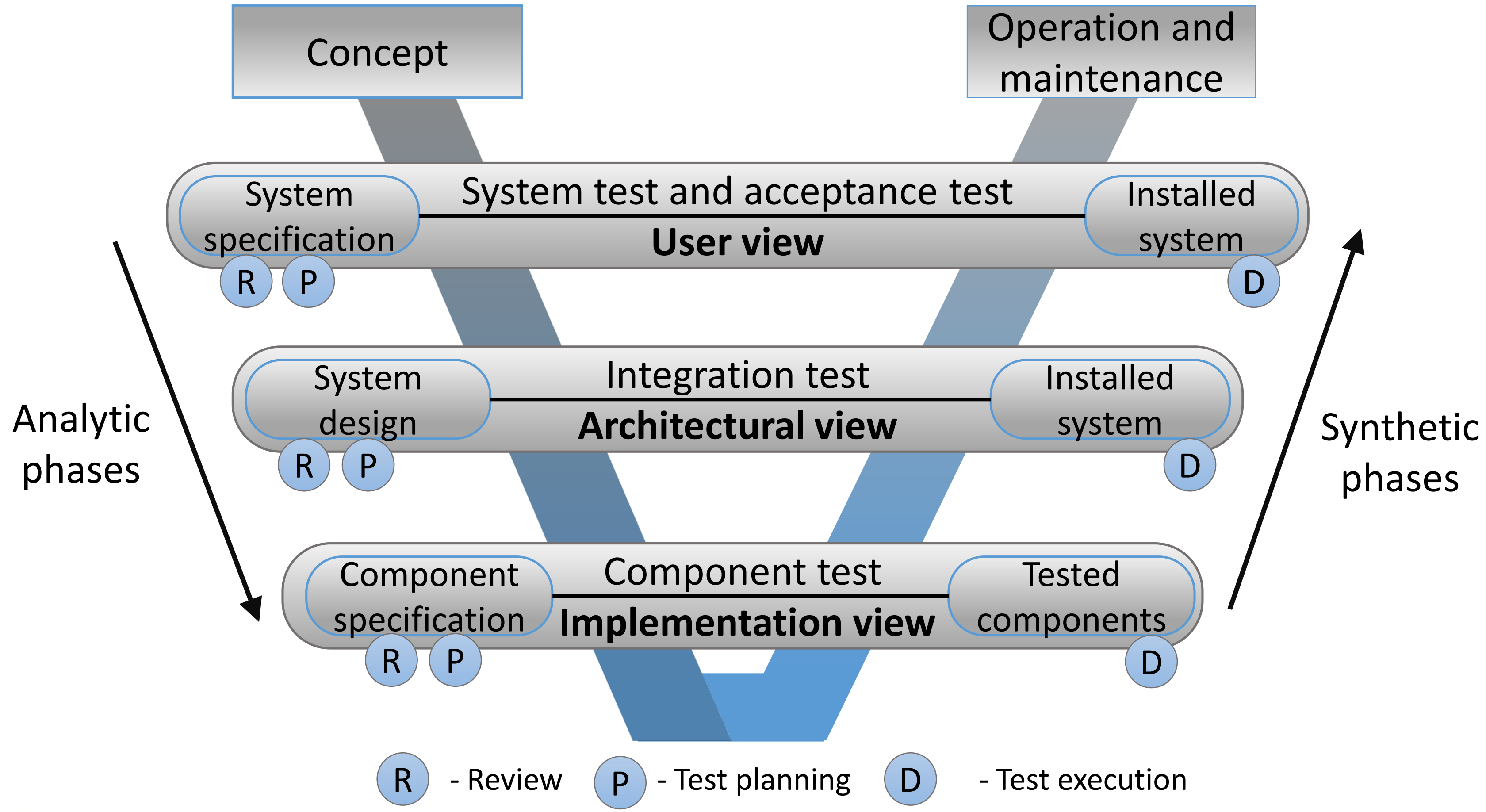
-

- low flexibility
- it is difficult to respond to (later) changes within the project
- errors are recognized too late
- late availability of software solutions due too sequential process

Co-funded by the Erasmus+ Programme of the European Union

Source: SOMMERVILLE, IAN: "Software Engineering" [8]
Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [22]

Examples of procedure models
V-Model



Co-funded by the Erasmus+ Programme of the European Union

Source: ADOLF-PETER BRÖHL and WOLFGANG DRÖSCHEL: “Das V-Modell: Der Standard für die Softwareentwicklung mit Praxisleitfaden” [10]
Source: J. Richenhagen, Lecture “Software for Combustion Engines”, RWTH Aachen University, 2019 [22]

Evaluation of procedure models
V-Model

+

- Particularly suitable for well-defined projects with stable and complete requirements
- Good predictability

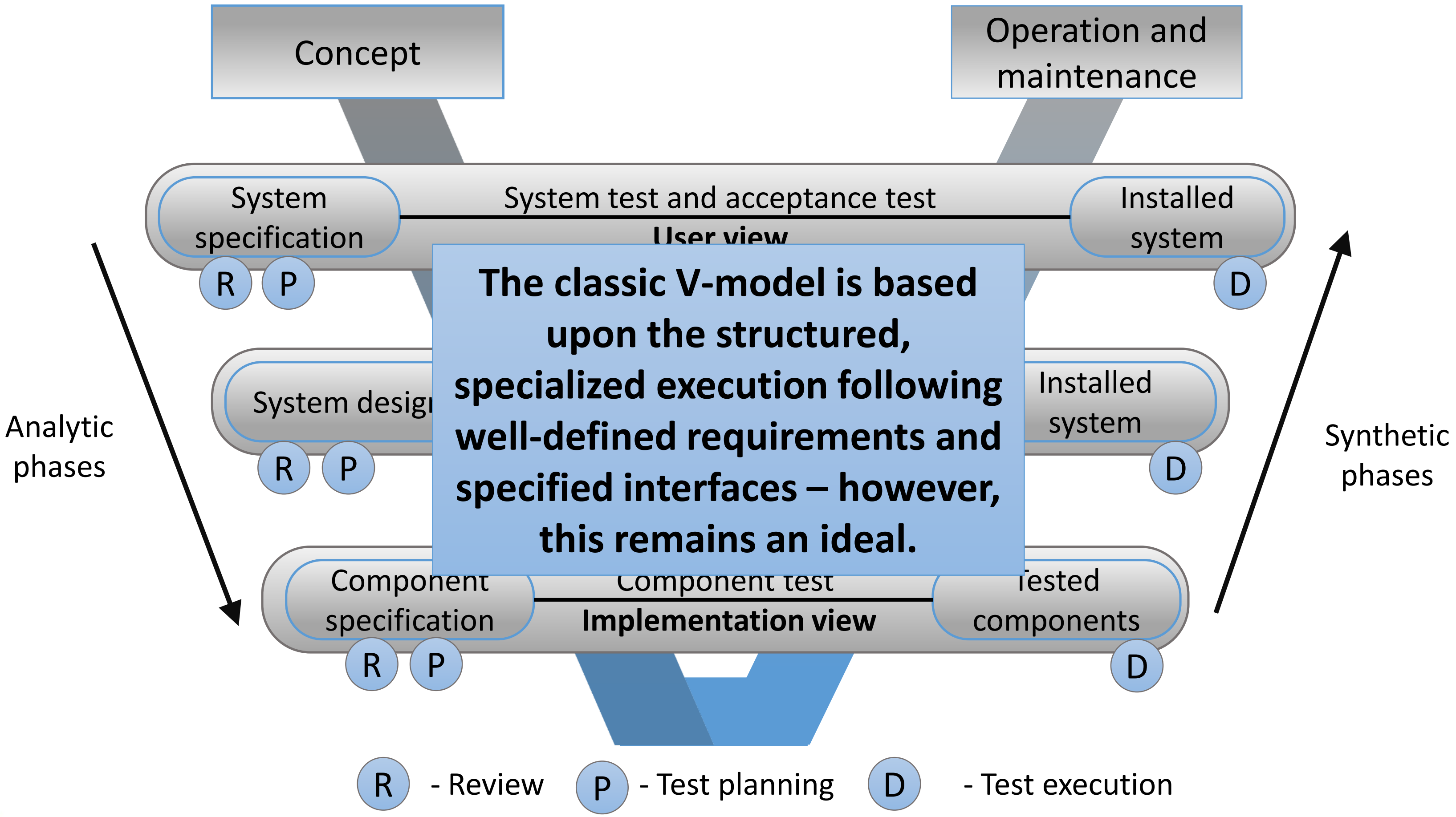
-

- Only limited possibility for regression
- Errors may lead to very high efforts for correction

Co-funded by the Erasmus+ Programme of the European Union

Source: ADOLF-PETER BRÖHL and WOLFGANG DRÖSCHEL: "Das V-Modell: Der Standard für die Softwareentwicklung mit Praxisleitfaden" [10]
Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [22]

Examples of procedure models
V-Model



The classic V-model is based upon the structured, specialized execution following well-defined requirements and specified interfaces – however, this remains an ideal.

Co-funded by the Erasmus+ Programme of the European Union

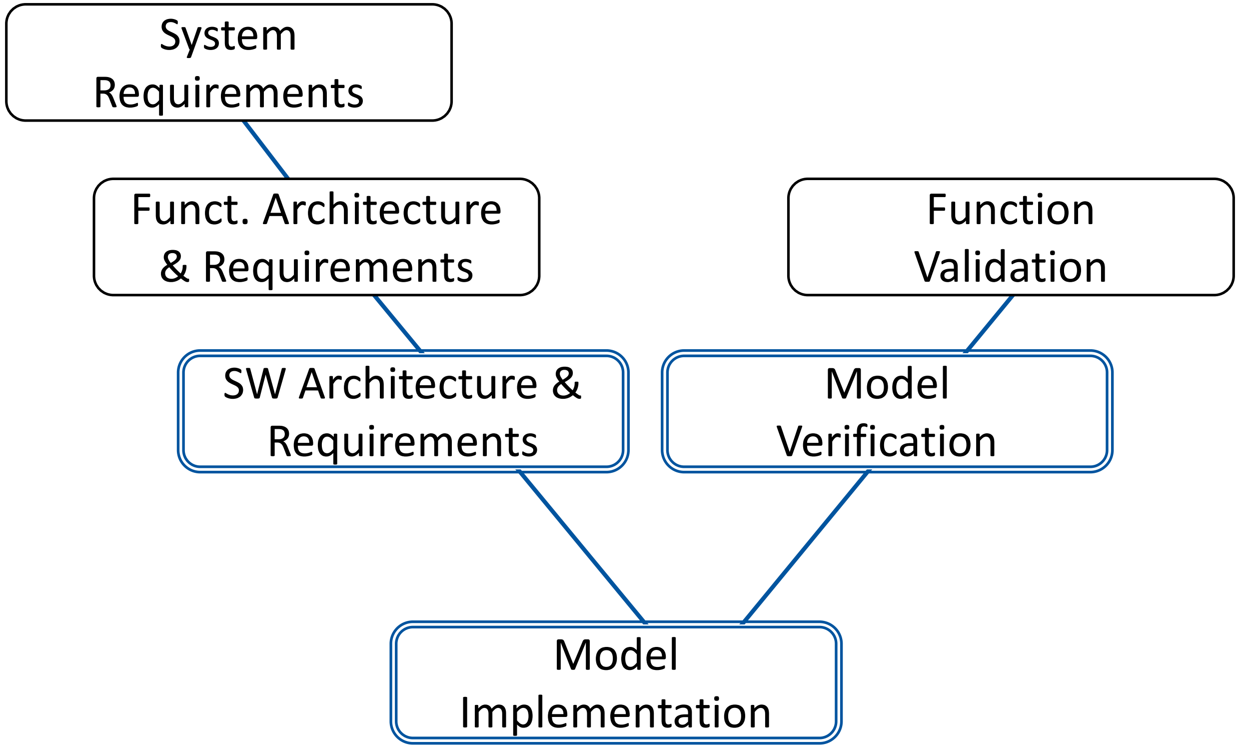
Source: ADOLF-PETER BRÖHL and WOLFGANG DRÖSCHEL: "Das V-Modell: Der Standard für die Softwareentwicklung mit Praxisleitfaden" [10]
Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [22]

V-Model in model-driven software engineering

I) MODEL DEVELOPMENT

Objectives

- Transition System → Software
- Knowledge of all requirements (e.g. control functions)
- Creation of correct models



Co-funded by the Erasmus+ Programme of the European Union

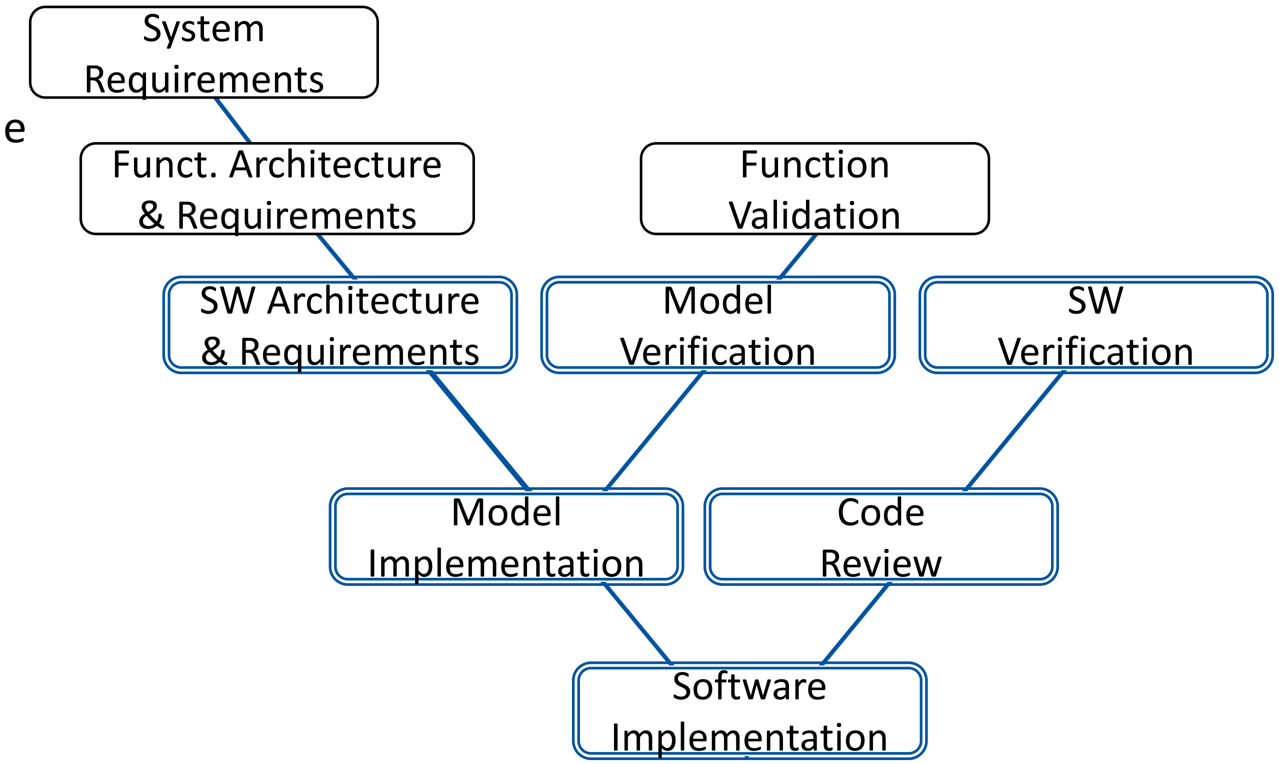
Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [22]

V-Model in model-driven software engineering

II) CODE DEVELOPMENT

Objectives

- Correct representation of requirements in the code
- Representation of required functionality (no overhead)
- Traceability

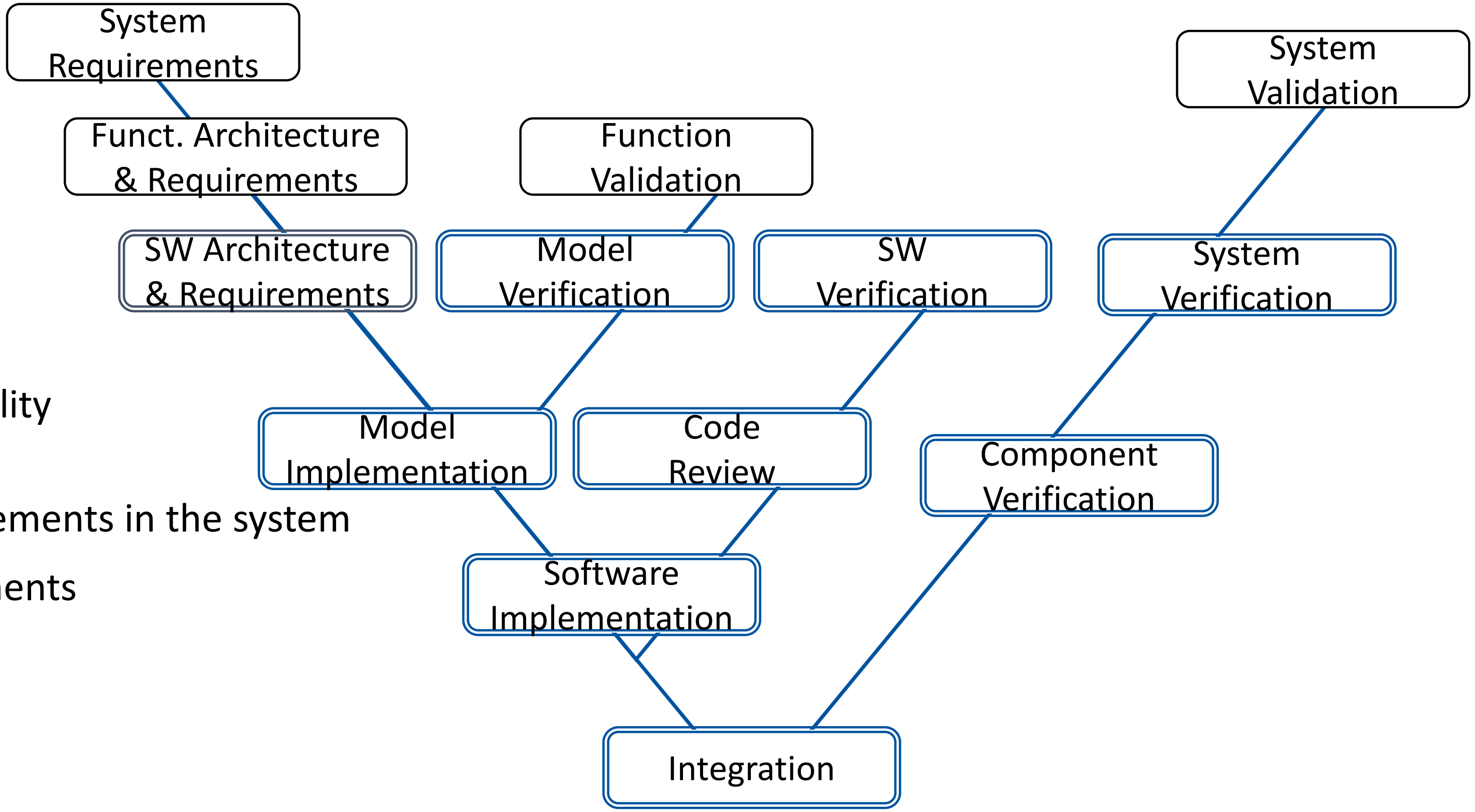


Co-funded by the Erasmus+ Programme of the European Union

Source: J. Richenhagen, Lecture “Software for Combustion Engines”, RWTH Aachen University, 2019 [22]

V-Model in model-driven software engineering

III) SYSTEM INTEGRATION



Objectives

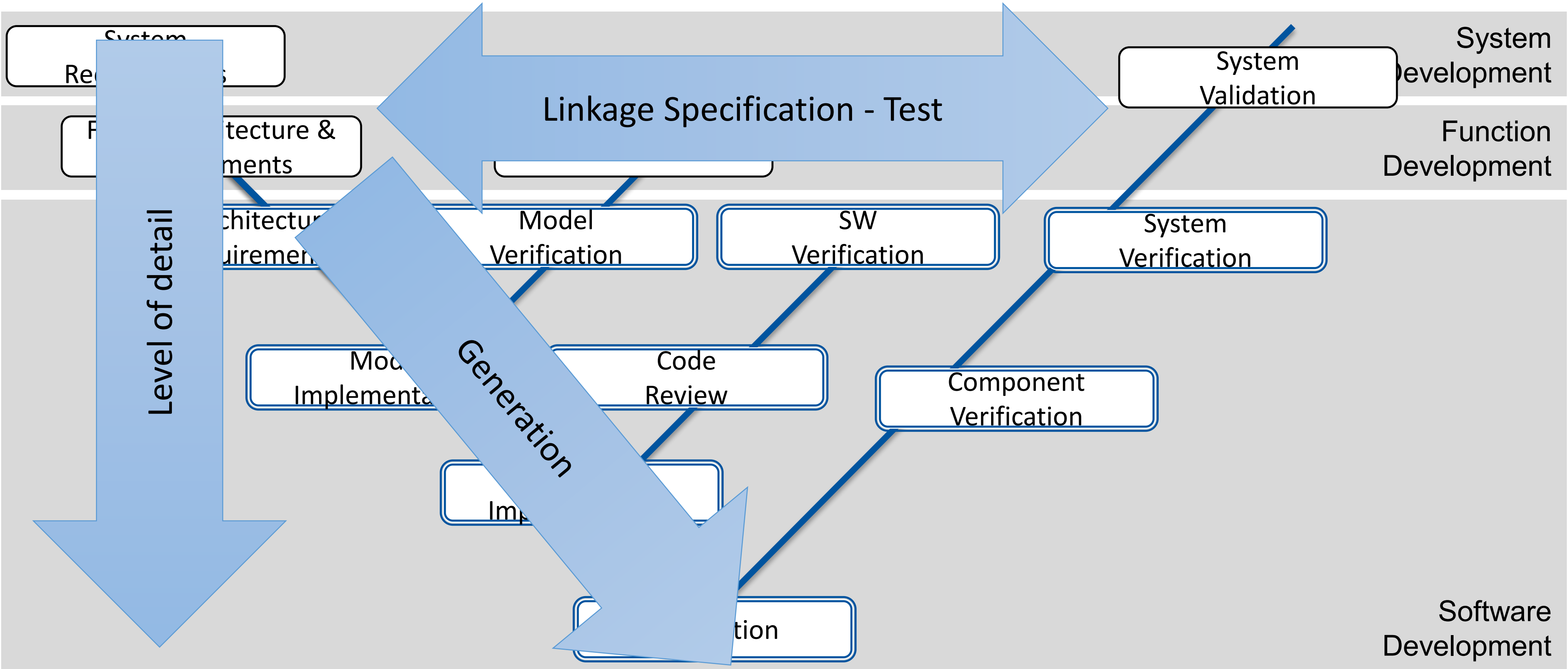
- Integration of functionality into the E/E system
- Realization of all requirements in the system
- Acceptance of requirements

Co-funded by the Erasmus+ Programme of the European Union

Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [22]

V-Model in model-driven software engineering

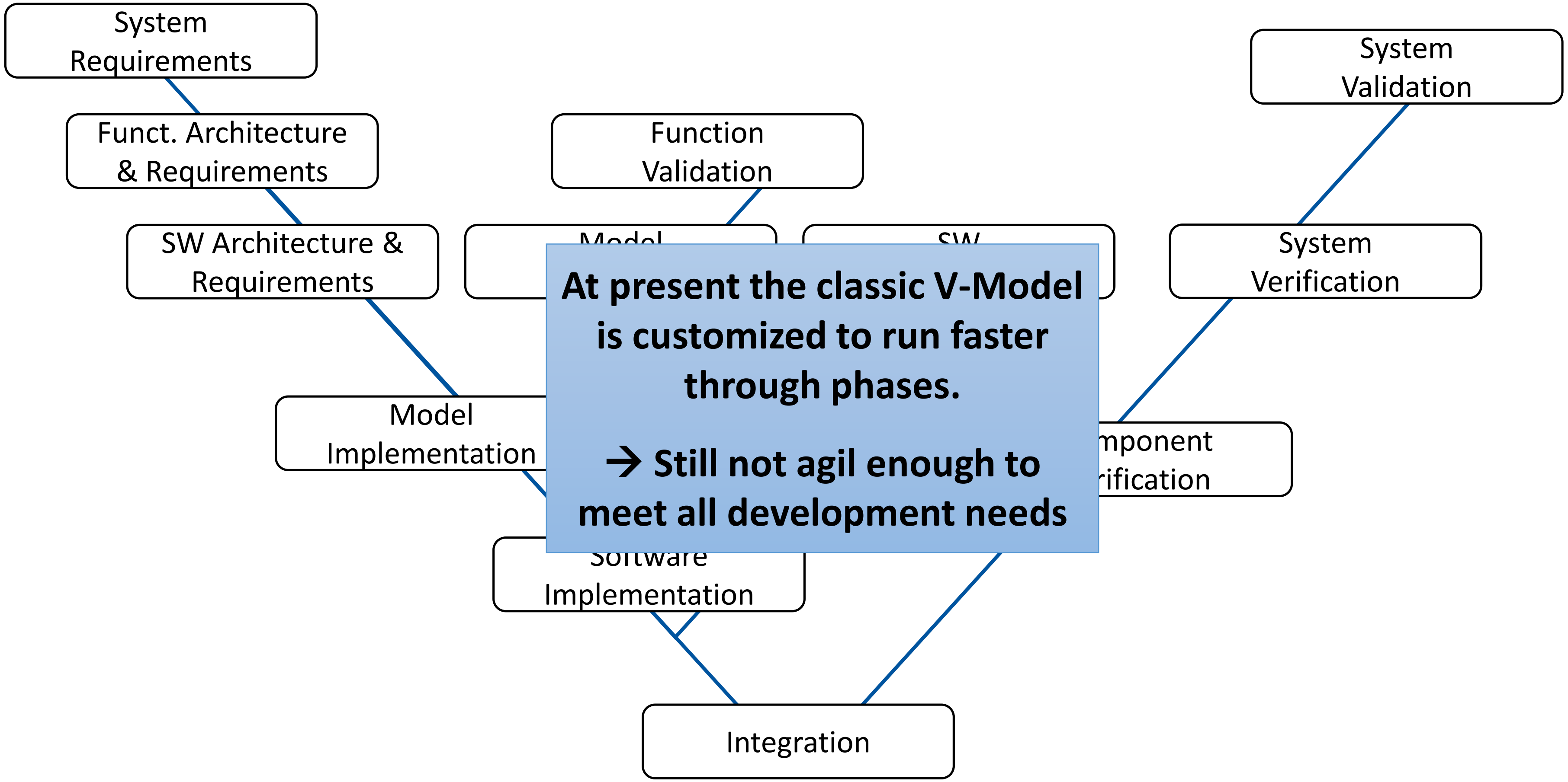
WHY "V" INSTEAD OF A "LINE"? – ABSTRACTION LEVELS AND CROSS-LINKAGES



Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [22]

Co-funded by the Erasmus+ Programme of the European Union

V-Model in model-driven software engineering



At present the classic V-Model is customized to run faster through phases.

→ Still not agil enough to meet all development needs

Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [22]

Co-funded by the Erasmus+ Programme of the European Union

Agenda

- Introduction and motivation
- Definition of technical terms
- Conventional development approaches
- Agile development approaches
- Challenges of agile engineering
- References

Co-funded by the Erasmus+ Programme of the European Union

Learning objectives

- Agile development
  - What does the agile manifesto say and what are the principles of agile development?
  - What is SCRUM, what describes this agile development approach and what is it used for?

Agility is an attitude towards (software) development

AGILE MANIFESTO (FOWLER, 2001)

Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

- First steps towards **agile software development** noted already in the early 1990s.
- First popular publication by Kent Beck et al. in 1999 on **Extreme Programming**.
- Increasing interest also in business environment led to further development and integration of **agile processes and methods** throughout the last 20 years.

Source: http://agilemanifesto.org/iso/en/manifesto.html [20]

Co-funded by the Erasmus+ Programme of the European Union

Which principles to follow in agile development?

THE AGILE MANIFESTO – 12 PRINCIPLES

- Our highest priority is to **satisfy the customer** through early and continuous delivery of valuable software.
- **Welcome changing requirements**, even late in development. Agile processes harness change for the customer's competitive advantage.
- **Deliver working software frequently**, from a couple of weeks to a couple of months, giving preference to the shorter timescale.
- Business people and developers must **work together** daily throughout the project.
- Build projects around **motivated individuals**. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is **face-to-face conversation**.
- **Working software** is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a **constant pace** indefinitely.
- Continuous attention to **technical excellence** and good design enhances agility.
- **Simplicity** – the art of maximizing the amount of work not done – is essential.
- The best architectures, requirements, and designs **emerge** from **self-organizing** teams.
- At **regular intervals, the team reflects** on how to become more effective, then tunes and adjusts its behavior accordingly.

Source: http://agilemanifesto.org/iso/en/manifesto.html [20]

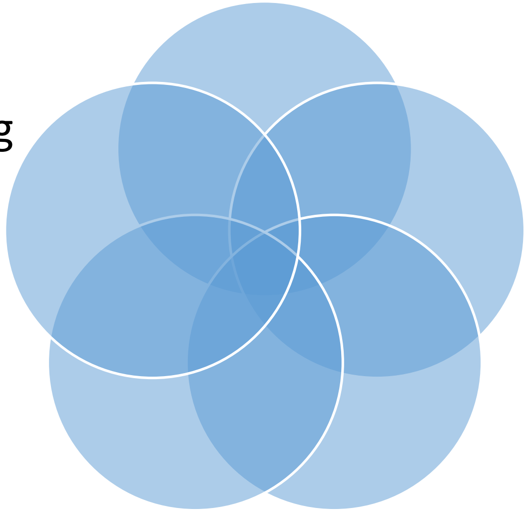
Co-funded by the Erasmus+ Programme of the European Union

Why agile practices work

FIVE REASONS FOR AGILE DEVELOPMENT

People closest to the problem are the ones to solve it

Teams measure progress by delivery of working product increments



People are motivated when they are self-organized

Teams and stakeholders learn via fast feedback

Teams have frequent opportunities to inspect and adapt

Co-funded by the Erasmus+ Programme of the European Union

What is the benefit of agile development?

UNITED

RESULTS OF A SUCCESSFUL IMPLEMENTATION OF AGILE DEVELOPMENT

Faster product development

Rework activities (e.g. due to changes) are reduced / avoided (elimination of process limits, closer interaction with users / customers → use cases)

Achievement of a higher product quality

According to customer needs
(closer interaction with users / customers, cooperation over different competencies)

A higher level of satisfaction is emerging

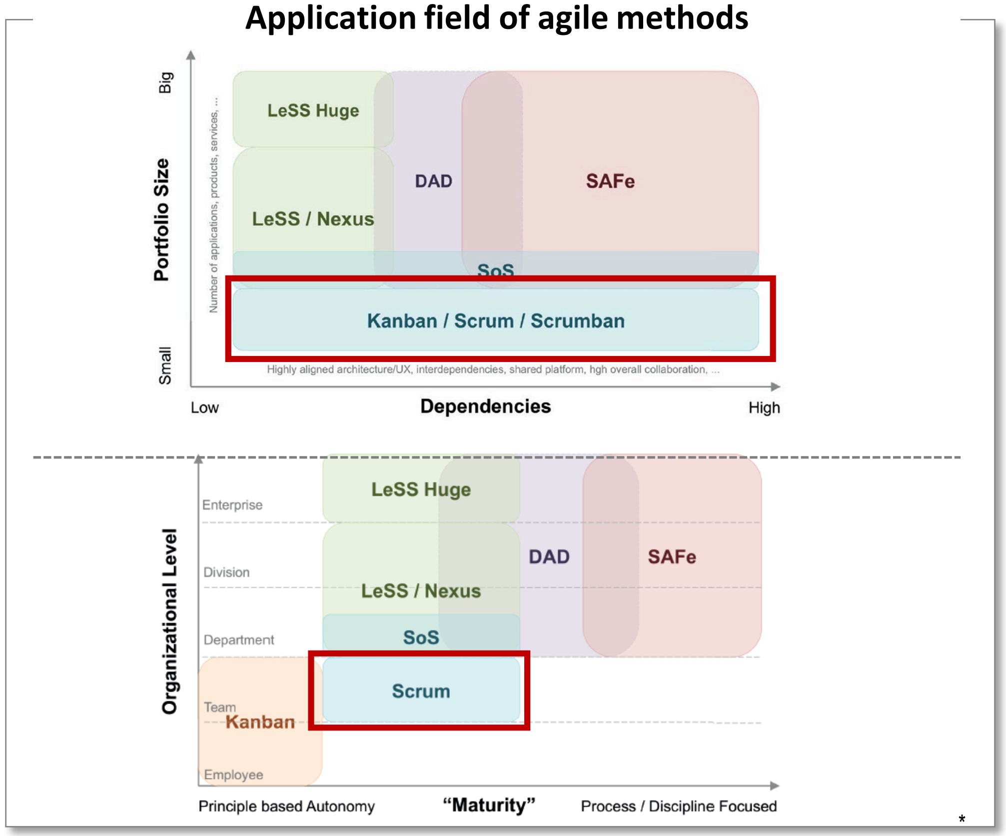
with developers
(due to close cooperation, freedom of discretion)

and with customers
(per above)

Co-funded by the Erasmus+ Programme of the European Union

SELECTION OF A FRAMEWORK DEPENDING ON DIFFERENT CRITERIA

SUITABILITY OF AGILE DEVELOPMENT APPROACHES



*Source: SwissQ Consulting AG | "Agile Modelle im Vergleich: [11]

- The term "Scrum" is derived from the scrum formation used by rugby teams.
- Manage complex work using an agile process framework .
- Break large projects into smaller fragments, review, adapt.

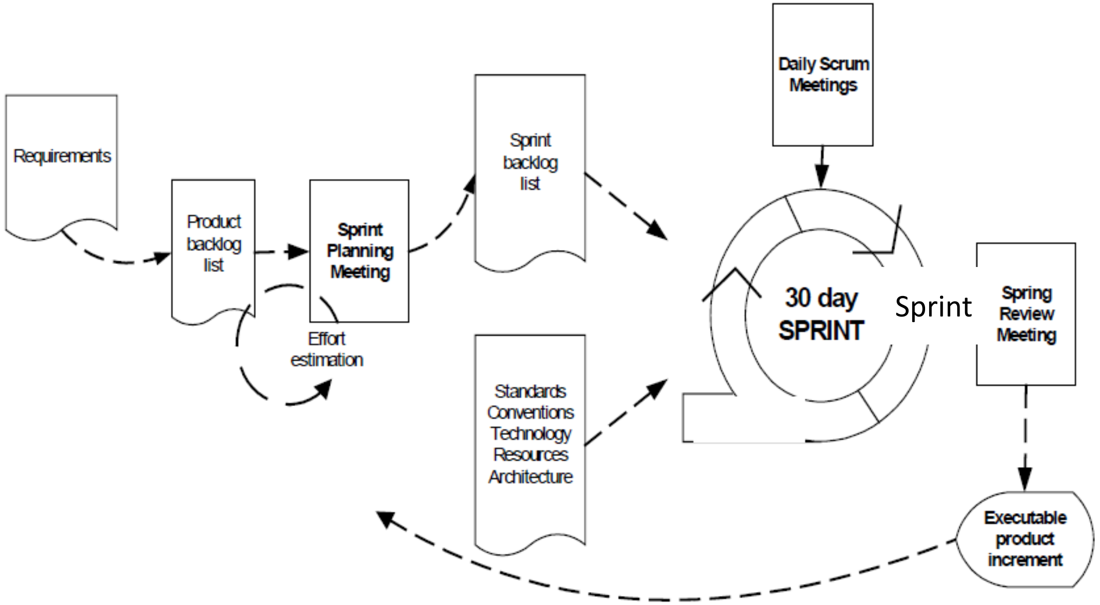
  Repeat.

**Source: https://www.scrumalliance.org/learn-about-scrum - Learn About Scrum - Use Scrum to continuously improve, rapidly respond to change, and deliver early. Scrum Alliance. [25]

Co-funded by the Erasmus+ Programme of the European Union

What is the concept of SCRUM?

ORGANIZATION OF SPRINTS



- Proposed as software development method by Schwaber and Sutherland in early 1990
- Testing and documentation as part of “development”, there is no following step
- Work units are done in “sprints” and are derived from a “backlog” of changing prioritized requirements that are already existing
- Short-term changes are not included into sprints, but rather into the backlog
- Five major Scrum events: Sprint Planning, Daily Scrum, The Sprint, Sprint Review, Sprint Retrospective
- Daily stand-up meetings are very short (15 minutes)
- Sprint planning, sprint retrospective are very intense and may be long (e.g. one day)
- Software releases are delivered to the customer within an assigned time slot. There may be not all functionality included, enabling the customer to assess and give feedback.

Source: P. Abarahamsson et al.: Agile software development methods - Review and analysis [13]

Co-funded by the Erasmus+ Programme of the European Union

Application of Scrum requires new responsibilities in development teams

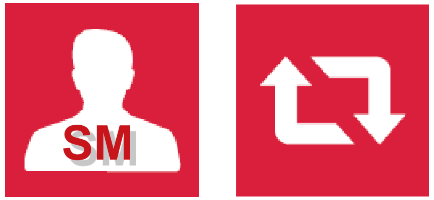
ROLES IN SCRUM TEAMS

Product Owner

- Represents the customer
- Responsible to maximize the ROI
- Manages the Product Backlog
- Is a person, not a committee

Development Team

- Delivers a potentially releasable increment of product at the end of each sprint
- Is self-organizing
- Is interdisciplinary
- Is empowered



Scrum Master

- Promotes and supports the Scrum Process
- Serves the PO and the Dev-Team
- Facilitates the Scrum Events
- Coaches the Dev-Team

Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

Different artefacts in SCRUM cycle

Product Backlog

- An ordered list of work-items
- Describes the upcoming work on the product
- Managed by the Product Owner
- Dev-Team estimates the work items

Sprint Backlog

- A set of work items selected for the Sprint
- Owned by the Dev-Team
- Is a forecast by the Dev-Team
- Scope is stable during the Sprint

Increment

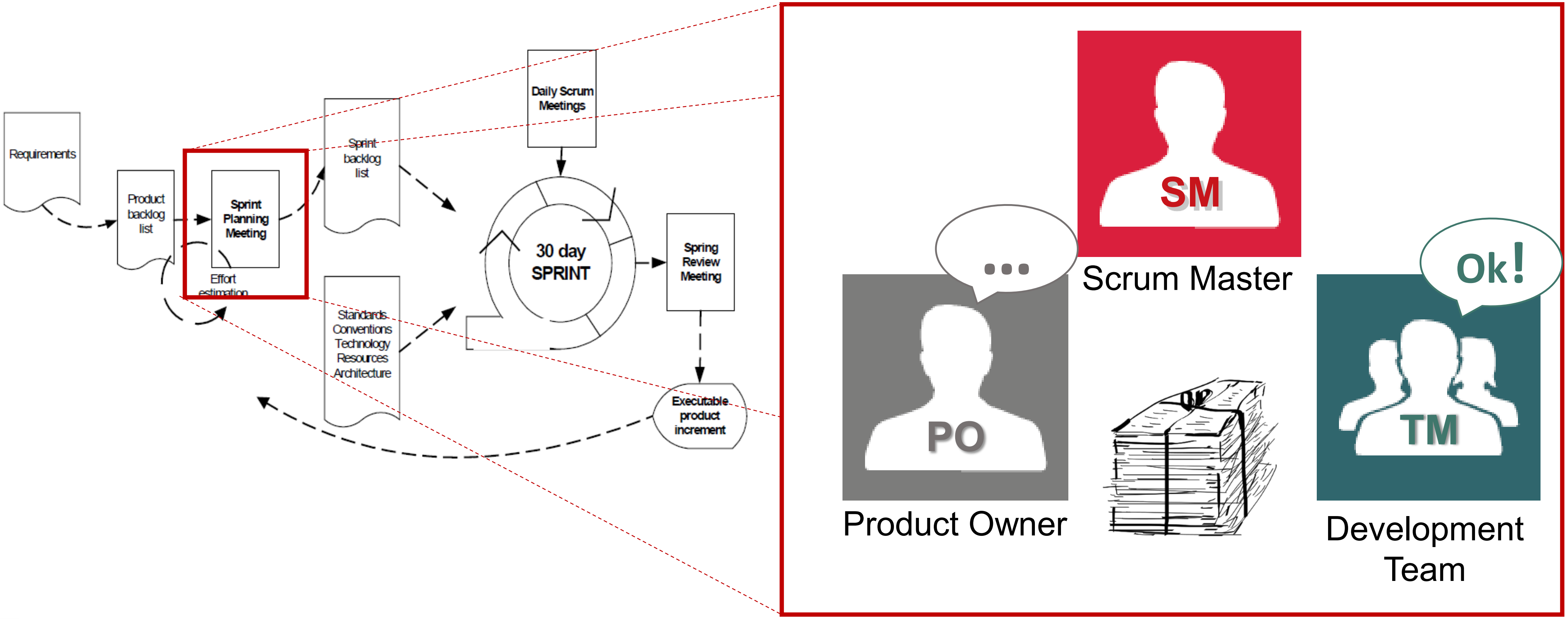
- Backlog items completed during a Sprint
- Meets the Scrum Team’s definition of “Done”
- Is developed by the Dev-Team and reviewed by the PO

Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

Scrum – Sprint Planning

ORGANIZATION OF SPRINTS

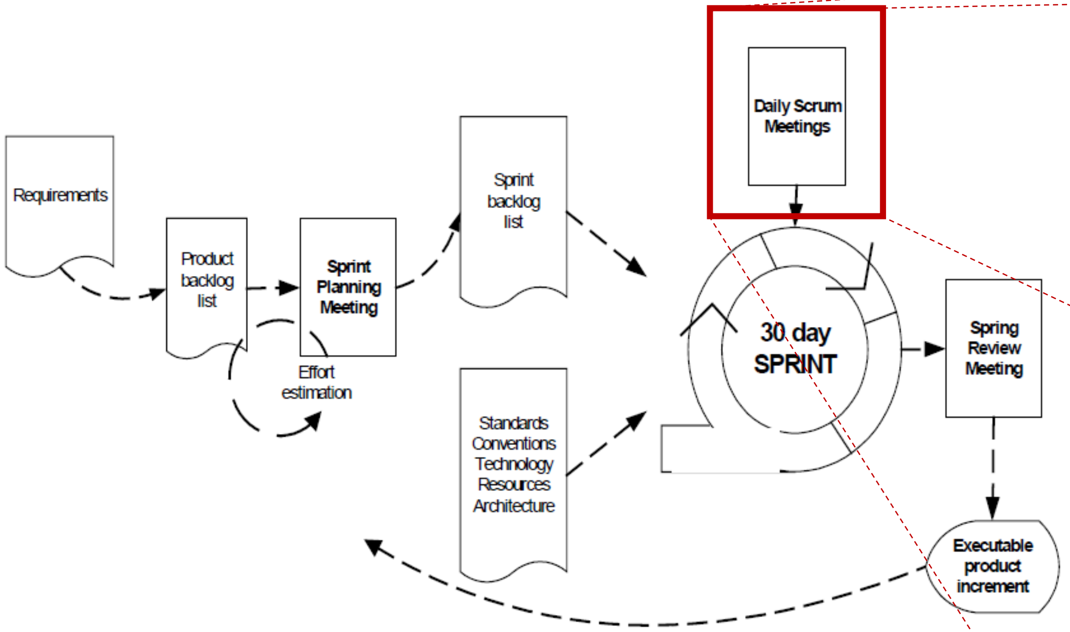


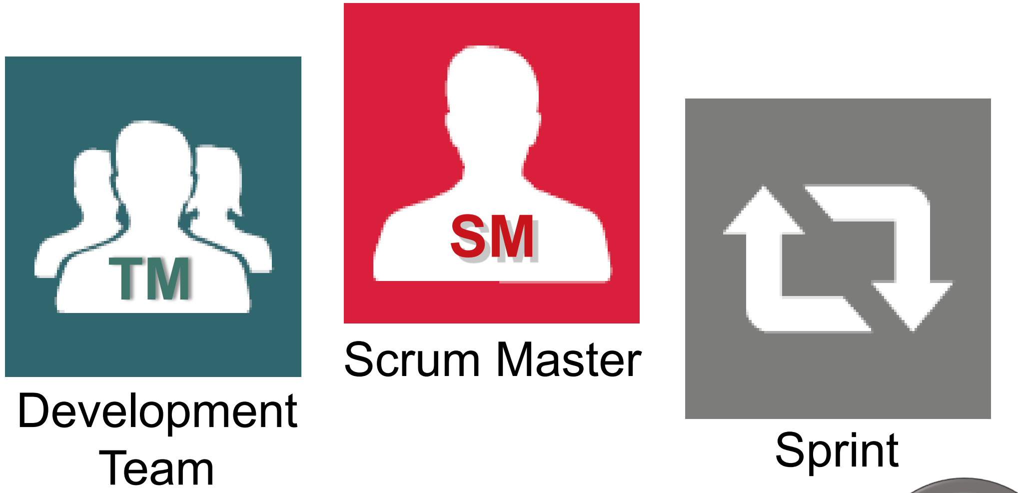
Source: P. Abrahamsson et al.: Agile Software Development Methods: Review and Analysis [13]

Co-funded by the Erasmus+ Programme of the European Union

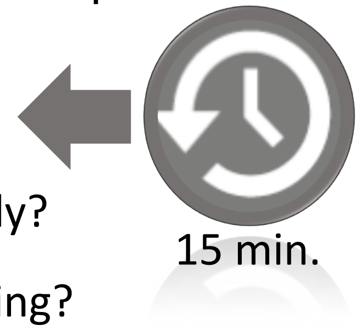
Scrum – Daily Scrum Meetings

ORGANIZATION OF SPRINTS





- ✓ What have I done since the last Daily?
- ✓ What will I do until the next Daily?
- ✓ What are the problems I am facing?

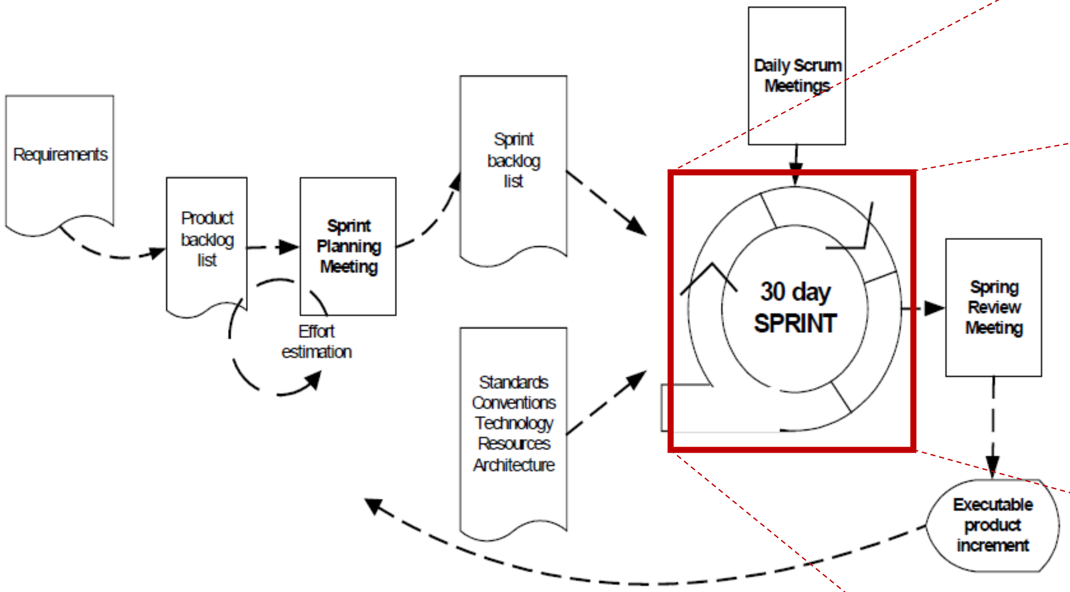


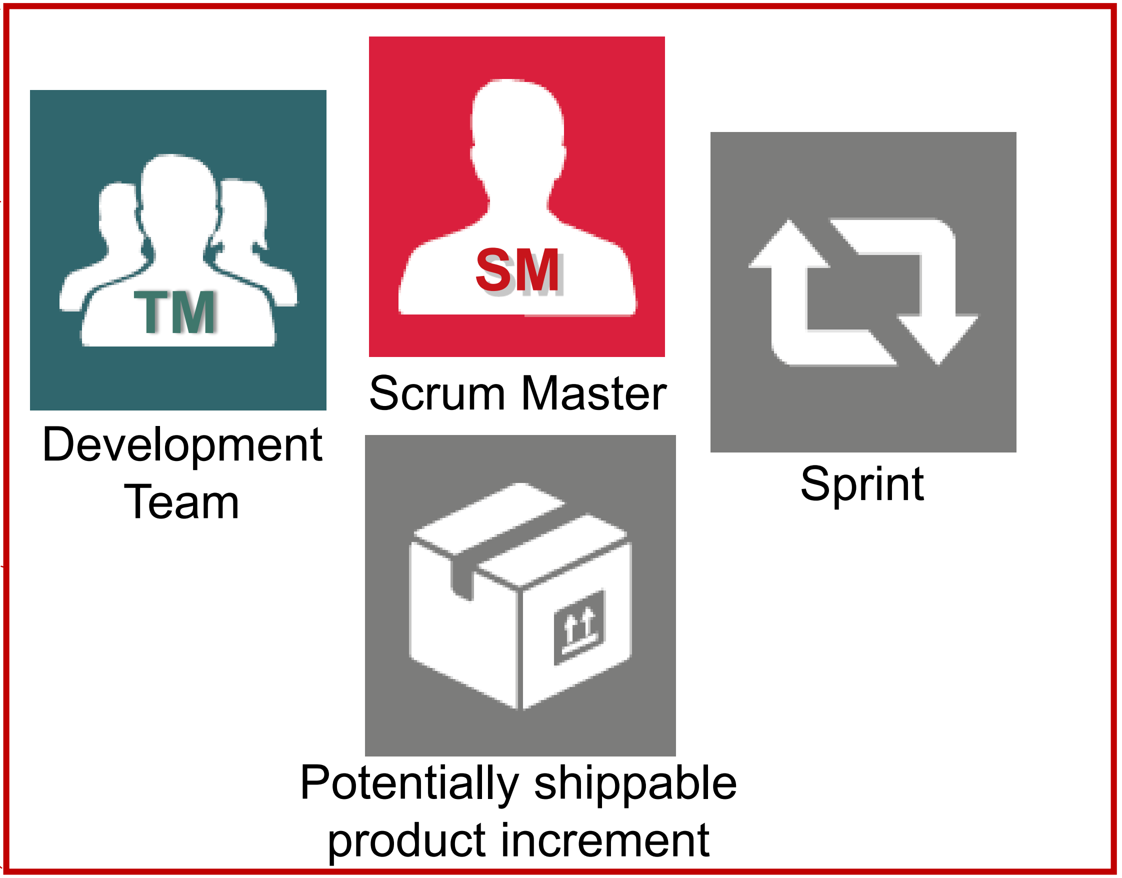
Source: P. Abrahamsson et al.: Agile Software Development Methods: Review and Analysis [13]

Co-funded by the Erasmus+ Programme of the European Union

Scrum – The Sprint

ORGANIZATION OF SPRINTS



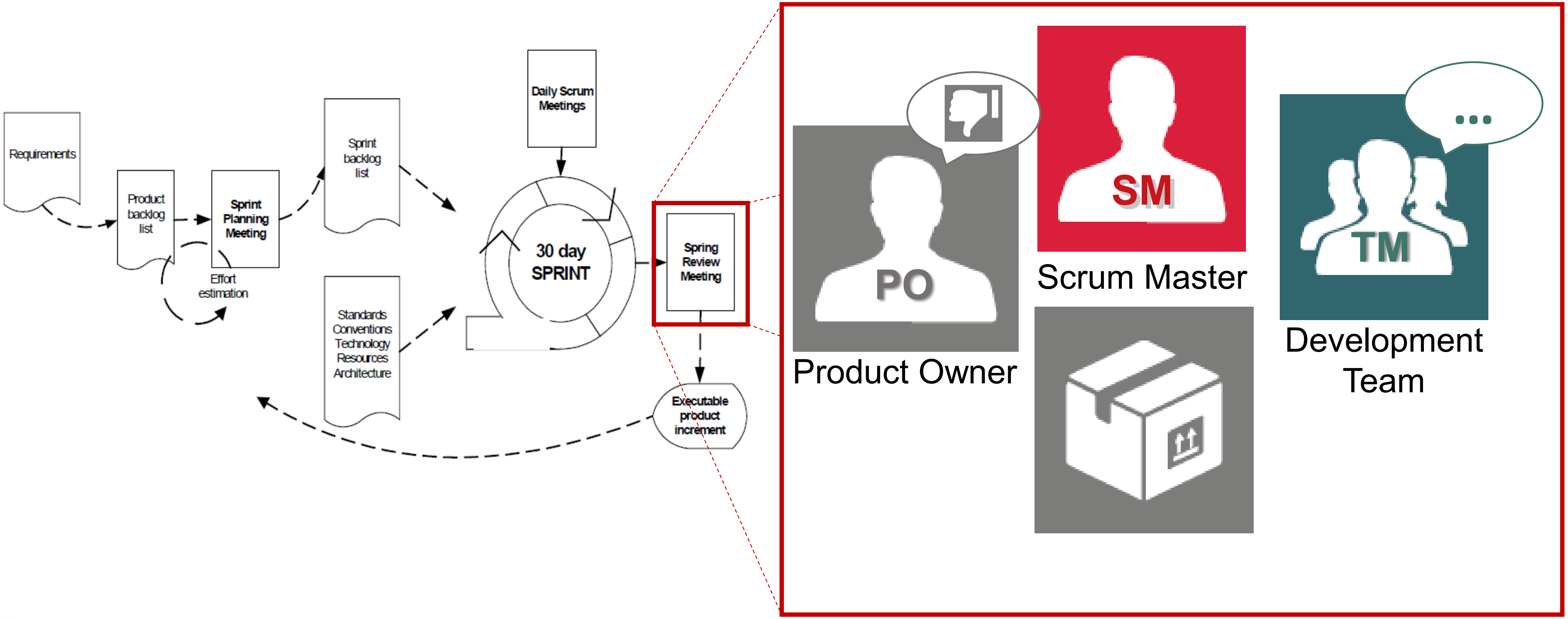


Source: P. Abrahamsson et al.: Agile Software Development Methods: Review and Analysis [13]

Co-funded by the Erasmus+ Programme of the European Union

Scrum – Sprint Review Meeting

ORGANIZATION OF SPRINTS

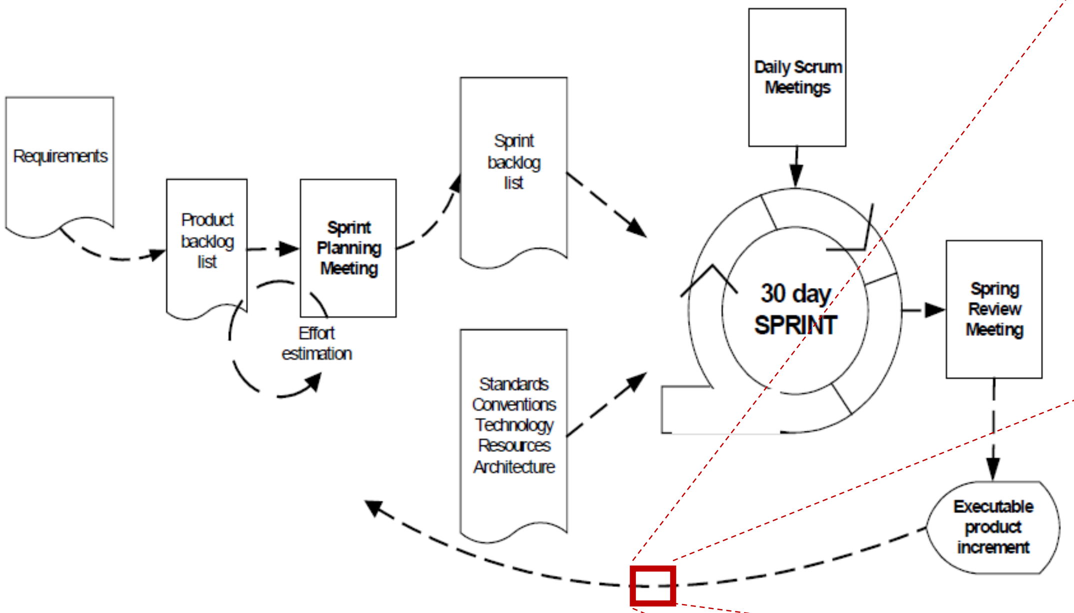


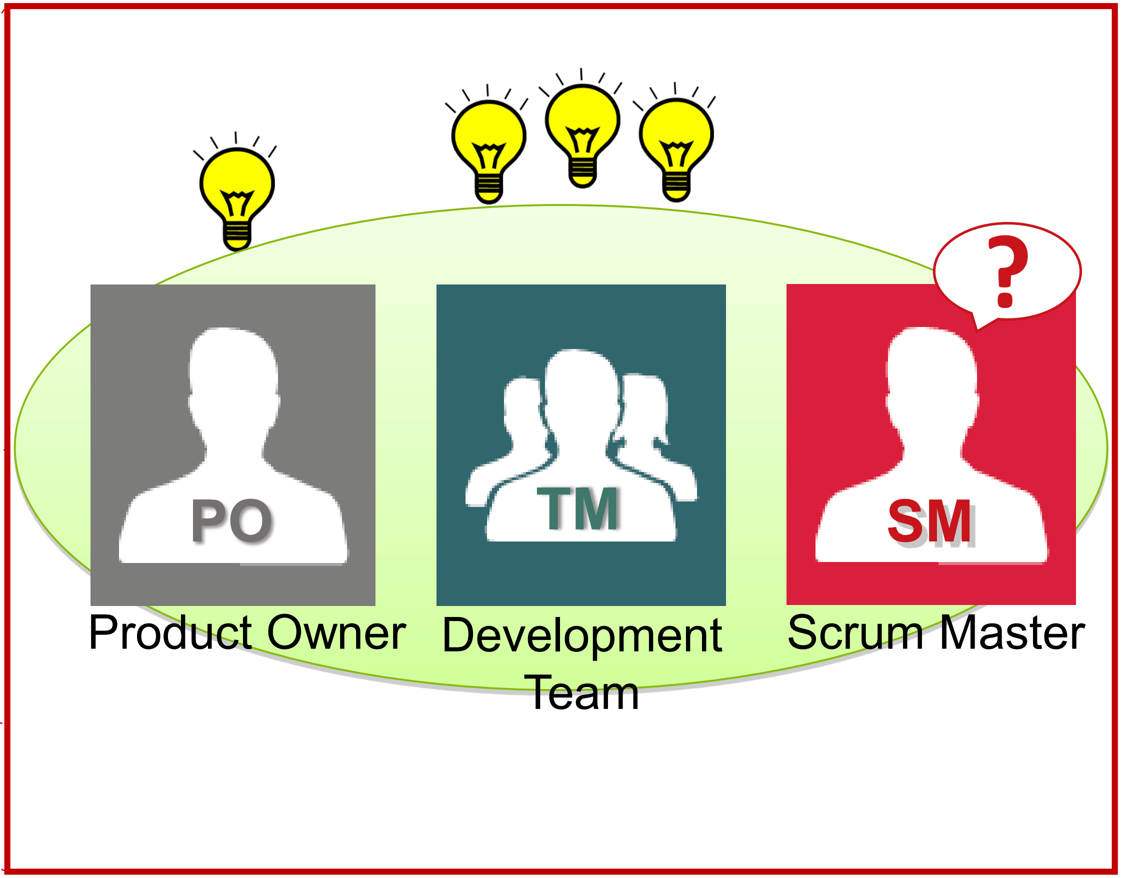
Source: P. Abrahamsson et al.: Agile Software Development Methods: Review and Analysis [13]

Co-funded by the Erasmus+ Programme of the European Union

Scrum – Retrospective

ORGANIZATION OF SPRINTS

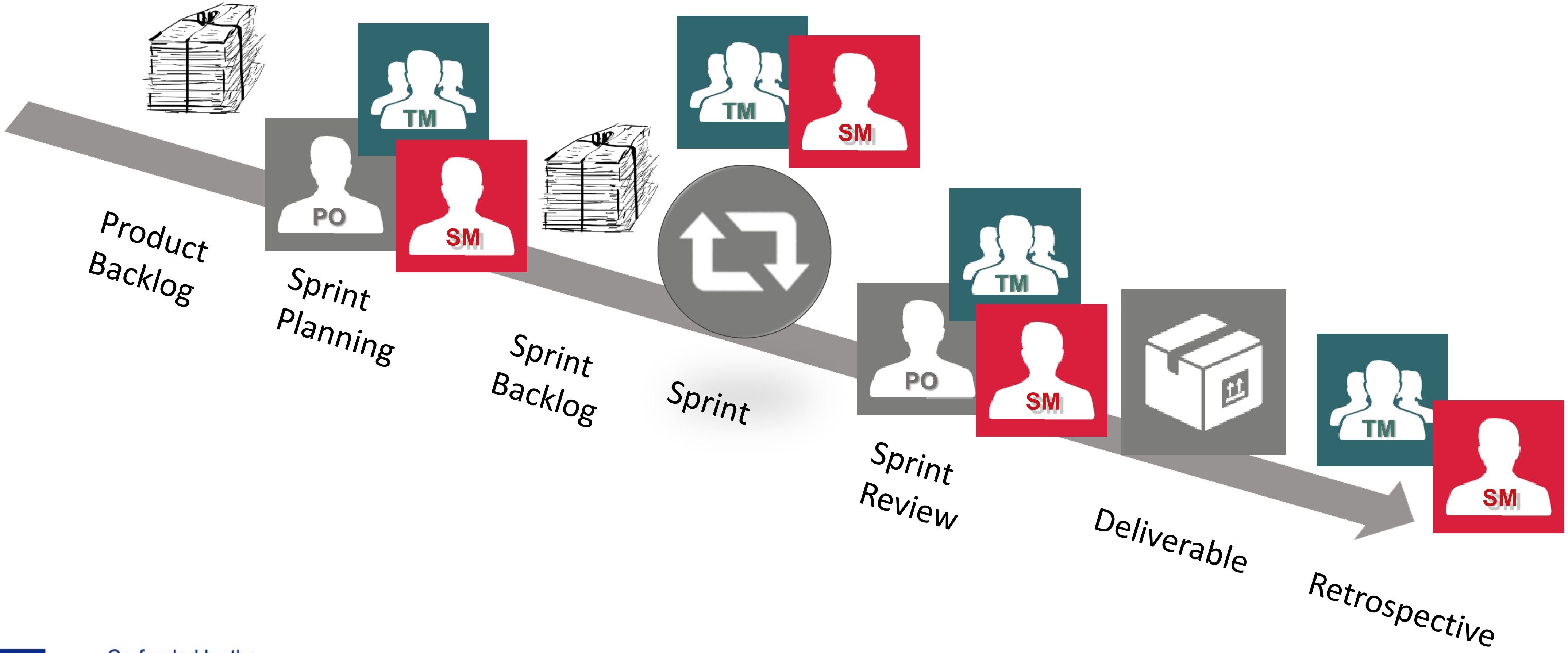




Source: P. Abrahamsson et al.: Agile Software Development Methods: Review and Analysis [13]

Co-funded by the Erasmus+ Programme of the European Union

Scrum process overview



Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

Agenda

- Introduction and motivation
- Definition of technical terms
- Conventional development approaches
- Agile development approaches
- Challenges of agile engineering
- References

Co-funded by the Erasmus+ Programme of the European Union

Learning objectives

- Challenges of agile development
  - What are, in short, the differences between agile and conventional development approaches?
  - What are the challenges of agile transformation for an enterprises?

Co-funded by the Erasmus+ Programme of the European Union

CHANGE IS ACCEPTED & ACTIVELY MANAGED, NEW LEADERSHIP REQUIRED

UNITED

COMPARISON OF CONVENTIONAL AND AGILE DEVELOPMENT

	Mindset "Established"	Mindset "Agile"
Planning horizon	▪ Long-term planning ▪ Detailed project planning for months, years ▪ Decisions as early as possible ▪ Change if necessary	▪ Efficient change management ▪ Detailed project planning for weeks, months ▪ Decisions as quick and late as possible ▪ Long-term planning if necessary
Division of labor	▪ Hierarchical organization ▪ separated KPI ▪ Project management, line management, … ▪ New cross-functions to gain synergies	▪ Cross-functional teams ▪ common KPI ▪ "T Shape Profile": every employee is specialist for one topic ("I") and supports other areas ("_")
Customer orientation	▪ Contract negotiation ▪ Contract is working base ▪ Acceptance against contract ▪ Change of scope if customer escalates	▪ Customer is part of the team ▪ Common understanding before starting to work ▪ Deep involvement for intermediate results ▪ Shared responsibility & trust
Leadership	▪ Planner, decision maker, controller ▪ Task definition & decision by hierarchy ▪ Individual targets and internal challenge ▪ Change where necessary	▪ Servant Leadership: communicator, convincer ▪ Decision where required → project team! ▪ Work on ecosystem & boundary conditions ▪ Change driver

Source: S. Kriebel, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [23]

Co-funded by the Erasmus+ Programme of the European Union

Are the conventional approaches now obsolete?

COMPARISON OF CONVENTIONAL AND AGILE DEVELOPMENT

Conventional processes are based on guidelines defined as best practice (low level of self-determination) → less preservation of experiences by agility?	Conventional processes are designed maximum risk-averse, esp. regarding safety → more risk by agility?	Until now the emphasis was put on structured approaches → less overview and short-term thinking with agility?	Conventional processes are designed for coordination between parties → Does agility cause chaos? How are "conventional" and agile organizations collaborating?
↓	↓	↓	↓
Agile working is not contradictory to process models → iterations and planning horizon are different	Agile working improves product quality by cross-functional / interdisciplinary coordination	Agile working is not possible without capturing interactions, otherwise the effects / impact cannot be assessed	Agile working with non-agile organizations needs a good differentiation of processes, agreements are different, collaboration mindset is crucial

Co-funded by the Erasmus+ Programme of the European Union

Source: S. Kriebel, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [23]

Change needs a common objective, time and patience

DIGITAL TRANSFORMATION: CHANGE

- **Based on John P. Kotter**, expert in the field of change management**, 70 percent of change projects are failing.** Cause of this low success rate mainly are **resistance** against change from **staff side** and a **fallback into old habits**.

John P. Kotter – The 8-Step Process

1. Increase urgency.
2. Build guiding team.
3. Develop vision and strategy.
4. Communicate vision.
5. Remove barriers.
6. Create short-term wins.
7. Don't let up.
8. Institute change.

Human is the biggest obstacle when creating change. →

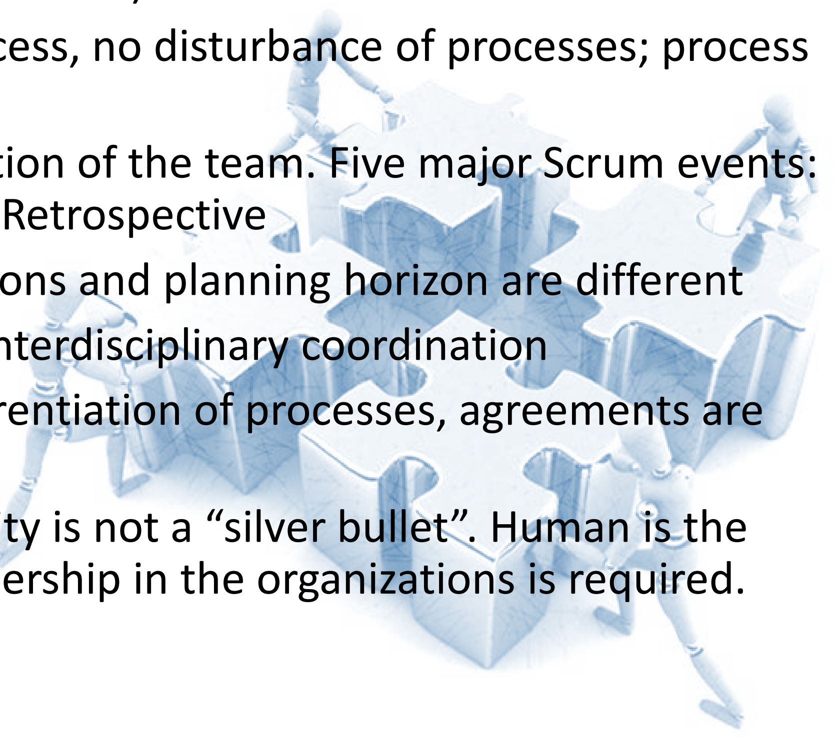
→ Knowledge is not understanding !!
→ Change must be wanted, it cannot be payed !!
→ Known things need to be challenged !!
→ New procedures must be developed and followed !!
→ Experiences, reflexes and automatisms need to be trained once again !!

Co-funded by the Erasmus+ Programme of the European Union

Source: S. Kriebel, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [23]
Source: John P. Kotter, "Accelerate: Building Strategic Agility for a Faster-Moving World", Harvard Business Review Press, 2014 [24]

Summary

- Agile development
  - Motivation: Fast response to changes, especially unclear requirements and solutions. (Even an adaptation of the conventional V-model is often no longer sufficient.)
  - Paradigm shift: Change of requirements is the core of the process, no disturbance of processes; process boundaries between crafts are eliminated
  - Scrum: Agile approach for small teams based on self-organization of the team. Five major Scrum events: Sprint Planning, Daily Scrum, The Sprint, Sprint Review, Sprint Retrospective
  - Agile working is not contradictory to process models → iterations and planning horizon are different
  - Agile working improves product quality by cross-functional / interdisciplinary coordination
  - Agile working with non-agile organizations needs a good differentiation of processes, agreements are different, collaboration mindset is crucial
  - Challenges of the transformation: Change of the mindset, agility is not a “silver bullet”. Human is the biggest obstacle when creating change. Additionally, new leadership in the organizations is required.



Co-funded by the Erasmus+ Programme of the European Union

Agenda

- Introduction and motivation
- Definition of technical terms
- Conventional development approaches
- Agile development approaches
- Challenges of agile engineering
- References

Co-funded by the
Erasmus+ Programme
of the European Union

References

[1] MC KINSEY & COMPANY
Monetizing car data
New service business opportunities to create new customer benefits
Advanced Industries, September 2016

[2] DIN 55350-11
Begriffe zum Qualitätsmanagement - Teil 11: Ergänzung zu DIN EN ISO 9000:2005
zitiert nach Balzert 2008, S. 460

[3] INTERNATIONAL STANDARDIZATION ORGANISATION (ISO)
Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models
Genf, 01.03.2011

[4] DUMKE & EBERT
Software Measurement
Berlin, 2007

[5] SCHÄUFFELE
Automotive Software Engineering
2014

[6] SCHÄUFFELE & ZURAWKA
Automotive Software Engineering: Grundlagen, Prozesse, Methoden und Werkzeuge effizient einsetzen
4th ed. Wiesbaden: Vieweg+Teubner Verlag / GWV Fachverlage
Wiesbaden, 2010.

Co-funded by the
Erasmus+ Programme
of the European Union

References

[7] SCHATTEN, DEMOLSKY, WINKLER et al.
Best Practice Software-Engineering
Spektrum Akademischer Verlag
Heidelberg, 2010

[8] SOMMERVILLE, I.
Software Engineering.
Addison-Wesley, 2007

[9] BOEHM, B.
A Spiral Model of Software Development and Enhancement.
IEEE Computer, 21(5):61–72, 1988

[10] BRÖHL, A. & DRÖSCHEL, W.
Das V-Modell: Der Standard für die Softwareentwicklung mit Praxisleitfaden.
Oldenburg, 1993

[11] SWISSQ CONSULTING AG
Agile Modelle im Vergleich: Wo passt welches?
Sacha Czudek, Head Agile, 2015
https://swissq.it/de/agile/unternehmensweite-agilitaet-ein-muss/

[12] COLLABNET
VersionOne 12th annual State of Agile Report, 2018
https://explore.versionone.com/state-of-agile/versionone-12th-annual-state-of-agile-report

Co-funded by the
Erasmus+ Programme
of the European Union

References

[13] ABRAHAMSSON at al.
Agile Software Development Methods: Review and Analysis
Espoo, 2002

[14] GRANRATH, C.
HIFI-ELEMENTS – Modellbasiertes Systems Engineering
Turin, 2017

[15] BECKER, H.
Auf Crashkurs
Springer Verlag, 2007; VDA Band 32,
Fast 2015 – Die neue Arbeitsteilung in der Automobilindustrie

[16] FEV
Prognose 05/2019

[17] DIN Deutsches Institut für Normung e.V.
EN ISO 8402
Beuth Verlag GmbH, 10772 Berlin
1995

[18] ISO International Standardization Organization
ISO 25010
2014-03

[19] https://www.scrum-tips.com/

[20] http://agilemanifesto.org/iso/en/manifesto.html

Co-funded by the
Erasmus+ Programme
of the European Union

References

[21] CARNEGIE MELLON UNIVERSITY & WIBAS IT MATURITY SERVICES GMBH
Capability maturity model integration (CMMI) für Entwicklung
Version 1.3
2006

[22] J. Richenhagen
Lecture: Software for Combustion Engines
RWTH Aachen University
2019

[23] S. Kriebel
Lecture: Software for Combustion Engines
RWTH Aachen University
2019

[24] John P. Kotter
Accelerate: Building Strategic Agility for a Faster-Moving World
Harvard Business Review Press
2014

[25] Scrum Alliance
Learn About Scrum - Use Scrum to continuously improve, rapidly respond to change, and deliver early
https://www.scrumalliance.org/learn-about-scrum (retrieved September 5, 2019)

Co-funded by the Erasmus+ Programme of the European Union

References

[26] International Organization for Standardization
"Systems and software engineering — Vocabulary," ISO/IEC/IEEE 24765:2017(E), ISO copyright office, Rev. Sep. 2017
"Systems and software engineering — Requirements for designers and developers of user documentation," ISO/IEC 26514:2008(E), ISO copyright office, Rev. Jun. 2008.
"Quality management systems — Fundamentals and vocabulary," ISO 9000:2015(en), ISO copyright office, Rev. Nov. 2015.
"Systems and software engineering — Software life cycle processes," ISO/IEC/IEEE 12207:2017(E), ISO copyright office, Rev. Nov. 2017.
"Information technology — Systems and software engineering — Guide for requirements engineering tool capabilities," ISO/IEC TR 24766:2009(E), ISO copyright office, Rev. Dec. 2019.
"Systems and software engineering — Information technology project performance benchmarking framework — Part 1: Concepts and definitions," ISO/IEC 29155-1:2017(en), 2nd ed., ISO copyright office, Rev. Dec. 2017.
"Systems and software engineering — Life cycle process-es — Requirements engineering," ISO/IEC/IEEE 29148:2018(E), ISO copyright office, Rev. Nov. 2018.
"Information technology — Modeling Languages — Part 2: Syntax and Semantics for IDEF1X97 (IDEFobject)," ISO/IEC/IEEE 31320-2:2012(en), 1st ed., ISO copyright office, Rev. Sep. 2012.
"Systems and software engineering — Content of life-cycle information items (documentation)," ISO/IEC/IEEE 15289:2017(E), ISO copyright office, Rev. May. 2017.

Institute of Electrical and Electronics Engineers
"IEEE Standard for Software Quality Assurance Processes," 730-2014, IEEE, Rev. Jun. 2014.
"IEEE Standard for Developing Software Life Cycle Processes," 1074-1991, IEEE, Rev. Jan. 1992.
"IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X/Sub 97/ (IDEF/Sub Object/)," 1320.2-1998, IEEE, Rev. 1998.

Co-funded by the Erasmus+ Programme of the European Union

Engineering Knowledge Transfer Units to Increase Student´s Employability and Regional Development

https://www.facebook.com/unitederasmus/

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
598710-EPP-1-2018-1-AT-EPPKA2-CBHE-JP

UNITED

Engineering Knowledge Transfer Units to Increase Student´s Employability and Regional Development

Model-based Systems Engineering

Day 2 – Slot 2

Christian Granrath, M.Sc.

FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
598710-EPP-1-2018-1-AT-EPPKA2-CBHE-JP

Agenda

- Motivation of systems engineering
- Systems engineering methodology
- Requirements specification guidelines
- Introduction to SysML
- Exemplary application of CUBE methodology
- Challenges and benefits
- References

Co-funded by the
Erasmus+ Programme
of the European Union

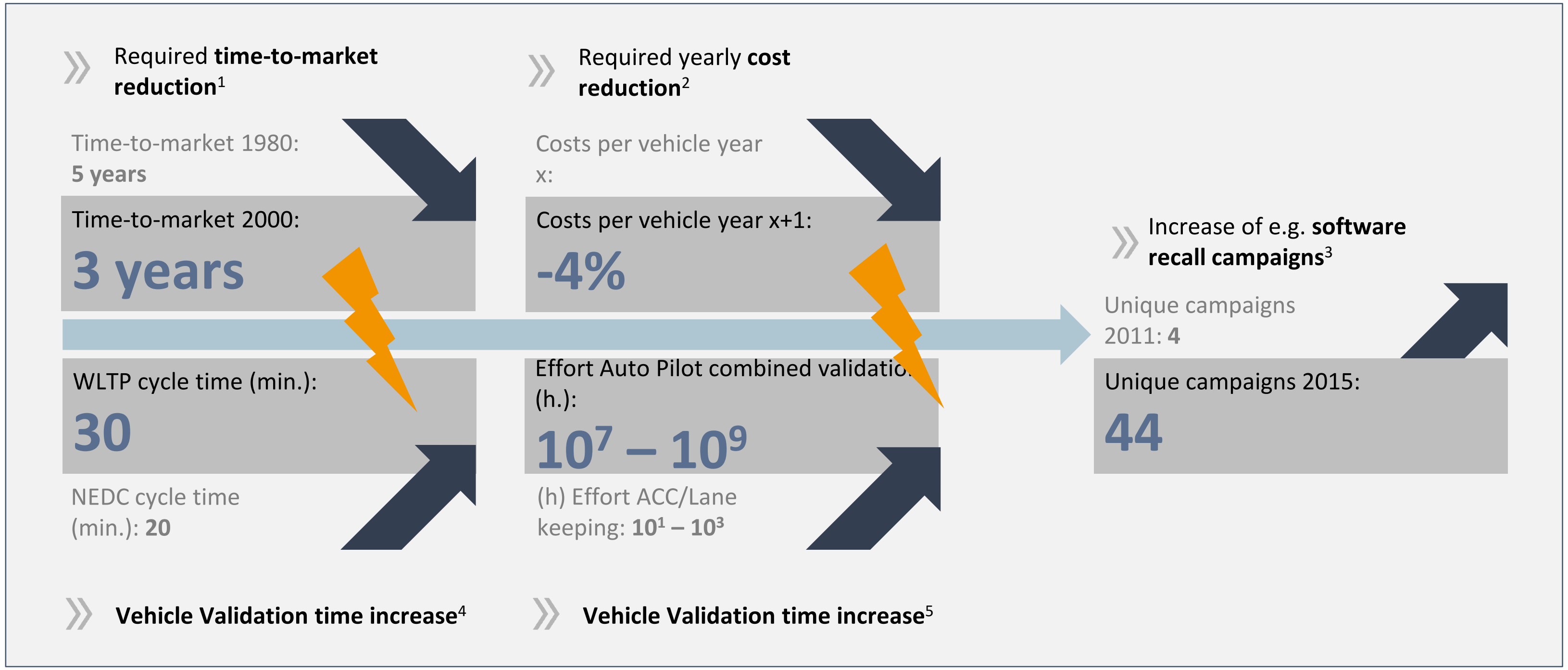
Learning objectives

- Motivation of Systems Engineering
  - What are the current gaps in automotive development approaches?
  - What is the focus of systems engineering?

Co-funded by the Erasmus+ Programme of the European Union

Required quality cannot be achieved by more vehicle tests

- CHALLENGES IN AUTOMOTIVE DEVELOPMENT



WLTP: Worldwide Harmonised Light-Duty Vehicles Test Procedure

NEDC: New European Driving Cycle

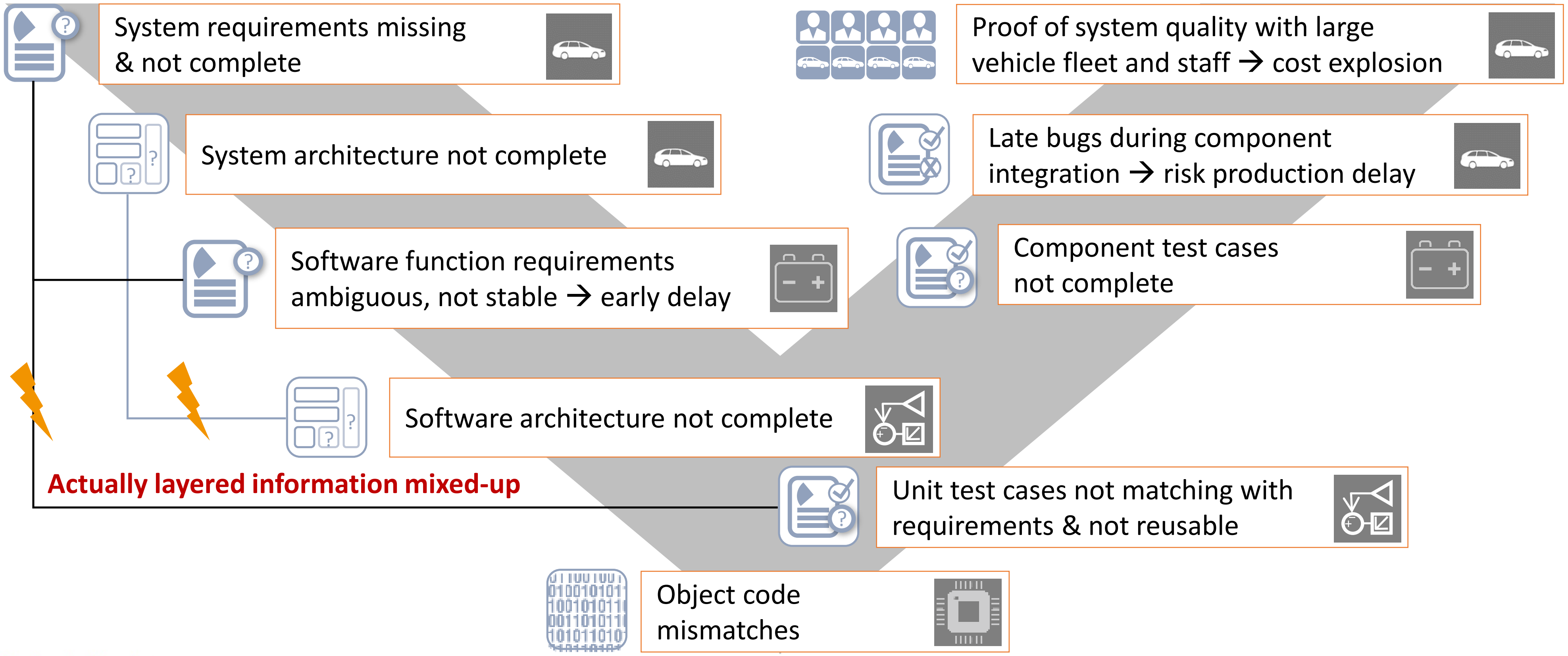
ACC: Adaptive Cruise Control

Co-funded by the Erasmus+ Programme of the European Union

Sources: [1] - [5]; Kriebel, Richenhagen et al.: "High Quality Electric Powertrains by model-based systems engineering", Aachen Colloquium 2017 [6]

Gaps in a company's engineering approach may cause serious quality issues

THE QUALITY DILEMMA OF PRODUCT DEVELOPMENT



Co-funded by the Erasmus+ Programme of the European Union

Source: Kriebel, Richenhagen et al.: "High Quality Electric Powertrains by model-based systems engineering", Aachen Colloquium 2017 [6]

Agenda

- Motivation of systems engineering
- Systems engineering methodology
- Requirements specification guidelines
- Introduction to SysML
- Exemplary application of CUBE methodology
- Challenges and benefits
- References

Co-funded by the
Erasmus+ Programme
of the European Union

Learning objectives

- Systems engineering methodology
  - What is systems engineering and what is addressed by it?
  - Which Systems engineering approaches do exist and what do these address?
  - How to use systems engineering for system specification?

Co-funded by the Erasmus+ Programme of the European Union

Different approaches for development of complex systems influence SE definitions

DEFINITION OF SYSTEMS ENGINEERING

“An **interdisciplinary** approach and means to enable the realization of **successful systems**.”
- INCOSE Handbook, 2004 [8]

"The Art and Science of creating effective systems, using **whole system, whole life principles**“
- Derek Hitchins [9]

System engineering is a **robust approach** to

- **design, creation, and operation of systems**
- **identification and quantification of system goals**
- creation of **alternative system design concepts**
- **implementation of the best design**
- **integration and verification** of the design
- **post-implementation assessment** of how well the system meets the goals

- NASA Systems Engineering Handbook, 1995 [10]

“Systems engineering is a **multidisciplinary** approach that is intended to transform a set of **stakeholder needs** into a **balanced system solution** that meets those needs. […] The systems engineering process includes activities to **establish top-level goals** that a system must support, **specify system requirements**, **synthesize alternative system designs**, **evaluate the alternatives**, **allocate requirements** to the components, **integrate the components** into the system, and **verify** that the system **requirements are satisfied**. It also includes essential **planning and control processes** needed to **manage a technical effort**.”
- A Practical Guide to SysML, S. Friedenthal, A. Moore, R. Steiner, 2015 [11]

“Systems engineering is an **interdisciplinary** approach to building **complex and technologically diverse systems**.”
- Agile Systems Engineering, Bruce Powel Douglass Ph.D., 2016 [12]

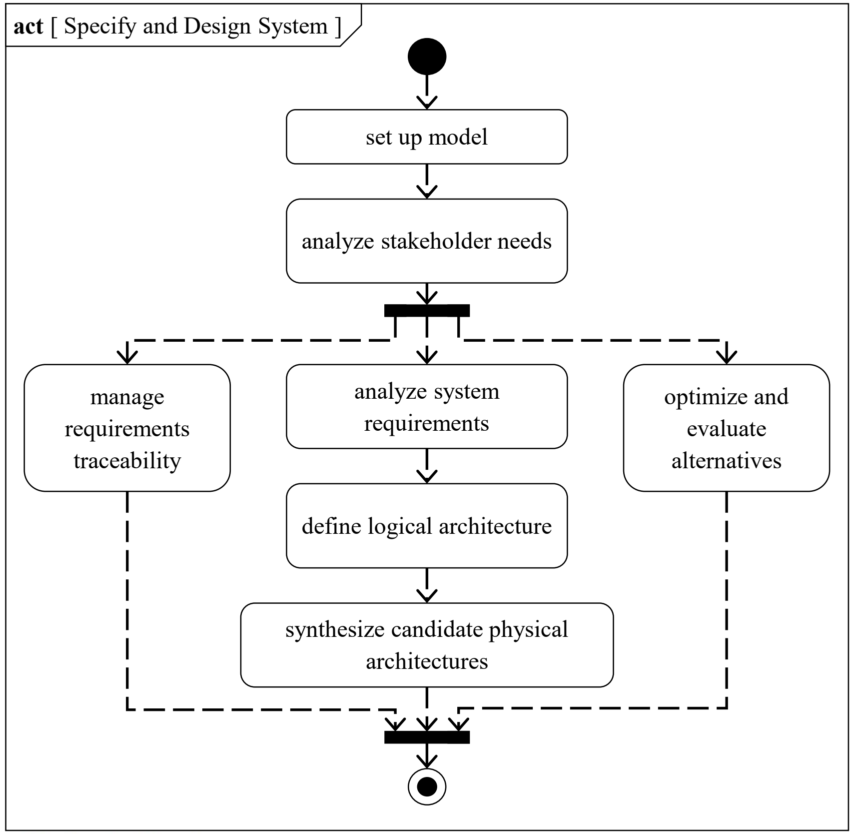
“Systems Engineering is an interdisciplinary and holistic engineering approach for the conception, realization and evaluation of complex technical systems as well as for engineering management over the complete life cycle of the systems.“

Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

Fundamental steps for model-based development

OBJECT-ORIENTED SYSTEMS ENGINEERING METHOD (OOSEM)



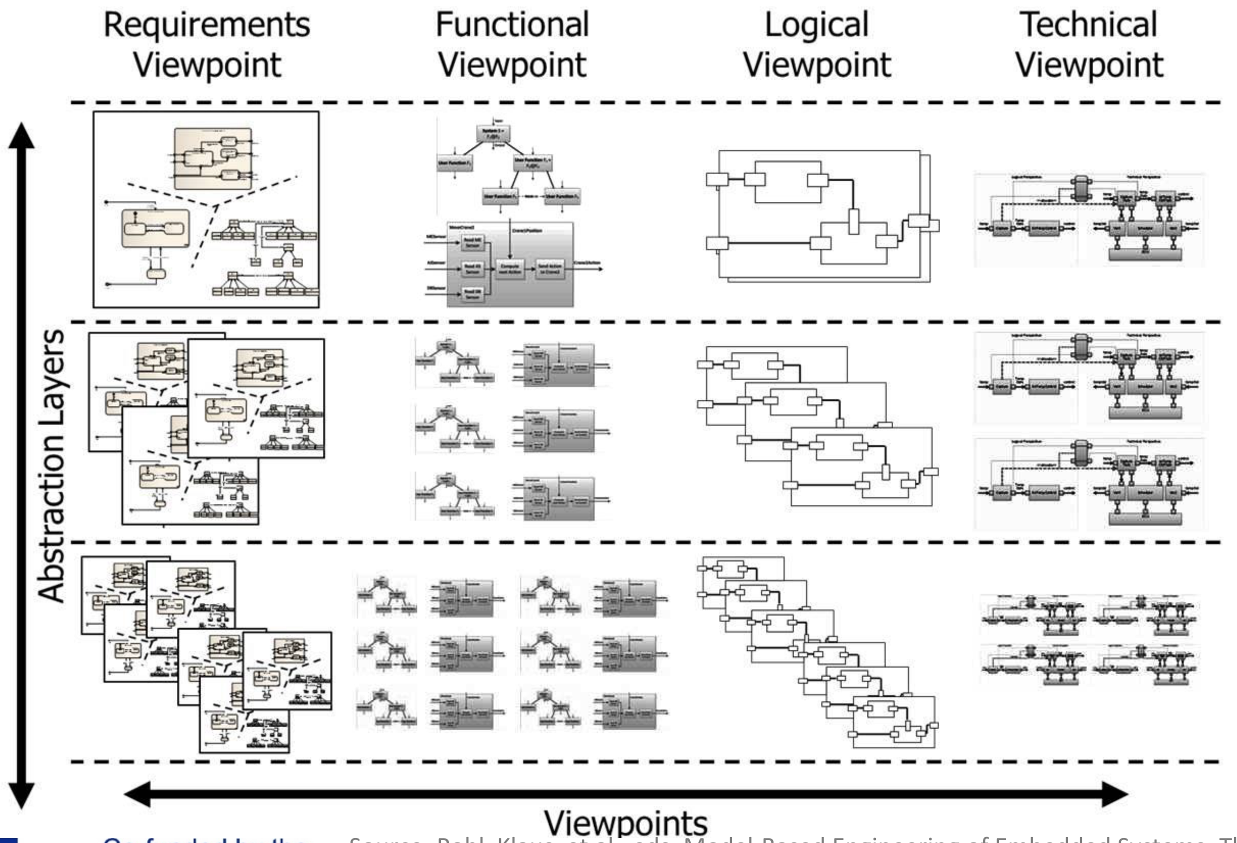
- System and environment analysis according to the customer and stakeholder needs
- Identification of system requirements, regarding the pre-defined needs
- Decomposition of the system in logical components
- Allocation of physical components and definition of their relationship

Source: Weilkiens, Tim. Systems engineering with SysML/UML: [18].

Co-funded by the Erasmus+ Programme of the European Union

Usage of identical view points on different abstraction layers

SPES 2020 METHODOLOGY



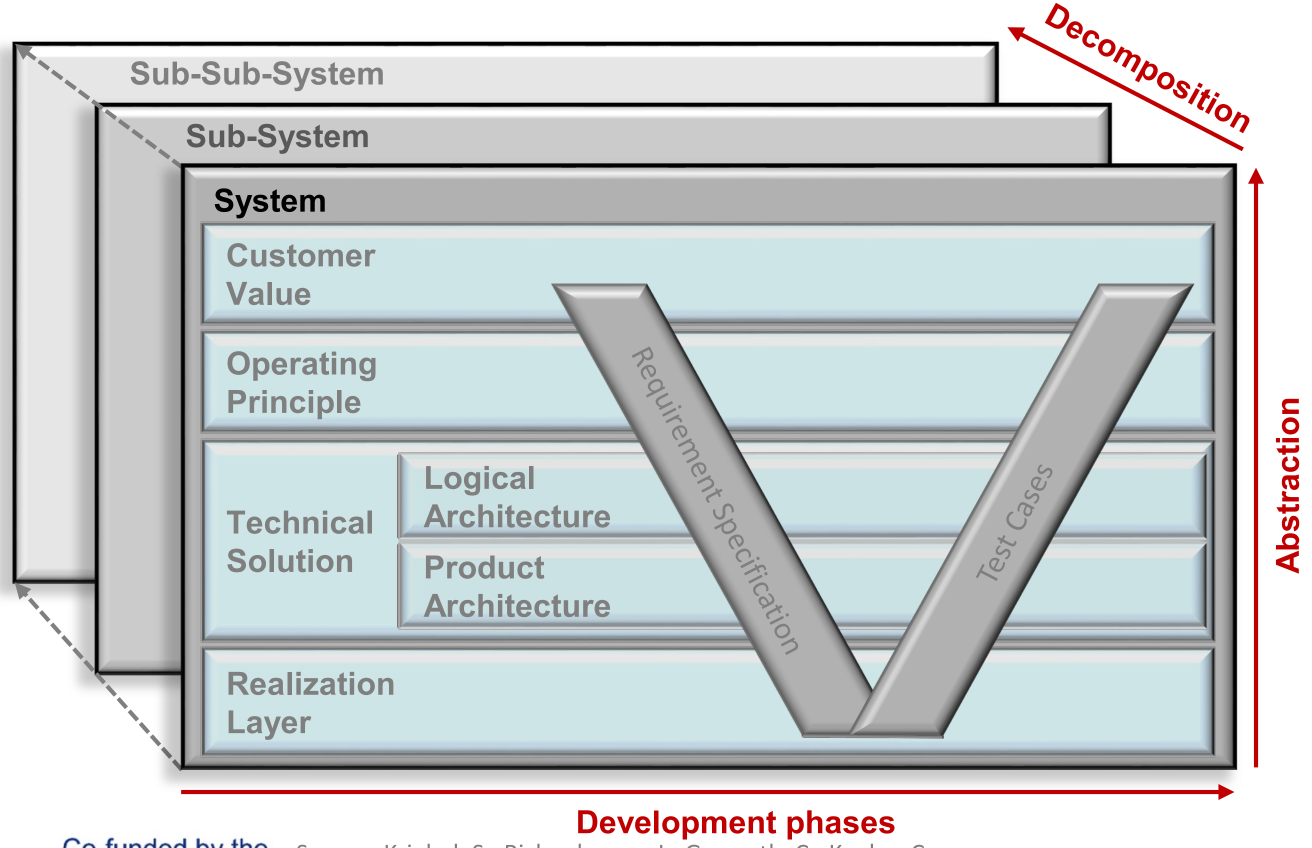
- Seamlessly integrated model-based, cross-domain engineering approach for embedded systems and their development process:
  - Analysis of system's context and customer requirements
  - Definition of system requirements
  - Specification of system, architecture design and implementation
  - Verification and certification of the system
- Iterative methodology for every abstraction layer

Source: Pohl, Klaus, et al., eds. Model-Based Engineering of Embedded Systems: The SPES 2020 Methodology. [23]

Co-funded by the Erasmus+ Programme of the European Union

CUBE model to ensure model- and feature-based systems engineering

SYSTEMS ENGINEERING C.U.B.E. - COMPOSITIONAL UNIFIED SYSTEM-BASED ENGINEERING



- System approach for the entire system development life cycle
- Decomposition of complex systems in manageable subsystems
- Model-based architecture specification in conjunction with textual requirements
- Reduced system integration risk
- Reduced future development efforts and costs

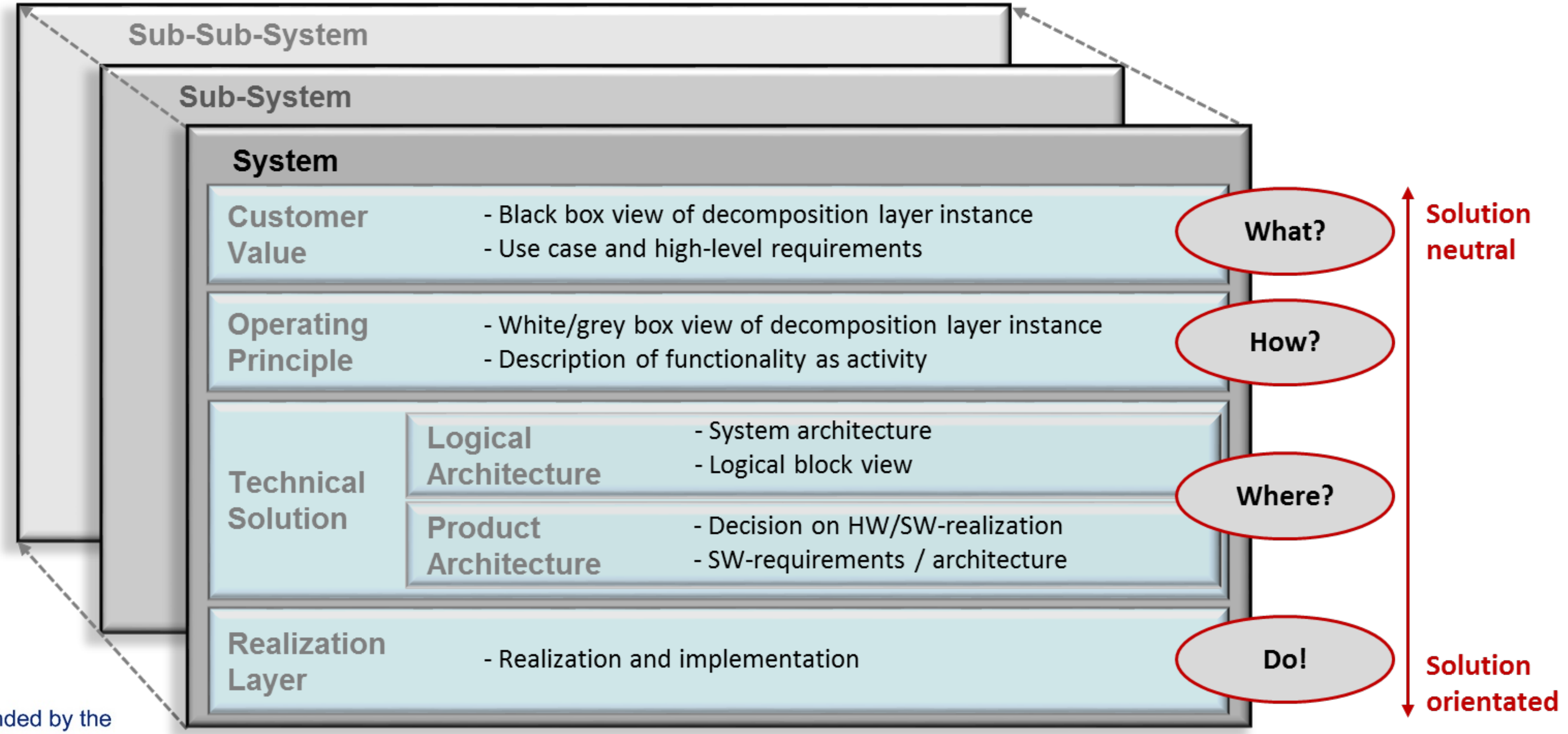
Co-funded by the Erasmus+ Programme of the European Union

Source: Kriebel, S., Richenhagen, J.; Granrath, C., Kugler, C.: "Systems Engineering with SysML The Path to the Future?" [13]

Abstraction and decomposition layers result in an appropriate holistic system description

UNITED

LAYERED STRUCTURE OF SYSTEMS ENGINEERING

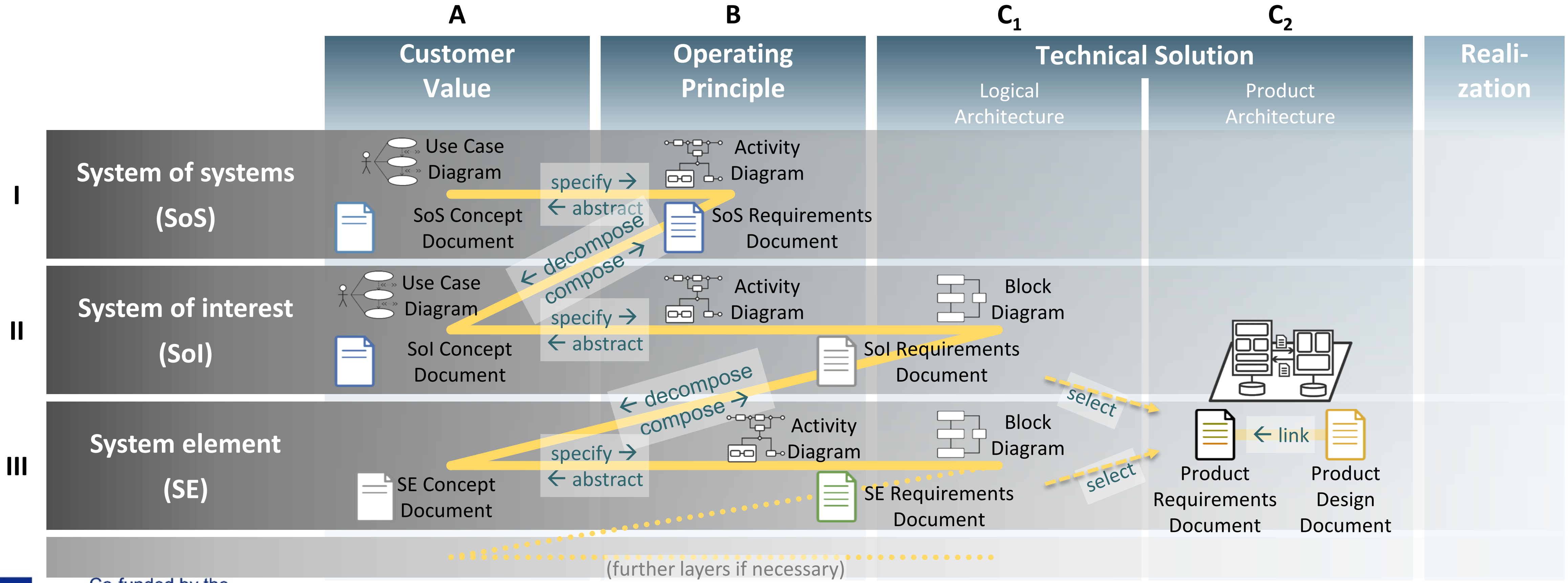


Source: FEV Europe GmbH

Co-funded by the Erasmus+ Programme of the European Union

Standardized procedure for specification of system requirements

CUBE REQUIREMENTS SPECIFICATION PROCEDURE



Source: FEV Europe GmbH

Co-funded by the Erasmus+ Programme of the European Union

Agenda

- Motivation of systems engineering
- Systems engineering methodology
- Requirements specification guidelines
- Introduction to SysML
- Exemplary application of CUBE methodology
- Challenges and benefits
- References

Co-funded by the
Erasmus+ Programme
of the European Union

Learning objectives

- Requirements specification guidelines
  - What defines a well-formed requirement?
  - How to formulate unambiguous requirements?
  - How does a good and bad requirement look like? And what are the quality criteria for requirements?

Co-funded by the Erasmus+ Programme of the European Union

What defines a well-formed requirement?

REQUIREMENTS CHARACTERISTICS

A requirement is a statement that:

- Must be fulfilled by a system to solve a **specific stakeholder problem**

- Describes a **specific** system capability or performance

- Is defined by **measurable** conditions and bounded by constraints

- Can be **verified**

- ISO/IEC/IEEE 29148 [15]

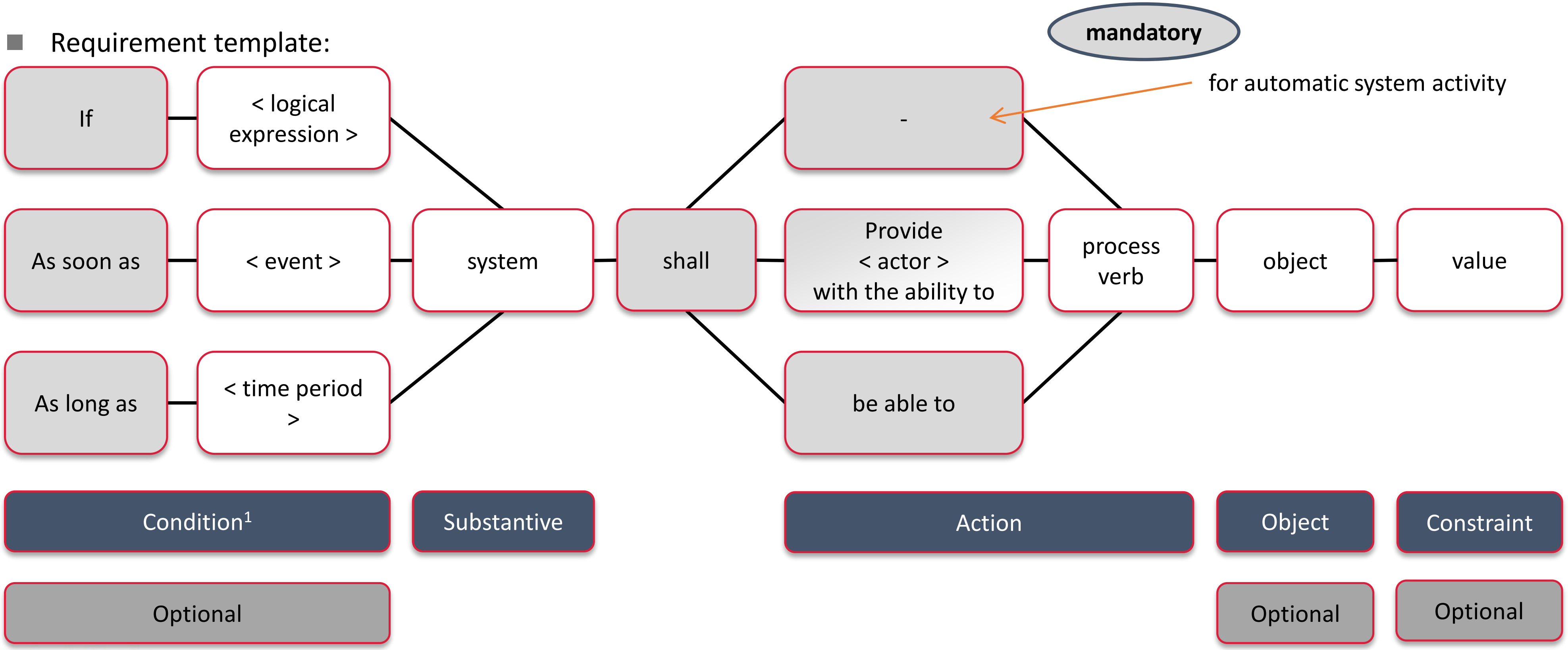
Co-funded by the Erasmus+ Programme of the European Union

Formulating unambiguous requirements

UNITED

REQUIREMENTS SPECIFICATION TEMPLATE

- Requirement template:



1 – It is also possible to attach the condition at the end of the sentence, as long as readability and understanding do not suffer.

Source: Die SOPHISTen: "Schablonen für alle Fälle" [16]

Co-funded by the Erasmus+ Programme of the European Union

Formulating unambiguous requirements

UNITED

GOOD EXAMPLES FOR REQUIREMENTS SPECIFICATION

- Example 1:

Condition[1]		Substantive		Action	Object
If	the absolute speed is below 50 km/h,	the vehicle	shall	deactivate	the Lane Keeping Assist functionality

- Example 2:

Condition[2]		Substantive		Action		Object	Constraint
If	the uphill gradient is less than or equal to 3%,	the vehicle	shall	be able to	reach	the creeping velocity	of 7 km/h

- Example 3:

Substantive		Action	Object	Constraint
The vehicle	shall	have	a maximum kerb weight	of 1500 kg

Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

Formulating unambiguous requirements

BAD EXAMPLES FOR REQUIREMENTS SPECIFICATION

- Example 1:

Substantive		Action	Constraint	Object
The vehicle	shall	be	lighter than	other vehicles in its class

- Example 2:

Substantive		Action	Object	Constraint
The torque	shall	be limited		

- Example 3:

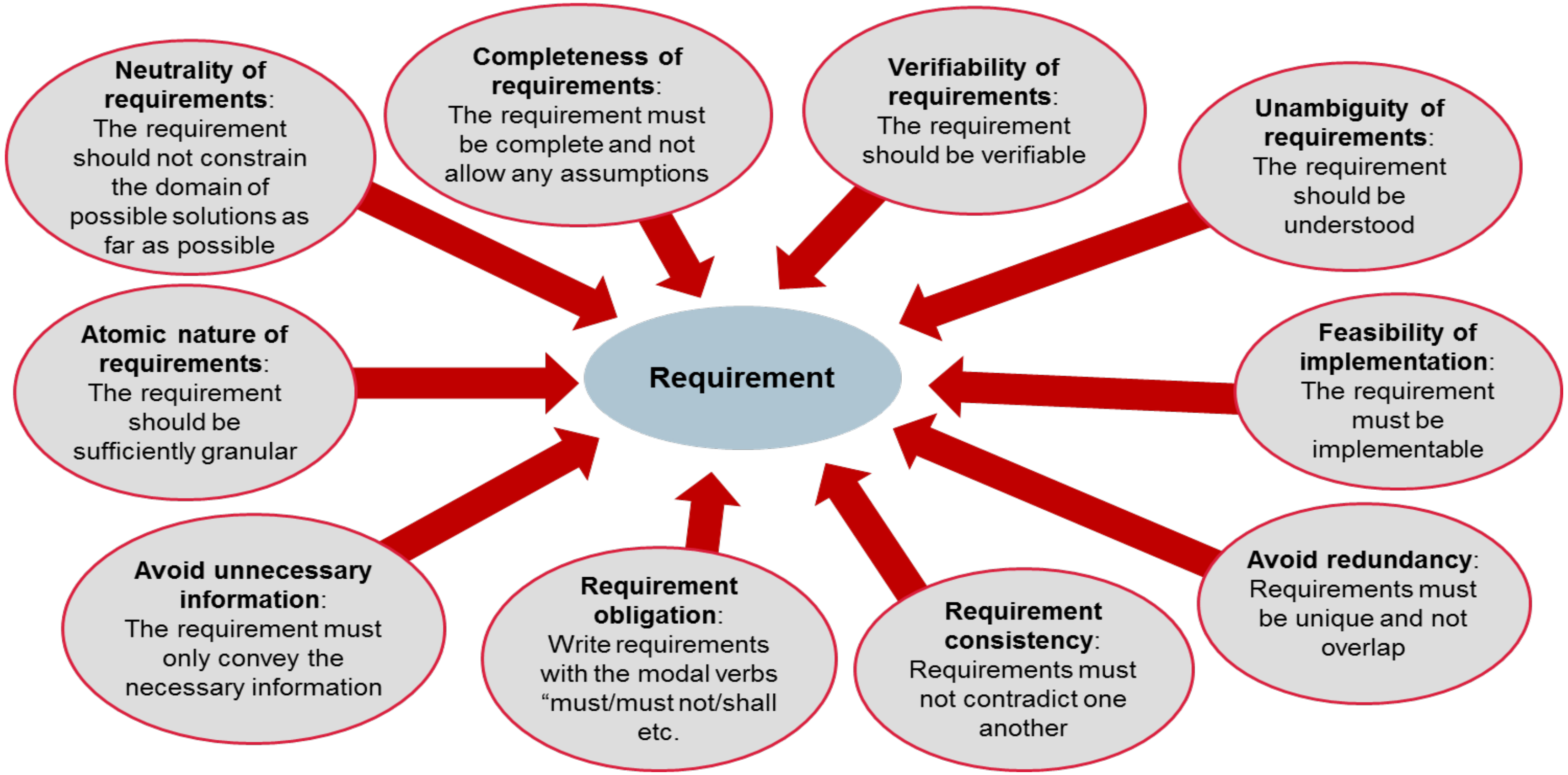
Substantive		Action	Object	Constraint
The software	shall	limit	the torque request	in most driving conditions.

Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

Evaluating formulated requirements

QUALITY CRITERIA TO EVALUATE REQUIREMENTS



Source: FEV Europe GmbH

Agenda

- Motivation of systems engineering
- Systems engineering methodology
- Requirements specification guidelines
- Introduction to SysML
- Exemplary application of CUBE methodology
- Challenges and benefits
- References

Co-funded by the
Erasmus+ Programme
of the European Union

Learning objectives

- Introduction to SysML
  - What is the benefit of model-based development compared only using textual specification?
  - What is SysML and what is it used for?
  - What is a use case diagram, what is it used for and how is it modeled?
  - What is a use activity diagram, what is it used for and how is it modeled?

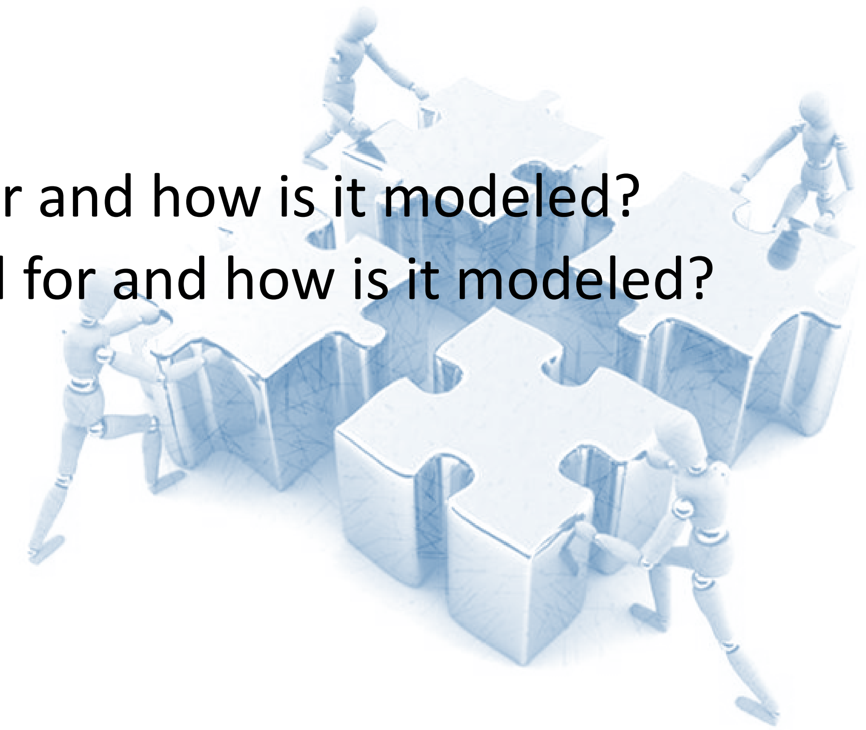


Image: Jasmin Hamadeh (http://www.jasmin-hamadeh.de/wp-content/uploads/2013/03/team_puzzle_clipart.jpg), https://creativecommons.org/licenses/by-nd/3.0/deed.de

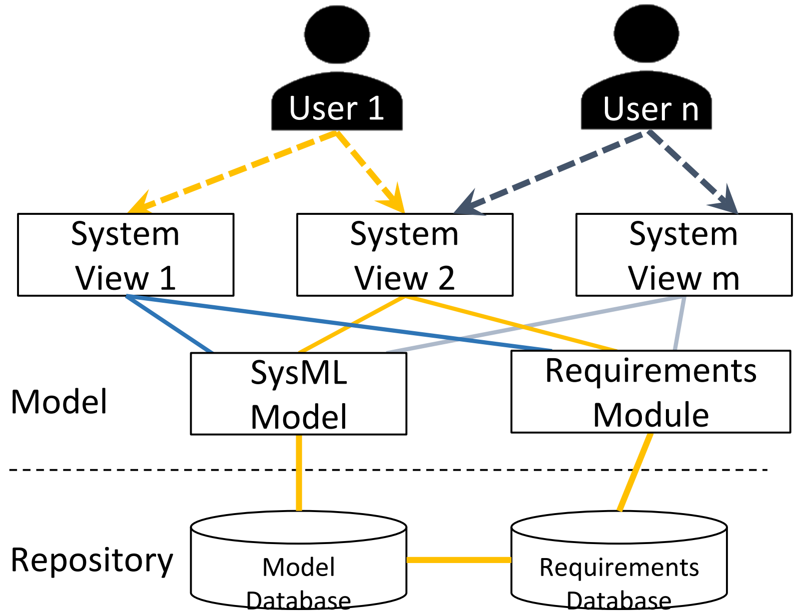
Co-funded by the Erasmus+ Programme of the European Union

Unambiguous requirement specifications using models

MOTIVATION FOR MODEL-BASED SYSTEMS ENGINEERING

Model

- Relationships within the system / function architecture easier to recognize
- Sequence errors easier to find
- Machine-readable format
- Reduced ambiguity through common language (eg. SysML)



Text

- Non-functional properties easier to describe
- Content errors easier to find
- Relationships searchable
- Boundary conditions clearly definable

Both

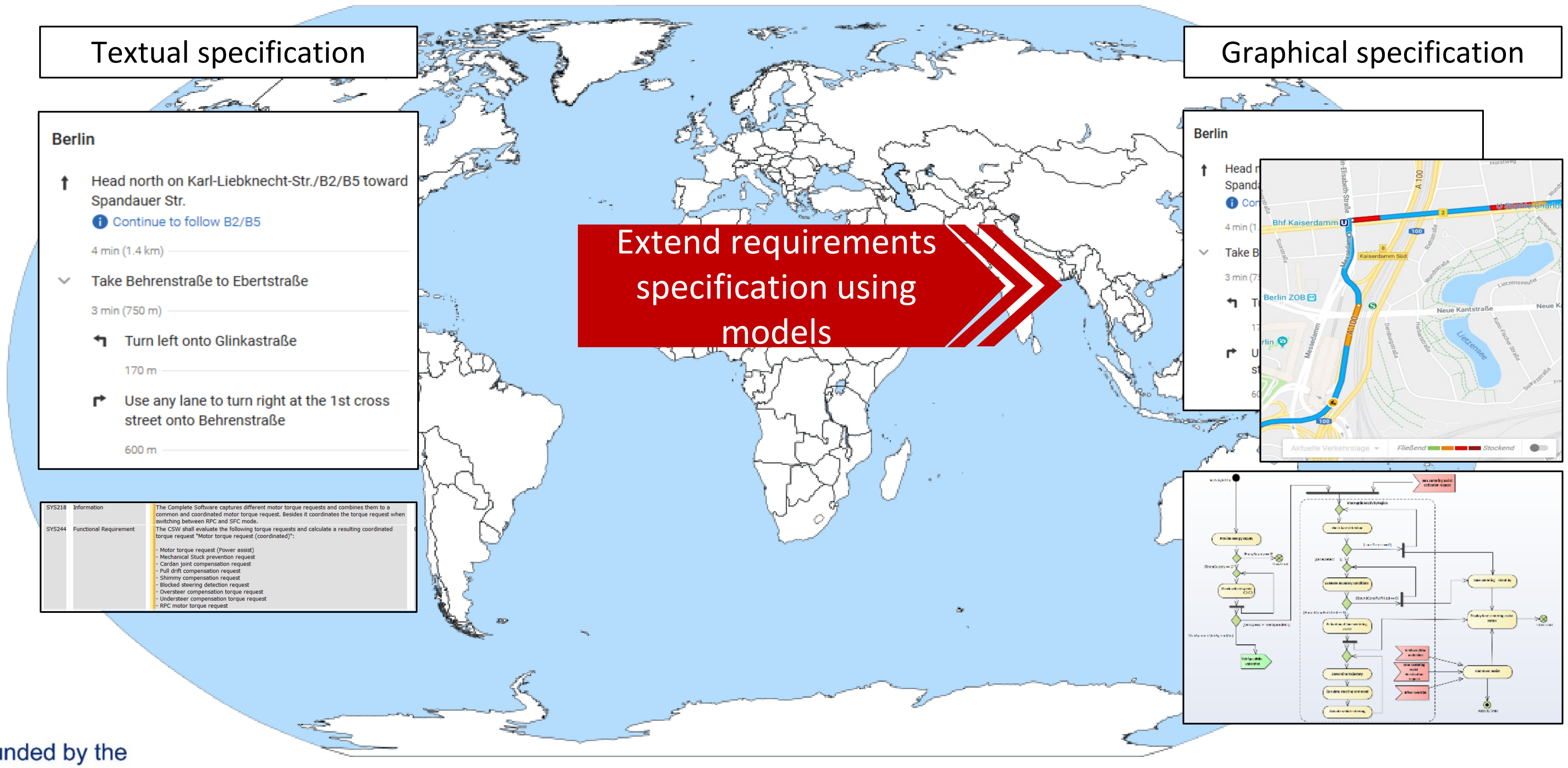
- Make knowledge explicit
- Unite customers, architects and developers
- Be a living, breathing document

Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

Unambiguous requirement specifications by usage of model-based approach

BENEFITS OF MODEL-BASED SYSTEMS ENGINEERING



Co-funded by the Erasmus+ Programme of the European Union

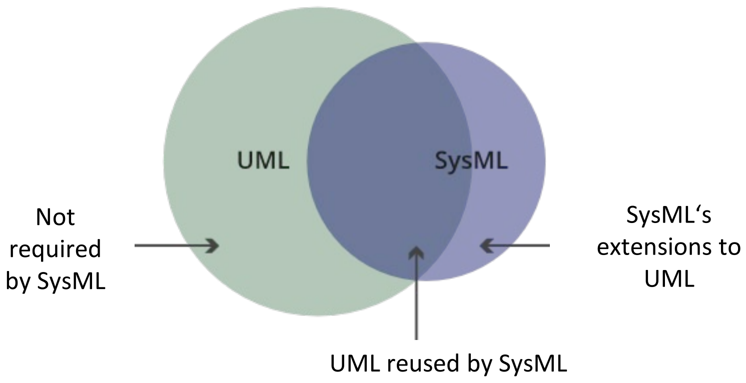
Source: https://www.google.de/maps/preview

FOR EDUCATIONAL PURPOSE ONLY

Standardized language for modern model-based systems engineering

SYSTEMS MODELING LANGUAGE

- SysML™ is a general-purpose graphical modeling language for specifying, analyzing, designing and verifying complex systems that may include hardware, software, information, personnel, procedures and facilities. It is a specialized UML profile targeted to systems engineering.
- Standardized by Object Management Group (OMG)
  - Current version 1.5
- Based on the Unified Modeling Language (UML 2)
  - Extension of existing UML diagram types
  - Specification of new diagram types
- SysML is a semiformal modeling language and therefore combines the exploitation of domain models as well as textual notation elements like written requirements
  - SysML's semantics are flexible and expandable via stereotyping in order to tailor it for domain or project specific needs

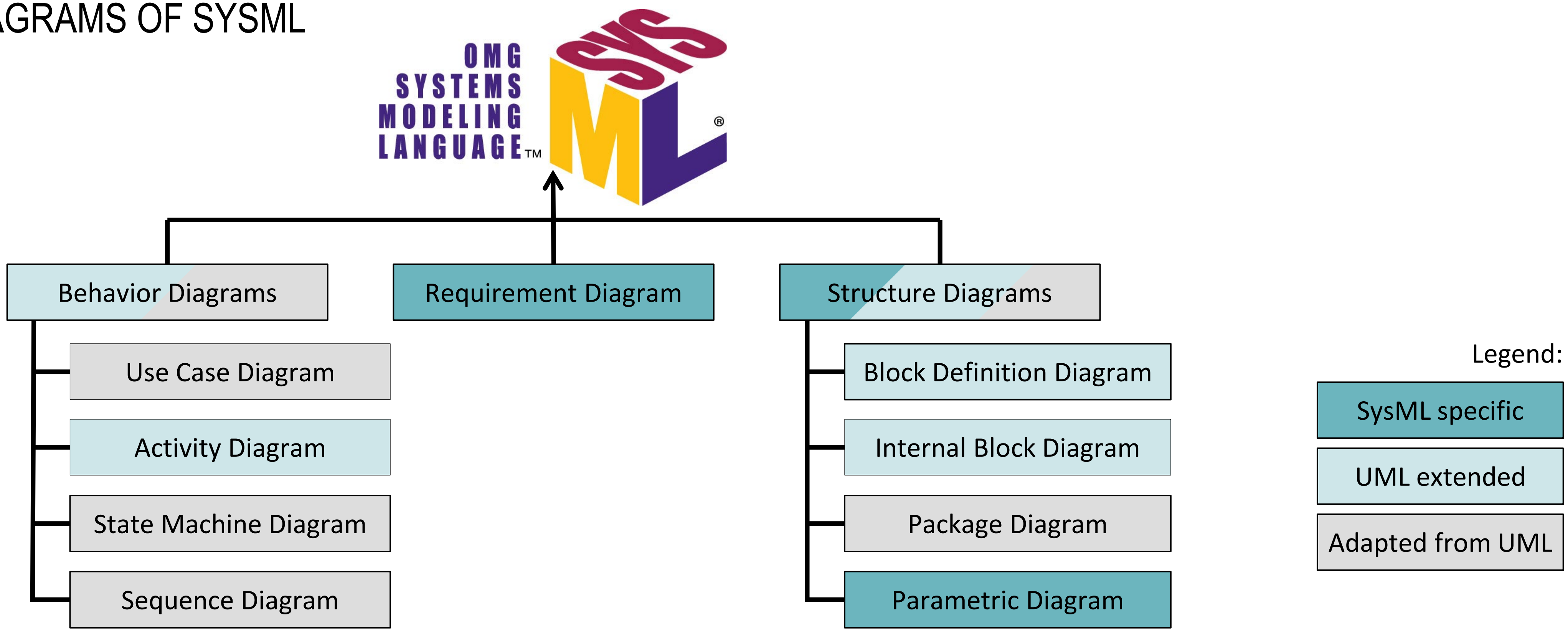


Co-funded by the Erasmus+ Programme of the European Union

Sources: https://re-magazine.ireb.org; Wikipedia; https://omg.org/spec/SysML/

SysML diagrams are inspired, extended and advanced from the UML standard

DIAGRAMS OF SYSML



Co-funded by the Erasmus+ Programme of the European Union

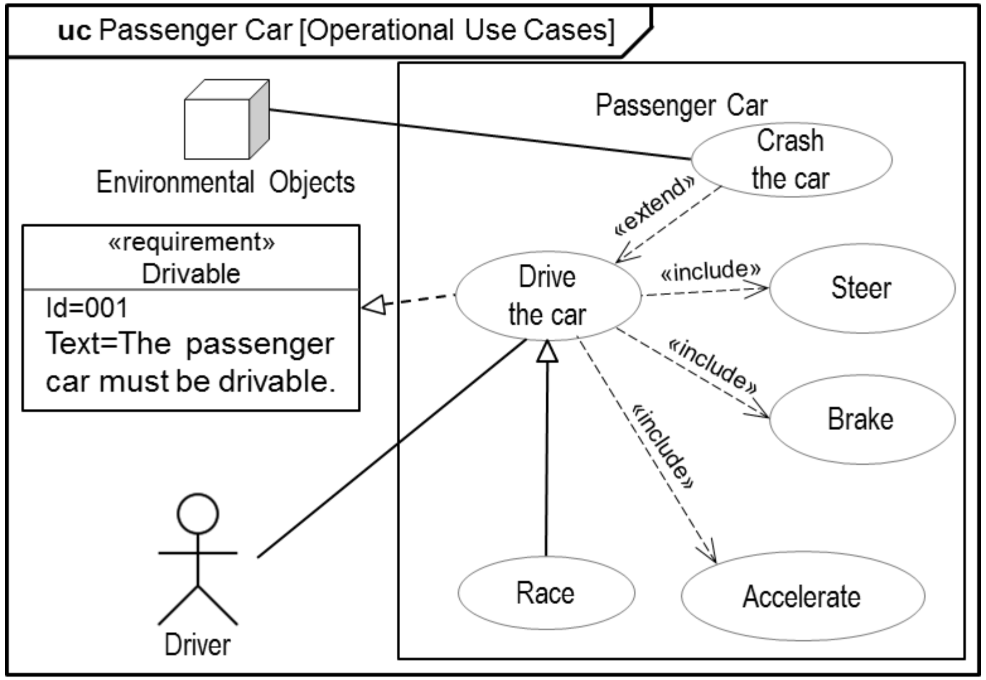
Source: https://sysml.org/

UML: Unified Modeling Language

High-level overview of system functionality and system stakeholders

USE CASE DIAGRAM - OVERVIEW

- **Black-box** (high-level) description of the system and its operational context
  - Demarcation of system boundary to ensure separation of concerns
- Describes the basic functionalities performed by a system (subject) through the interaction with its environment (actors) to achieve a goal.
- Includes the use case and actors and the associated communications between them.
- Use cases can be extended to include use case descriptions.

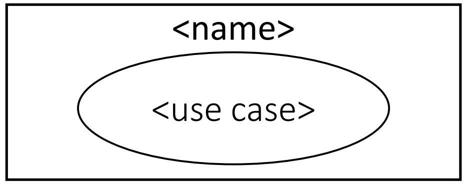
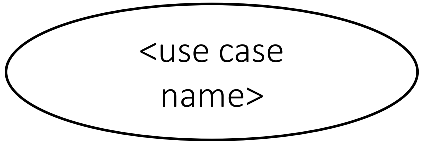
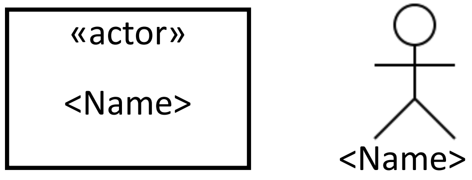


Source: FEV Europe GmbH

Co-funded by the Erasmus+ Programme of the European Union

Which notation elements are available?

USE CASE DIAGRAM – DIAGRAM ELEMENTS (PART I)

Element name	Notation	Description
Subject (System Boundary)		A subject represents the system that is being developed. It supports different functionalities through its use cases.
Use Case		A use case describes the function that the system offers regarding how its users use the system to achieve their goal.
Actor		An actor represents the role of a human (user), an organization, or any external system that interacts with the subject (system).

Source: FEV Europe GmbH

Co-funded by the Erasmus+ Programme of the European Union

Which notation elements are available?

UNITED

USE CASE DIAGRAM – DIAGRAM ELEMENTS (PART II)

Element name	Notation	Description
Association		Association is a relationship between an actor and a use case that indicates the interaction between the actor and the system.
Include	«include»	Include is a relationship that indicates the function of the included use case is a part of the base use case functionality.
Extend (with condition)	Condition: {<Constraint>} Extension points: <ExtensionPoint> «extend»	Extend allows a base use case to be extended by an extending use case, the functionality of which is not part of the base use case functionality.
Generalization		Generalization relationship shows one element (child) inherits the properties of another element (parent) but is more specialized.

Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

High-level overview of vending machine functionality

USE CASE DIAGRAM – A PRACTICAL EXERCISE

- Cold Drink Vending Machine
  - **Aim**: The aim of the following exercise is to practice drawing of a use case diagram:
  - **Problem definition**: Assume that the design of a cold drink vending machine is under development i.e. the system (subject) is a vending machine.
- To put the vending machine into operation, the maintenance employee will setup the machine and then load the machine with products (cold drinks). Assume that a customer (user) wants to purchase a cold drink from the vending machine. To do so, the user must select an item, and then pay the amount related to that item. The machine will take the item from the storage automatically and drops it in the container from which the user can extract the item. In case that the machine is out-of-order, the maintenance employee will repair the machine.

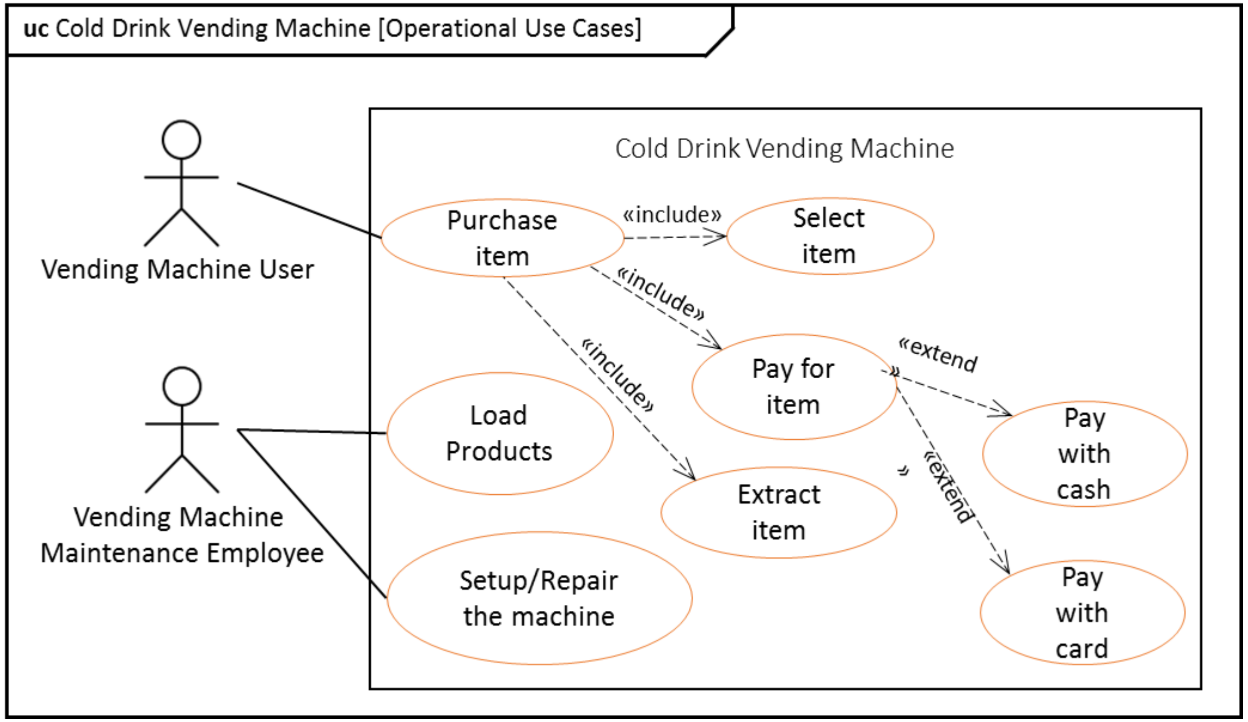
Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH
Image: Shig ISO (https://commons.wikimedia.org/wiki/File:沖縄コーラ_Okinawan_Cola_(4230611559).jpg), „沖縄コーラ Okinawan Cola (4230611559)“, https://creativecommons.org/licenses/by/2.0/legalcode

High-level overview of vending machine functionality

USE CASE DIAGRAM – A PRACTICAL EXERCISE

- Possible Solution for the Use-Case diagram



Source: FEV Europe GmbH

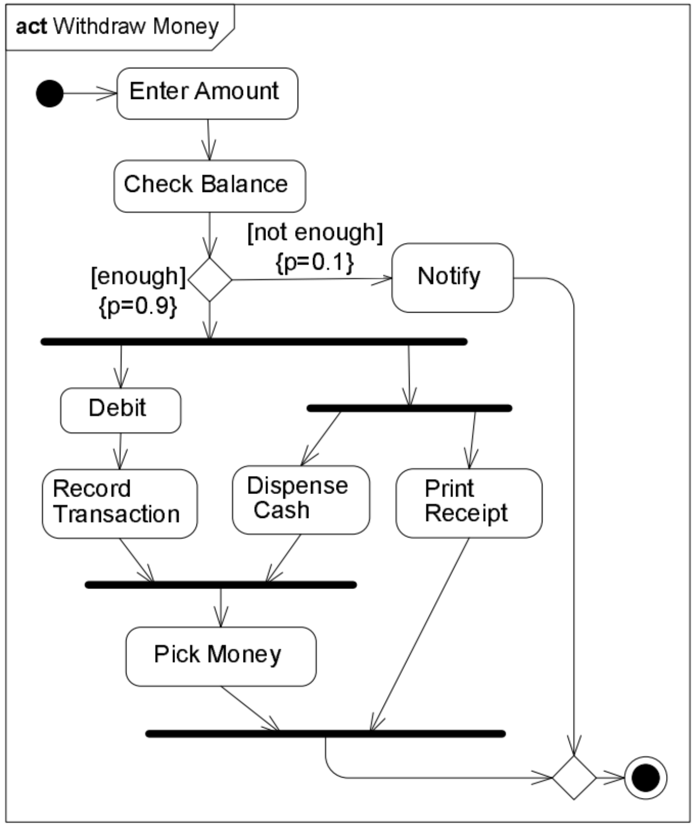
To put the vending machine into operation, the maintenance employee will setup the machine and then load the machine with products (cold drinks). Assume that a customer (user) wants to purchase a cold drink from the vending machine. To do so, the user must select an item, and then pay the amount related to that item. The machine will take the item from the storage automatically and drops it in the container from which the user can extract the item. In case that the machine is out-of-order, the maintenance employee will repair the machine.

Co-funded by the Erasmus+ Programme of the European Union

Functional sequence of system actions to describe system operation

ACTIVITY DIAGRAM - OVERVIEW

- **Dynamic view** of a system, expressing its behavior in a particular **order of actions**
- Mainly used to specify expectation of behavior (e.g. the transformation of inputs to outputs)
- Primary representation for modelling workflow in SysML
  - Focuses on:
    - Execution flow of actions
    - Actions' dependency between themselves
    - Capturing activities made up of smaller actions
- Can include constructs depicting allocation/grouping of actions
    - Action partitioning (e.g. swim lanes)

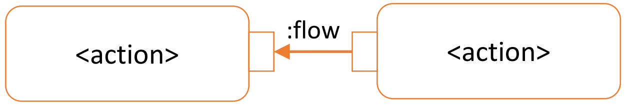
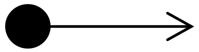
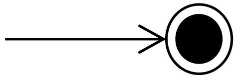
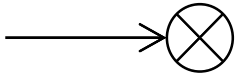


Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH
Diagram Source: On the Meaning of SysML Activity Diagrams, Jarraya et al. [19]

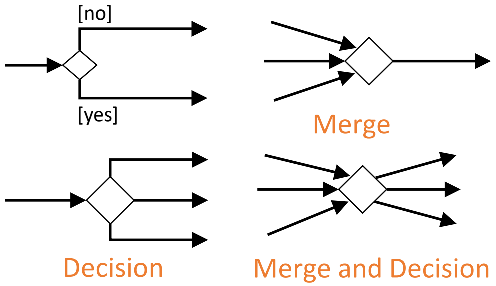
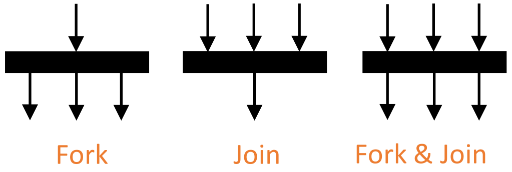
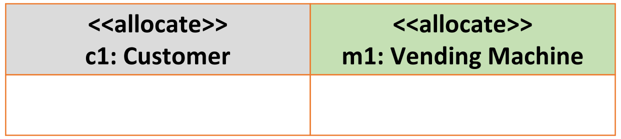
Which notation elements are available?

ACTIVITY DIAGRAM – DIAGRAM ELEMENTS (PART I)

Element name	Notation	Description
Actions	<action>	Actions are round-cornered rectangle shaped. They represent the single (atomic) steps within the activity. That means that an activity depicts a behavior composed by individual elements, the actions.
Object (nodes)	<object>	Object nodes are rectangle shaped. They are used to define objects flowing within an activity. These can be extended to include pins, buffers, parameters and expansions which will not be covered here.
Pins		Pins are a kind of Object Nodes, which are used to specify inputs and outputs for actions. The Pin symbol is a square which is attached to the outside edge of the Action block.
Initial State		The Initial State node provides the starting point for the behavior execution of an Activity with one or more control flows coming out of it. It is however possible to have both multiple or no initial state nodes.
Final State		When the control flow reaches the Final State node, the execution of the entire activity is terminated.
Flow Final State		When the control flow reaches the Flow Final State node, the flow of actions entering the node is terminated. However, the entire activity is not.

Source: FEV Europe GmbH

Which notation elements are available?

Element name	Notation	Description
Decision and Merge node(s)		A decision node is a control node that accepts tokens on one or two of the (incoming) edges and selects one outgoing edge for which a condition is fulfilled. A merge node is a control node that brings together multiple incoming flows to give out a single outgoing flow. Edges must be either object or control flows.
Call Behavior Action		A call behavior action can invoke a behavior diagram of another activity.
Fork and Join Node(s)		Fork nodes are control nodes that split one incoming edge into multiple edges (parallelism). Join nodes are control nodes that join multiple incoming edges into one outgoing edge.
Swim Lanes		Swim lanes can be used for allocation of behaviors based on responsibilities.

Co-funded by the
Erasmus+ Programme
of the European Union

Source: FEV Europe GmbH

Functional sequence of system actions for ticket vending machine

ACTIVITY DIAGRAM – A PRACTICAL EXERCISE

- Ticket Vending Machine
  - The aim of the following exercise is to practice drawing of an activity diagram:
  - Assume that the design of a ticket vending machine is under development. Draw an activity diagram describing the operation of the ticket vending machine.
  - The operation is initialized by the Customer who is then requested by the Machine to provide the trip and payment info.
  - Payment by card or cash can be chosen. In case of card payment, the Machine sends information to the Bank for processing.
  - In case card payment is declined, a “*Card declined*” message has to be displayed.
  - If payment is successfully processed, the ticket is dispensed as well as any change (if cash payment has been chosen). Operation ends with one of the following notifications:
    - A “*Thank you*” message has to be displayed on the screen if the process has been successfully finished (with either card or cash).

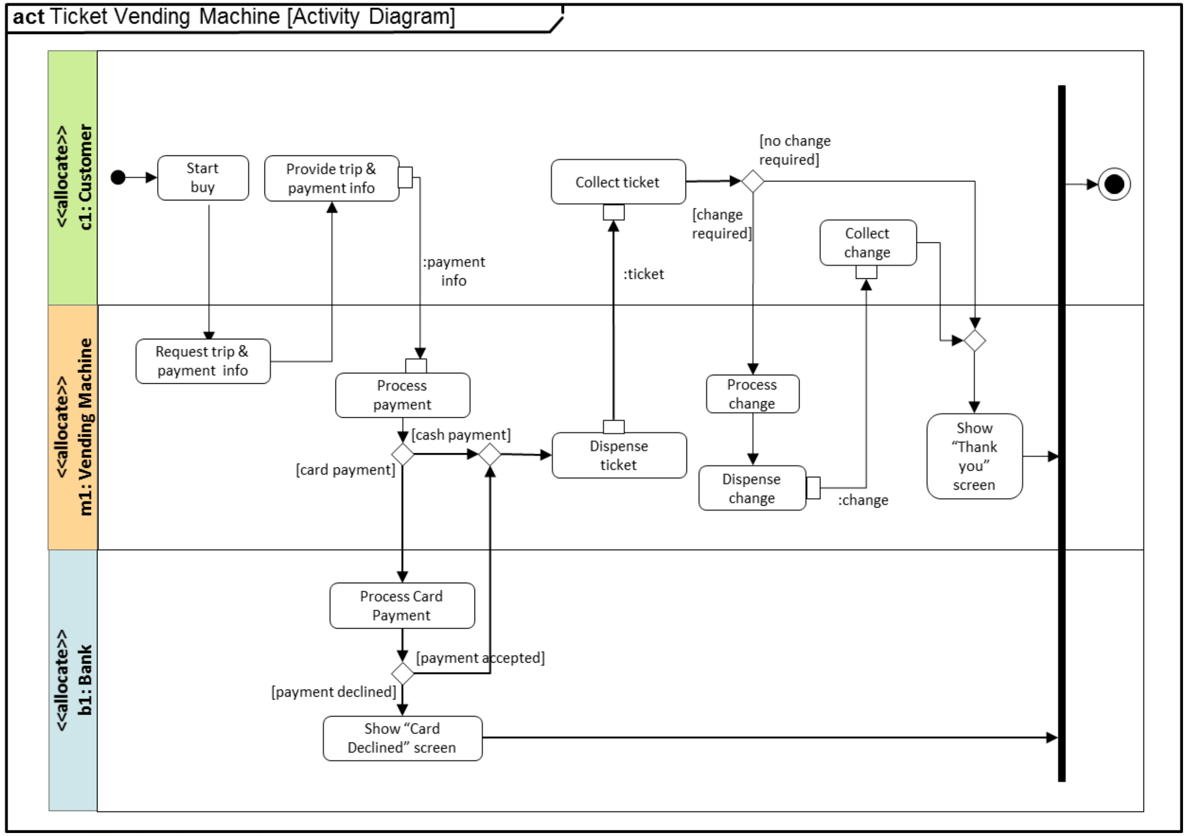
Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH
Image: Public Domain (https://commons.wikimedia.org/wiki/File:Parkscheinautomat_Skidata.jpg)

Functional sequence of system actions for ticket vending machine

UNITED

ACTIVITY DIAGRAM – A PRACTICAL EXERCISE



Source: FEV Europe GmbH

- The operation is initialized by the Customer who is then requested by the Machine to provide the trip and payment info.
- Payment by card or cash can be chosen. In case of card payment, the Machine sends information to the Bank for processing.
- In case card payment is declined, a "*Card declined*" message has to be displayed.
- If payment is successfully processed, the ticket is dispensed as well as any change (if cash payment has been chosen). Operation ends with one of the following notifications:
  - A "*Thank you*" message has to be displayed on the screen if the process has been successfully finished (with either card or cash).

Co-funded by the Erasmus+ Programme of the European Union

Agenda

- Motivation of systems engineering
- Systems engineering methodology
- Requirements specification guidelines
- Introduction to SysML
- Exemplary application of CUBE methodology
- Challenges and benefits
- References

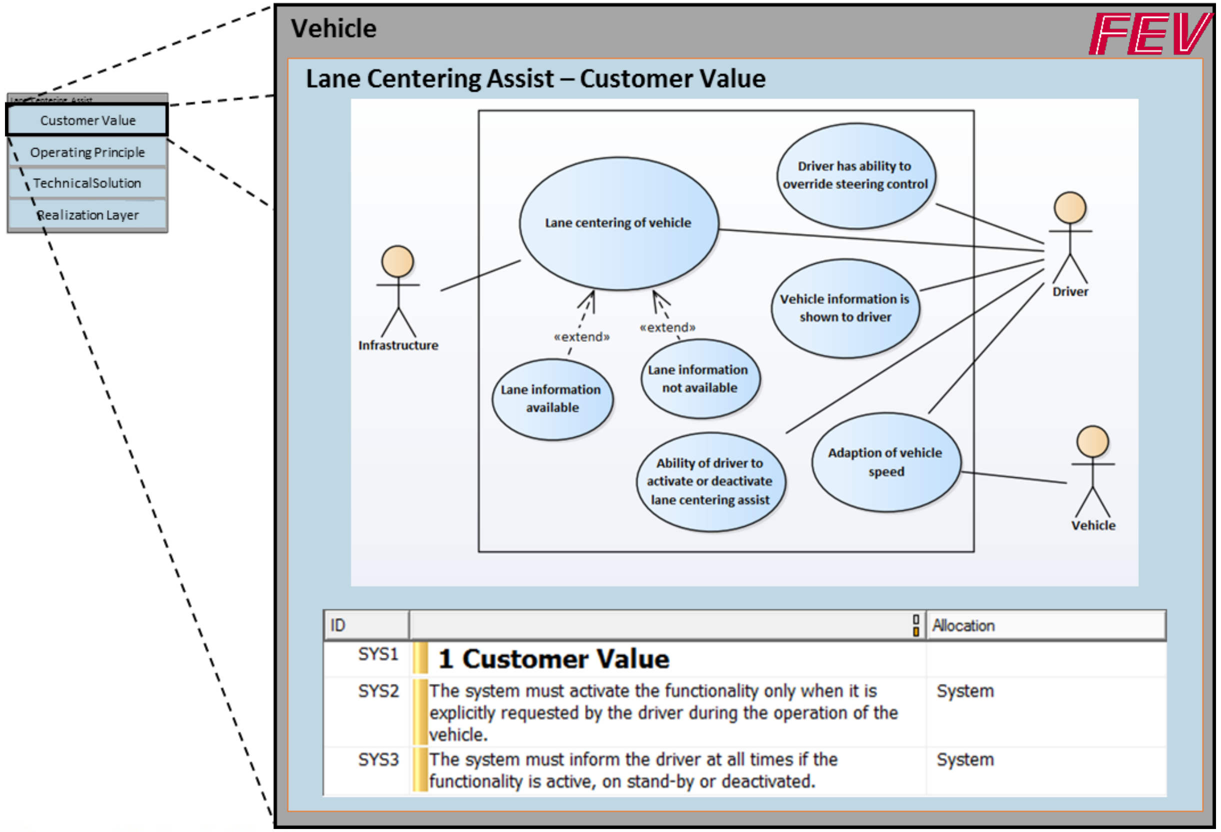
Co-funded by the Erasmus+ Programme of the European Union

Learning objectives

- Exemplary application of CUBE methodology
  - How to use SysML for the model-based specification of system?
  - How to use the CUBE methodology in combintion with SysML?

A use-case diagram captures the expectations of all system actors

CUSTOMER VALUE - LANE CENTERING ASSIST



ID		Allocation
SYS1	1 Customer Value	
SYS2	The system must activate the functionality only when it is explicitly requested by the driver during the operation of the vehicle.	System
SYS3	The system must inform the driver at all times if the functionality is active, on stand-by or deactivated.	System

Source: FEV Europe GmbH

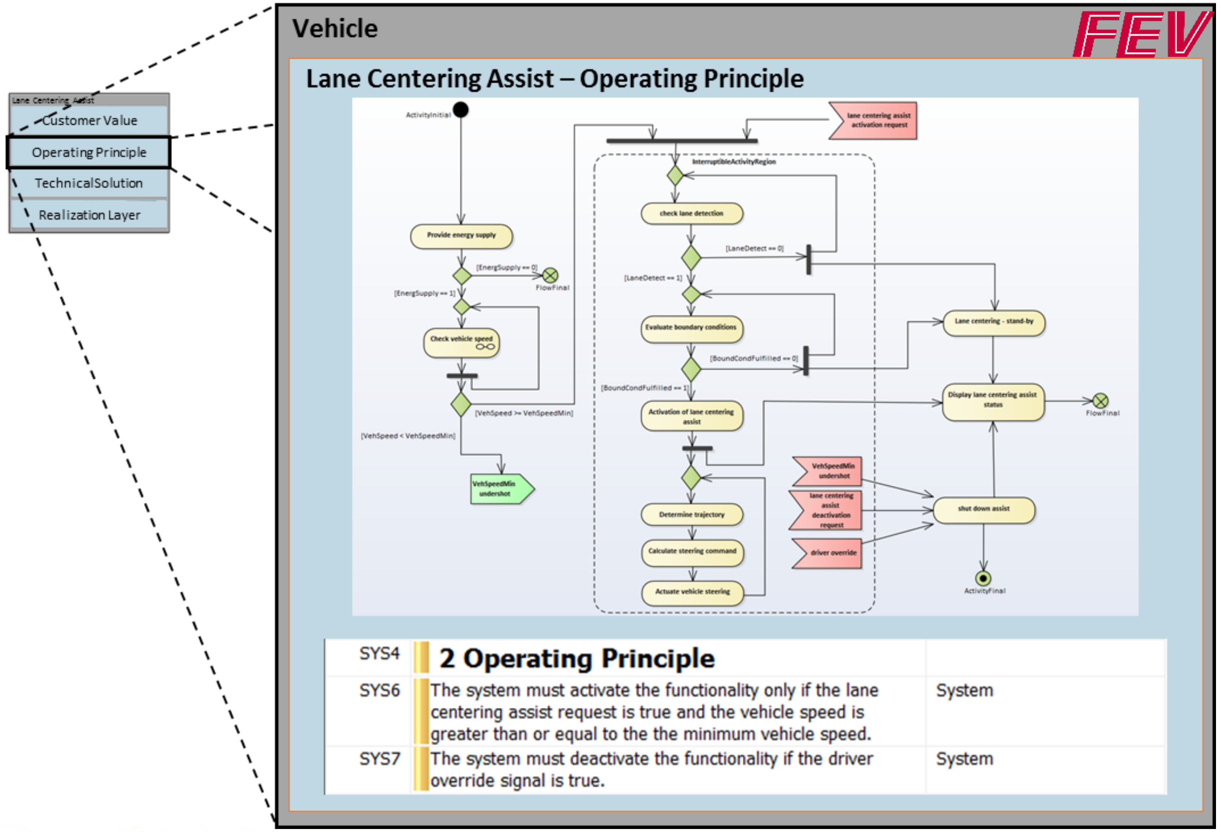
- Black-box view on system, describing the customer value of the system
- Most abstract and solution neutral description of the system
- Graphical, easy-understandable representation
- Overview of all actors of the system
- Representation of dependencies between system use cases

Co-funded by the Erasmus+ Programme of the European Union

The operating principle is expressed in an activity diagram

UNITED

OPERATING PRINCIPLE - LANE CENTERING ASSIST



Source: FEV Europe GmbH

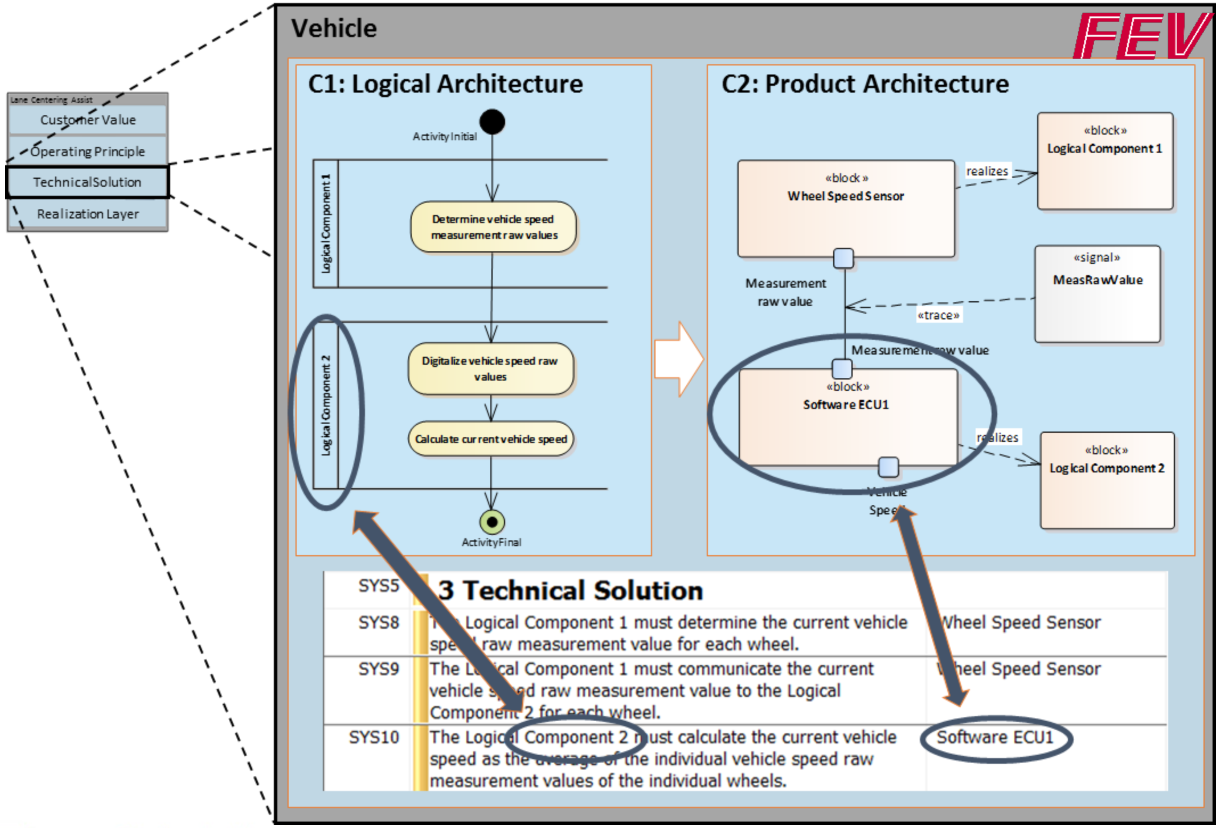
- White-box view on system, describing the operating principle / functionalities of the system
- Abstract description of the system with graphical, easy-understandable representation
- Solution agnostic description of system functionality including:
  - system actions
  - constrains and transitions
  - dependencies to other composition elements
  - decision paths

Co-funded by the Erasmus+ Programme of the European Union

Allocation of activities to a logical architecture and final product architecture

UNITED

TECHNICAL SOLUTION - LANE CENTERING ASSIST



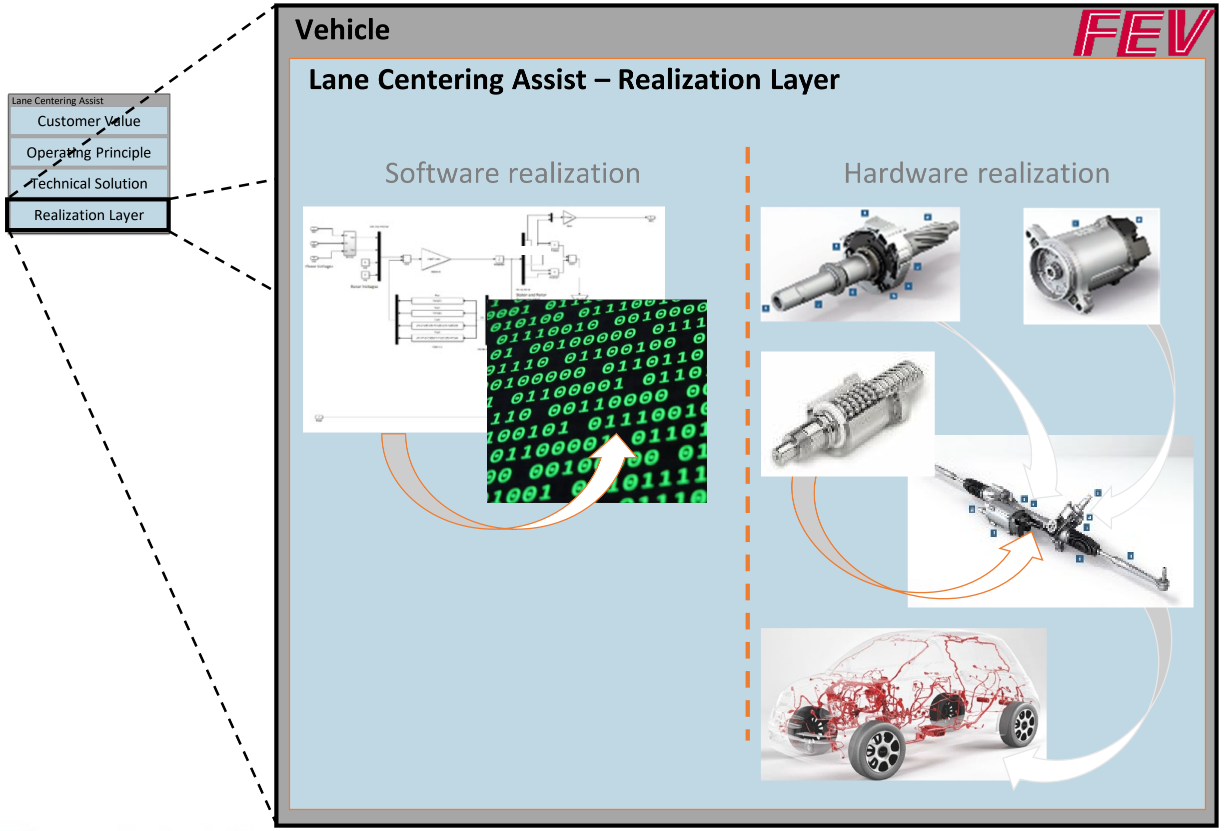
Source: FEV Europe GmbH

- Solution oriented description of the system
- Technical solution of the system describing system architecture (including HW and SW)
- Decision on HW and SW realization including assignment of system actions to system realization elements (“partitions”)
- Description of information flows between partitions including ports and signals

Co-funded by the Erasmus+ Programme of the European Union

Realization of lane centering assist is completed in multiple product artefacts

EXEMPLARY REALIZATION - LANE CENTERING ASSIST



- Most detailed and solution oriented description of the system
- Best practice for artefacts is very specific to the product at hand
- Implementation result of described technical solution of the system
- Software or hardware product/ realization artifact of the system

Co-funded by the Erasmus+ Programme of the European Union

Source Pictures: https://www.flickr.com/photos/christiaancolen/20607150556 ; https://docplayer.org/12015020-Servolectric-elektromechanisches-lenksystem-fuer-ein-dynamisches-fahrgefuehl-und-hochautomatisierte-funktionen.html; FEV Europe GmbH

Agenda

- Motivation of systems engineering
- Systems engineering methodology
- Requirements specification guidelines
- Introduction to SysML
- Exemplary application of CUBE methodology
- Challenges and benefits
- References

Co-funded by the
Erasmus+ Programme
of the European Union

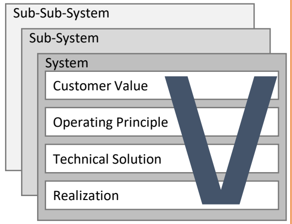
Learning objectives

- Challenges and benefits
  - What are the challenges by introducing systems engineering as development approach?
  - What are the benefits by using systems engineering?

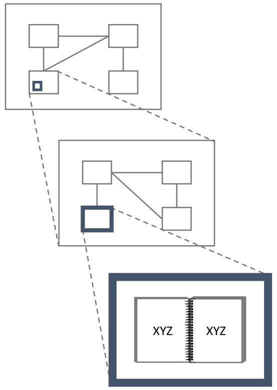
Image: Jasmin Hamadeh (http://www.jasmin-hamadeh.de/wp-content/uploads/2013/03/team_puzzle_clipart.jpg), https://creativecommons.org/licenses/by-nd/3.0/deed.de

Improved project results with systems engineering

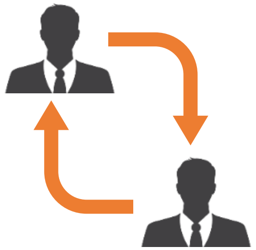
BENEFITS OF SYSTEMS ENGINEERING



- Holistic system development and management over product life cycle
- Reduced risk during system integration
- Reduced costs through reusable platform solutions



- Structured derivation of unambiguous requirements
- Complete traceability of requirements ensured
- Formally analyzable system architecture and requirements



- Improved interdisciplinary collaboration and communication

- Verified and validated performance of functionality through test case generation

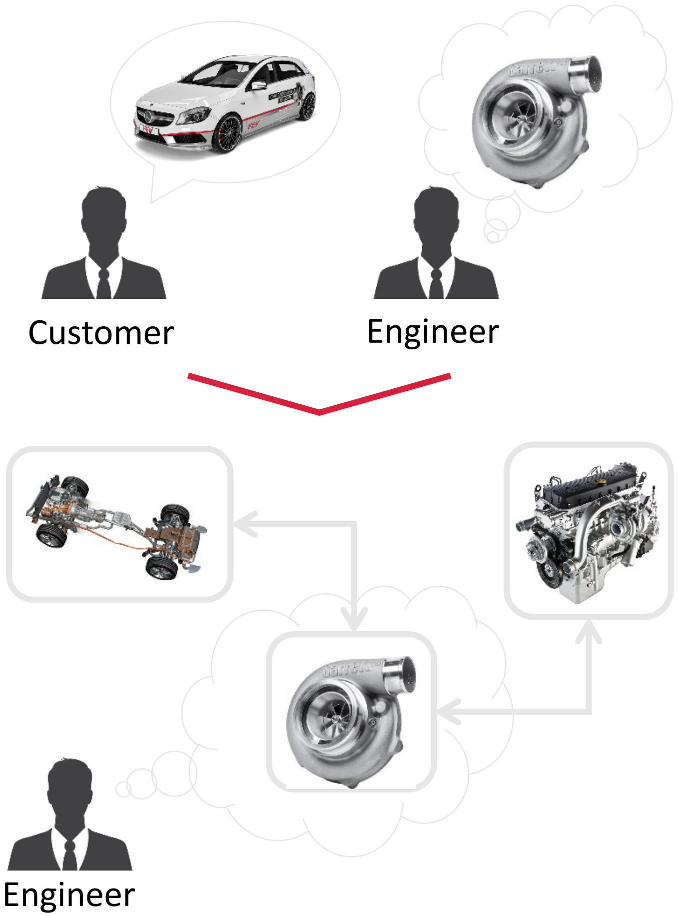
Co-funded by the Erasmus+ Programme of the European Union

Source: Systems Engineering in der industriellen Praxis (2013) [17];
INCOSE Systems Engineering Handbook (2015) [8];
Systems Engineering mit SysML/UML (2008) [18]

Correctly implementing systems engineering

CHALLENGES AND RISKS

- Encouraging a **mindset shift** from component-based to function-based engineering on system level
- Common understanding of systems engineering requires **build-up of know-how**
- Effective **collaboration** and **communication** mandatory
- Definition of **project responsibilities** necessary
- Increased initial effort due to frontloading of **development tasks**
- Clearly **defined abstraction levels** necessary for system specification and definition of **maturity levels**
- Aligned **documentation** of methodology required
- Interdisciplinary collaboration requires appropriate **team coordination** across different development areas



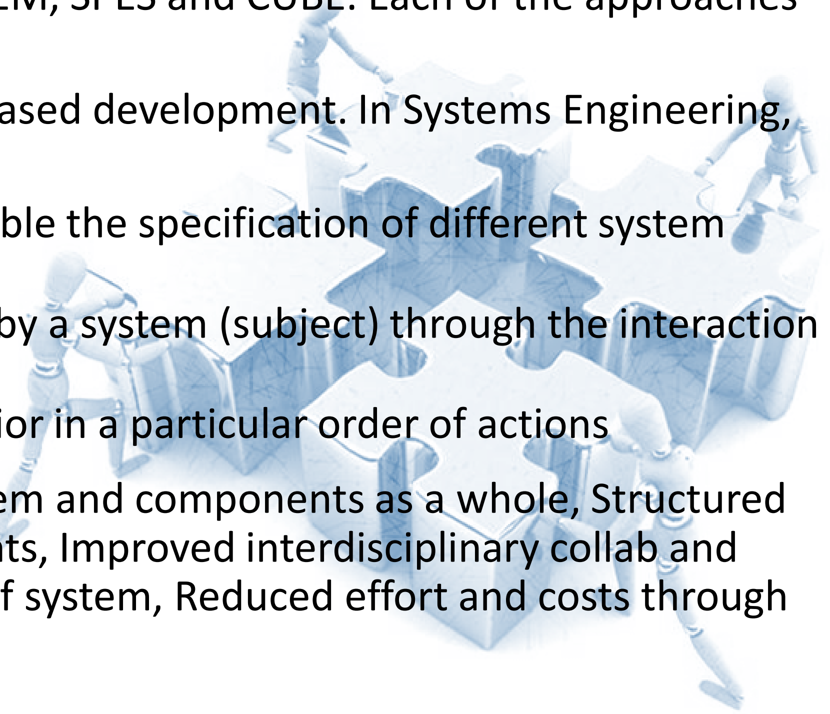
Co-funded by the Erasmus+ Programme of the European Union

Source: https://www.tzr-motorsport.de/Garrett-Turbolader-GTX3071R-GEN-2-A/R-063; https://www.autobild.de/artikel/vw-touareg-hybrid-gegen-lexus-rx-450h-1142265.html; http://www.directindustry.de/prod/fpt-industrial-spa/product-98569-1875345.html
Image: FEV Europe GmbH

Summary & Conclusion

UNITED

- "Systems Engineering is an interdisciplinary and holistic engineering approach for the conception, realization and evaluation of complex technical systems as well as for engineering management over the complete life cycle of the systems."
- Several approaches exist for the system-based development e.g. OOSEM, SPES and CUBE. Each of the approaches has its focus on slightly different aspects.
- Unambiguous requirement specification can be supported by model-based development. In Systems Engineering, SysML is very common for model-based specification.
- SysML is a derivation of the UML and provides several diagrams to enable the specification of different system views.
  - Use case Diagram: Describes the basic functionalities performed by a system (subject) through the interaction with its environment (actors) to achieve a goal.
  - Activity Diagram: Dynamic view of a system, expressing its behavior in a particular order of actions
- Benefits: Holistic development over product life cycle → focus on system and components as a whole, Structured derivation of requirements → unambiguous and complete requirements, Improved interdisciplinary collab and communication, formal analyzability of requirements through model of system, Reduced effort and costs through platform and variant management, Reduced risk at system integration



Co-funded by the Erasmus+ Programme of the European Union

Image: Jasmin Hamadeh (http://www.jasmin-hamadeh.de/wp-content/uploads/2013/03/team_puzzle_clipart.jpg), https://creativecommons.org/licenses/by-nd/3.0/deed.de

Agenda

- Motivation of systems engineering
- Systems engineering methodology
- Requirements specification guidelines
- Introduction to SysML
- Exemplary application of CUBE methodology
- Challenges and benefits
- References

Co-funded by the Erasmus+ Programme of the European Union

References

[1] PRASAD, B.
Analysis of pricing strategies for new product introduction
Pricing Strategy and Practice, Vol. 5 Issue: 4, pp.132-141
Bingley (UK), 1997

[2] MOHR, D. et al.
The road to 2020 and beyond: What's driving the global automotive industry?
McKinsey & Company, Inc.
Stuttgart, 2013

[3] STEINKAMP, N.
2016 Automotive Warranty & Recall Report: Industry Insights for the Road Ahead
Chicago, 2015

[4] HOFACKER, A. and KÖLLNER, C.
WLTP und NEFZ im Vergleich
2017

[5] HÜBNER, H.-P.
Automatisiertes Fahren – Wohin geht die Fahrt?
Proc. 18. Kongress Fortschritte in der Automobilelektronik
Ludwigsburg, 2014

[6] KRIEBEL, S., RICHENHAGEN, J. et al.
High Quality Electric Powertrains by model-based systems engineering
Aachen Colloquium, 2017

[7] KRIEBEL, S., RICHENHAGEN, J. et al.
High Quality Electric Powertrains by model-based systems engineering
Aachen Colloquium, 2017, presentation

Co-funded by the
Erasmus+ Programme
of the European Union

References

[8] INCOSE
Systems Engineering Handbook
2004

[9] DEREK HITCHINS (1995)
cited in: HERBERT NEGELE (2000)
Systems Engineering - A Key to Competitive Advantage for All Industries. p. 166.

[10] NASA
Systems Engineering Handbook
1995

[11] FRIEDENTHAL, S., MOORE, A., STEINER, R.
A Practical Guide to SysML
2015

[12] DOUGLASS, B. P.
Agile Systems Engineering
2016

[13] KRIEBEL, S., RICHENHAGEN, J.; GRANRATH, C., KUGLER, C.
Systems Engineering with SysML: The Path to the Future?.
MTZ worldwide ed. 05/2018 p.44-47.

[14] GRANRATH, C., MEYER, M., ORTH, PH., BADER, B., RICHENHAGEN, J., KRIEBEL, S., ANDERT, J.
The next generation of electrified powertrains: Smart digital systems engineering for safe and reliable products
SIA Paris 2019

[15] ISO/IEC/IEEE 29148:2018: Systems and software engineering - Life cycle processes - Requirements engineering
2018-11

Co-funded by the
Erasmus+ Programme
of the European Union

References

[16] DIE SOPHISTEN
Schablonen für alle Fälle
2016

[17] GAUSEMEIER, J. et al.
Systems Engineering in der industriellen Praxis.
9. Paderborner Workshop: "Entwurf mechatronischer Systeme“, 2013

[18] WEILKIENS, T.
Systems Engineering with SysML/UML
2008

[19] JARRAYA et al.
On the Meaning of SysML Activity Diagrams
Concordia University, IEEE, 2009

[20] FRIEDENTHAL, MOORE & STEINER
A Practical Guide to SysML: Systems Modeling Language
2012

[21] KOSSIAKOFF, A. et al.
Systems engineering principles and practice
Vol. 83. John Wiley & Sons
2011

[22] HABERFELLNER, R. et al.
Systems Engineering–Grundlagen und Anwendung.
12. Aufl. Zürich
2012

Co-funded by the
Erasmus+ Programme
of the European Union

References

[23] POHL, K. et al.
Model-Based Engineering of Embedded Systems: The SPES 2020 Methodology.
Springer Science & Business Media
2012.

[24] KRIEBEL, S. et al.
The next generation of BMW's electrified powertrains: Providing software features quickly by model-based system design.
26th Aachen Colloquium Automobile and Engine Technology.
2017.

Co-funded by the Erasmus+ Programme of the European Union

Engineering Knowledge Transfer Units to Increase Student´s Employability and Regional Development

https://www.facebook.com/unitederasmus/

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
598710-EPP-1-2018-1-AT-EPPKA2-CBHE-JP

Engineering Knowledge Transfer Units to Increase Student´s Employability and Regional Development

Exercise - MBSE and Scrum

Day 2 – Slot 3

Christian Granrath, M.Sc.

FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
598710-EPP-1-2018-1-AT-EPPKA2-CBHE-JP

Agenda

- Simulating Scrum – A practical example
- Modeling a Use Case Diagram
- Modeling an Activity Diagram
- References

Co-funded by the
Erasmus+ Programme
of the European Union

Learning objectives

- Agile development
  - What is SCRUM, what describes this agile development approach and what is it used for?

  → Practical application of SCRUM: Ball Point Game

Image: Jasmin Hamadeh (http://www.jasmin-hamadeh.de/wp-content/uploads/2013/03/team_puzzle_clipart.jpg), https://creativecommons.org/licenses/by-nd/3.0/deed.de

Co-funded by the Erasmus+ Programme of the European Union

Simulating Scrum – The Ball Point Game

SCRUM SIMULATION GAME – TASK

- Split group into teams of 8-15 people.
- Your sprint product are “balls”.
- Estimate how many balls your team can deliver, and then deliver them (see rules on next slide).
- Collect data for each iteration: Estimated and actual number of balls delivered.

Co-funded by the Erasmus+ Programme of the European Union

Simulating Scrum – The Ball Point Game

UNITED

SCRUM SIMULATION GAME – RULES

- To ‘deliver’ a ball, every person must touch the ball.
- When you pass the ball, it must have air time.
- You cannot pass the ball to your neighbor on the right or left.
- One person has to introduce the balls into the system and the balls have to return to this person in order to be counted.

- 1 minute: Plan and estimate.
- 1 minute: Deliver. Count how many you delivered.
- 1 minute: What were your problems? Focus on problems, not solutions.
- 1 minute: Plan your next sprint, estimate again.
- Repeat 4 times!

Co-funded by the Erasmus+ Programme of the European Union

Simulating Scrum – The Ball Point Game

UNITED

SCRUM SIMULATION GAME – SUMMARY

- What were your experiences?
- What is the correlation of the game and the Scrum theory?

Co-funded by the Erasmus+ Programme of the European Union

Agenda

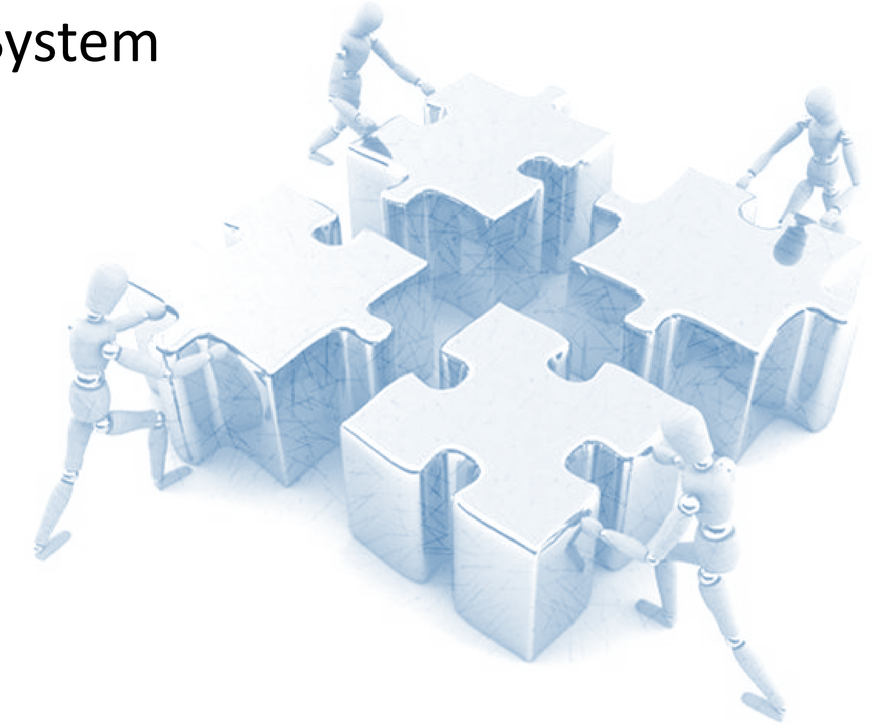
- Simulating Scrum – A practical example
- Modeling a Use Case Diagram
- Modeling an Activity Diagram
- References

Co-funded by the
Erasmus+ Programme
of the European Union

Learning objectives

- Modeling a Use Case Diagram
  - What is a use case diagram, what is it used for and how is it modeled?

  → Practical application: Case Study - Elevator System



Task 1: Defining use cases

CASE STUDY: ELEVATOR SYSTEM – USE CASE DIAGRAM MODELING

- You have been contracted to design a prestigious building project – the tallest skyscraper in the world! One of your tasks is to design the high-speed elevator system for the skyscraper. This involves carrying out an analysis of the elevator system before finalizing the design.
- During your interaction with various stakeholders (customer, end-user etc.), you have been able to elicit a list of concerns. The following are examples from the list:
    - Passengers want to be transported quickly and safely.
    - The elevator system shall be able to pick up passengers travelling in the same direction.
    - The system shall make an “emergency stop” (at nearest floor), in case of an emergency (e.g. fire alarm in the building).
    - The elevator system shall have a dedicated function to prepare the system for maintenance. This function can only be initiated by authorized maintenance personnel.
    - There is demand for a “VIP version” of the elevator system, i.e. the standard functionality shall be extended by additional “VIP features” – for example, VIP passengers have priority and can override transportation commands set by others.
- You have already drawn a Context Diagram to show the interaction of the elevator system with entities from its environment (see slide “Additional Material: Context Diagram”).

Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

Task 1: Defining use cases

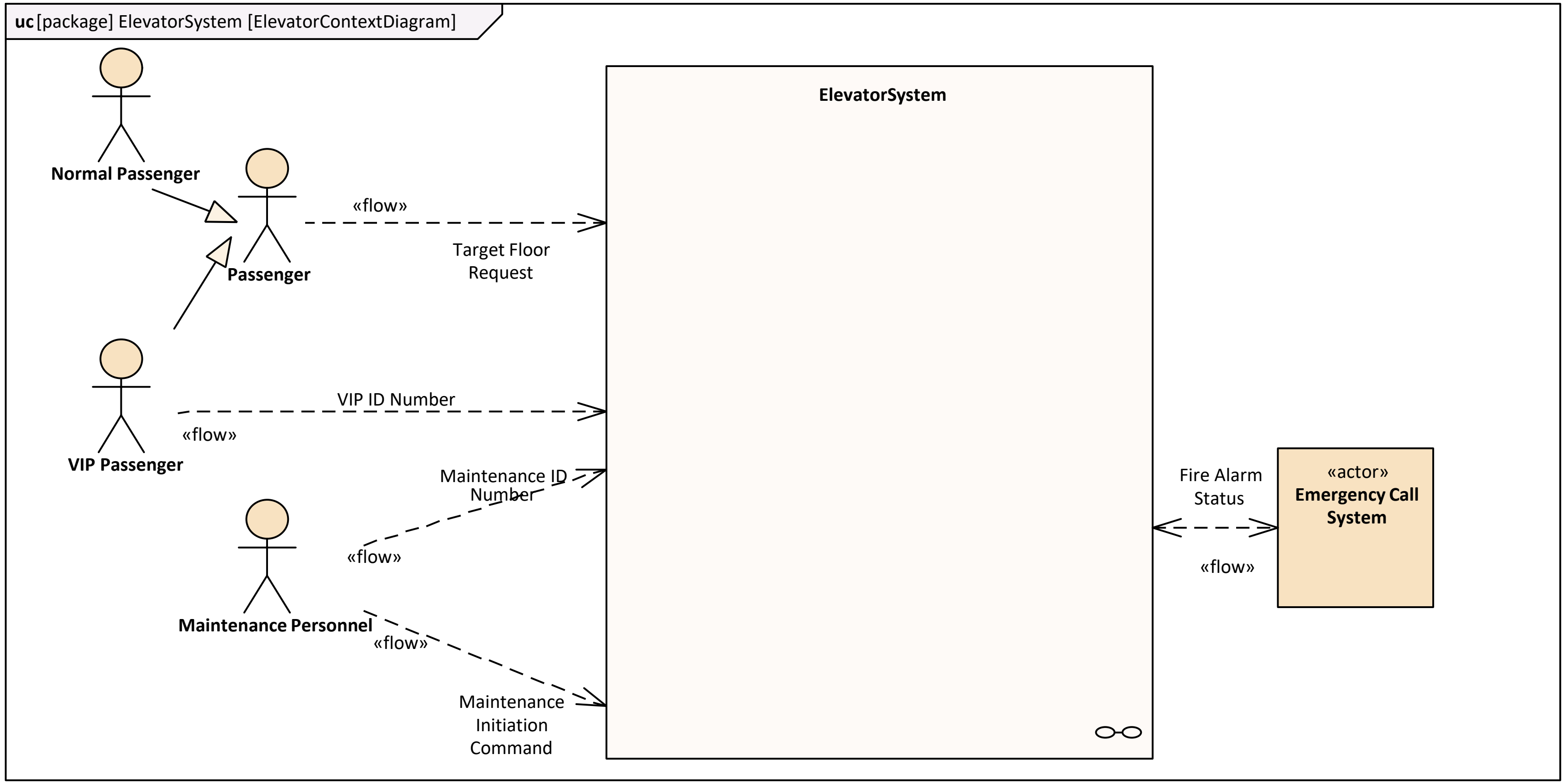
CASE STUDY: ELEVATOR SYSTEM – USE CASE DIAGRAM MODELING

- **Task:** Model a Use Case (UC) Diagram that shows the system, actors and the use-cases the system must fulfill.
  - Identify the system boundary and the actors for the UC Diagram. Refer to the Context Diagram for guidance.
  - Identify use cases based on the stakeholder concerns mentioned above.

Co-funded by the Erasmus+ Programme of the European Union

Additional material: Context diagram

CASE STUDY: ELEVATOR SYSTEM – USE CASE DIAGRAM MODELING

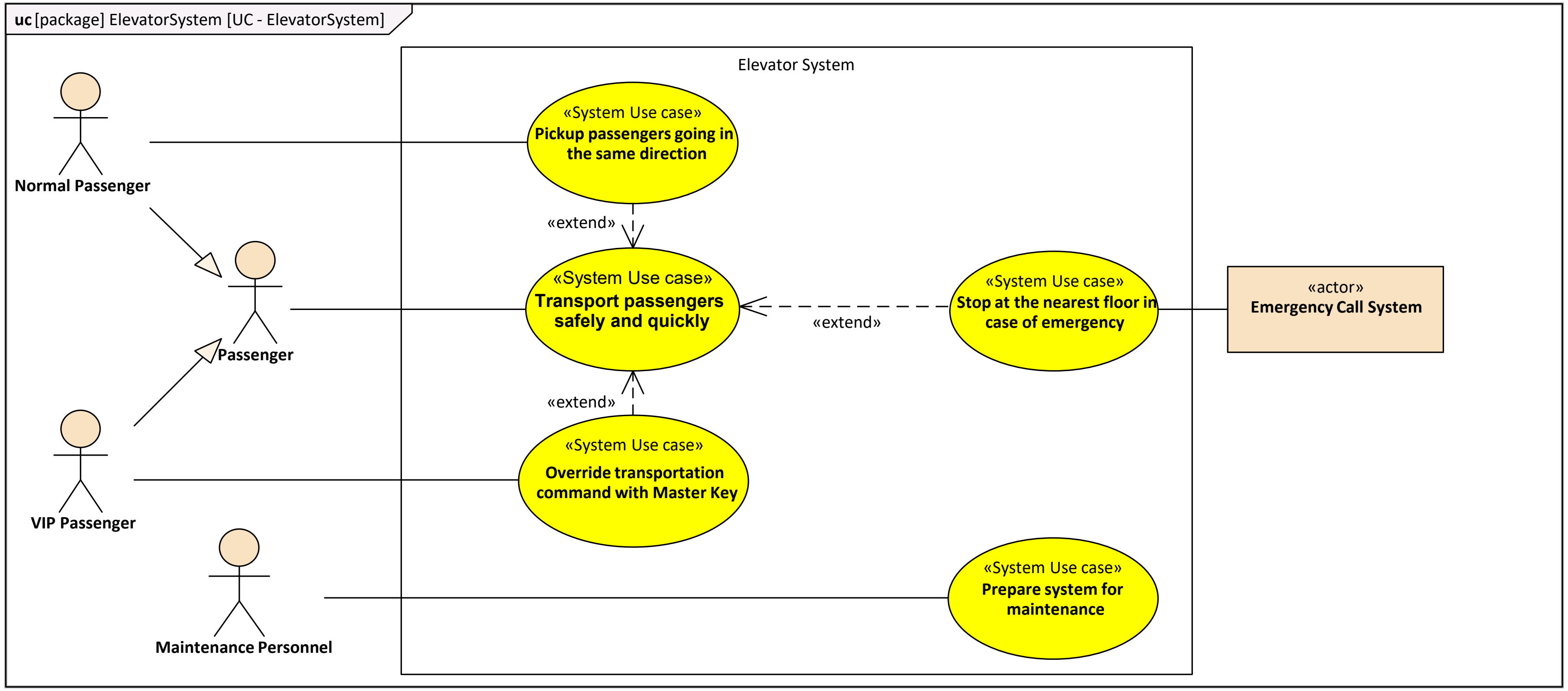


Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

Solution: Use Case (UC) Diagram

CASE STUDY: ELEVATOR SYSTEM – USE CASE DIAGRAM MODELING



Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

Agenda

- Simulating Scrum – A practical example
- Modeling a Use Case Diagram
- Modeling an Activity Diagram
- References

Co-funded by the
Erasmus+ Programme
of the European Union

Learning objectives

- Modeling an Activity Diagram
  - What is a activity diagram, what is it used for and how is it modeled?
  - → Practical application: Case Study - Elevator System

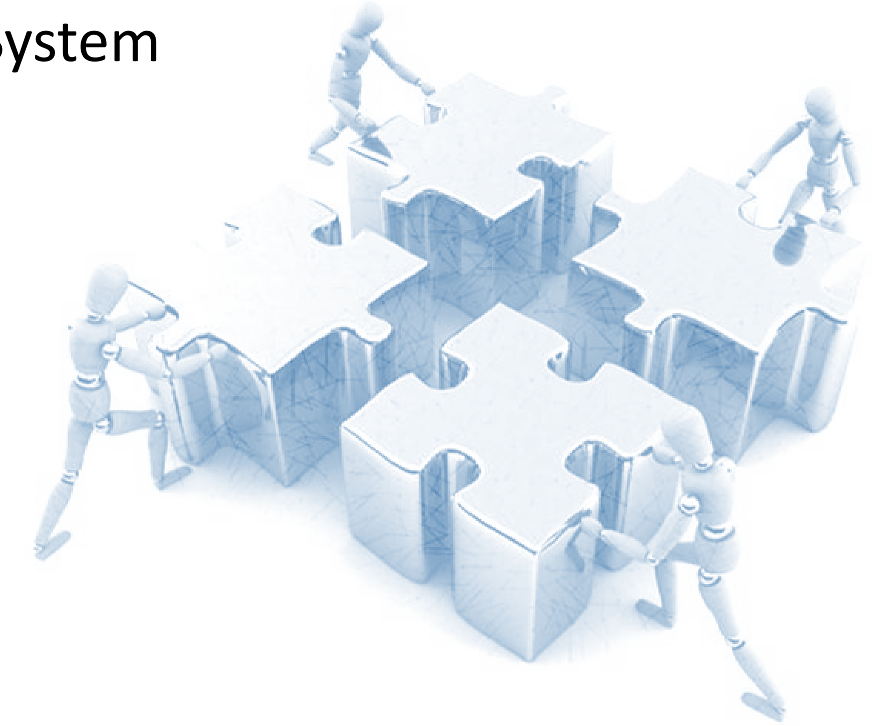


Image: Jasmin Hamadeh (http://www.jasmin-hamadeh.de/wp-content/uploads/2013/03/team_puzzle_clipart.jpg), https://creativecommons.org/licenses/by-nd/3.0/deed.de

Co-funded by the Erasmus+ Programme of the European Union

Task 2.1: Defining functional behavior

CASE STUDY: ELEVATOR SYSTEM – ACTIVITY DIAGRAM MODELING

- For the next step, you have to analyze the system behavior with respect to the function:
  - Deactivate system for maintenance.
- Function summary and context:
  - The function shall fulfill the use cases “Prepare system for maintenance” and “Override transportation command with master key”.
  - The maintenance personnel is equipped with a Maintenance Key (e.g. RFID card) to identify themselves. The key contains a unique ID number.
  - The personnel must press a button to send a Maintenance Initiation Command to initiate the function.
  - After the elevator system has confirmed their identity and received the command, it shall perform a set of actions to bring the elevator to safety so that maintenance can be carried out.
  - The elevator system shall indicate that it is in maintenance using a dedicated warning lamp.

Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

Task 2.1: Defining functional behavior

CASE STUDY: ELEVATOR SYSTEM – ACTIVITY DIAGRAM MODELING

- After careful analysis, you have defined the following system behavior if the function is carried out:
  1. Before deactivation, the elevator shall check for the Maintenance ID Number and Maintenance Initiation Command. After confirmation, a Transportation Command is sent to transport the elevator to the ground floor.
  2. After being transported to the ground floor, the elevator system shall open its doors. A status signal is sent indicating that the doors are open. Simultaneously, a Warning Lamp Activation Request is sent out.
  3. After the doors are open, the elevator system shall activate a Maintenance Mode. The elevator system shall communicate the Maintenance Mode Status. For the sake of simplicity, this action will not be detailed further for workshop purposes.
  4. After Maintenance Mode is active and the Warning Lamp Activation Request is received, the system shall switch on a Warning Lamp to indicate visually that it is in Maintenance Mode. During this time, the maintenance personnel shall perform their duties.

Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

Task 2.1: Defining functional behavior

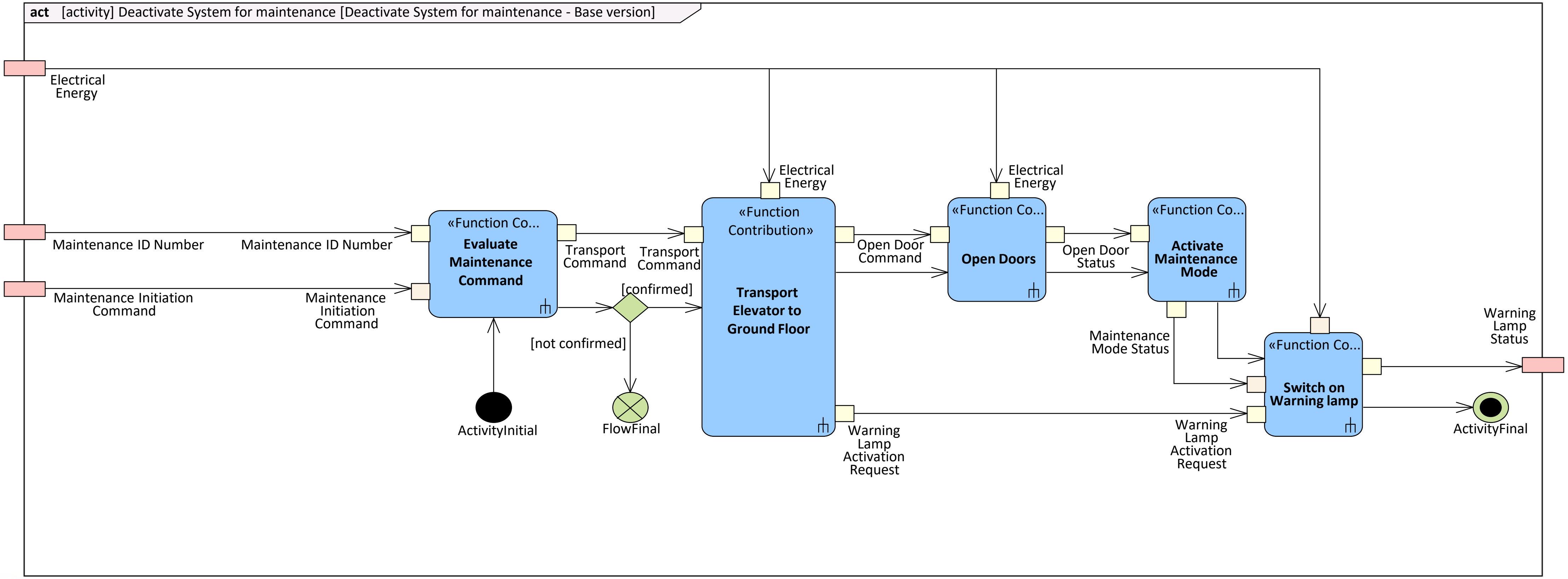
CASE STUDY: ELEVATOR SYSTEM – ACTIVITY DIAGRAM MODELING

- **Task:** Perform the following tasks:
  - Model an Activity Diagram showing the system behavior for the above-mentioned function. Refer to the description of the functional behavior. Consider the following aspects during modelling:
    - What are the sub-functionalities? → Actions
    - What information is exchanged? → Object flows
    - What is the execution order and which decisions are taken? → Control flows

Co-funded by the Erasmus+ Programme of the European Union

Solution 2.1: Activity Diagram

CASE STUDY: ELEVATOR SYSTEM – ACTIVITY DIAGRAM MODELING



Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

Task 2.2 (Optional): Defining functional behavior

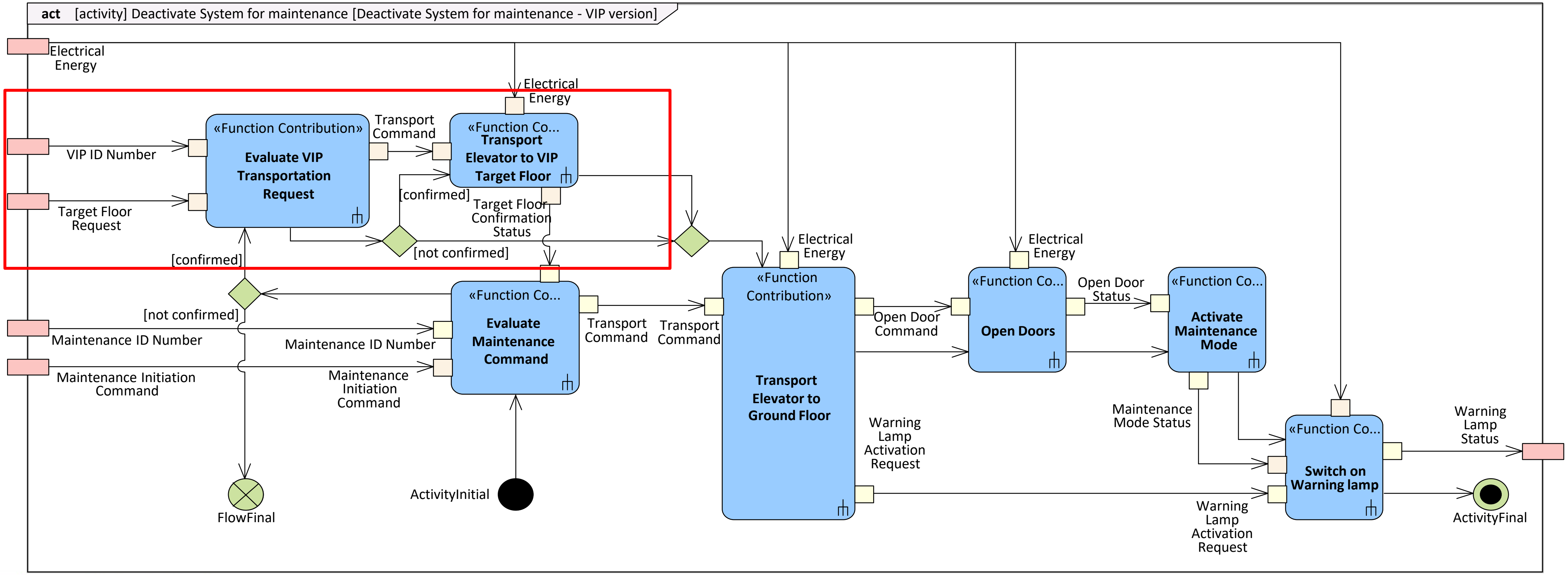
CASE STUDY: ELEVATOR SYSTEM – ACTIVITY DIAGRAM MODELING

- **VIP feature:** For the sake of additional comfort, VIP passengers already inside the elevator are given priority.
  - A VIP passenger also has a special VIP Priority Key with a unique ID number.
  - The elevator system shall check for an existing VIP command (with the help of the VIP ID number). If the command is confirmed, the VIP has priority and is transported to the target floor.
  - After transportation of the VIP, a Confirmation Status shall be sent and the elevator system shall perform the basic maintenance function.

Co-funded by the Erasmus+ Programme of the European Union

Solution 2.2: Activity Diagram – VIP Feature

CASE STUDY: ELEVATOR SYSTEM – ACTIVITY DIAGRAM MODELING



Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

Task 3: Requirements specification

CASE STUDY: ELEVATOR SYSTEM – REQUIREMENTS SPECIFICATION

- **Task:** Specify requirements regarding the activities **Evaluate VIP Transportation Request** and **Transport Elevator to Ground Floor**.
    - Specify requirements explicitly for the functional behavior of the activities.
    - Do not focus on non-functional aspects.
    - Please consider the restrictions from the previous tasks.

Co-funded by the Erasmus+ Programme of the European Union

Solution 3: Requirements specification

CASE STUDY: ELEVATOR SYSTEM – REQUIREMENTS SPECIFICATION

- **Exemplary functional requirements**
  - **Evaluate VIP Transportation Request**
    - If VIP ID Number is verified and Target Floor Request is valid, the elevator system shall send Transport Command to Floor {Target Floor}.
  - **Transport Elevator to Ground Floor**
    - If the Transport Command is valid, the elevator system shall transport the elevator cabin to Ground Floor.
    - If the elevator cabin has reached Ground Floor, the elevator system shall send an Open Door Command.
    - If the elevator cabin has reached Ground Floor, the elevator system shall send a Warning Lamp Activation Signal.

Co-funded by the Erasmus+ Programme of the European Union

Source: FEV Europe GmbH

Agenda

- Simulating Scrum – A practical example
- Modeling a Use Case Diagram
- Modeling an Activity Diagram
- References

Co-funded by the
Erasmus+ Programme
of the European Union

References

[1] Indicated parts of this presentation have been licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Germany license (CC BY-NC-SA 3.0 DE - http://creativecommons.org/licenses/by-nc-sa/3.0/de/deed.en).
Author: agile42 GmbH.

Co-funded by the
Erasmus+ Programme
of the European Union

Engineering Knowledge Transfer Units to Increase Student´s Employability and Regional Development

https://www.facebook.com/unitederasmus/

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
598710-EPP-1-2018-1-AT-EPPKA2-CBHE-JP

UNITED

Engineering Knowledge Transfer Units to Increase Student´s Employability and Regional Development

Verification and validation

Day 2 – Slot 4

Christian Granrath, M.Sc.

FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
598710-EPP-1-2018-1-AT-EPPKA2-CBHE-JP

Agenda

- Motivation
- Terminology
- Approaches for software evaluation and testing
- Test management
- References

Co-funded by the
Erasmus+ Programme
of the European Union

Learning objectives

- Motivation
  - Why are software errors critical? What types of software errors might occur? How can we locate and prevent software errors?

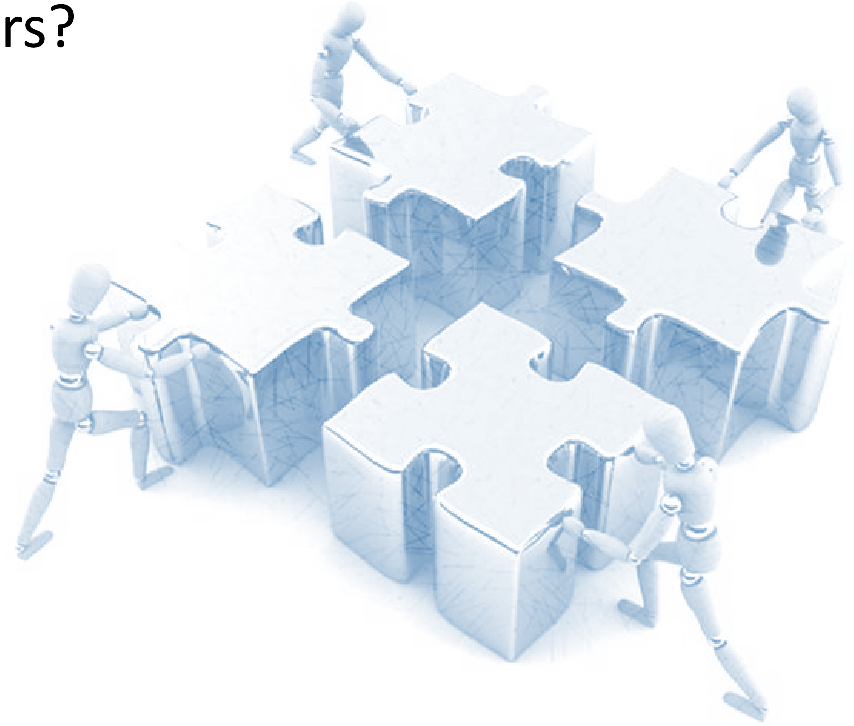


Image: Jasmin Hamadeh (http://www.jasmin-hamadeh.de/wp-content/uploads/2013/03/team_puzzle_clipart.jpg), https://creativecommons.org/licenses/by-nd/3.0/deed.de

Problem with fuel pump system

CANADIAN AIR FUEL LEAK, 2002

A computer problem on the Canadian Air Transat flight caused an emergency landing in the Azores last summer. Apparently, as early reports describe, **a "computer program" incorrectly reported a fuel leak as an "imbalance"**. To correct the "imbalance" the "computer program" diverted fuel from a good tank to the tank that was leaking thus both tanks were emptied. Inflight. The skill of the pilot and the availability of an island with an airport in the Atlantic Ocean averted a disaster.

What stage of software development caused this failure?

Source: *Canadian Press, "Toronto Globe and Mail", "Toronto Star" & other Canadian newspapers*

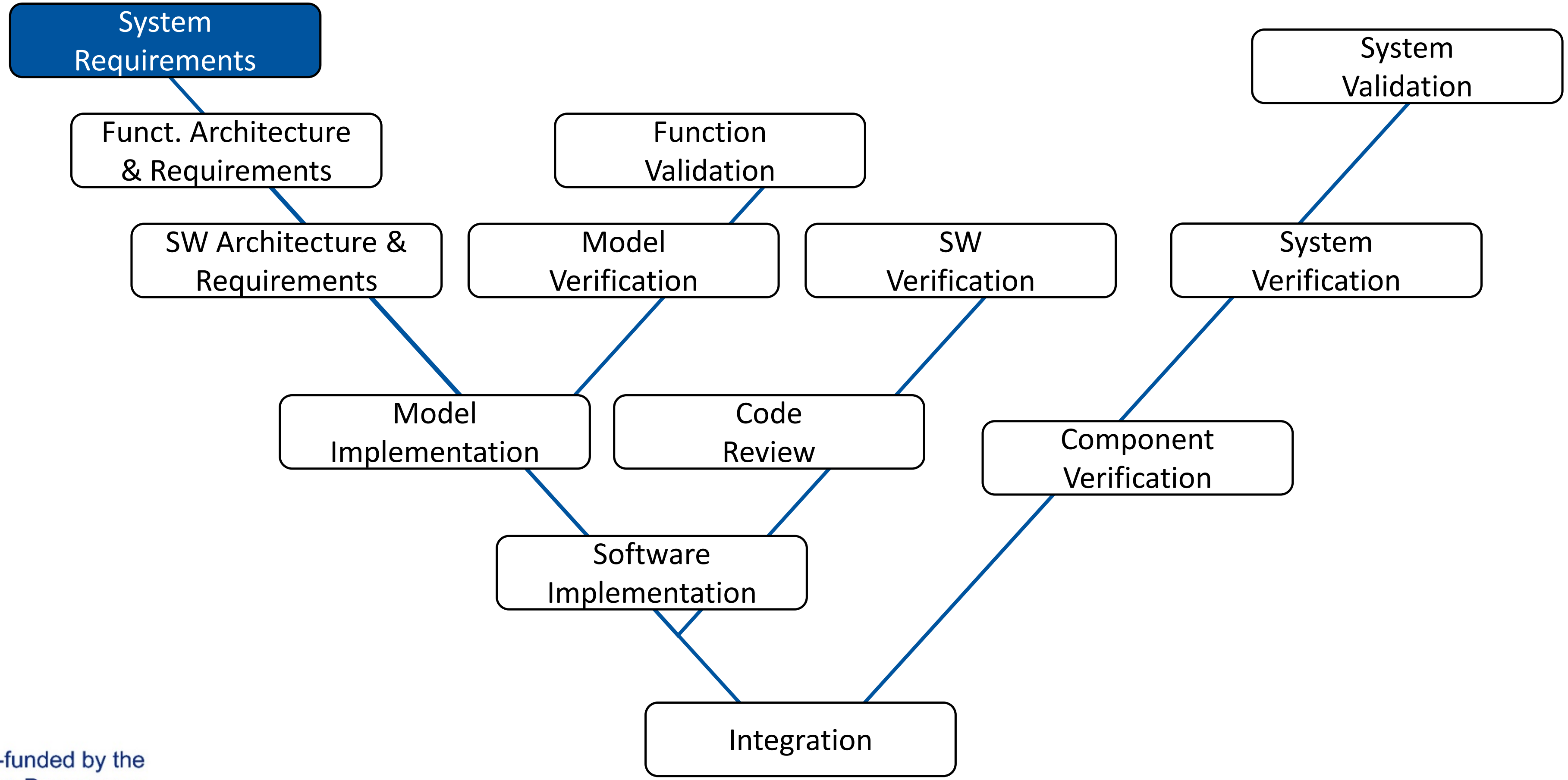
Taken from *John Johnson, Risk Forum, 4 March 2002*

Co-funded by the Erasmus+ Programme of the European Union

Source: S. Kowalewski, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [1]
Image: Shawn from Airdrie, Canada (https://commons.wikimedia.org/wiki/File:Air_Transat_A330.jpg), „Air Transat A330", Cropped by FH Aachen, https://creativecommons.org/licenses/by-sa/2.0/legalcode

Where does the software testing take place?

V-MODEL IN MODEL-DRIVEN SOFTWARE ENGINEERING



Source: S. Kowalewski, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [1]

Co-funded by the Erasmus+ Programme of the European Union

Problem with SBC Braking System

Mercedes Recalls 680,000 Models

AP

BERLIN (AP) _ DaimlerChrysler AG's Mercedes-Benz unit said Tuesday it is recalling some 680,000 E- and SL-class vehicles worldwide to examine potential problems with a brake control system.

Mercedes said it is asking owners of the vehicles to visit their local service centers for a precautionary check of the so-called Sensotronic Brake Control system.

The company said it was aware of a ``very small number'' of complaints, but that braking was assured by the system's additional hydraulic function.

The recall applies to E-class Limousines built after March 2002, T model cars built after March 2003 and SL-class vehicles made after October 2001, Mercedes said.

Spokesman Norbert Giesen said the recall affects about 680,000 vehicles worldwide, including 225,000 in Germany.

What stage of software development caused this failure?

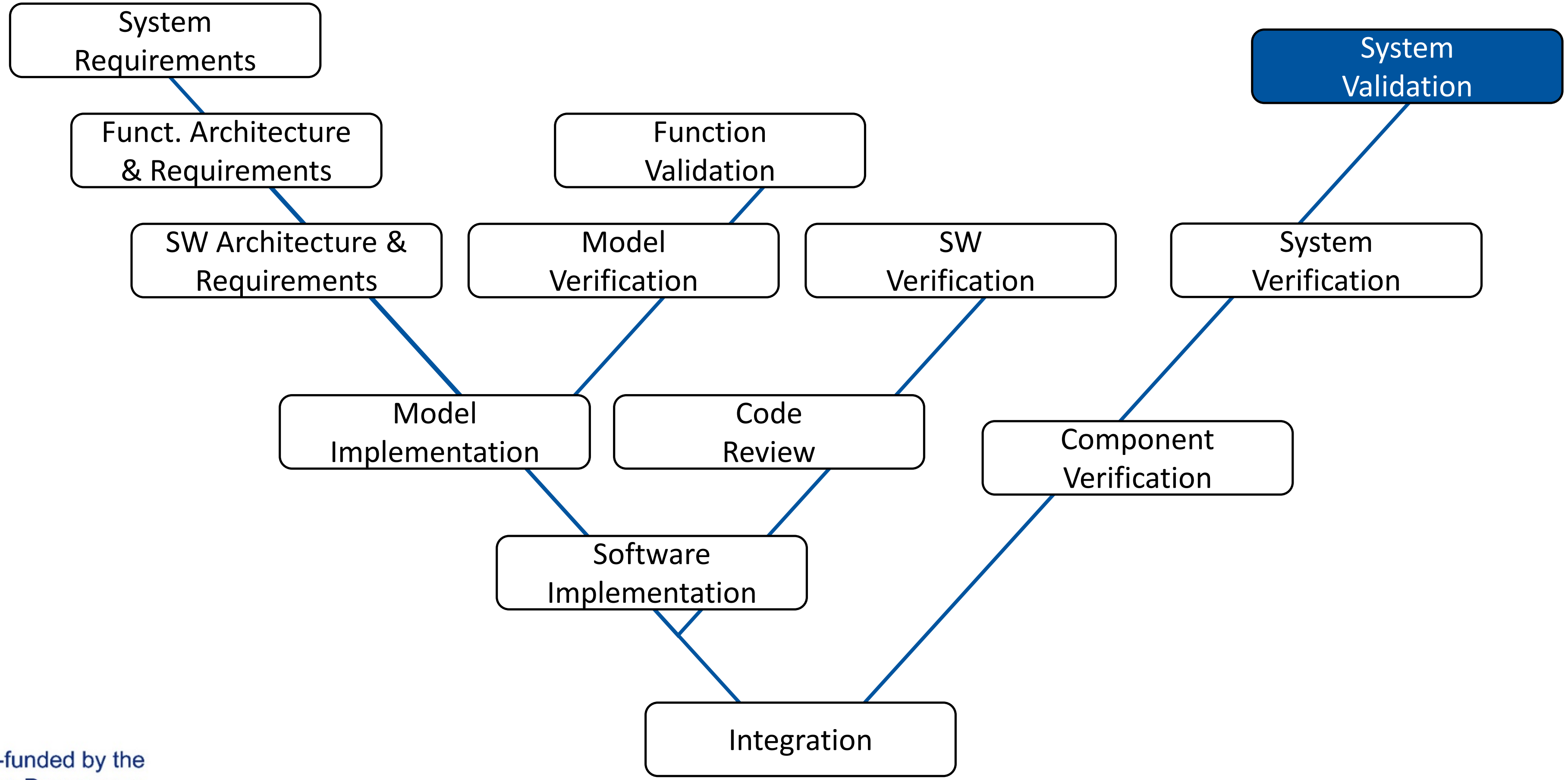
Source: The Associated Press - May 11, 2004

Co-funded by the Erasmus+ Programme of the European Union

Source: S. Kowalewski, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [1]

Where does the software testing take place?

V-MODEL IN MODEL-DRIVEN SOFTWARE ENGINEERING



Co-funded by the Erasmus+ Programme of the European Union

Source: S. Kowalewski, Lecture “Software for Combustion Engines”, RWTH Aachen University, 2019 [1]

Automotive recalls due to software flaws – a summary

	Issue	Impact
FIAT / CHRYSLER	▪ SW security flaw made cars vulnerable to hackers	▪ Recall of ~1.4M cars
	▪ SW incompatibility between EV control unit and battery control module may cause propulsion system to shut down	▪ Recall of ~5,600 Electric Vehicles
TOYOTA	▪ SW flaw may cause the hybrid system to shut down while driving	▪ Recall of 1.9M hybrid cars
	▪ SW flaw may cause VSC, ABS and traction control functions to shut down	▪ Recall of ~260,000 Vehicles
HONDA	▪ Flaw in the continuously variable automatic transmission software may subject the drive pulley shaft to high stress	▪ Recall of 143,000 cars in the US
	▪ Cell voltage sensor incorrectly interprets electrical noise and may cause a sudden loss of power	▪ Voluntarily recall of 6,786 hybrid cars
LAND ROVER	▪ SW flaw may cause vehicle doors to be unlatched	▪ Recall of 65,000 cars
	▪ Flaw in the anti-lock braking system may disable stability and control safety systems	▪ Recall of 2,687 SUVs
Mercedes-Benz	▪ Flaw in engine control unit SW may cause engine to stop while stopping at a traffic light	▪ Recall of ~3,000 cars

Co-funded by the Erasmus+ Programme of the European Union

Source: S. Kowalewski, Lecture “Software for Combustion Engines”, RWTH Aachen University, 2019 [1];
Mc Kinsey, “Numetrics R&D Analytics”, April 2016 [2]

The later the error is detected, the higher the costs

RELATIVE COSTS OF AN ERROR DEPENDING ON THE TIME OF DETECTION

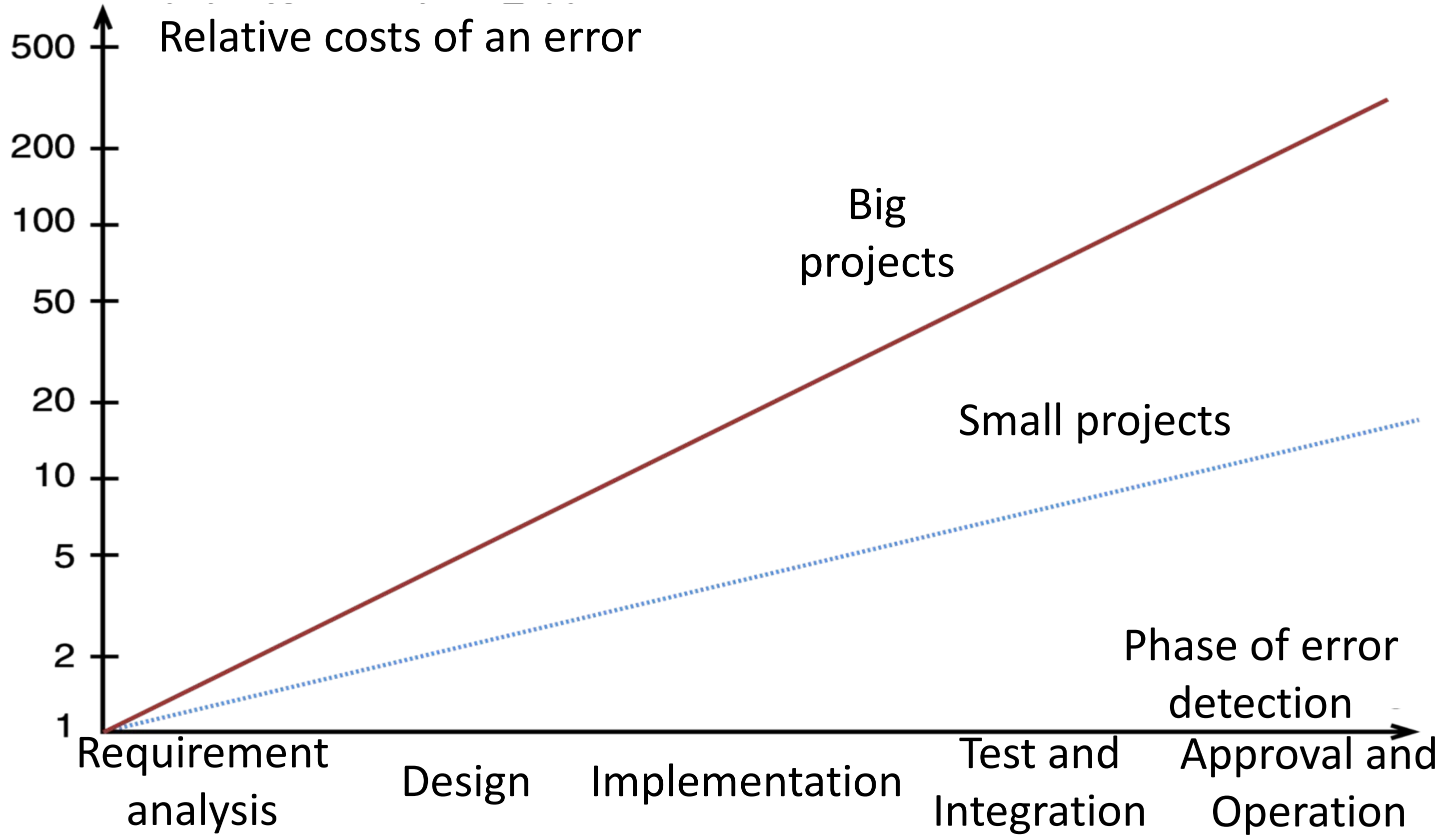


Image: Ludewig, Lichter: Software-Engineering

Co-funded by the Erasmus+ Programme of the European Union

How can we ensure software quality?

SOFTWARE DEVELOPMENT = REALIZATION OF CONSTRUCTIVE & ANALYTIC MEASURES

Constructive measures

- Quality assurance a priori (in advance)
- Examples: (Architectural, modelling) guidelines, processes/methods/standards

Analytic measures

4

- Quality assurance a posteriori (measurement, improvement)
- Examples: Screening processes (reviews, guideline examinations, testing, process reviews)

Mapping to slots

Co-funded by the Erasmus+ Programme of the European Union

Source: J. Richenhagen, Lecture “Software for Combustion Engines”, RWTH Aachen University, 2019 [3]

Agenda

- Motivation
- Terminology
- Approaches for software evaluation and testing
- Test management
- References

Co-funded by the
Erasmus+ Programme
of the European Union

Learning objectives

- Terminology
  - What do terms like software quality, quality assurance, testing, verification and validation mean?

Co-funded by the Erasmus+ Programme of the European Union

Image: Jasmin Hamadeh (http://www.jasmin-hamadeh.de/wp-content/uploads/2013/03/team_puzzle_clipart.jpg), https://creativecommons.org/licenses/by-nd/3.0/deed.de

Functional vs. non-functional attributes

- **Functional** attributes describe the elementary purpose of a system
  - Example: Adaptive Cruise Control - Adjustment of vehicle speed and distance from vehicles ahead, as well as switching between both modes
- **Non-functional** attributes describe features important for the economic success of a product, required in addition to accurate functionality.
  - Examples:
    - Limited response time for switching in distance control
    - Time of market introduction
    - Reusability of the software for future product generations

Functional attributes
- high performance / fast reaction time
- high reliability

vs.

Non-functional attributes
- Simple portability to other hardware systems
- Short time-to-market

Co-funded by the Erasmus+ Programme of the European Union

Source: S. Kowalewski, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [1]

Quality assurance and software evaluation

IEEE 610.12 STANDARD GLOSSARY OF SOFTWARE ENGINEERING TERMINOLOGY

- Quality Assurance: **... A planned and systematic pattern of all activities necessary to provide adequate confidence that an item conforms to established technical requirements.**
- Software evaluation: **Analysis of the quality of a given test object, i.e. if the test object is "qualified to fulfill given requirements"**
  → Prerequisite for a software evaluation is the presence of a **test object** and a **reference object** that can be checked against each other

Co-funded by the Erasmus+ Programme of the European Union

Source: IEEE 610.12-1990, "IEEE Standard Glossary of Software Engineering Terminology", Institute of Electrical and Electronics Engineers, New York [4];
S. Kowalewski, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [1]

Example pairs of test and reference objects

Test object	Reference object	
Overall system requirements (requirement specification document)	Customer expectations	virtually not recordable
Specification (functional specification document)	Consistency, integrity, ...	generic, not tied to product
Architecture	Qualification to support business objectives (e.g. reliability)	possibly covering several products, product lines
Model	Functional requirements	product-specific
Handwritten code	Coding guidelines, metrics, ...	possibly covering several products, company-specific
Generated code	Behavior of the corresponding model	product-specific
Modified code	Behavior of the code in its last executable version	product-specific

Co-funded by the Erasmus+ Programme of the European Union

Source: S. Kowalewski, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [1]

Validation vs. verification

- **Validation:**
  Make sure the test object complies with the customer's expectations
  „Are we doing the right things?"

- **Verification:**
  Prove the test object fulfills verifiable requirements
  „Are we doing the things right?"

- Another meaning:
  - Verification: formal (e.g. use techniques to prove)
  - Validation: informal (e.g. use techniques to review)

Co-funded by the Erasmus+ Programme of the European Union

Source: S. Kowalewski, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [1]

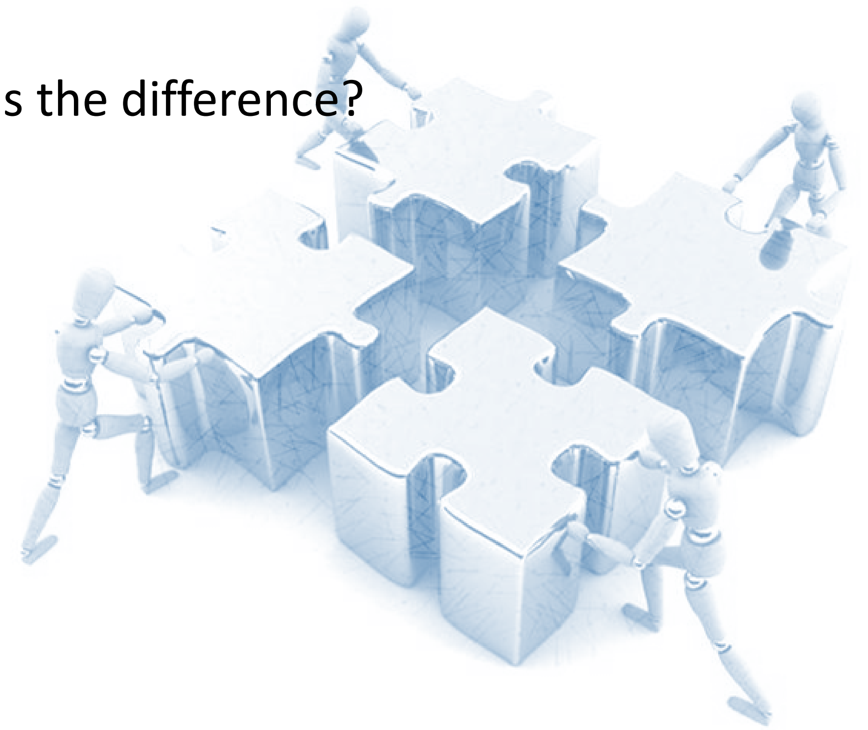
Agenda

- Motivation
- Terminology
- Approaches for software evaluation and testing
- Test management
- References

Co-funded by the
Erasmus+ Programme
of the European Union

Learning objectives

- Approaches for software evaluation and testing
  - Which types of software evaluation are available? And how do they differ?
  - What is software testing?
  - Which different testing approaches do exist and what is the difference?
  - What are future trends to reduce testing efforts?



Co-funded by the Erasmus+ Programme of the European Union

Which types of software evaluation approaches do exist?

DIFFERENT TYPES OF SOFTWARE EVALUATION

Example: Textual requirements
Example: Models
Modelled requirements

Software evaluation

By humans
"manual"

- **Reviews**
- **Scenario techniques**
- **Modelling**

By computers
"automated"

Import the testing object into an analyzer
"static"

- **Static analysis**
- **Linguistic analysis**
  - **Modelling guidelines**
  - **Metrics**
  - **Generic attributes**
- **Model checking**

Execution of the testing object
"dynamic"

- **Test (MiL, simulation)**
- **Simulation**

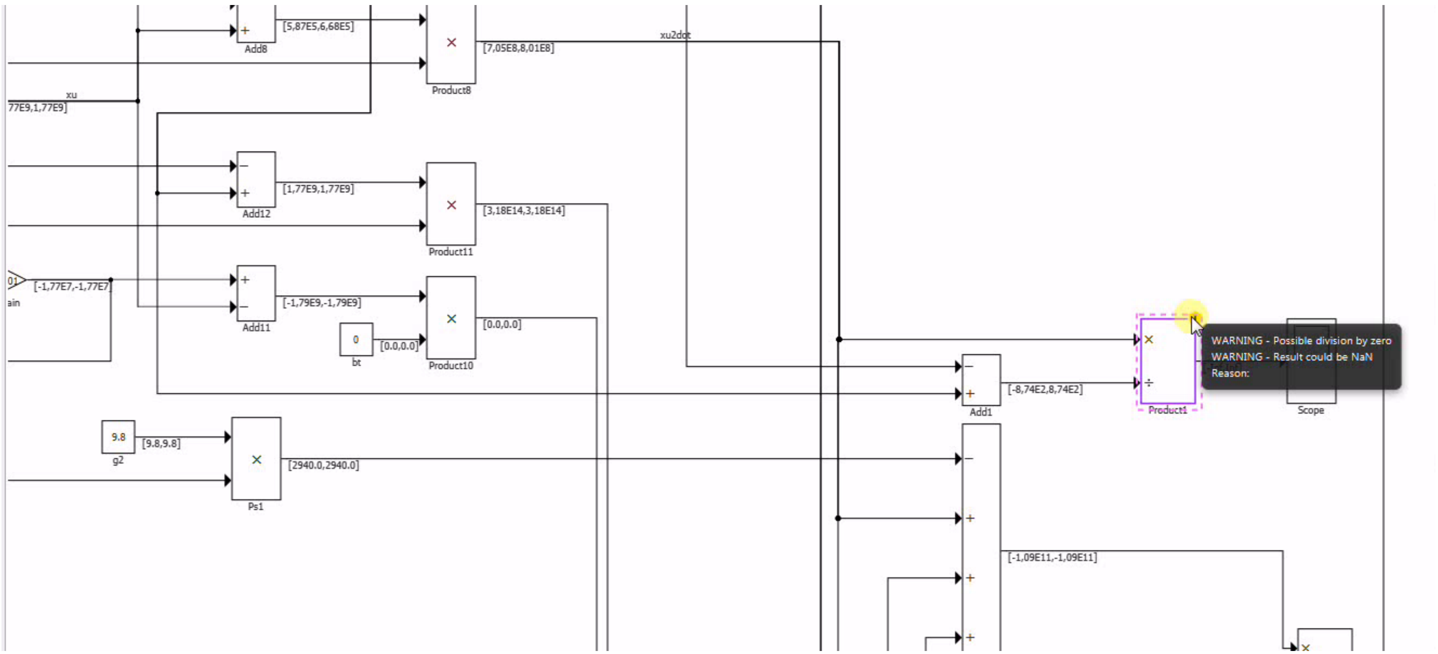
Co-funded by the Erasmus+ Programme of the European Union

Source: S. Kowalewski, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [1]

Static analysis of models is automatable

TYPICAL STATIC EVALUATIONS

- Division by zero/inf/NaN
- Data type under-/overflow
- NaN operations (asin, sqrt, etc…)
- Dead paths
- Saturation reached
- Connection of blocks
- …

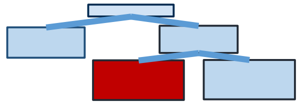


Source: S. Kowalewski, Lecture “Software for Combustion Engines”, RWTH Aachen University, 2019 [1]

Co-funded by the Erasmus+ Programme of the European Union

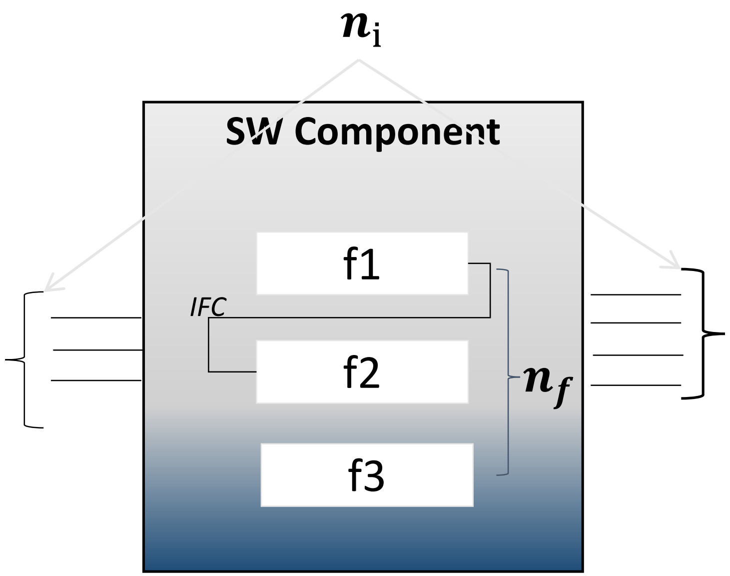
Definition of Metrics for SW Architecture

SOFTWARE ARCHITECTURE COMPLEXITY METRIC



- **Goal:** Complexity
- **Metric:**

  Software component complexity (S.C.C)
- **Formula:**
  - $SCC = log_{10}((n_f * (IFC + 1) * (n_i))$
    - n_f -> number of functions in the software component
    - *IFC*→ number of inter-functional interfaces
    - n_i -> number of external interfaces to the software component
- <u>**Note**</u>- Calibrations are treated as inputs to the component
- **Interpretation: High value of the metric →** Poor testability and maintainability

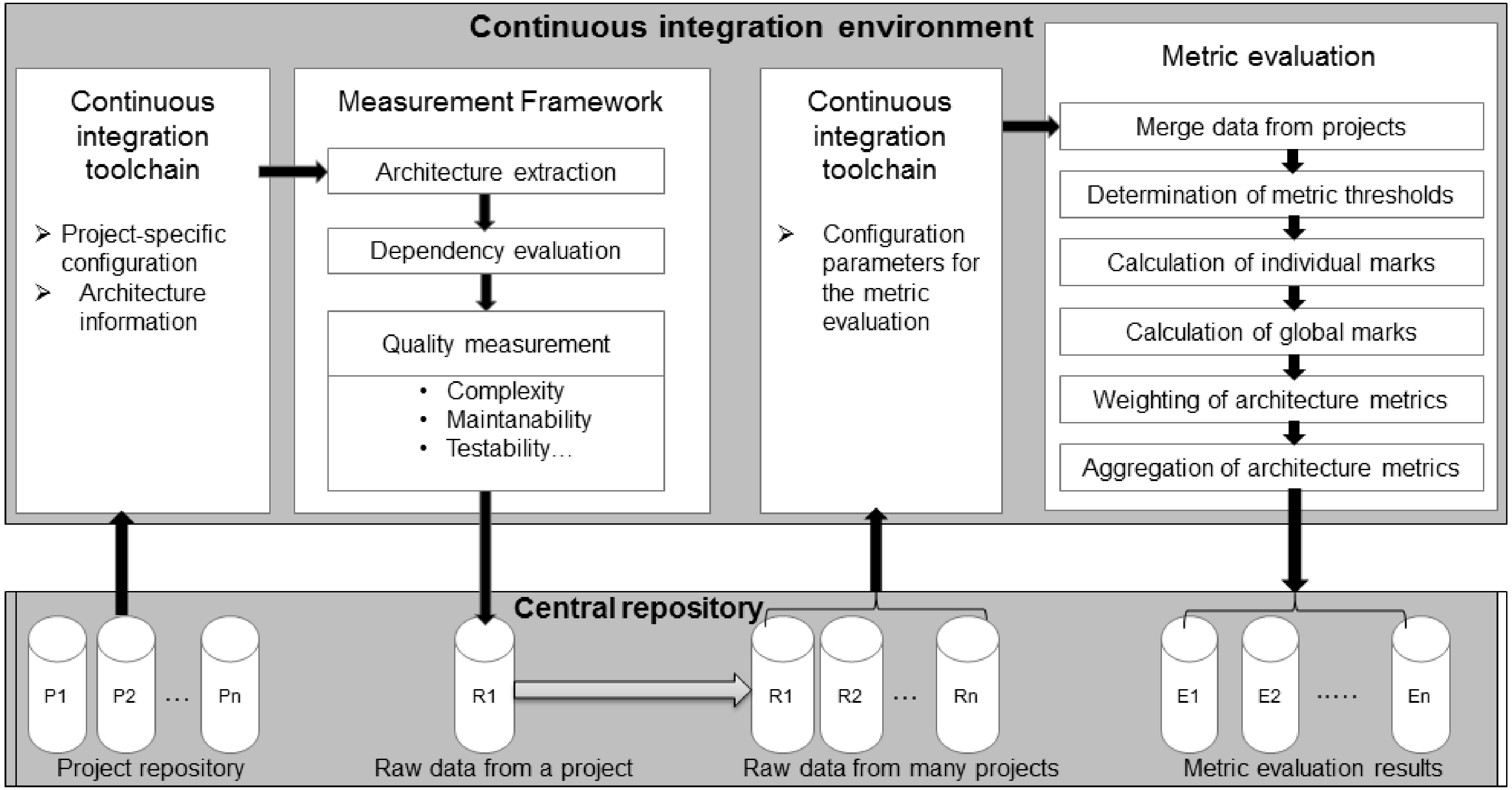


Co-funded by the Erasmus+ Programme of the European Union

Source: H. Venkitachalam, C. Granrath, G.V. Balachandar, J. Richenhagen, "Metric-based Evaluation of Hybrid Control Software", SAE World Congress, Detroit, 2017 [5]

Evaluation of measurement values for metrics mandatory

FRAMEWORK FOR THE EVALUATION OF METRICS



Co-funded by the Erasmus+ Programme of the European Union

Source: H. Venkitachalam, C. Granrath, G.V. Balachandar, J. Richenhagen, “Metric-based Evaluation of Hybrid Control Software“, SAE World Congress, Detroit, 2017 [5]

What is testing?

SOFTWARE ENGINEERING TERMINOLOGY

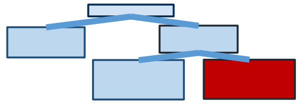
- Myers (1979):
  *"Testing is the process of **executing** a program with the **intent of finding errors**."*
  → A test is **successful** if errors are located.
- Testing is an element of quality assurance, meaning it is also targeting at building confidence in the qualification of a program to fulfill its requirements (= "there are as few errors as possible")
- **"Complete" tests are almost impossible.**
- Dijkstra:
  "Program testing can be a very effective way to show the presence of bugs, but is hopelessly inadequate for showing their absence."
  → Testing is a spot check → Planning of tests!

Co-funded by the Erasmus+ Programme of the European Union

Source: S. Kowalewski, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [1]

How do testing approaches differ?

TYPOLOGY OF TESTING



- Differentiation by **type of the required attribute:**
  - Function test, functional testing
  - Nonfunctional testing
    - Load testing, Interface testing, Memory testing, resource usage testing, Fault injection testing
- Differentiation by **involved stakeholders**:
  - Alpha testing
    - by developing organization
  - Beta testing
    - by customer, prior to finalization of the product
      (of less relevance to the automotive industry)
  - Acceptance testing
    - with the customer, upon project completion

Co-funded by the
Erasmus+ Programme
of the European Union

Source: S. Kowalewski, Lecture “Software for Combustion Engines”, RWTH Aachen University, 2019 [1]

How do testing approaches differ?

TYPOLOGY OF TESTING

- Differentiation by **reference objects**:
  - Requirements verification
    - comparison to requirements specification (PRD)
  - Back-to-back testing
    - Comparison to another executable realization of the required function
      - e.g. various programming
      - ISO 26262-relevant case: Comparison of generated code with the corresponding model
  - Regression testing
    - Comparison to previous version of the program after a limited change

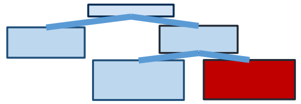
- Differentiation by **selection criteria for the catalog of test cases**:
  - ***Black-box testing*** (test cases are derived from specification by criterion of completeness)
    - Equivalence partitioning (classes for input data)
    - State-based testing (coverage of state-based specifications)
    - Use case-based testing
  - ***White-box testing*** (test cases are derived from a program's control flow by criterion of completeness)
    - Statement, branch, term, and path coverage
    - Decision, condition, decision/condition, modified decision/condition coverage
    - Function coverage, call coverage
  - ***Stochastic testing*** (test cases are generated stochastically)

Co-funded by the Erasmus+ Programme of the European Union

Source: S. Kowalewski, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [1]

How do testing approaches differ?

TYPOLOGY OF TESTING



- Differentiation by **testing infrastructure**:
  - Model-in-the-loop
  - Software-in-the-loop
  - Processor-in-the-loop
  - Hardware-in-the-loop
- Note: Testing infrastructure implies the type of the test object, e.g. model in MIL or code in SIL

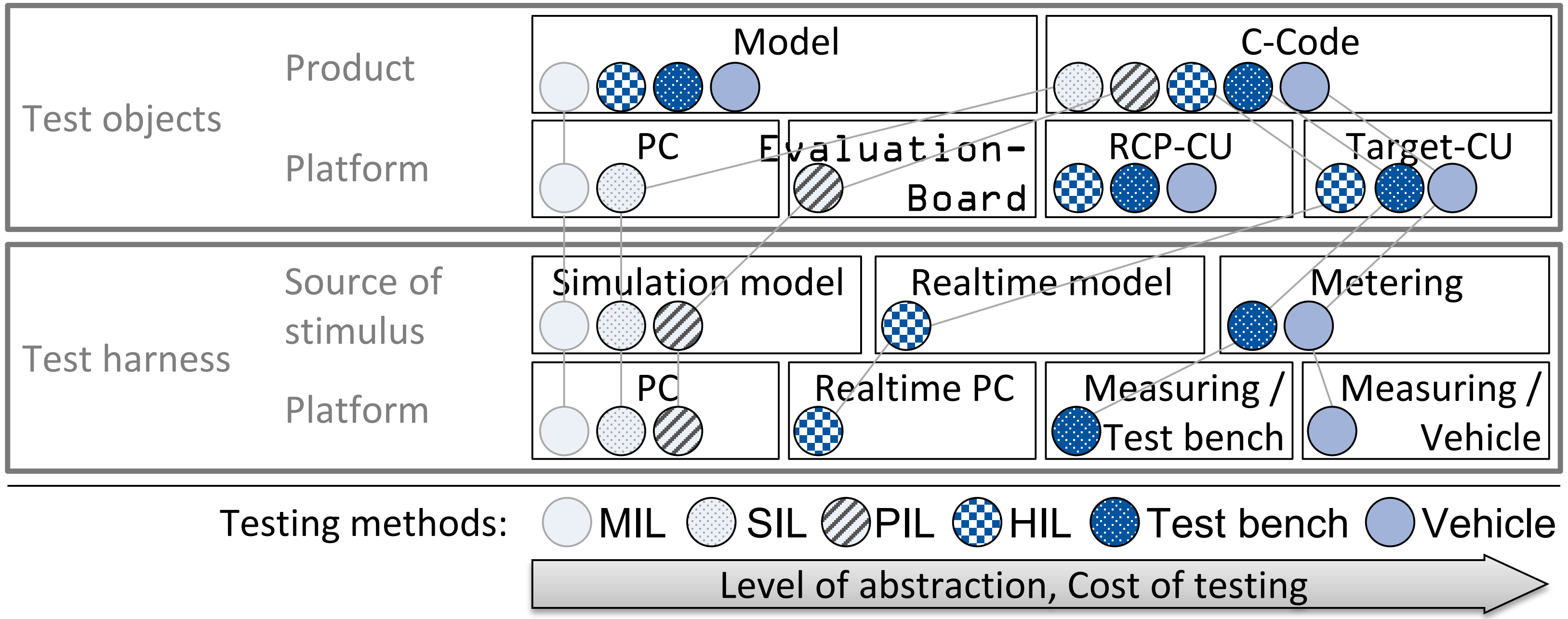
- Differentiation by **type of testing object throughout the development process** (referred to as testing stages or testing cycles):
  - Unit testing
  - (Module testing)
  - Integration testing
  - System testing
  - Field trial

Co-funded by the Erasmus+ Programme of the European Union

Source: S. Kowalewski, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [1]

Requirements-based software testing

ABSTRACTION LEVELS



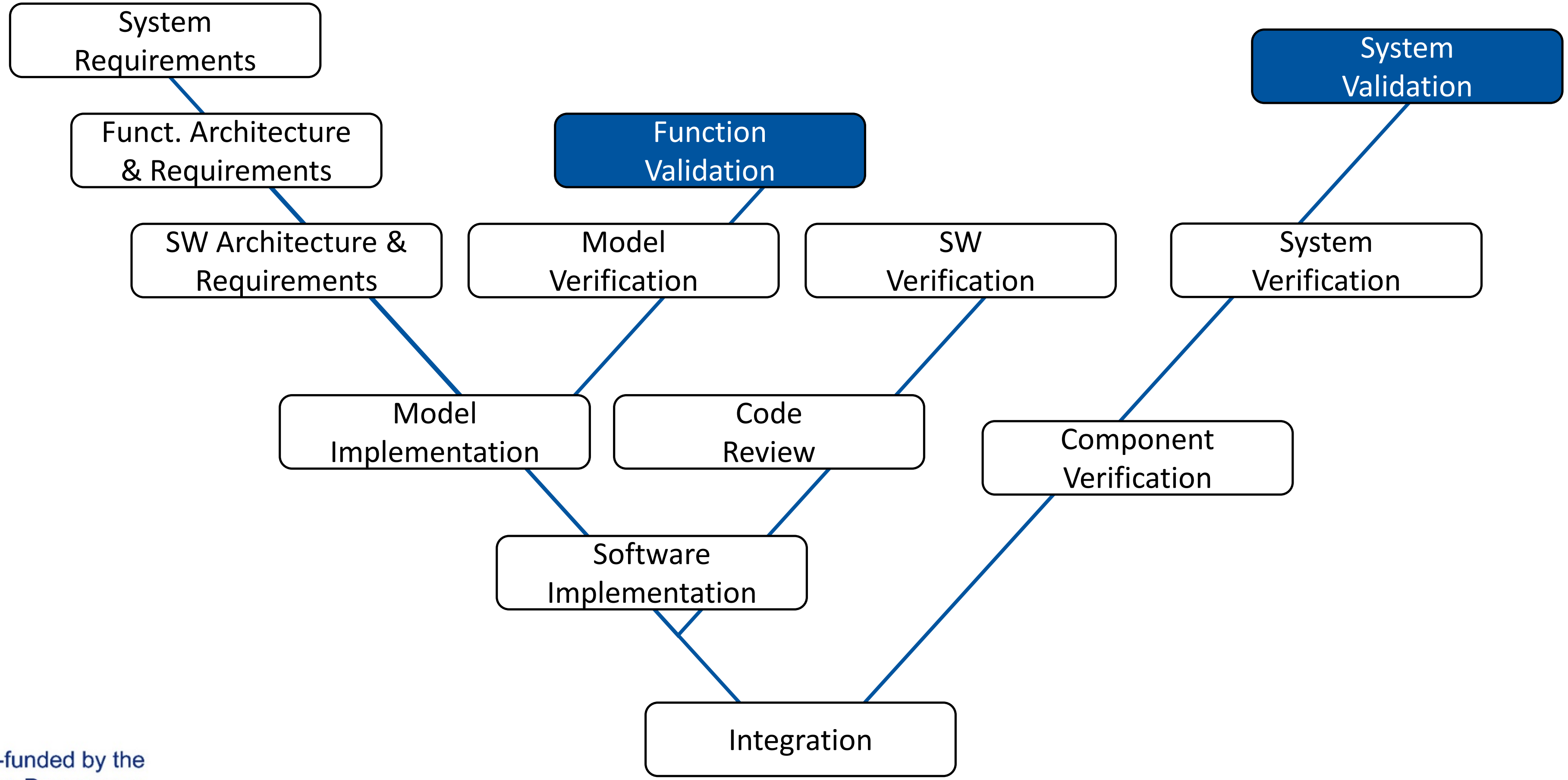
- Function tests can be executed at any stage
- Choice of tests according to project requirements
- Testing strategy and tools must be able to cover all methods

Co-funded by the Erasmus+ Programme of the European Union

Source: J. Richenhagen, "Entwicklung von Steuerungs-Software für den automobilen Antriebsstrang mit agilen Methoden", Doctoral thesis, 2014 [6]

How can be software validation performed?

V-MODEL IN MODEL-DRIVEN SOFTWARE ENGINEERING

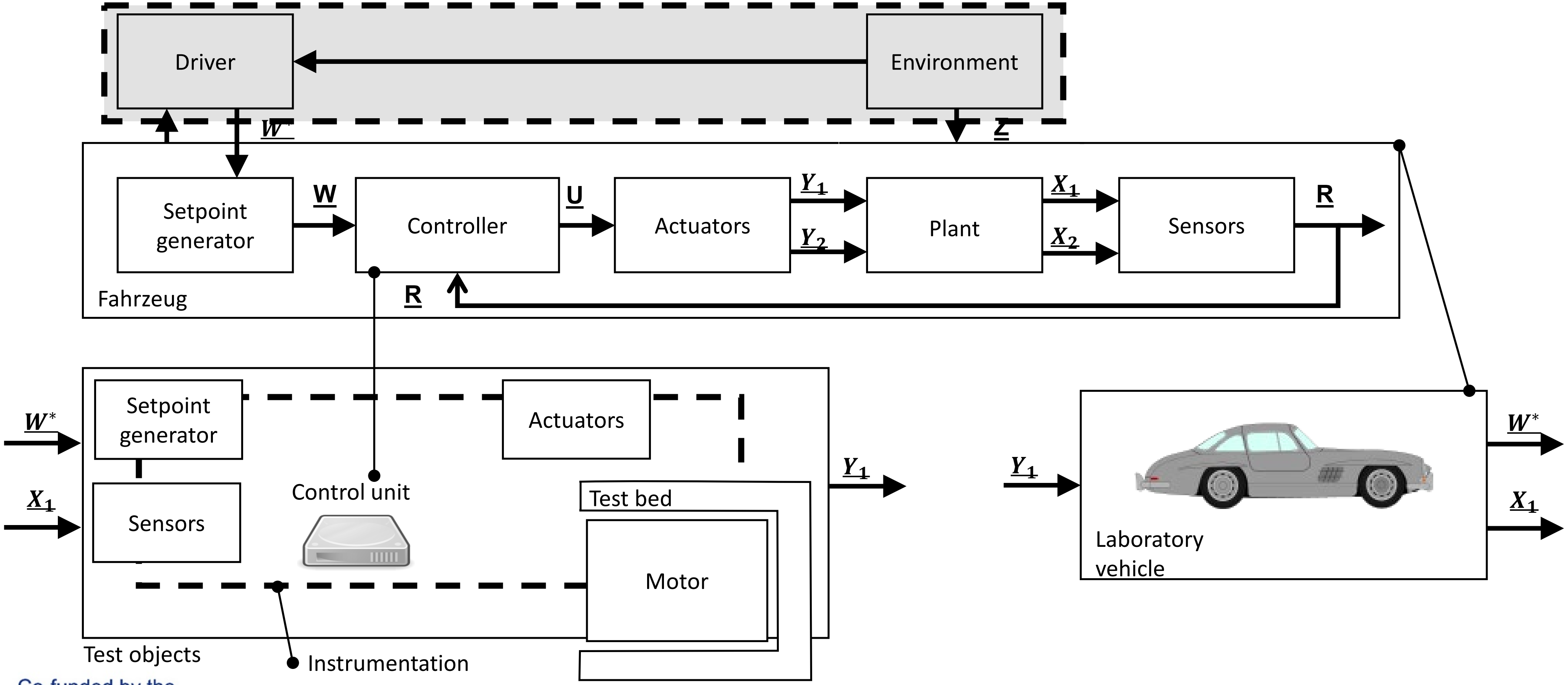


Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [3]

Co-funded by the Erasmus+ Programme of the European Union

Validation of the control unit in a virtual test-bed environment

TRACK PARTLY SIMULATED, DRIVER AND (ROUGH) ENVIRONMENT FULLY SIMULATED

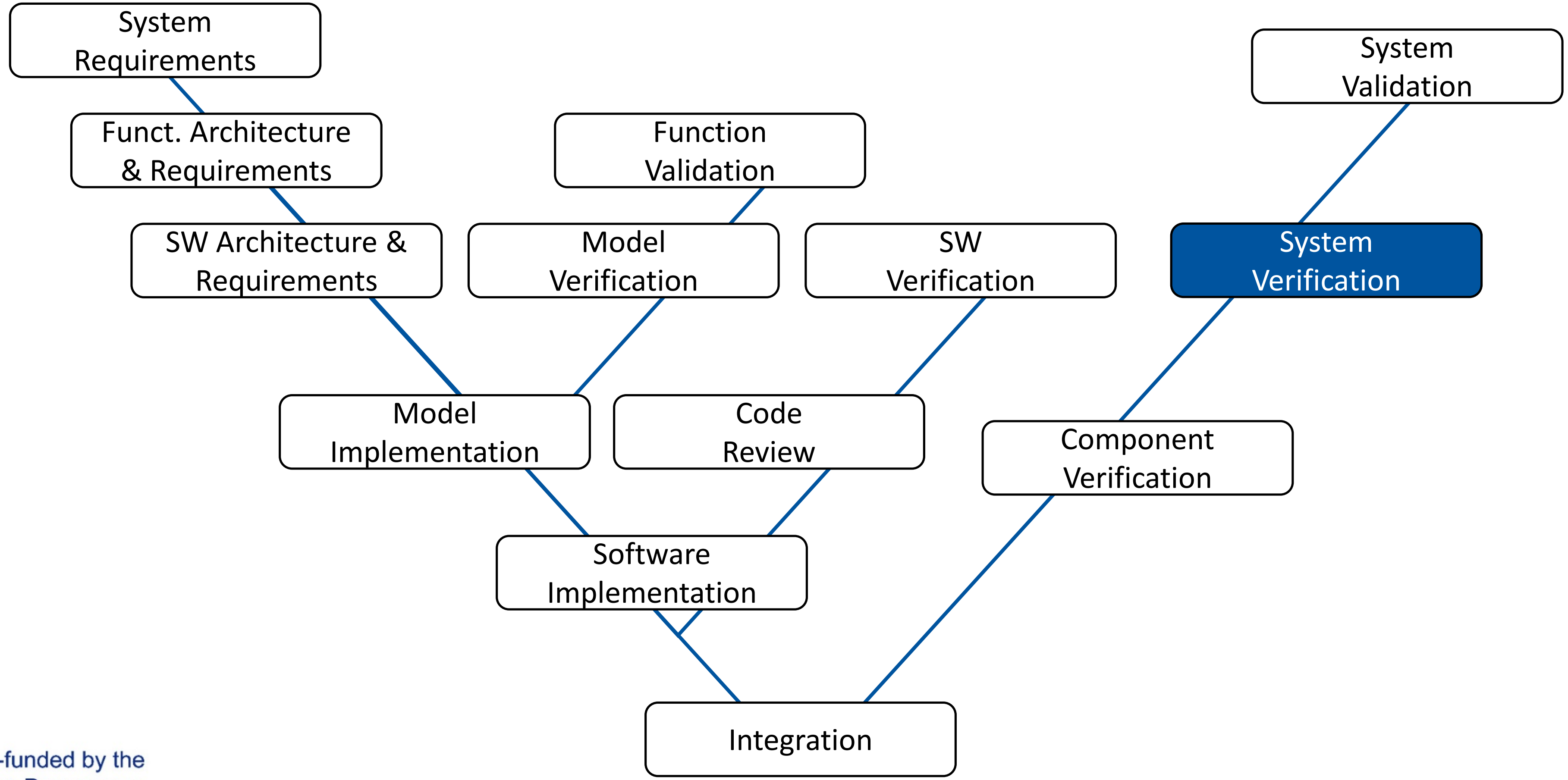


Co-funded by the Erasmus+ Programme of the European Union

Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [3]; Schäuffele and T. Zurawka, "Automotive Software Engineering: Grundlagen, Prozesse, Methoden und Werkzeuge effizient einsetzen", Vieweg+Teubner Verlag / GWV Fachverlage, 2010 [7]

How can be system verification performed?

V-MODEL IN MODEL-DRIVEN SOFTWARE ENGINEERING



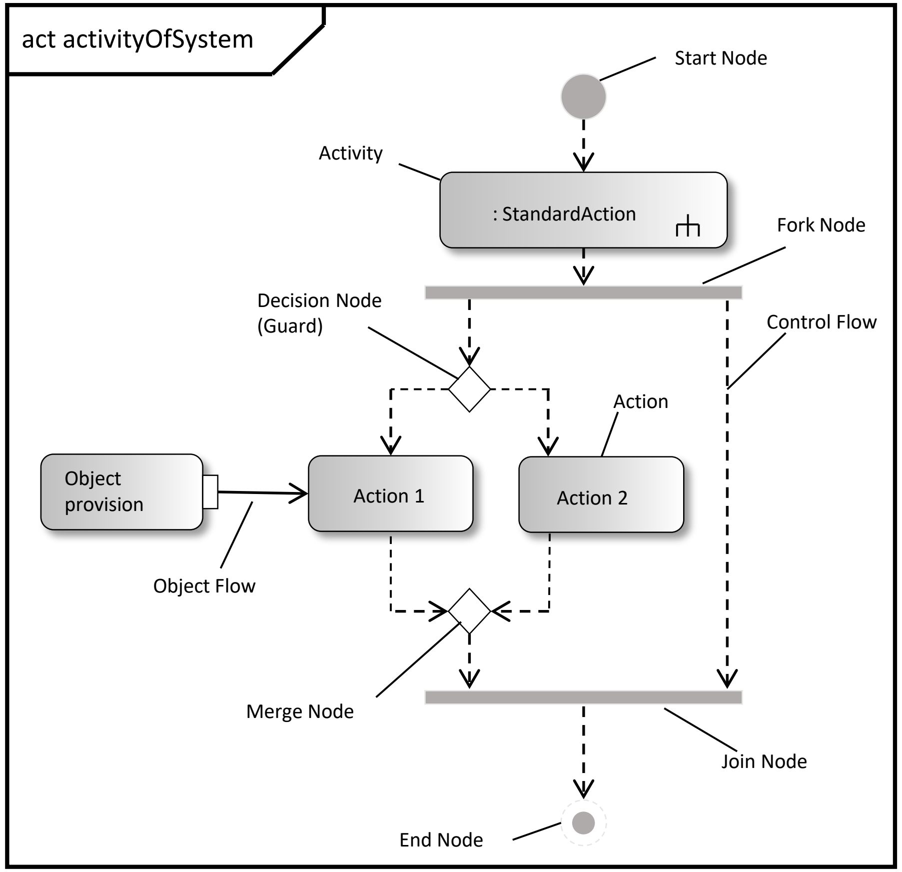
Co-funded by the Erasmus+ Programme of the European Union

Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [3]

Formalized SysML diagrams as base for test case generation

STRUCTURAL ARRANGEMENT – BEHAVIOR DIAGRAMS | ACTIVITY DIAGRAM

- Used to depict flow of control, inputs and outputs
- Control flow (dashed arrows) show order of execution
- Object flow (solid arrow) shows flow of an object (e.g. data, materials,…)

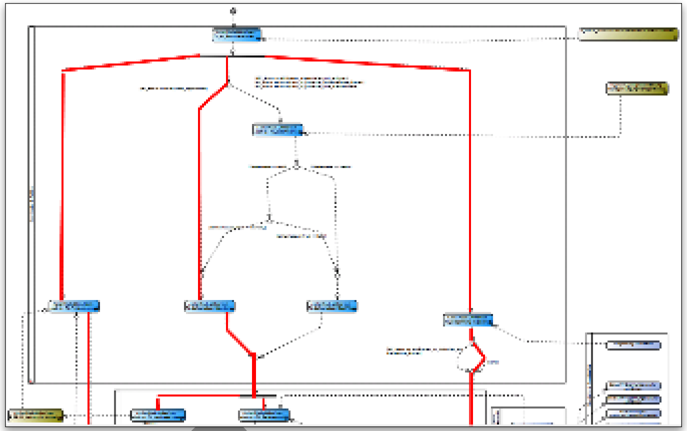


Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [3]

Results of automated test case extraction

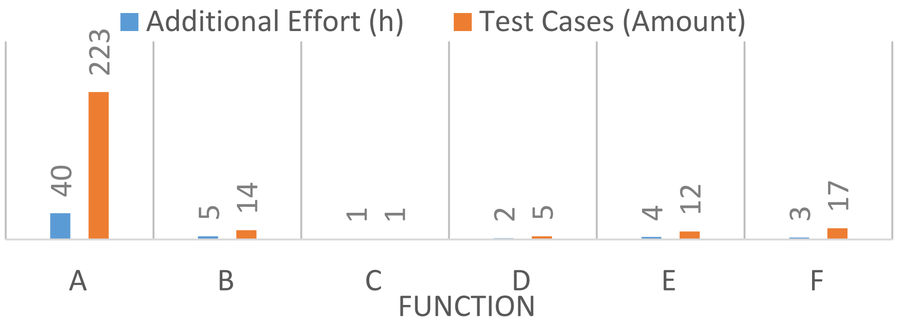
SIGNIFICANT EFFORT REDUCTION PER TEST CASE

Test depth specification



Action	Expected Value
Precondition	
Produce: HV_Status	HVSystem_im_Notbetrieb_HVBatterieloser_Betrieb
Produce: Werksmodus	aktiv
Produce: Sollbetriebsart_Werk	Buck
Produce: Sollspannung	Sollspannung_BatterieloserBetrieb
Action	
Status: Sollbetriebsart	Buck
Status: Leistungspotential_NVSpannungswandler	ermittelt
Status: Fehlerstatus	aktiv
Status: Fehlerstatus_NVSpannungswandler	aktiv
Status: Status_NV_Spannungswandler	Standby

Project results



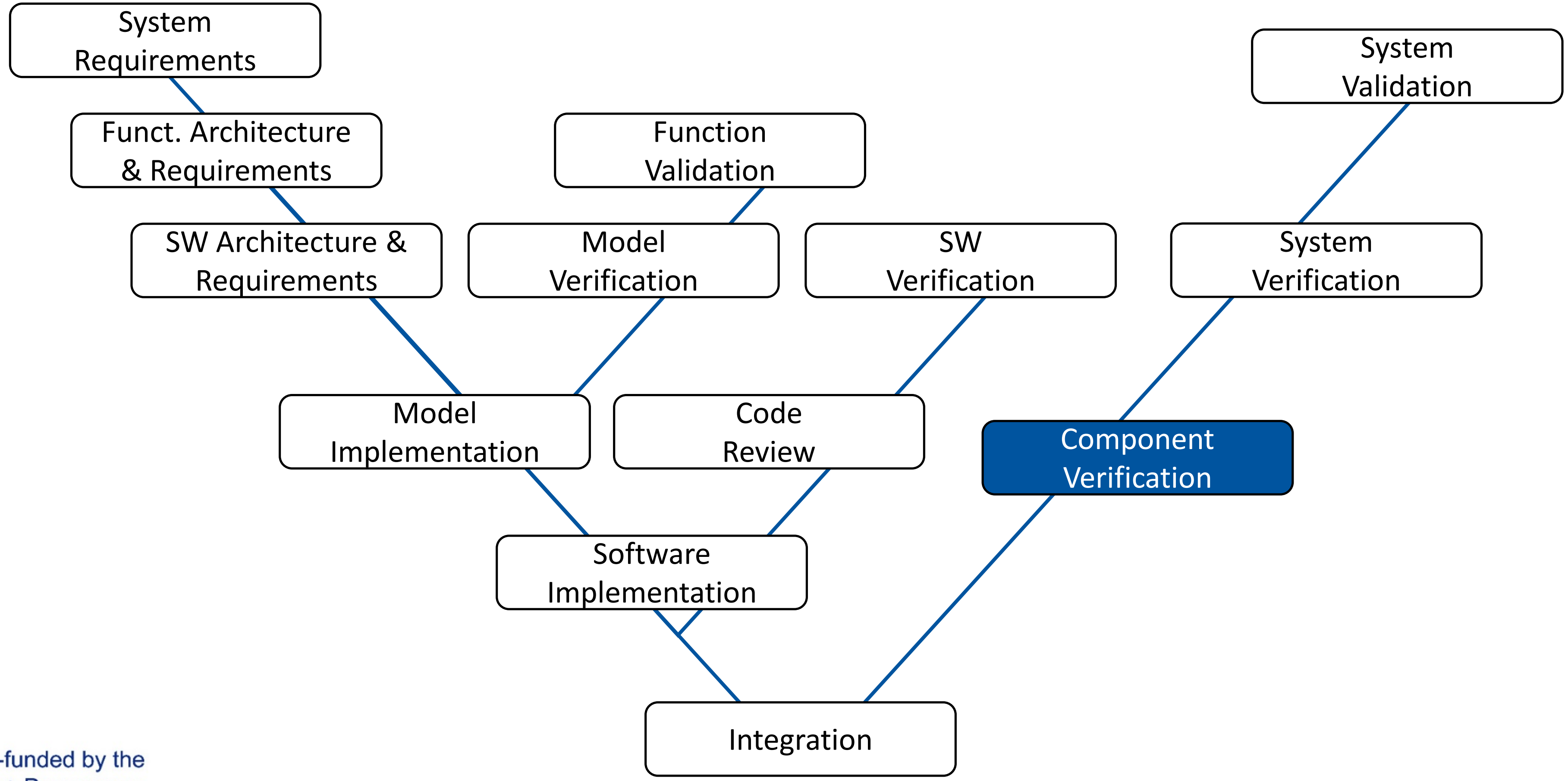
- No separate test model necessary
- Full traceability for diagnosis and safety tests
- Manual effort replaced by automation
- **Benefit:**
  - Reduced effort through right test selection

Co-funded by the Erasmus+ Programme of the European Union

Source: J. Richenhagen, Lecture “Software for Combustion Engines”, RWTH Aachen University, 2019 [3]

How can be component verification performed?

V-MODEL IN MODEL-DRIVEN SOFTWARE ENGINEERING



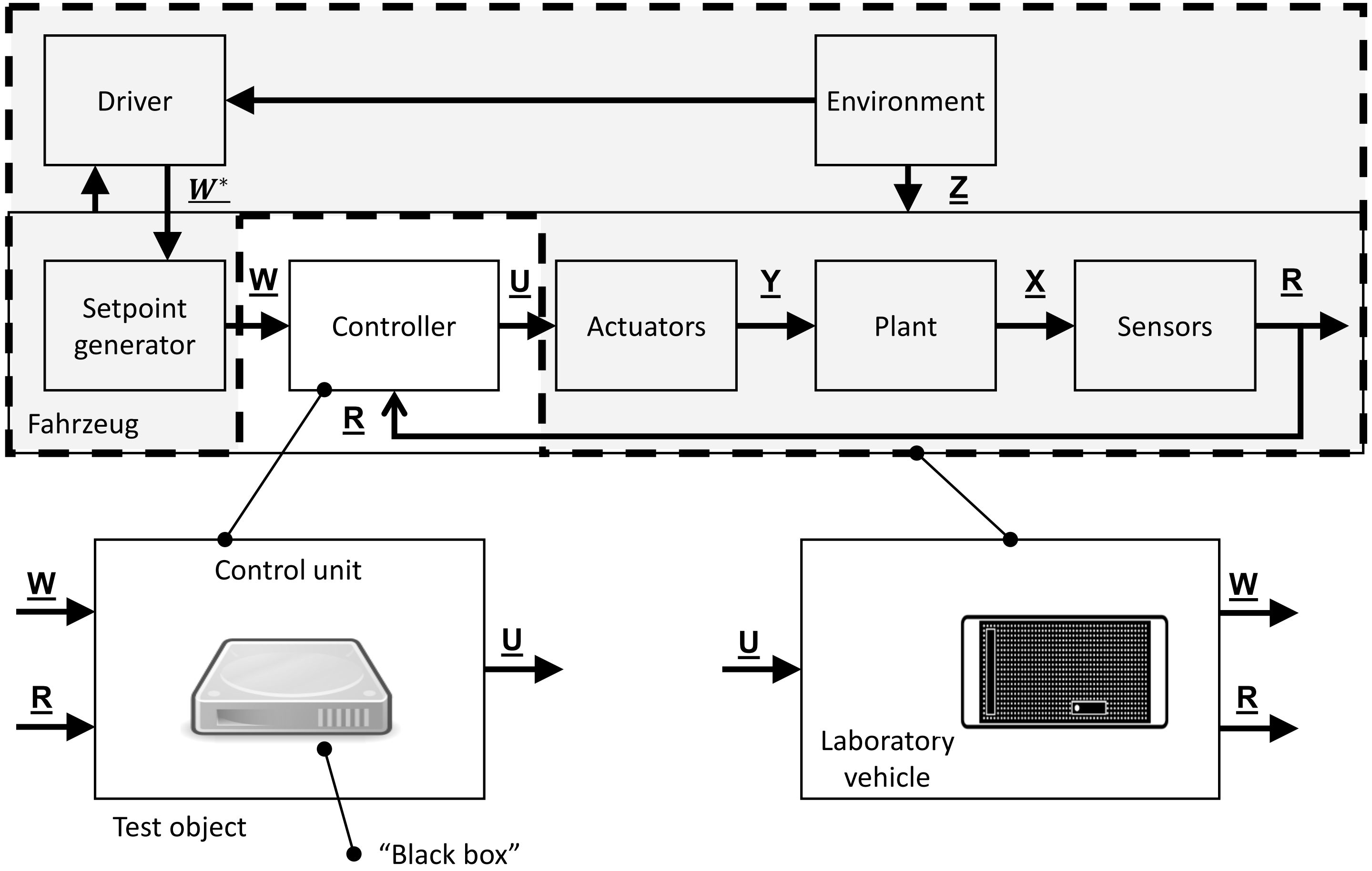
Co-funded by the Erasmus+ Programme of the European Union

Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [3]

Operation of the control device in a virtual environment with HIL

Simulation of:

- sensors/actuators,
- control devices,
- driver,
- track,
- environment

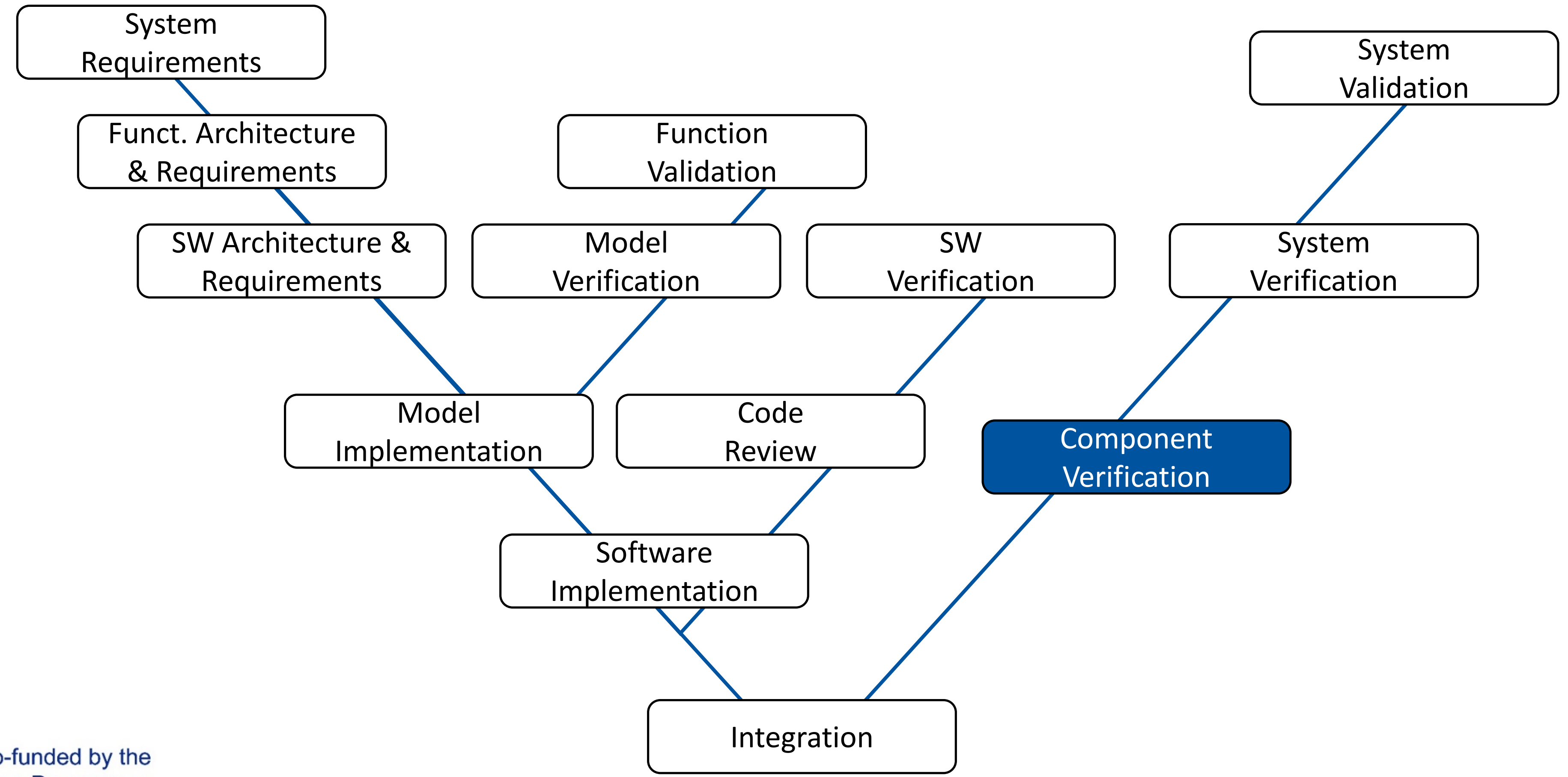


Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [3]; Schäuffele and T. Zurawka, "Automotive Software Engineering: Grundlagen, Prozesse, Methoden und Werkzeuge effizient einsetzen", Vieweg+Teubner Verlag / GWV Fachverlage, 2010 [7]

Co-funded by the Erasmus+ Programme of the European Union

How can be component verification performed?

V-MODEL IN MODEL-DRIVEN SOFTWARE ENGINEERING

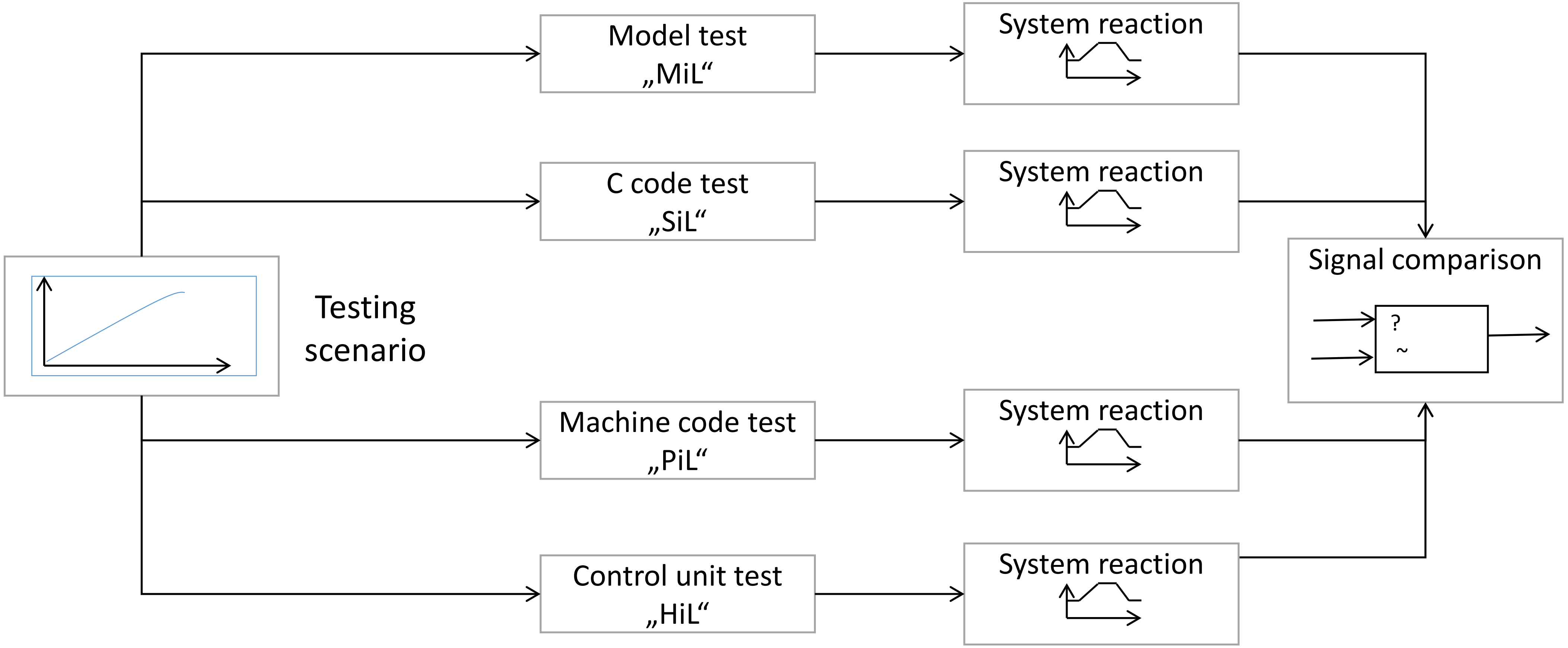


Co-funded by the Erasmus+ Programme of the European Union

Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [3]

What is the concept of Back-to-back testing?

CLASSIFICATION



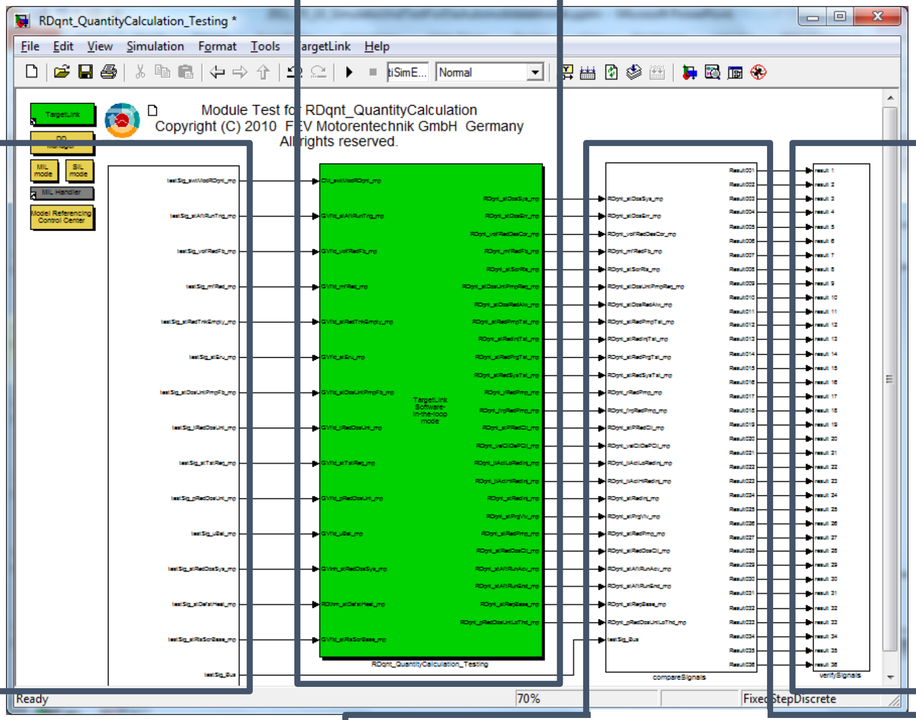
Co-funded by the Erasmus+ Programme of the European Union

Source: J. Richenhagen, Lecture “Software for Combustion Engines”, RWTH Aachen University, 2019 [3];
S. Sadeghipour, “Testautomatisierung: Ein akademisches Thema?”, SPES, 2010 [8]

Back-to-back testing example

TEST ARRANGEMENT AND EXECUTION VIA INTERFACE MATLAB/SIMULINK

Connection to software unit



Stimulation with test signals

Specification of desired output

Stimulation with test signals

Specification of desired output

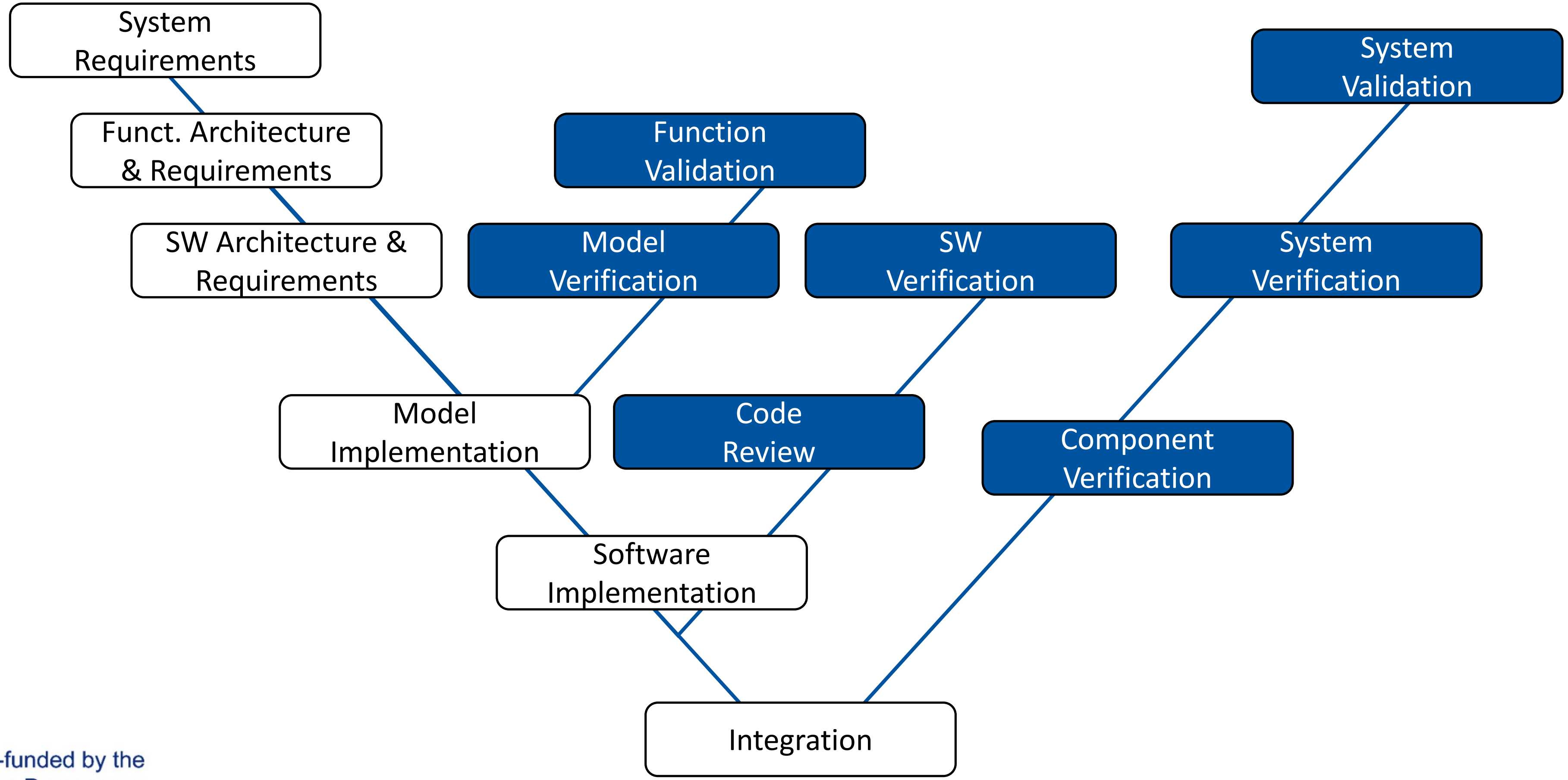
Recording of test results, validation of output

Co-funded by the Erasmus+ Programme of the European Union

Source: J. Richenhagen, Lecture “Software for Combustion Engines”, RWTH Aachen University, 2019 [3]

What are future trends to reduce testing efforts?

V-MODEL IN MODEL-DRIVEN SOFTWARE ENGINEERING



Source: FEV Europe GmbH

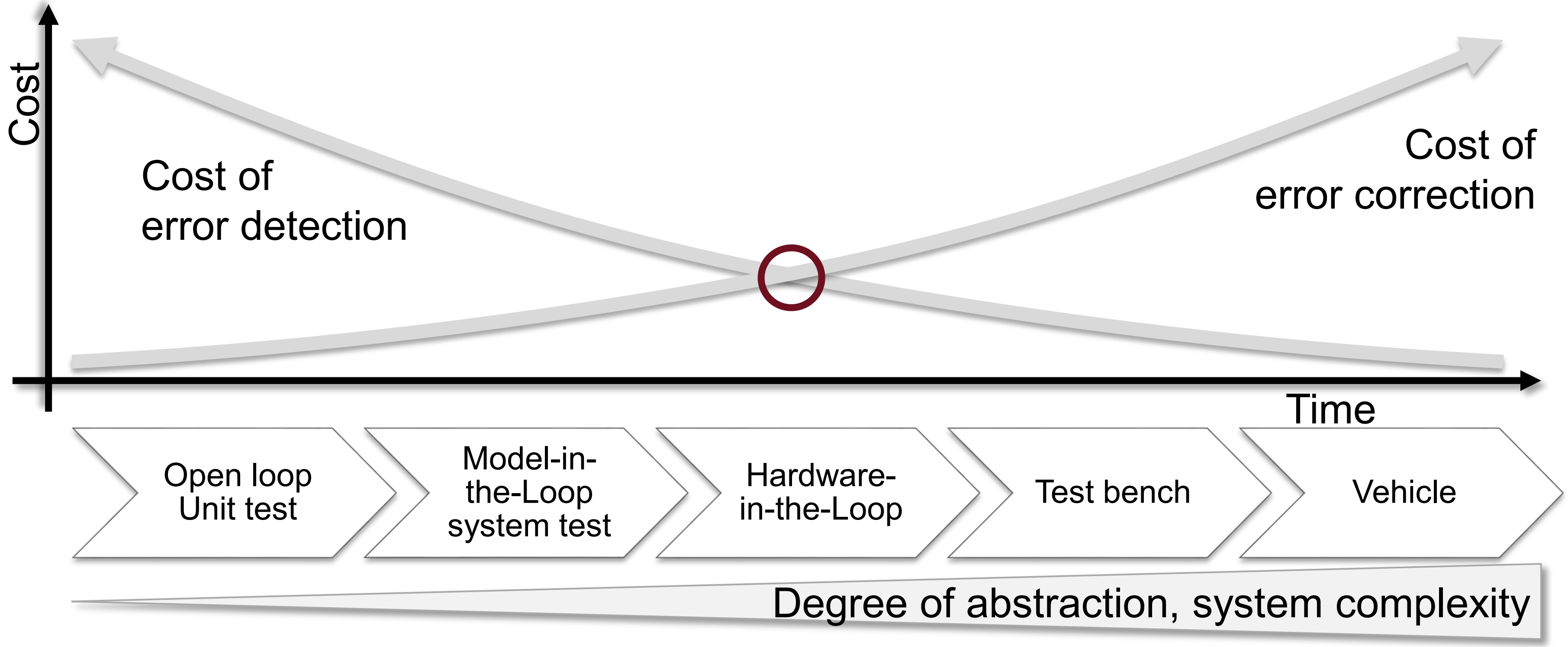
Co-funded by the Erasmus+ Programme of the European Union

Early error correction becomes efficient through virtualization

OPTIMIZATION OF FUNCTION TESTS IN ACOSAR, MODELISAR AND HIFI-ELEMENTS

Balance between the cost of virtualization and benefit of early error detection:

A typical contemporary trade-off



Source: P. Sternberg, J. Richenhagen, Dr. A. Schloßer, FEV GmbH, “The Challenge of early functional software tests”, 6. VDI/VDE Fachtagung AUTOREG 2013 [9]

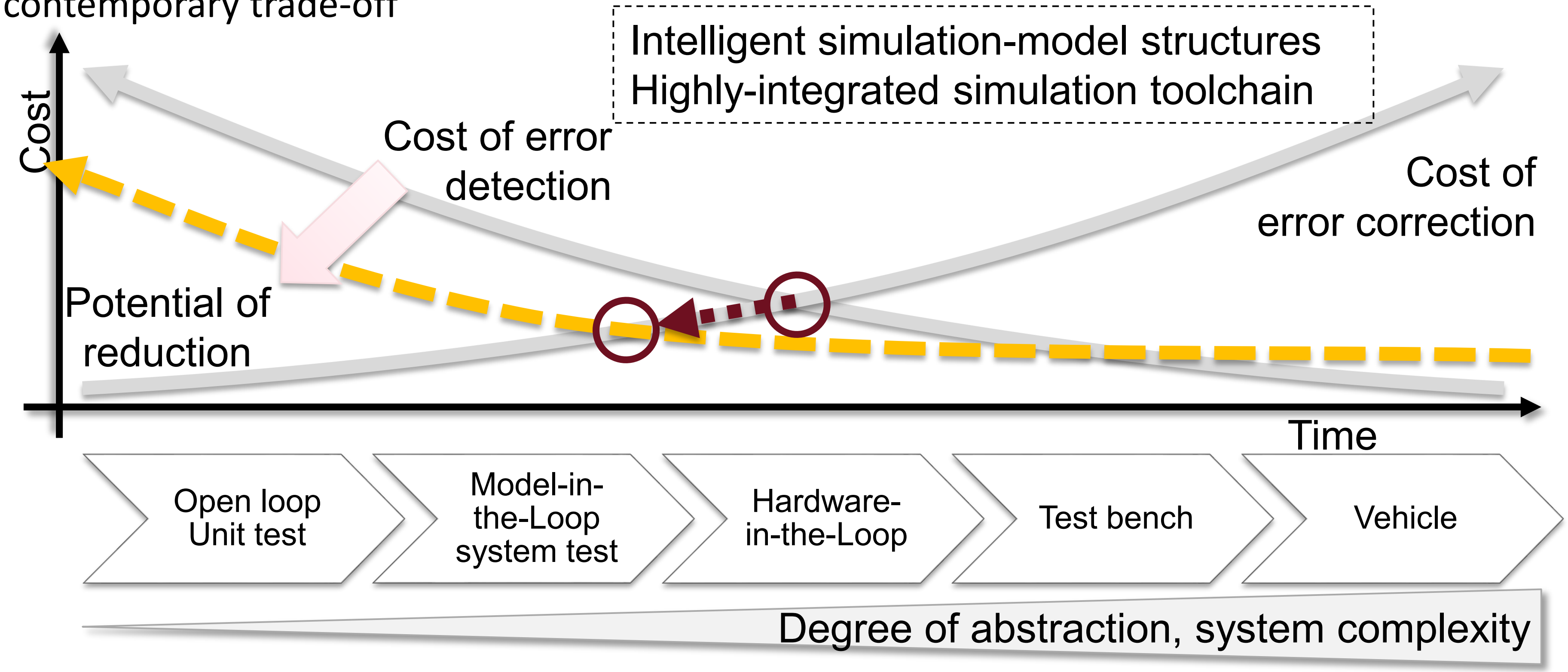
Co-funded by the Erasmus+ Programme of the European Union

Early error correction becomes efficient through virtualization

OPTIMIZATION OF FUNCTION TESTS IN ACOSAR, MODELISAR AND HIFI-ELEMENTS

Balance between the cost of virtualization and benefit of early error detection:

A typical contemporary trade-off

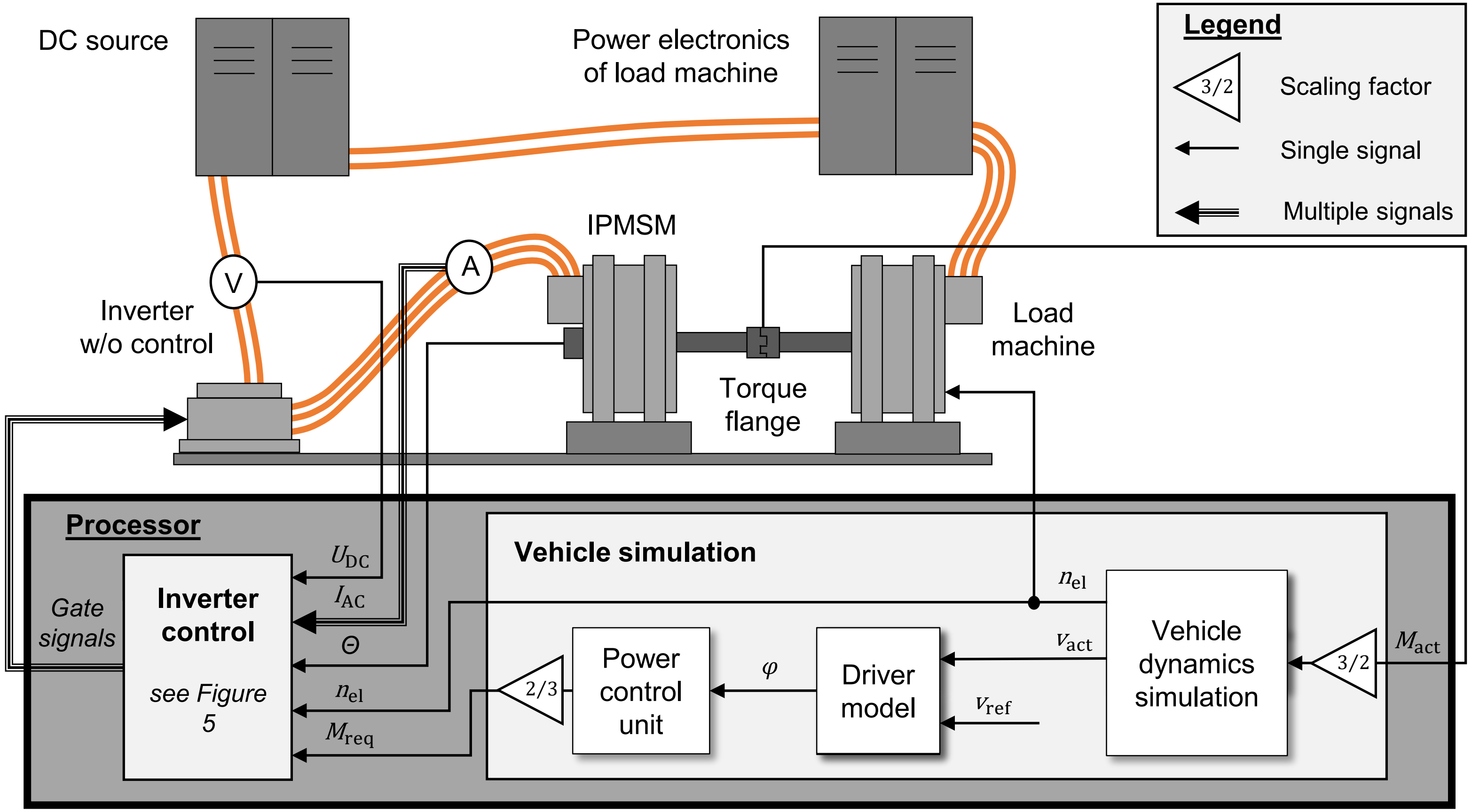


Co-funded by the Erasmus+ Programme of the European Union

Source: P. Sternberg, J. Richenhagen, Dr. A. Schloßer, FEV GmbH, “The Challenge of early functional software tests”, 6. VDI/VDE Fachtagung AUTOREG 2013 [9]

Virtualization by E-Motor-in-the-Loop

HARDWARE SETUP OF E-MOTOR-IN-THE-LOOP TEST BENCH

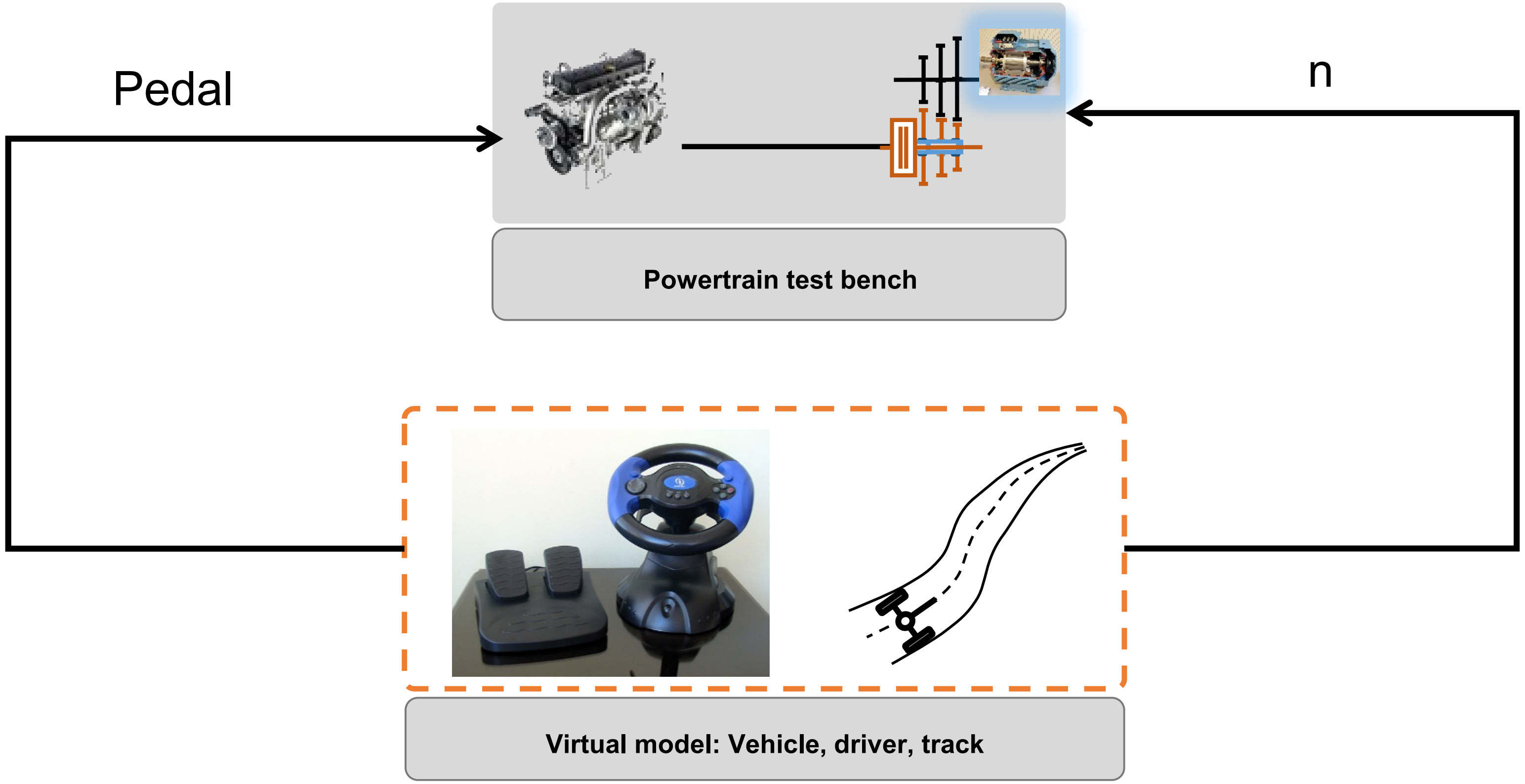


Source: Etzold, K., Kürten, C., Thul, A., Müller, L. et al., "Efficient Power Electronic Inverter Control Developed in an Automotive Hardware-in-the-Loop Setup," SAE Technical Paper 2019-01-0601, 2019 [10]

Operation of the control unit in a virtual test-bed environment

FUTURE: FULL SIMULATION OF TRACK, DRIVER AND ENVIRONMENT

- System test by integration of subsystems
- New setup necessary, i.e. high effort and expensive
- Only possible in a late stage of the project since all powertrain components need to be physically available



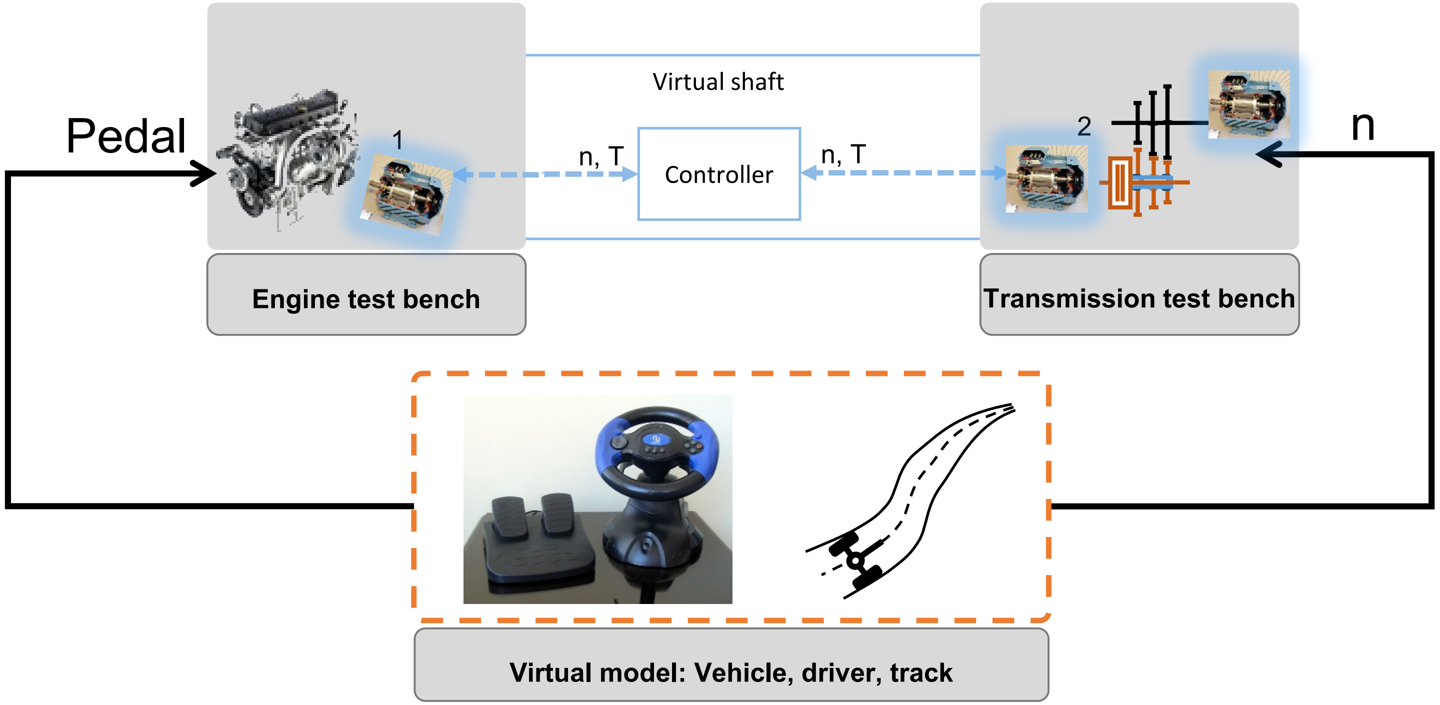
Co-funded by the Erasmus+ Programme of the European Union

Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [3]
Image: Martin Iturbide (https://commons.wikimedia.org/wiki/File:InterAct_V-Thunder_Racing_Wheel_for_GameCube.JPG), „InterAct V-Thunder Racing Wheel for GameCube", https://creativecommons.org/licenses/by-sa/3.0/legalcode

Operation of the control unit in a virtual test-bed environment

FUTURE: FULL SIMULATION OF SYSTEM, DRIVER AND ENVIRONMENT "VIRTUAL SHAFT"

- Necessary matching condition for ideal shaft connection
  → $n_1 = n_2, T_1 = T_2$
- Closed-loop controller for number of revolutions and torque
- Virtual shaft enables powertrain testing at early project stages



Co-funded by the Erasmus+ Programme of the European Union

Source: J. Richenhagen, Lecture "Software for Combustion Engines", RWTH Aachen University, 2019 [3]
Image: Martin Iturbide (https://commons.wikimedia.org/wiki/File:InterAct_V-Thunder_Racing_Wheel_for_GameCube.JPG), „InterAct V-Thunder Racing Wheel for GameCube", https://creativecommons.org/licenses/by-sa/3.0/legalcode

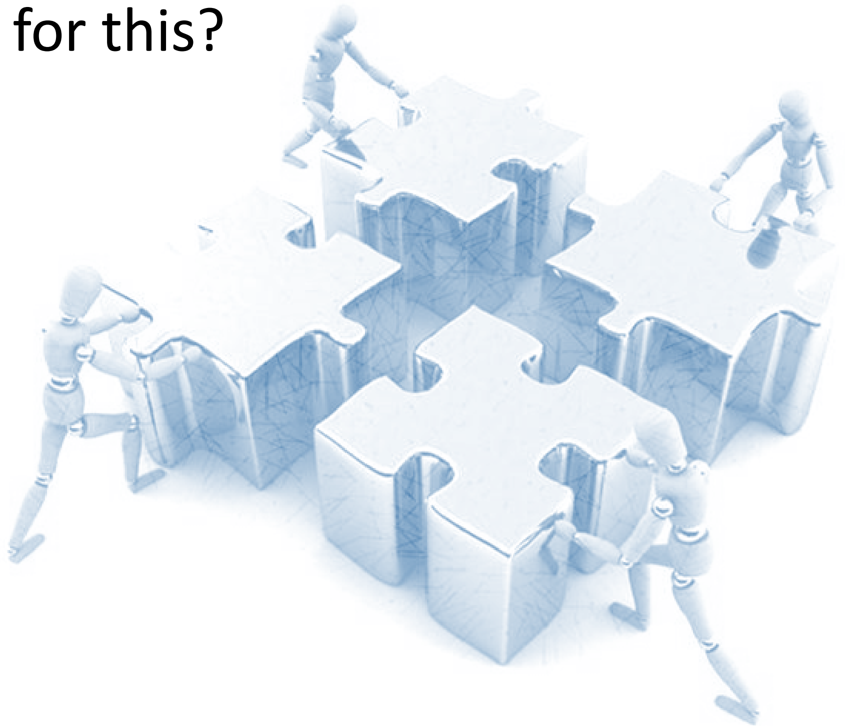
Agenda

- Motivation
- Terminology
- Approaches for software evaluation and testing
- Test management
- References

Co-funded by the Erasmus+ Programme of the European Union

Learning objectives

- **Know the demand of test scheduling**
  - Why do we need test scheduling and a test strategy?
  - What are the guidelines and standards for this?



Co-funded by the
Erasmus+ Programme
of the European Union

Test management is relevant to ensure test coverage at acceptable costs

AUTOMOTIVE TEST MANAGEMENT - MOTIVATION & CHALLENGES

- Importance of software verification & validation
  - ISO 25010: product quality assurance
  - ISO 26262: functional safety
    → legal obligations

- but: resources are limited
  - restrictions on budget, time, personnel
  - classic “squeeze” on testing
  - ➢ exhaustive testing often not feasible
  - ➢ strategic, intelligent testing on basis of defined processes required

Source: FEV Consulting / C. Kugler, V&V Management, FEV Europe GmbH

Co-funded by the Erasmus+ Programme of the European Union

What is a testing strategy?

AS DEFINED BY THE **INTERNATIONAL SOFTWARE TESTING QUALIFICATIONS BOARD** (ISTQB)

Definition

- “The test strategy describes the organization’s **general test methodology**.
- This includes
  - the way in which testing is used to **manage product and project risks**,
  - the **division of testing into levels**, and
  - the **high-level activities associated with testing**.”

Requirements

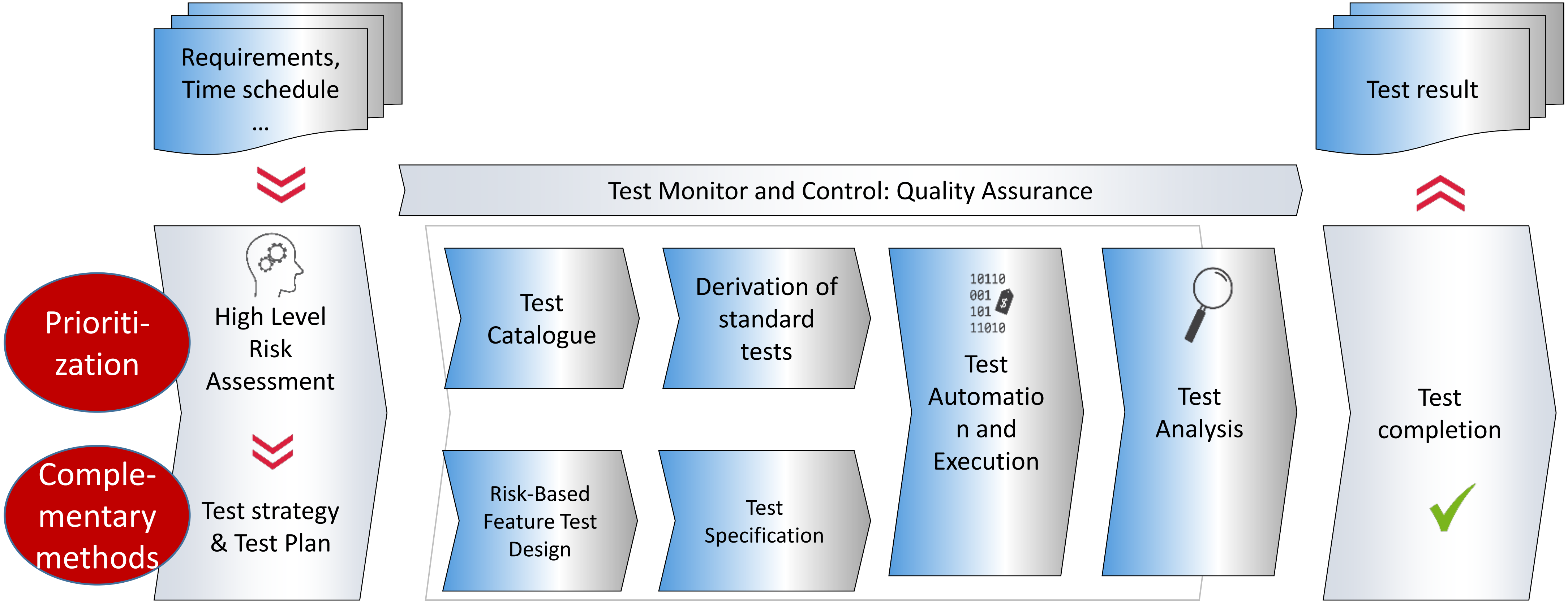
- **Flexibility**: “The same organization may have different strategies for different situations, such as different software development lifecycles, different levels of risk, or different regulatory requirements.”
- **Consistency**: “The test strategy, and the processes and activities described in it, should be consistent with the test […] [Objectives]. It should provide the generic test entry and exit criteria for the organization or for one or more programs.”

Source: ISTQB, “Certified Tester - Advanced Level Syllabus Test Manager”, Version 2012 [11]

Co-funded by the Erasmus+ Programme of the European Union

PROCESS ASSURES QUALITY OF SOFTWARE AT EACH DEVELOPMENT STAGE

TEST MANAGEMENT - INTEGRAL TO SOFTWARE DEVELOPMENT

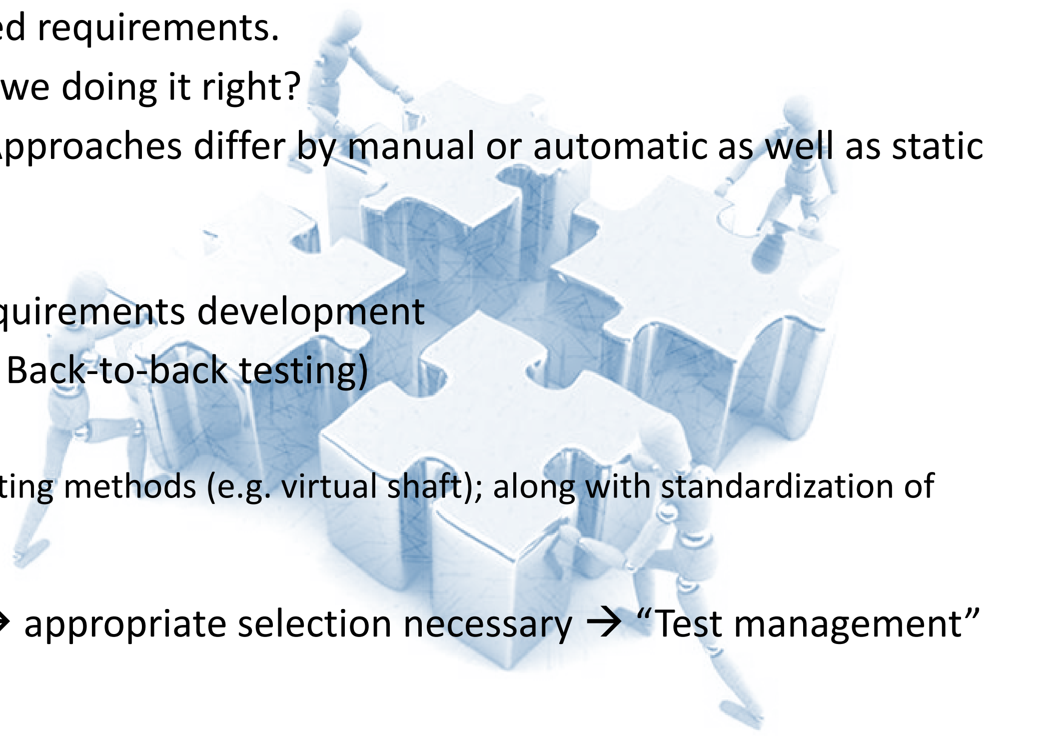


Source: FEST Services / FEV Europe GmbH

Co-funded by the Erasmus+ Programme of the European Union

Summary & Conclusion

- Verification and Evaluation
  - Analytic measures are important in each step of the development
  - The later the error is detected, the higher the costs
  - Quality is not an absolute concept, but is related to defined requirements.
  - Validation: Are we doing the right thing? Verification: Are we doing it right?
  - Different types of software evaluation approaches exist. Approaches differ by manual or automatic as well as static and dynamic execution.
  - Metrics can be used for static evaluation of software.
  - Derivation of test cases → advantages of model-based requirements development
  - Automatization of test execution and evaluation (e.g. HIL, Back-to-back testing)
  - Trends to reduce validation- and verification efforts
    - Modelling / simulation for quick evaluation of extensive testing methods (e.g. virtual shaft); along with standardization of interfaces
      → e.g. Modelisar, Acosar, HiFi-ELEMENTS
  - Verification and validation on different levels is possible → appropriate selection necessary → "Test management"



Co-funded by the Erasmus+ Programme of the European Union

Agenda

- Motivation
- Terminology
- Approaches for software evaluation and testing
- Test management
- References

Co-funded by the
Erasmus+ Programme
of the European Union

References

[1] S. KOWALEWSKI
Lecture: Software for Combustion Engines
RWTH Aachen University
2019

[2] MC KINSEY & COMPANY
Numetrics R&D Analytics
April 2016

[3] J. RICHENHAGEN
Lecture: Software for Combustion Engines
RWTH Aachen University
2019

[4] INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS
IEEE 610.12-1990
IEEE Standard Glossary of Software Engineering Terminology
New York, 1990

[5] H. VENKITACHALAM, C. GRANRATH, G.V. BALACHANDAR, J. RICHENHAGEN
Metric-based Evaluation of Hybrid Control Software
SAE World Congress
Detroit, 2017

Co-funded by the
Erasmus+ Programme
of the European Union

References

[6] J. RICHENHAGEN
Entwicklung von Steuerungs-Software für den automobilen Antriebsstrang mit agilen Methoden
Doctoral thesis
2014

[7] SCHÄUFFELE AND T. ZURAWKA
Automotive Software Engineering: Grundlagen, Prozesse, Methoden und Werkzeuge effizient einsetzen, 4th ed.
Vieweg+Teubner Verlag / GWV Fachverlage
Wiesbaden, 2010

[8] S. SADEGHIPOUR
Testautomatisierung: Ein akademisches Thema?
SPES
2010

[9] P. STERNBERG, J. RICHENHAGEN, DR. A. SCHLOSSER
The Challenge of early functional software tests
FEV GmbH – 6. VDI/VDE Fachtagung AUTOREG
2013

[10] ETZOLD, K., KÜRTEN, C., THUL, A., MÜLLER, L. ET AL.
Efficient Power Electronic Inverter Control Developed in an Automotive Hardware-in-the-Loop Setup
SAE Technical Paper 2019-01-0601
2019

[11] ISTQB
Certified Tester - Advanced Level Syllabus Test Manager
2012

Co-funded by the
Erasmus+ Programme
of the European Union

UNITED

Engineering Knowledge Transfer Units to Increase Student´s Employability and Regional Development

https://www.facebook.com/unitederasmus/

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
598710-EPP-1-2018-1-AT-EPPKA2-CBHE-JP

Engineering Knowledge Transfer Units to Increase Student´s Employability and Regional Development

Module 1 – Part 3

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
598710-EPP-1-2018-1-AT-EPPKA2-CBHE-JP

Summary

1. Automation in modern vehicles
2. Overview of ADAS and autonomous driving techniques
3. Advancement in environment sensing
4. Overview of control techniques for ADAS (FL, MPC, AI)
5. Examples of ADAS techniques
6. IT connectivity
7. Vehicle charging systems

Co-funded by the
Erasmus+ Programme
of the European Union

Requirements of modern car design

Connectivity

Style

Safety

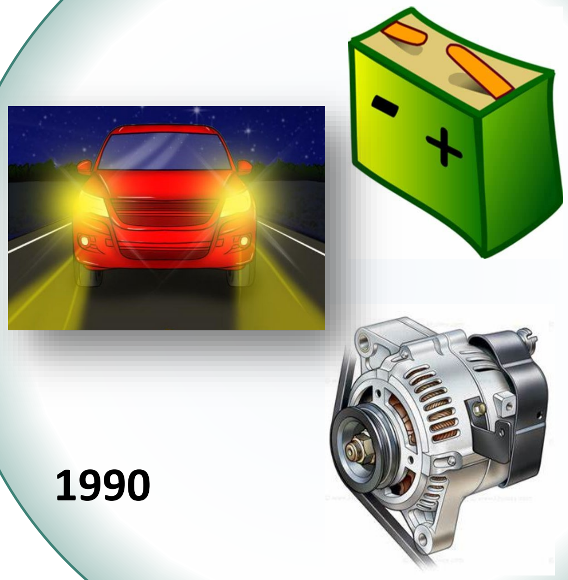
Affordability

Performance

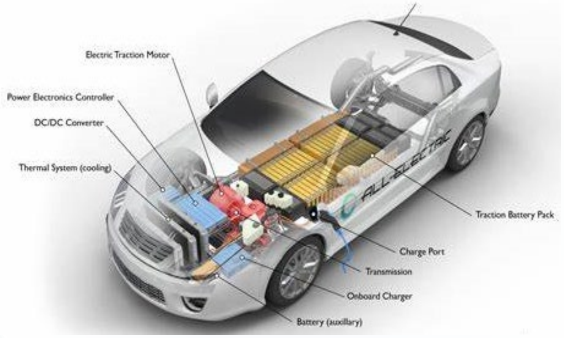
Sustainability

Co-funded by the
Erasmus+ Programme
of the European Union

Electrification process



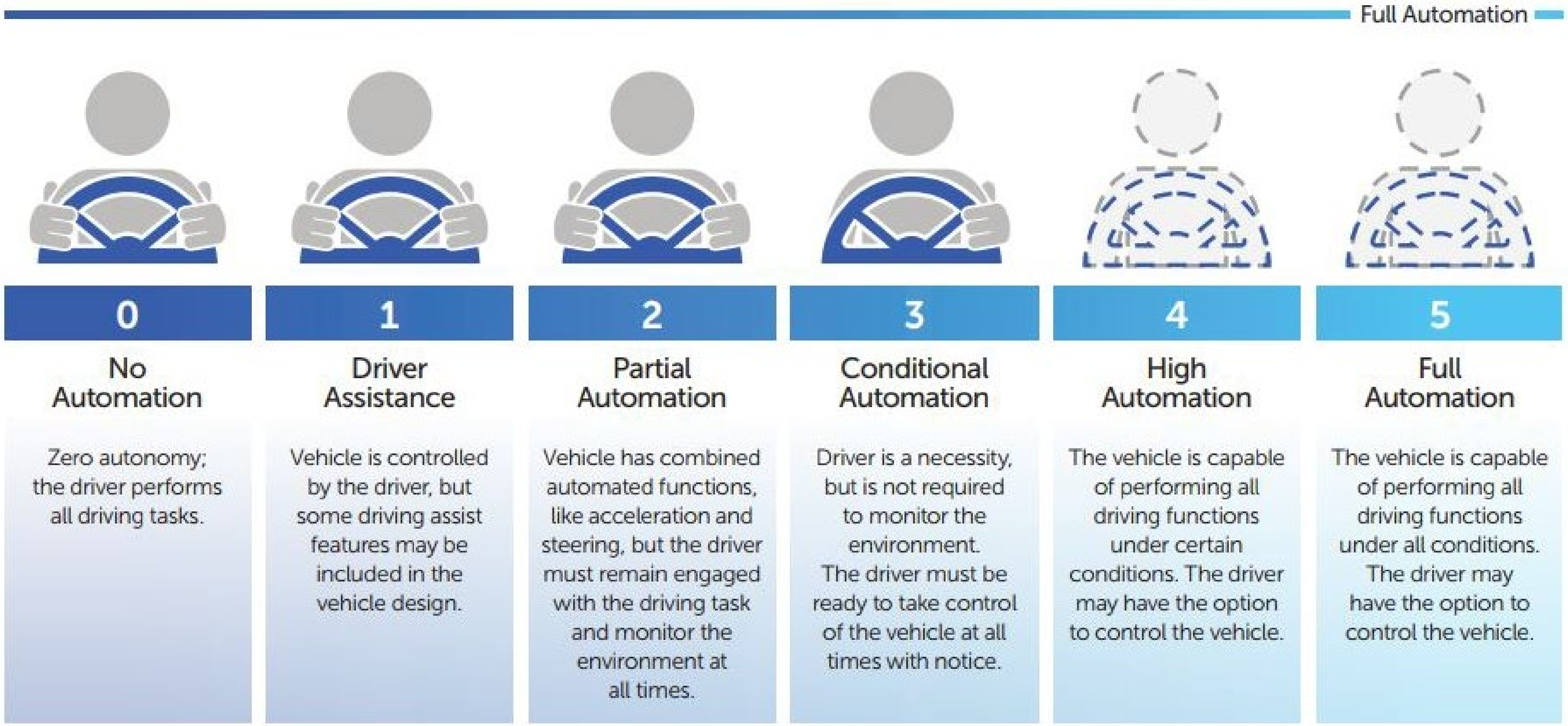
1990



2020

Co-funded by the
Erasmus+ Programme
of the European Union

Autonomous driving SAE levels



Co-funded by the Erasmus+ Programme of the European Union

Autonomous driving SAE levels

degree of automation →

	Driver only	Assisted	Partial automation	Conditional automation	High automation	Full automation
SAE	0	1	2	3	4	5
NHTSA	0	1	2	3		4
driver in the loop	yes (required)			not required		
time to take control back	-	~ 1s		several seconds	couple of minutes	
other activities while driving	not allowed			specific	all (even sleeping)	
examples	FCW, LDW	ACC, LKA	Traffic Jam Assistant	Highway Chauffeur	Valet Parking	Robot car

FCW ... Forward Collosion Warning
LDW ... Lane Departure Warning

ACC ... Adaptive Cruise Control
LKA ... Lane Keeping Assistant

Source: SAE, NHTSA, VDA

Co-funded by the Erasmus+ Programme of the European Union

Autonomous driving SAE levels

Partially automated – current status on some vehicles
The vehicle has to be monitored constantly
Driver is called to be in charge of the driving task at any moment

Highly automated – within 2020
Constant monitoring of the vehicle is not required
The driver can take over the driving tasks with lead time

Fully automated – within 2025
The monitoring of the vehicle is not required
The driver is exempt from driving duties

Co-funded by the Erasmus+ Programme of the European Union

- Vehicle dynamics
- Radar
- Lidar
- Camera
- Sensor fusion
- Virtual sensing techniques

Co-funded by the
Erasmus+ Programme
of the European Union

- **LIDAR**

+ High precision distance
+ 360° horizontal resolution
- Weather condition
- Low vertical resolution
- Cost

- **Tasks**
  - environment 3D point-cloud generation
  - wide range 3D environment reconstruction
  - 3D point-cloud semantic segmentation
  - objects frame detection and labeling

Co-funded by the Erasmus+ Programme of the European Union

- **Camera**

+ Cost
+ over 150° frontal Field-of-View
+ Stereo Image redundancy
- Limited 3D reconstruction
- No 360° Field-of-View

- **Tasks**
  - 2D environment perception
  - 2D environment reconstruction
  - object detection and labeling
  - limited 3D reconstruction

Why a Stereo Camera?

- ✓ Detect both images (left and right)
- ✓ Use each detection as comparison on the other image
- ✓ Search for correspondences
- ✓ Calculate disparities
- ✓ Calculate object distances from disparaties

Co-funded by the Erasmus+ Programme of the European Union

- **GPS**

+ Robust system worldwide validated

- Centimeter accuracy = higher price

- **Tasks**
  - Robust positioning
  - Vehicle Tracking
  - Route planning

- **IMU**

+ 9 DOF measurement (position and orientation)

+ Integration with GPS and LIDAR

+ Cost

- **Tasks**
  - Robust positioning
  - Vehicle Tracking
  - Integration with LIDAR and camera measurements

Co-funded by the Erasmus+ Programme of the European Union

- **Radar**

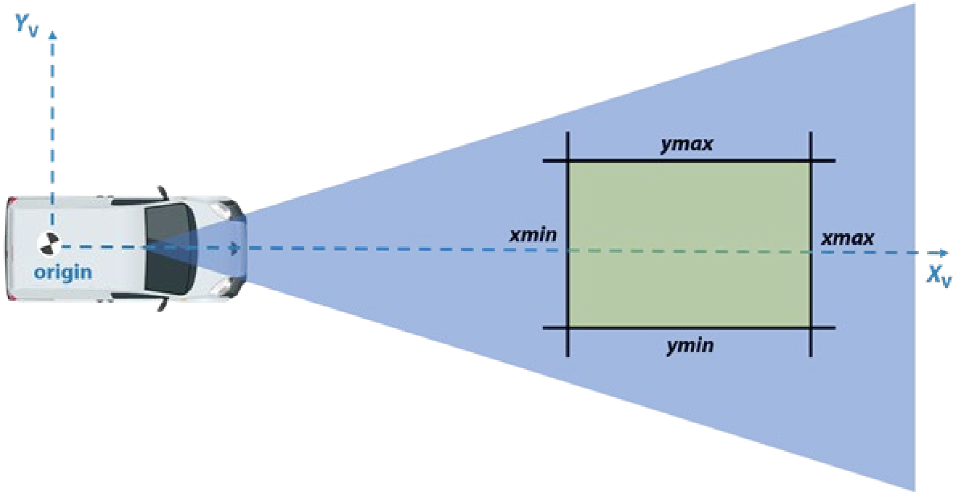
Radio detection and ranging
Origin: military from WW2
Adopted in automotive from 1998 for
  collision warning
  adaptive cruise control
  automatic emergency brake
  lane change assistance
  park assistance
  Blind spot assistance

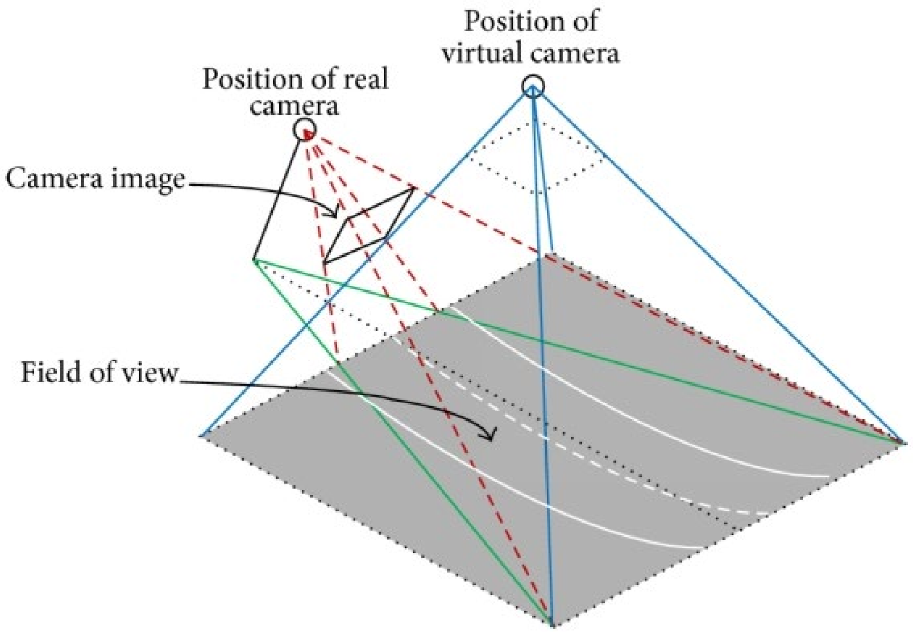
Commercial models:
…

+ Long range
+ Independent on the weather conditions
- The need of long-, mid- and short-range devices means higher price for each instrumented vehicle

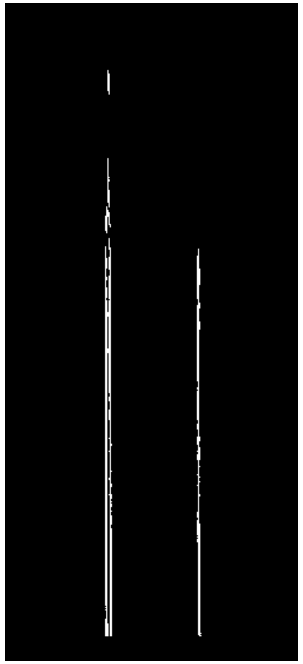
Co-funded by the Erasmus+ Programme of the European Union

Basic tasks – Lane detection





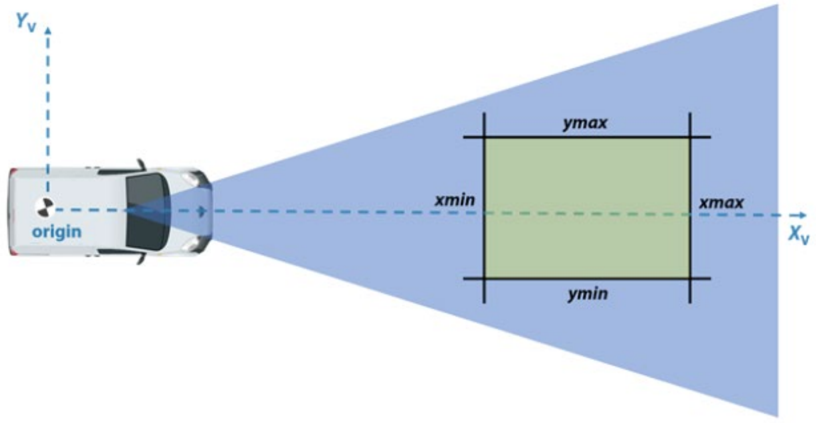
(b)



Co-funded by the
Erasmus+ Programme
of the European Union

Basic tasks – Lane detection

Camera mounted on the vehicle top

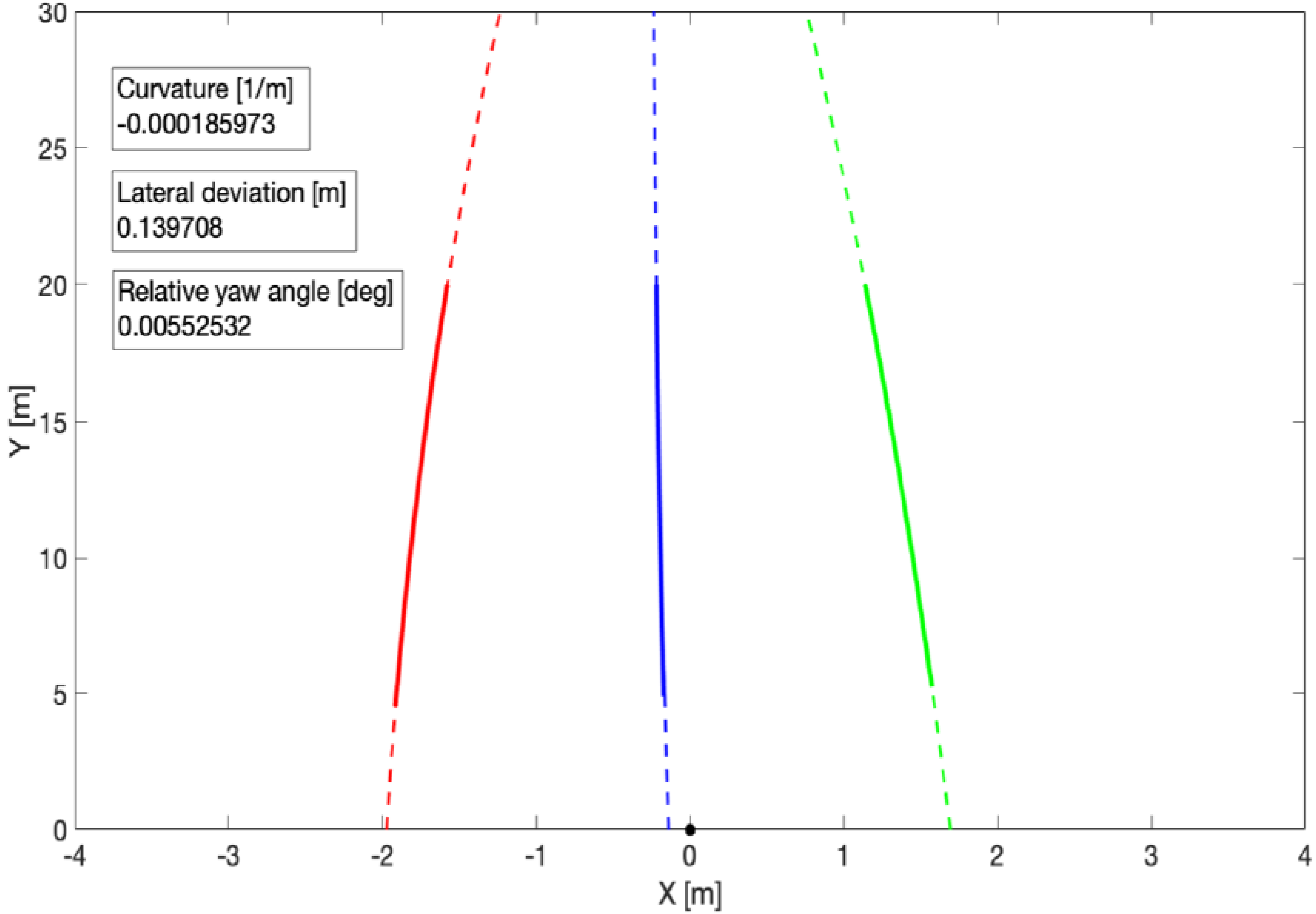


Bird's eye view

Lane detection

Co-funded by the
Erasmus+ Programme
of the European Union

Lane detection



Co-funded by the
Erasmus+ Programme
of the European Union

FOF

Artificial intelligence techniques

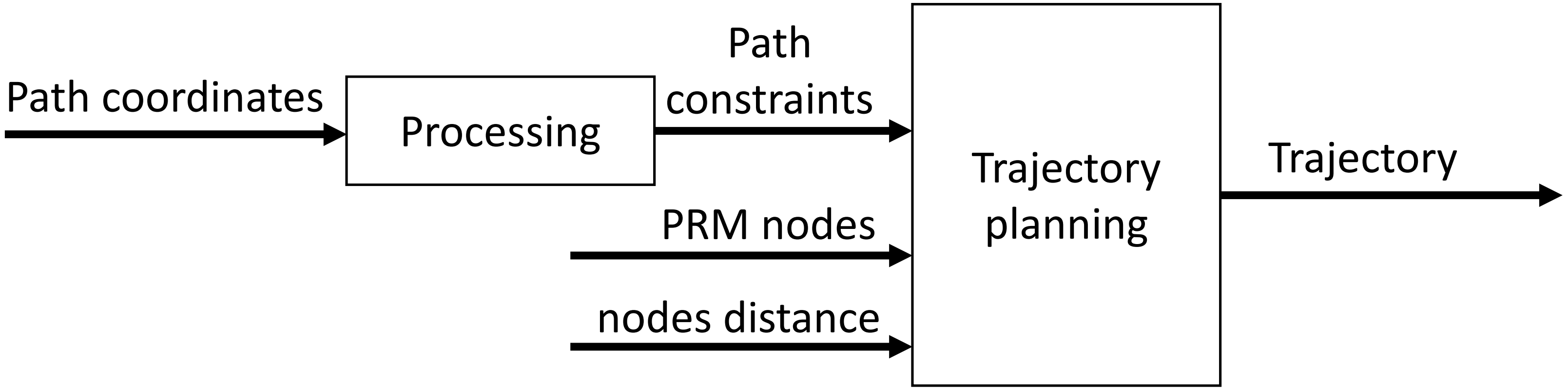
Regression tasks

Classification tasks

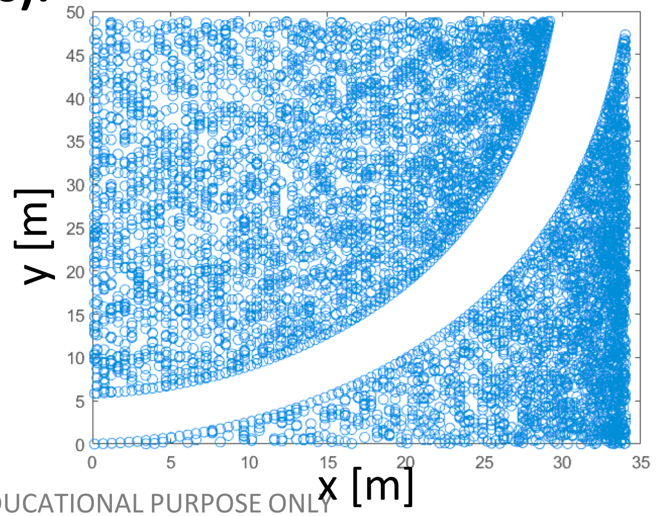
Co-funded by the
Erasmus+ Programme
of the European Union

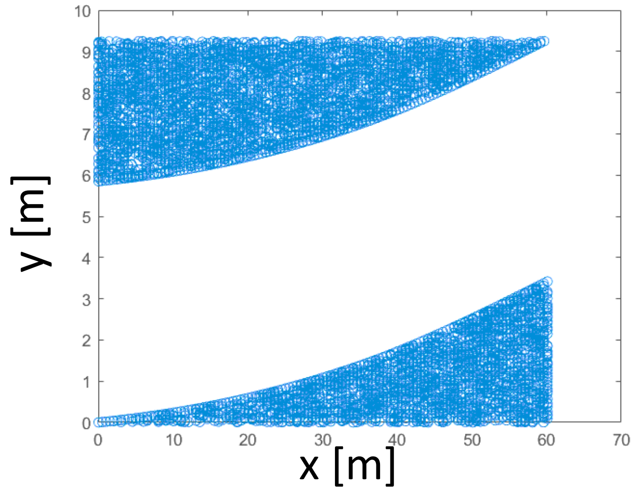
Trajectory planning

❑ Probabilistic Roadmap Algorithms (PRA) layout



❑ Maps (examples):

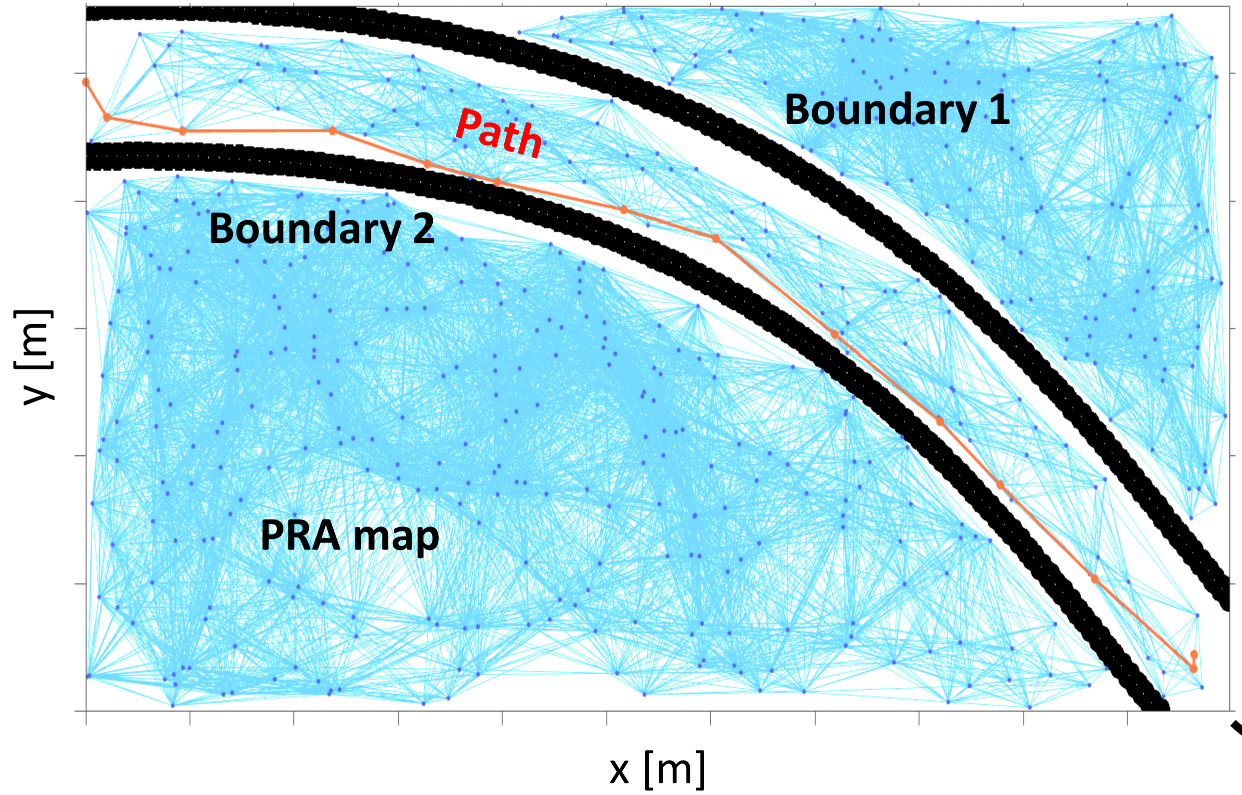


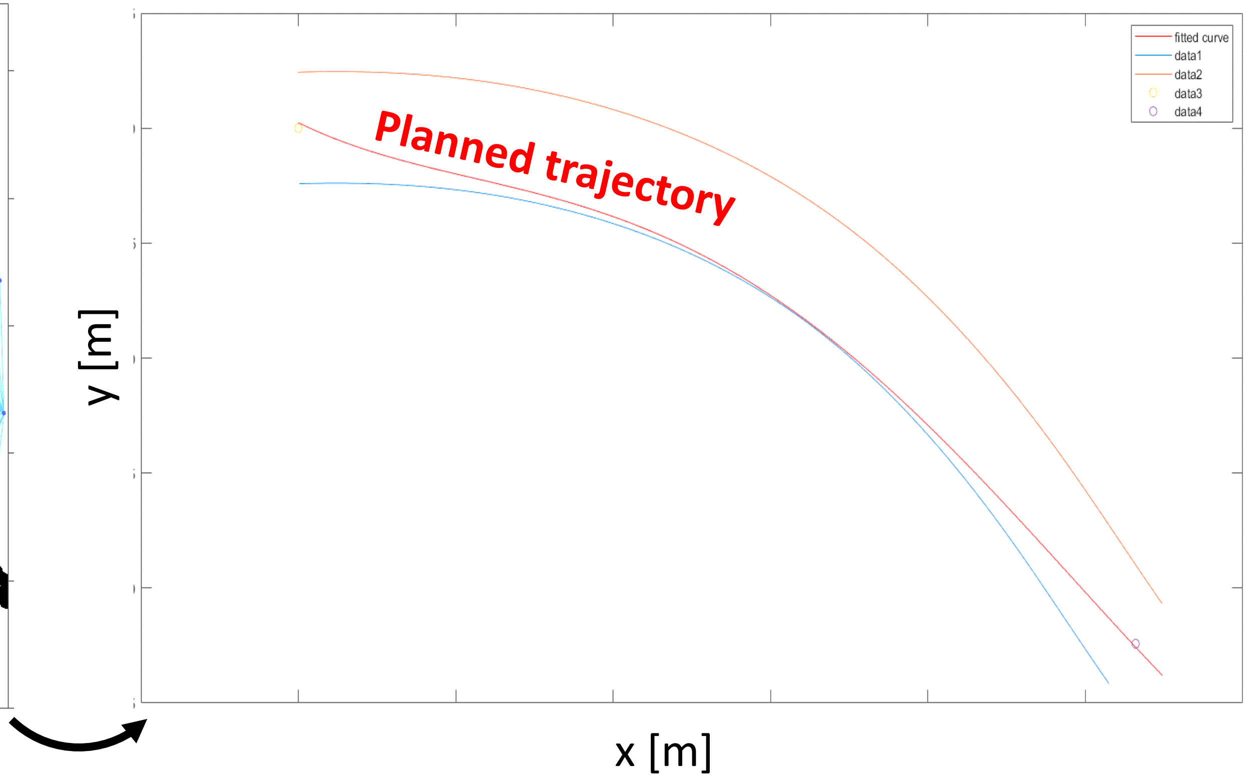


Co-funded by the
Erasmus+ Programme
of the European Union

Trajectory planning

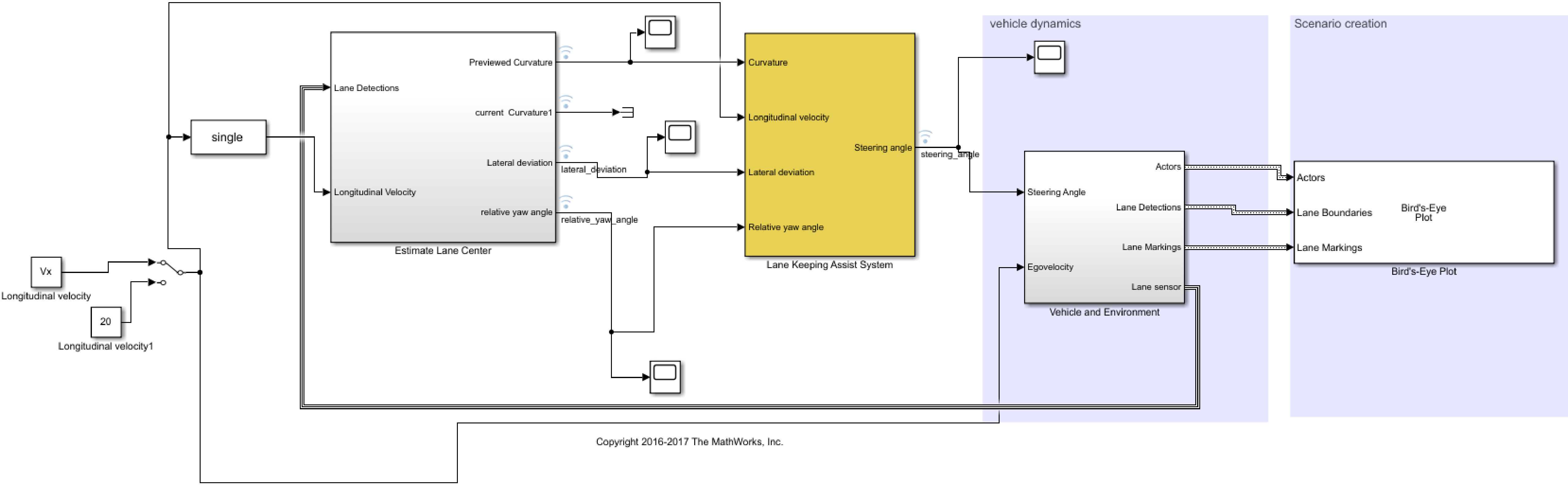
❑ Probabilistic Roadmap Algorithms (PRA) – results



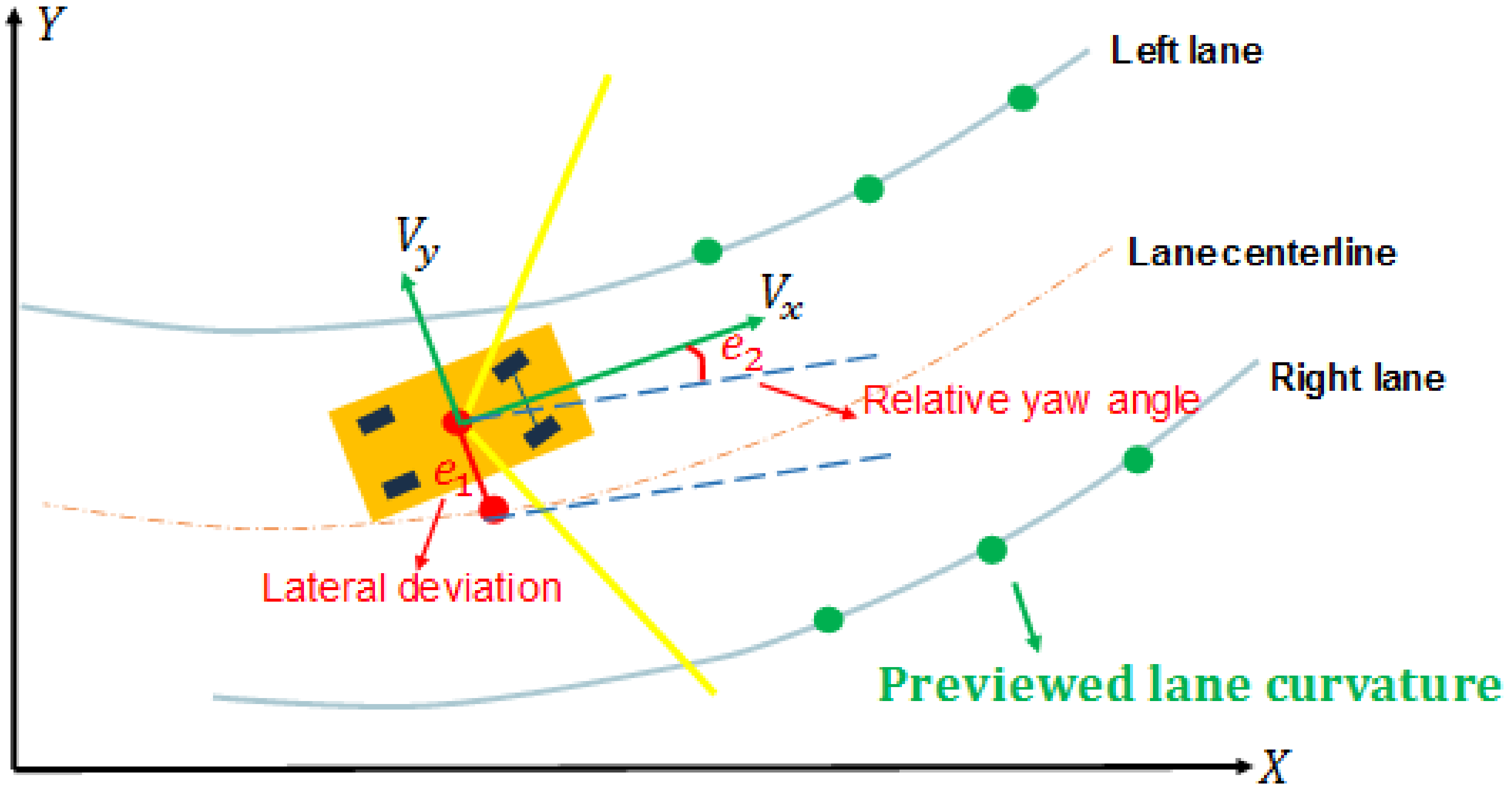


Co-funded by the
Erasmus+ Programme
of the European Union

Basic tasks – Lane keeping

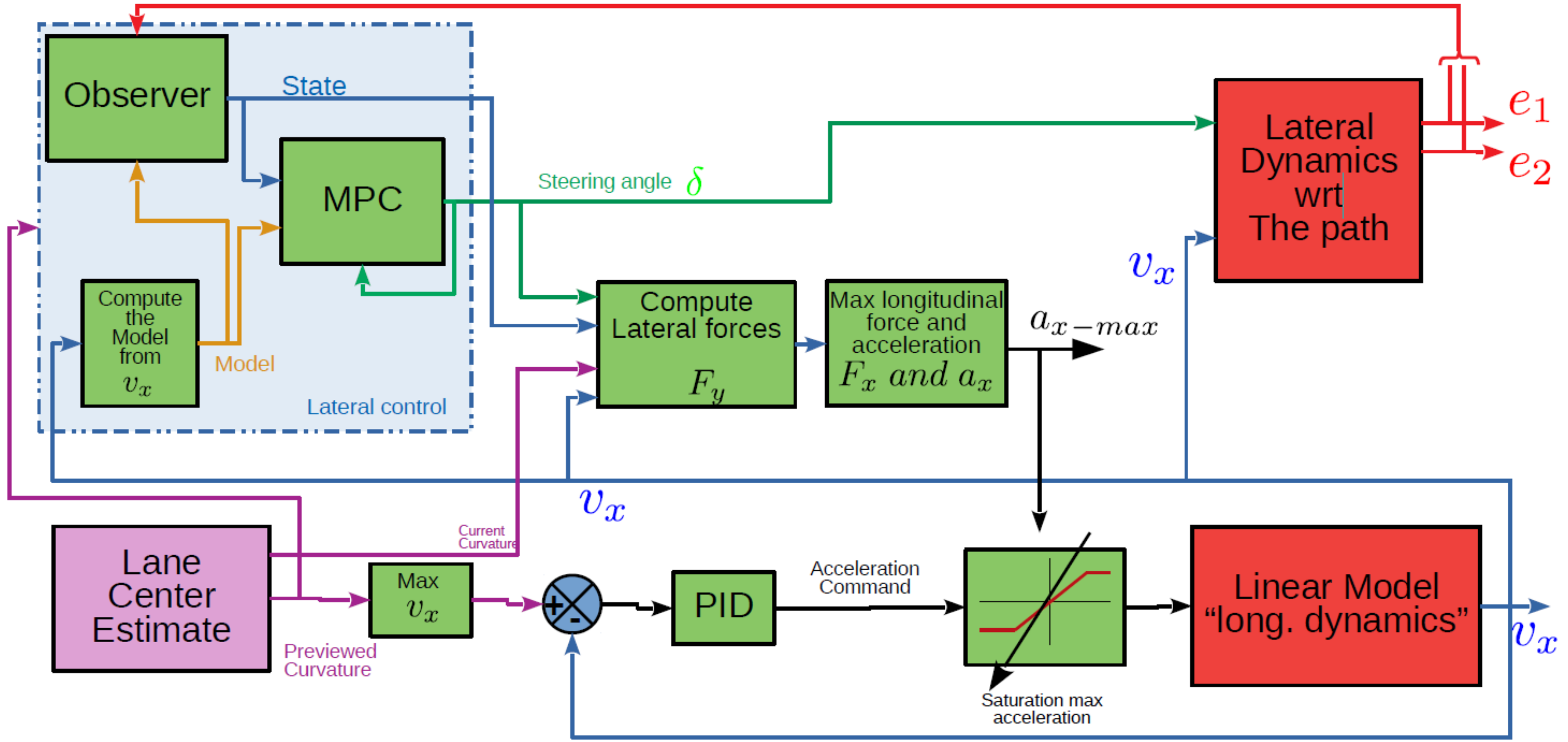


Co-funded by the Erasmus+ Programme of the European Union



Co-funded by the
Erasmus+ Programme
of the European Union

Advanced tasks - Longitudinal velocity profile generation



Virtual sensing techniques – Sideslip angle and vehicle speed

What is the Sideslip angle β?

- Angle between the vehicle longitudinal direction $\boldsymbol{u}$ and the actual speed vector $\boldsymbol{V}$
- Highly dependent from road condition
- Adopted in Active Control Systems to improve perfomance and safety
- Used as vehicle ***handling indicator***:
    - on-fly assessment of the vehicle conditions
    - detection of the proximity to the tire saturation region

Co-funded by the
Erasmus+ Programme
of the European Union

State of the art – Sideslip estimation

❑ **Methods:**

Sideslip Angle β

- ***Direct Measurements***
  - optical sensor (e.g. Correvit ® S-Motion from Kistler)
  - dual antenna sensor (e.g. Vbox Automotive)
- ***Indirect Method*** – Virtual sensing - Estimation
  - *Analytical techniques (model-based)* → State observer based vehicle dynamic model
  - *Empirical techniques (model-less)* → ***Artificial Network Networks (ANNs)***

❑ **Why β estimation?**

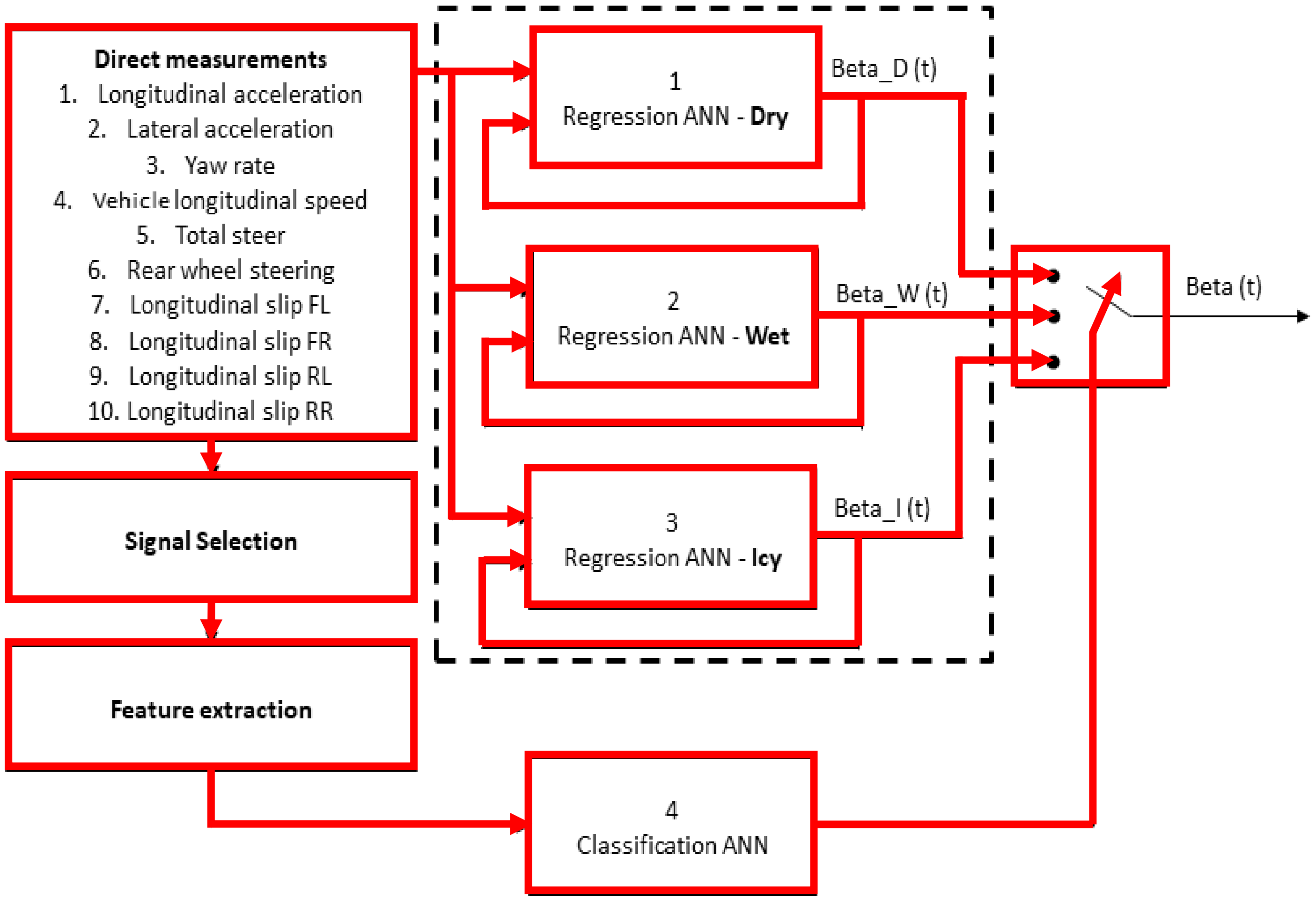
- It cannot be directly measured due to high cost or impracticality of the necessary sensor HW

❑ **Why ANNs for Sideslip estimation?**

- ✓ no vehicle model required
- ✓ no additional sensor required
- ✓ low computational cost when deployed on ECUs
- ✓ effective description of non-linear dynamics
- × huge amount of training data
- × high perfomance processor for training phase

Co-funded by the Erasmus+ Programme of the European Union

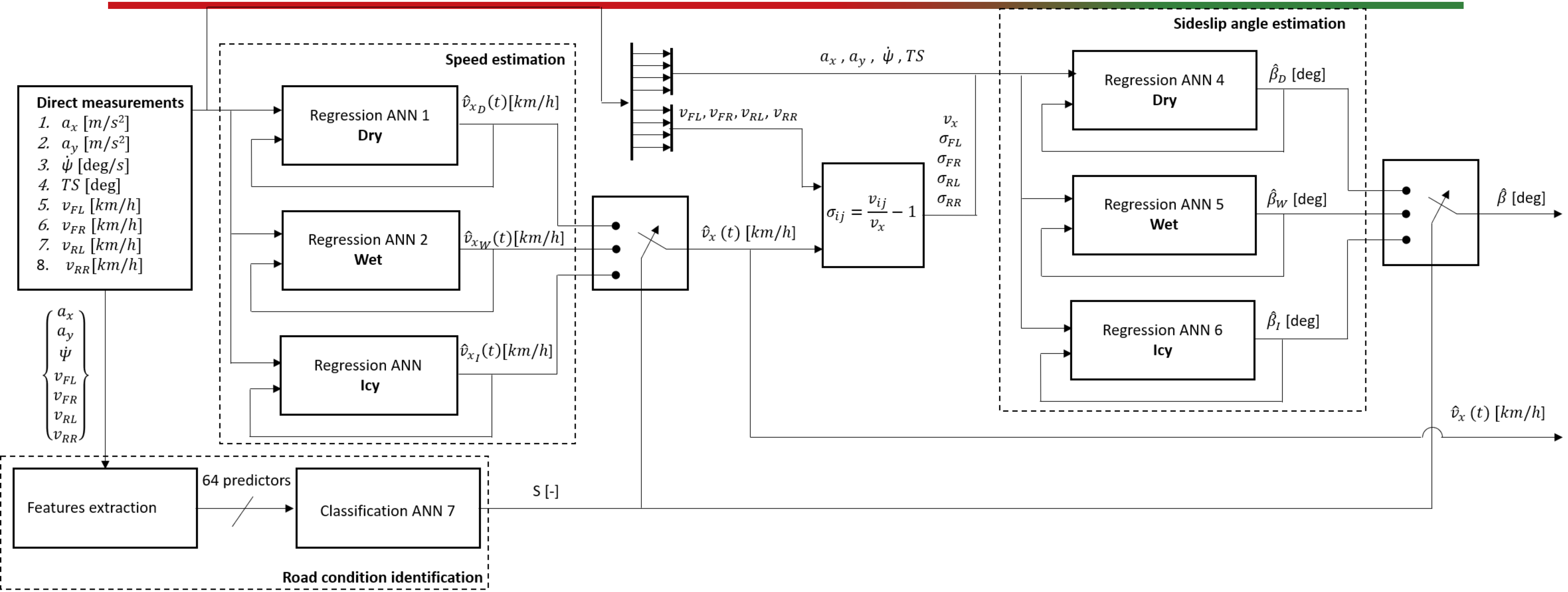
Overall system - Layout



1) **Sideslip Angle Estimation**
Regression ANNs

2) **Road Condition Identification**
Classification ANN

Overall layout



Co-funded by the
Erasmus+ Programme
of the European Union

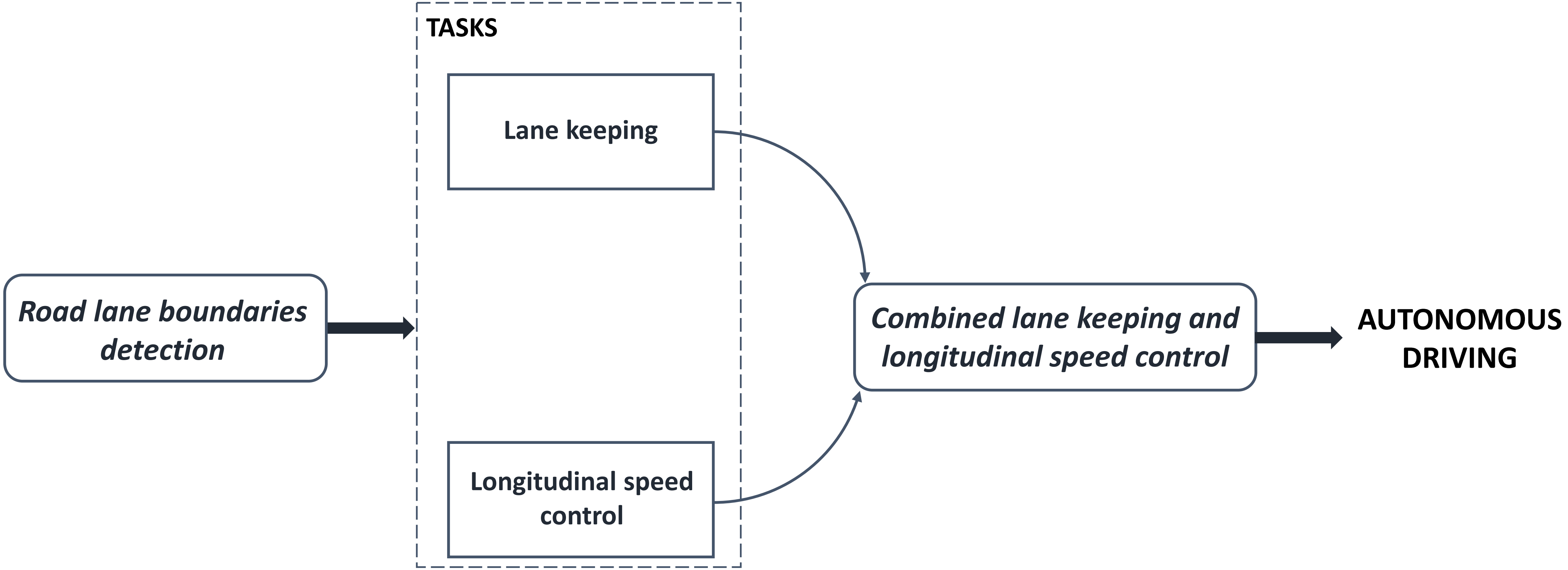
Overview of control techniques for ADAS

- Fuzzy Logic
- PID
- Model Predictive Control

Co-funded by the
Erasmus+ Programme
of the European Union

❑ Lane Keeping and longitudinal speed control

- the design of an algorithm for both lane keeping and longitudinal speed control for autonomous driving is investigated
- the algorithm must optimize the vehicle longitudinal speed, while respecting the vehicle dynamics constraints



Co-funded by the Erasmus+ Programme of the European Union

❑ Autonomous vehicle modeling

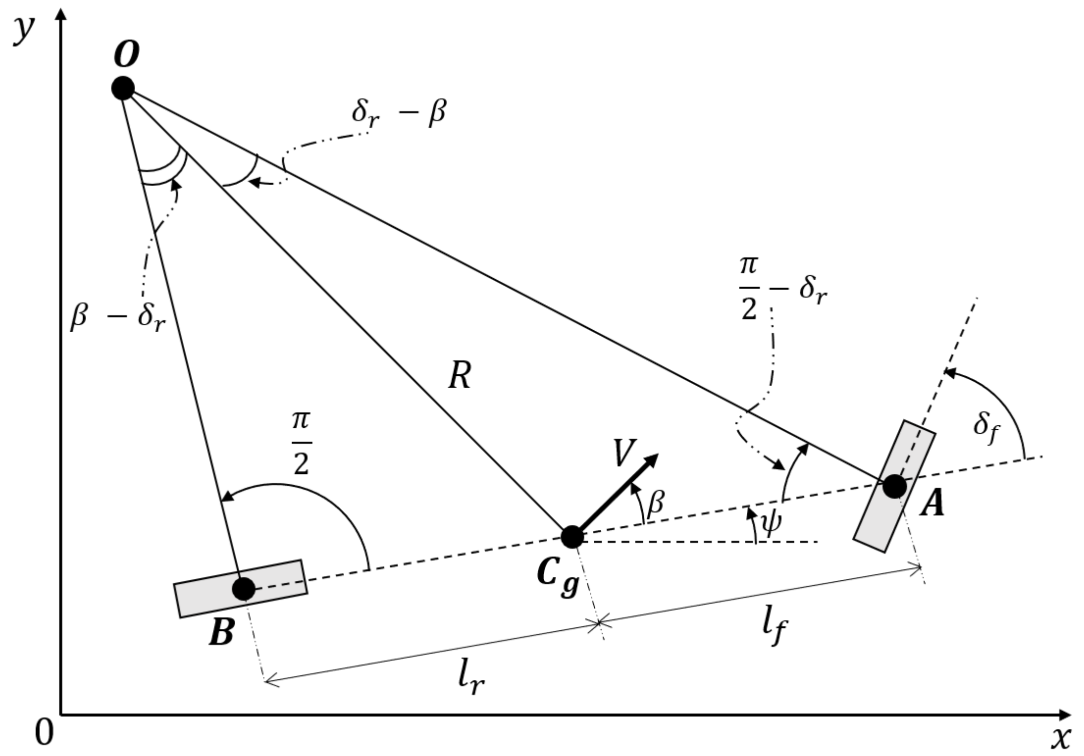
- The autonomous vehicle's dynamics is modeled with a 3 DoF Linear Kinematic Bicycle Model

- The vehicle plant $P_v(s)$ is a linear transfer function:

$$P_v(s) = \frac{1}{s(0.5s+1)}$$

Characteristic parameters of the considered vehicle.

Parameter	Value	Unit
m	1575	$[kg]$
I_z	2875	$[kg \cdot s^2]$
l_f	1.2	$[m]$
l_r	1.6	$[m]$
$C_{\alpha f}$	19000	$[N/rad]$
$C_{\alpha r}$	33000	$[N/rad]$



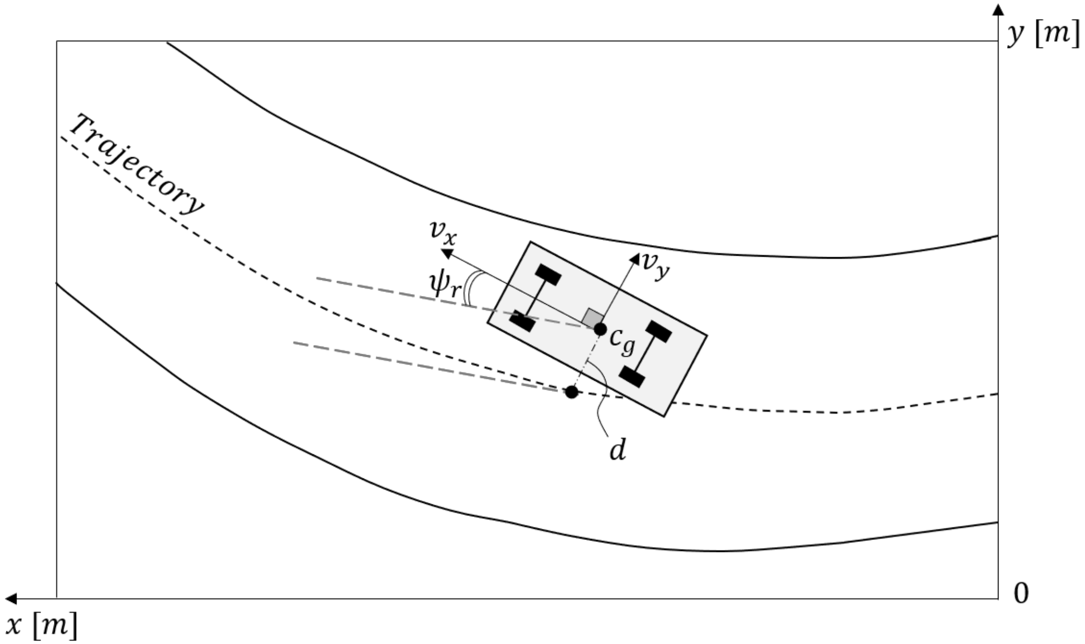
Co-funded by the
Erasmus+ Programme
of the European Union

❑ Autonomous vehicle modeling

- The controlled variables are:
  1. Lateral deviation d computed with respect to the reference trajectory
  2. Relative yaw angle ψ_r computed with respect to the reference vehicle's orientation

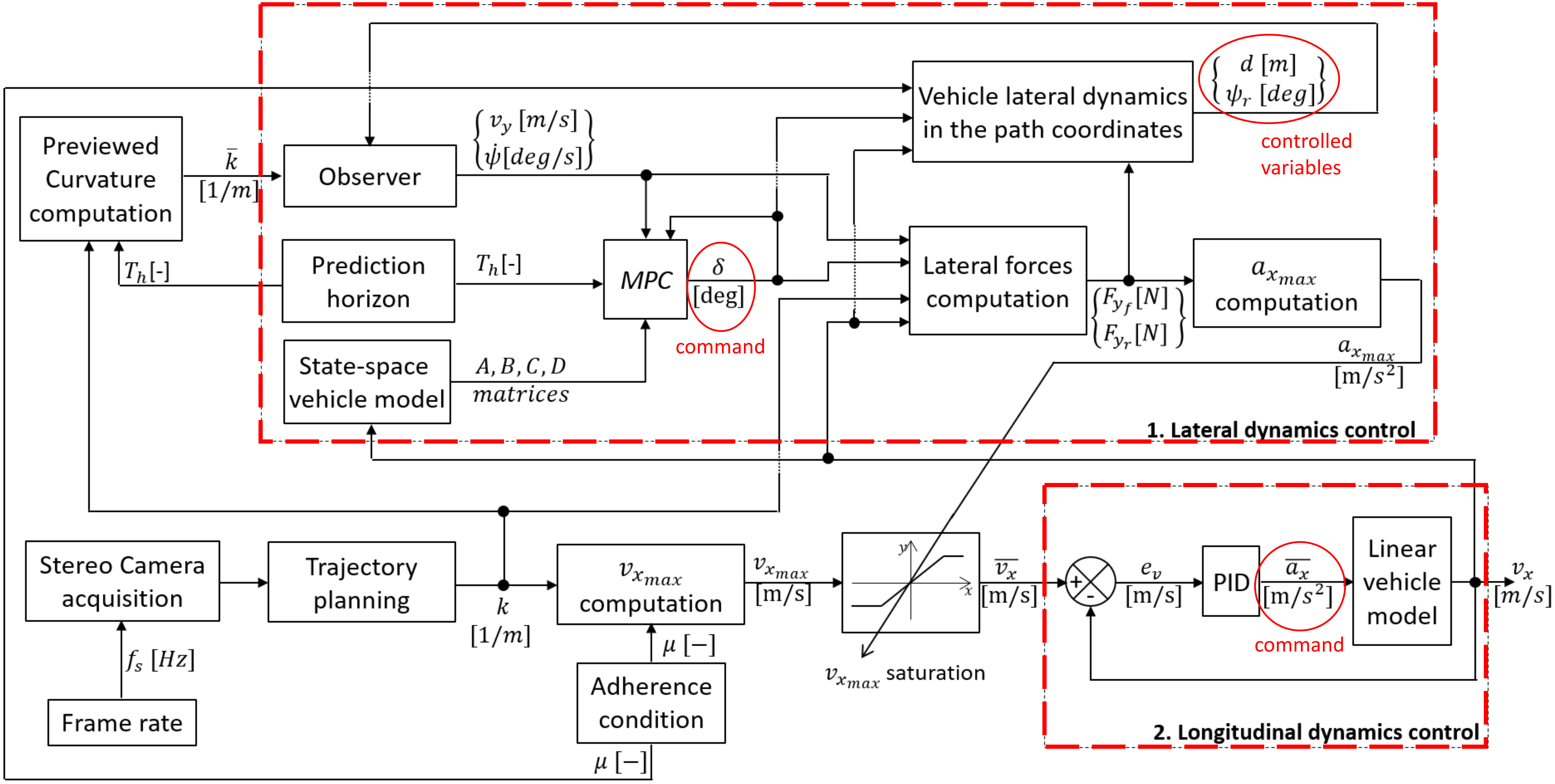
- The command signals are:
  1. Longitudinal acceleration $\overline{a_x}$
  2. Steering angle δ

- The command signals are generated considering the limitations resulting from the lateral and longitudinal dynamics of the vehicle, and the adherence limit constraints.



Co-funded by the Erasmus+ Programme of the European Union

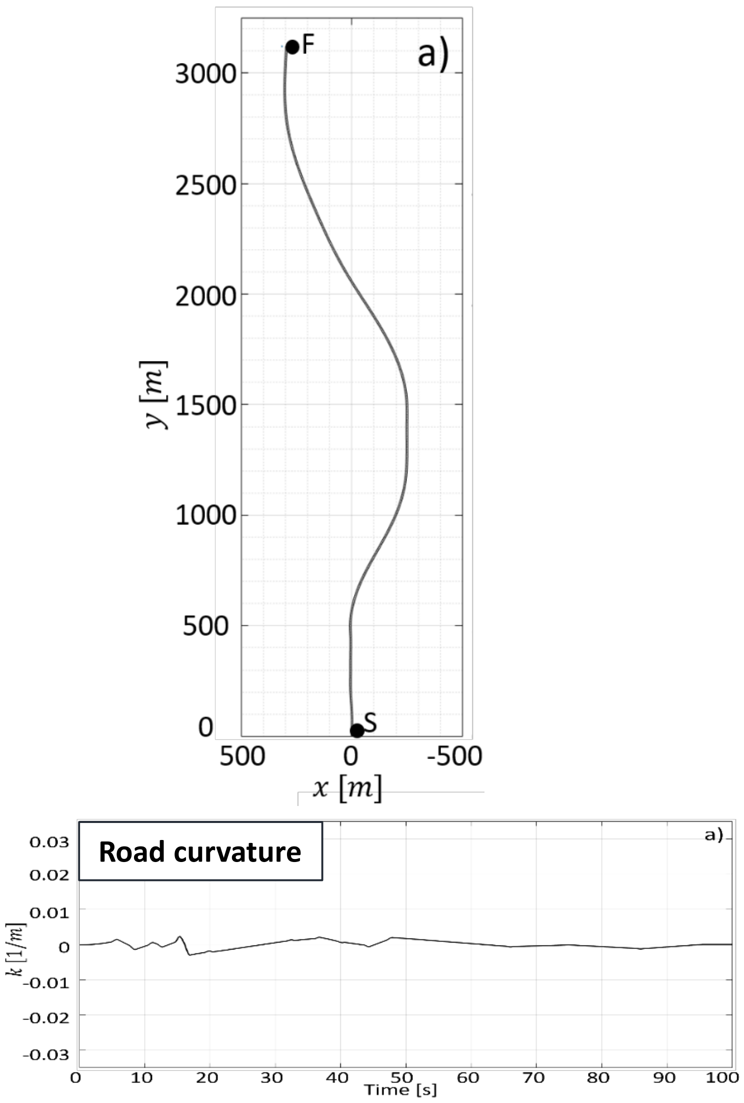
❑ Control architecture



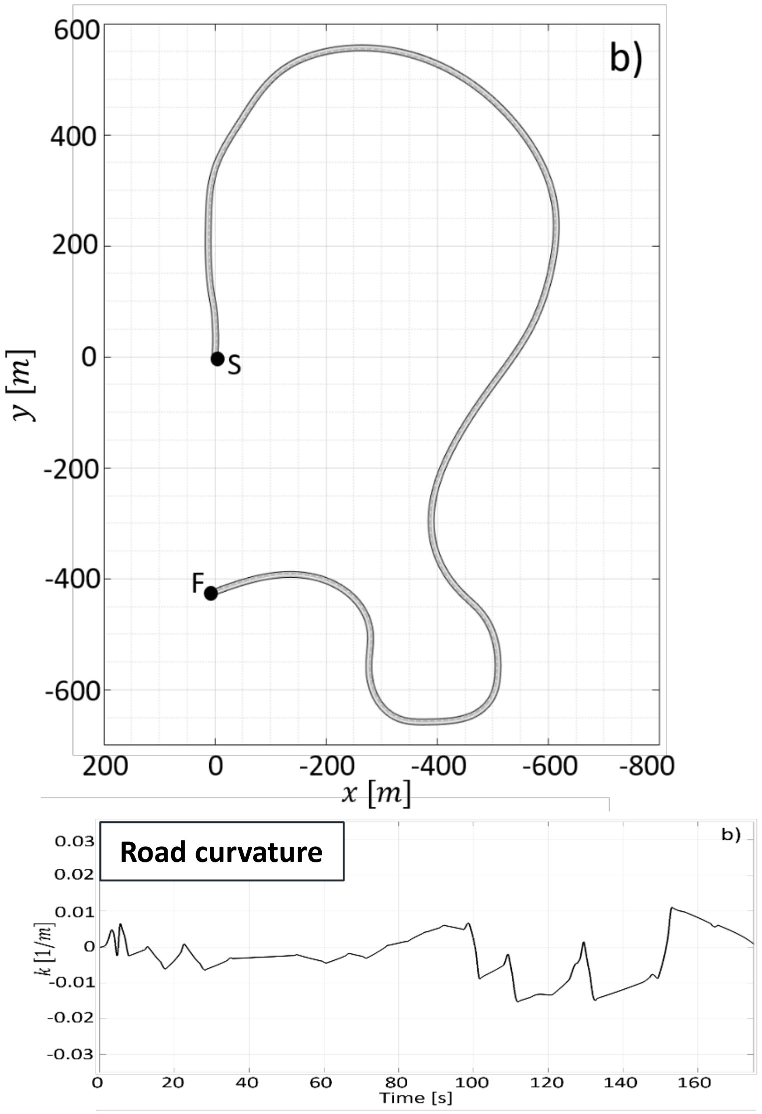
Co-funded by the Erasmus+ Programme of the European Union

❑ Driving scenarios

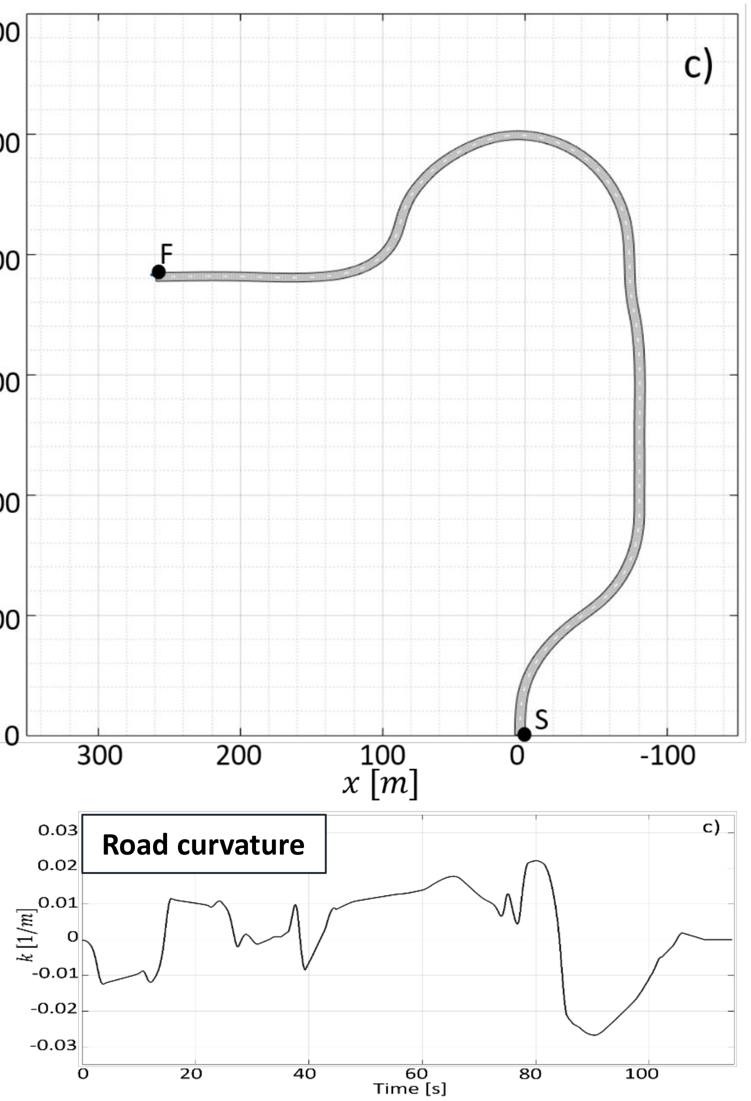
1 – Highway



2 – Inter-urban

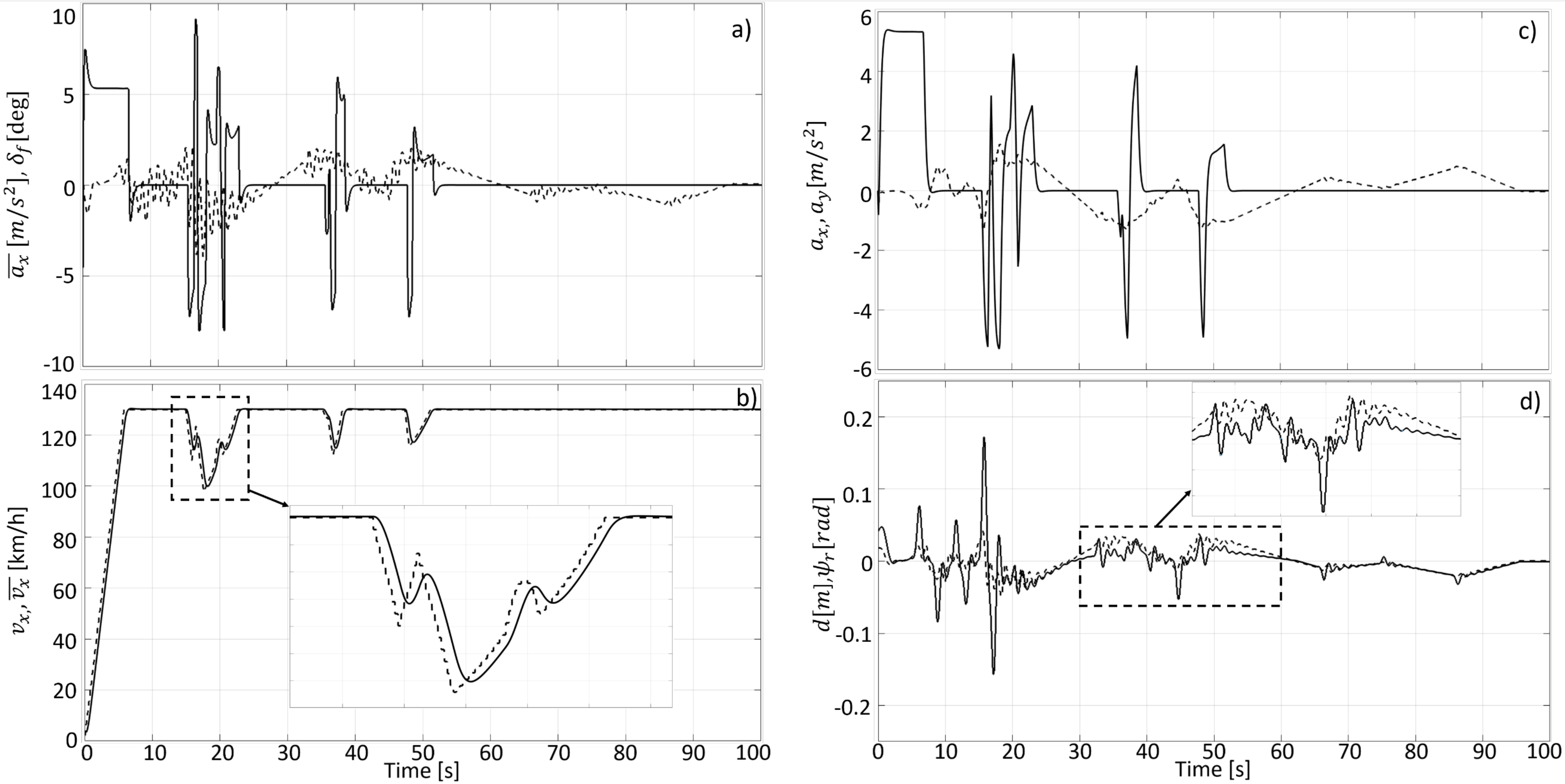


3 – Urban



Co-funded by the
Erasmus+ Programme
of the European Union

❑ Results – Driving scenario 1

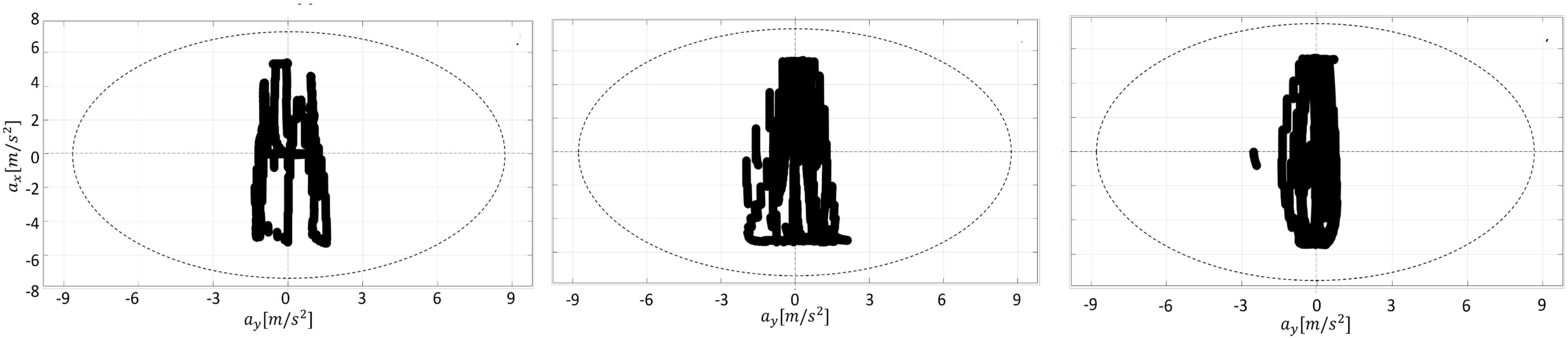


Co-funded by the
Erasmus+ Programme
of the European Union

❑ Results – Adherence limit constraints

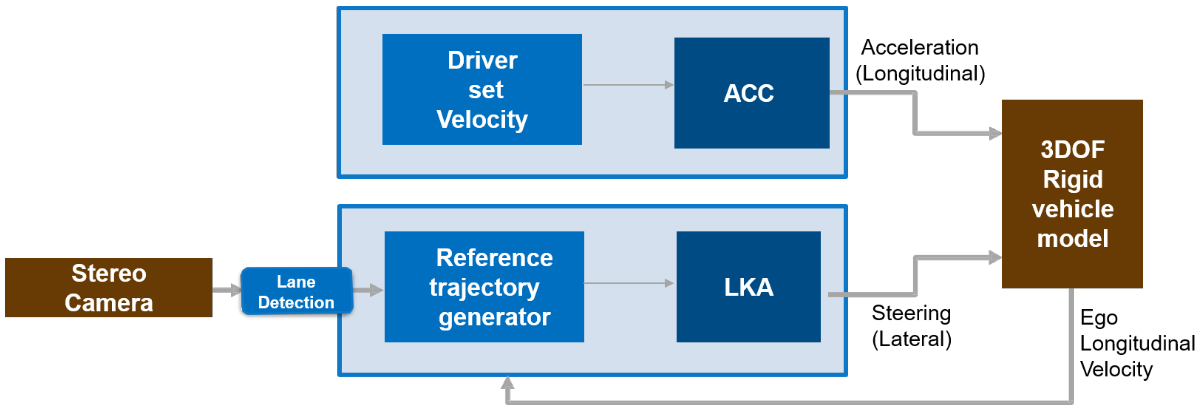
- The adherence limit constraints are respected in all the considered driving scenarios

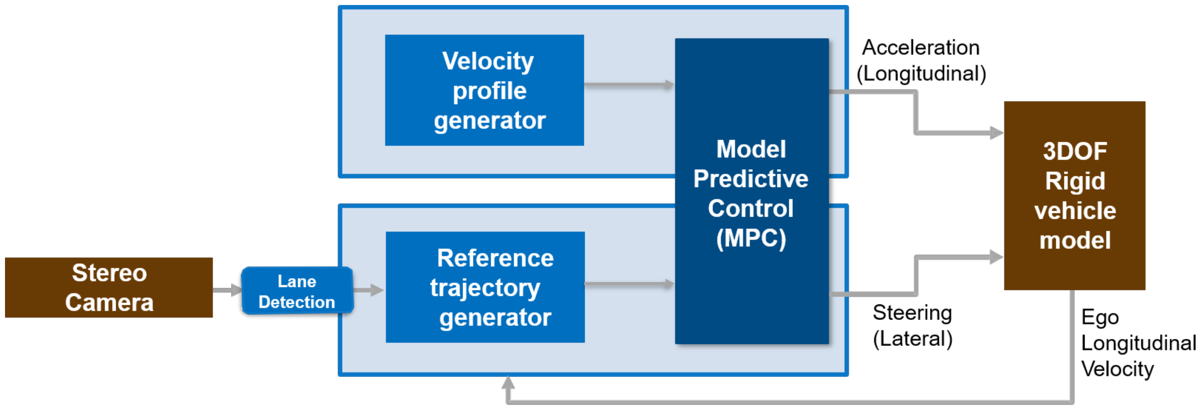
1 – Highway **2 – Inter-urban** **3 – Urban**



Co-funded by the Erasmus+ Programme of the European Union

Introduction: State of the art





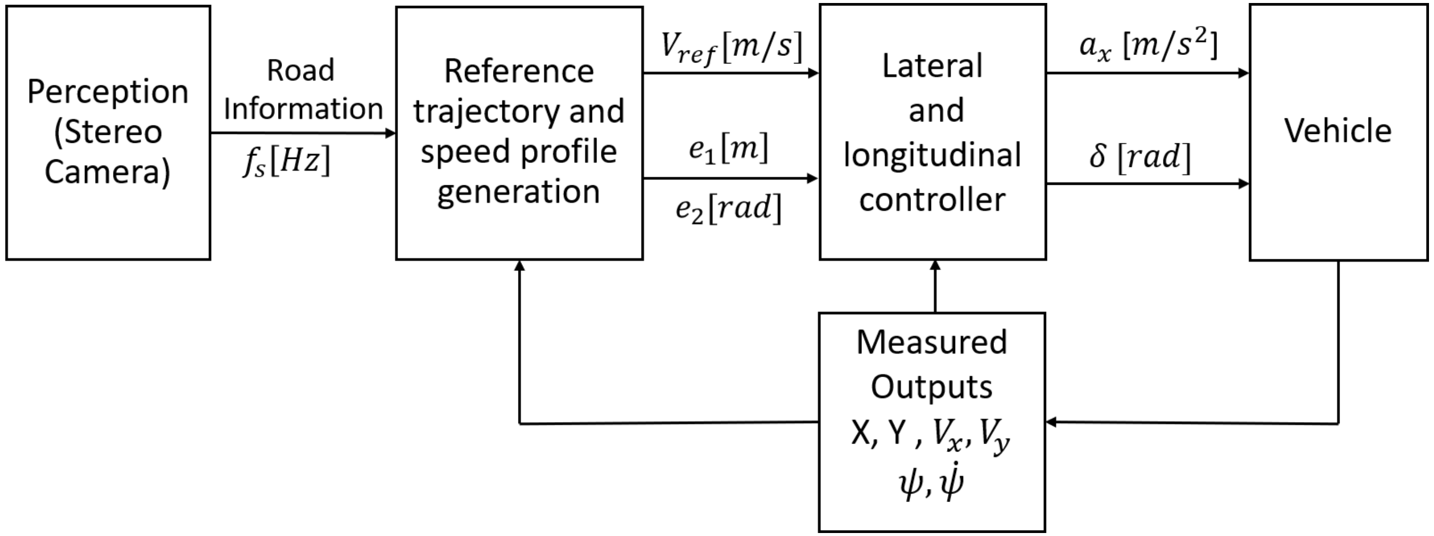
- Two different controllers are implemented for lateral and longitudinal control.
- Combined lateral and longitudinal control.
  - To improve the effectiveness of control in a large operating range.

- Non linear Model Predictive Control (NLMPC) are present in literature.
- An alternative approach based on Adaptive MPC with simpler optimization problem is considered.

Co-funded by the Erasmus+ Programme of the European Union

Introduction: Global architecture of the control strategy

- **Perception:** It defines the environment in which the vehicle drives.
- **Reference generation:** It calculates the geometric trajectory which defines the path to be followed as well as the reference speed profile.
- **Control:** It ensures the automated vehicle guidance along the generated trajectories providing the appropriate control signals.



Global architecture of the control strategy for autonomous driving presented in this thesis.

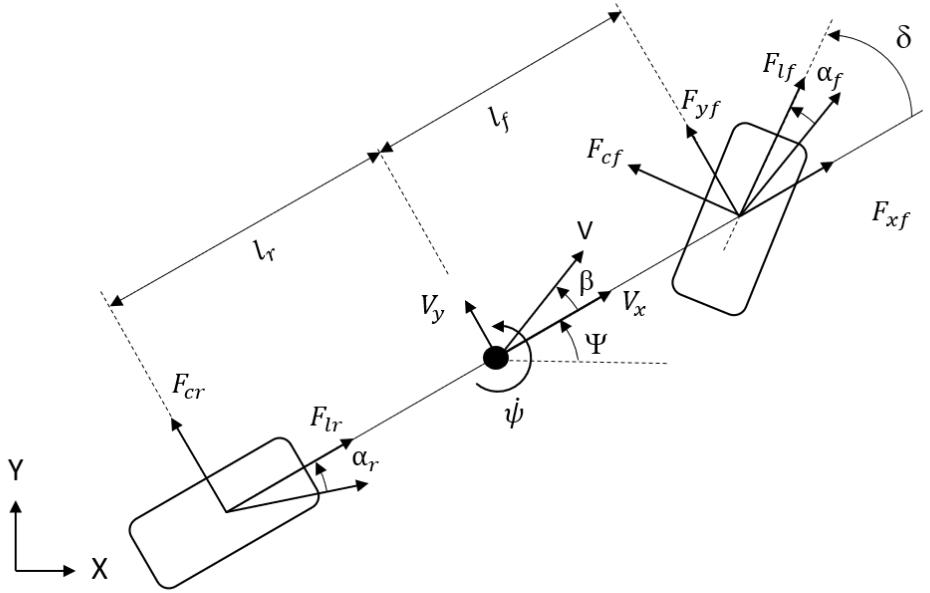
Co-funded by the Erasmus+ Programme of the European Union

Modelling: Vehicle model for validation and simulation

3 degree of freedom rigid vehicle model (Single Track).

Newton Euler equations (1), (2) denote the longitudinal and lateral momentum with respect to CG in the vehicle reference frame. While, yaw dynamics are considered by (3).

$$m\dot{V}_x = mV_y r + F_{xf} + F_{xr} - F_{aero} \quad (1)$$
$$m\dot{V}_y = -mV_x r + F_{yf} + F_{yr} \quad (2)$$
$$I_{zz}\ddot{\psi} = l_f F_{yf} - l_r F_{yr} \quad (3)$$



The forces acting on the vehicle center of gravity are related to tires forces and front steering angle δ by the equations (4)-(8).

$$F_{xf} = F_{lf}\cos\delta - F_{cf}\sin\delta \quad (4)$$
$$F_{yf} = F_{lf}\sin\delta + F_{cf}\cos\delta \quad (5)$$
$$F_{xr} = F_{lr} \quad (6)$$
$$F_{yr} = F_{cr} \quad (7)$$

Where,
F_l, F_c are the longitudinal and lateral tire forces, respectively.
F_x, F_y are the longitudinal and lateral forces acting on the vehicle center of gravity

Co-funded by the Erasmus+ Programme of the European Union

Vehicle model for MPC

The longitudinal dynamics of the plant model used for control design is approximated by the following model (Rajamani, 2012)

$$\tau \ddot{V}_x + \dot{V}_x = a \qquad (14)$$

The transfer function between desired acceleration (a) and actual vehicle speed (V_x) is given by:

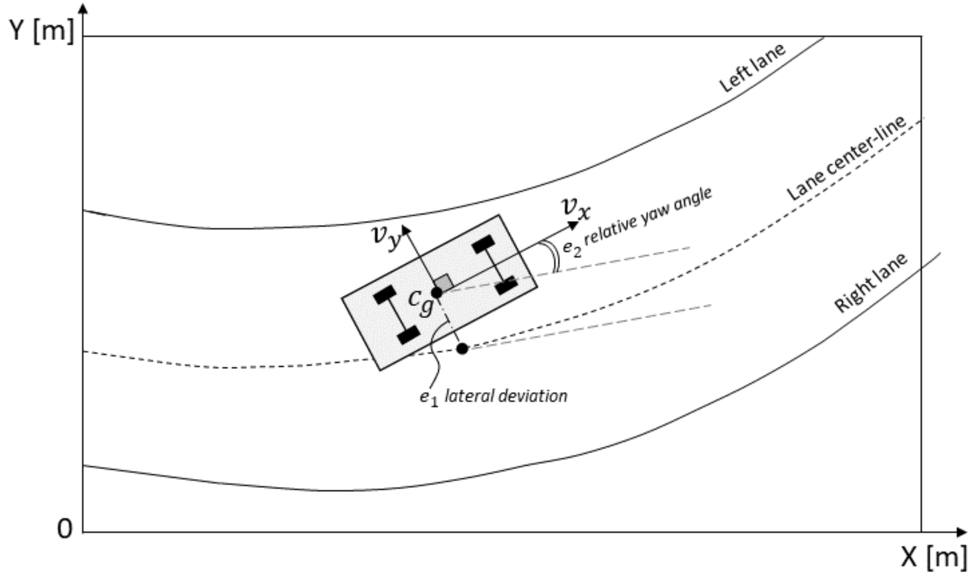
$$P(s) = \frac{1}{s(\tau s+1)} \qquad (15)$$

A 2 degree of freedom vehicle model is used to define the lateral dynamics of the vehicle for controller's internal plant model in terms of errors with respect to the reference trajectory.

$$\dot{e_1} = V_x e_2 + V_y \qquad (16)$$

$$e_2 = \psi - \psi_d \qquad (17)$$

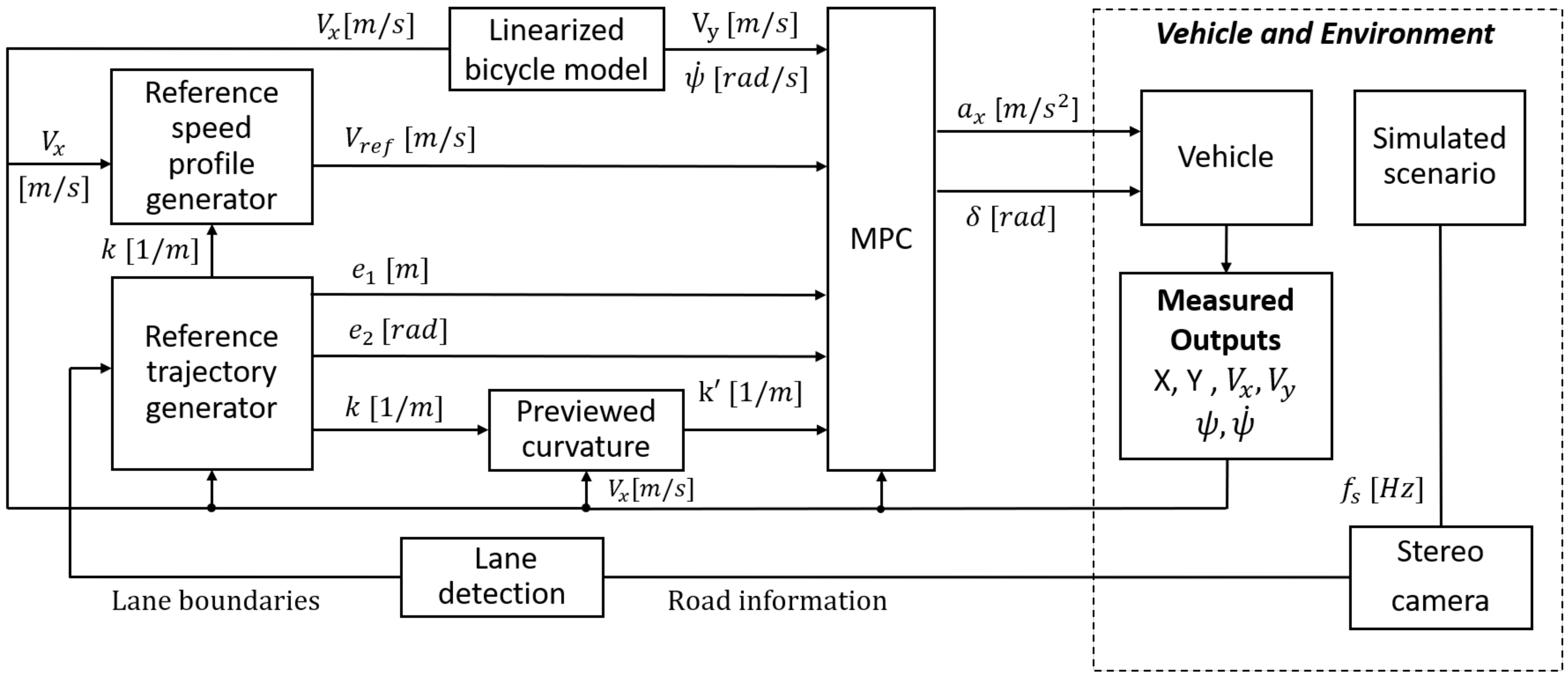
$$\dot{\psi}_d = \frac{V_x}{R} = V_x k \qquad (18)$$



Definition of lateral deviation (e_1) and relative yaw angle (e_2) with respect the center line of the lane

Co-funded by the Erasmus+ Programme of the European Union

Control: Detailed architecture of the control strategy



Co-funded by the Erasmus+ Programme of the European Union

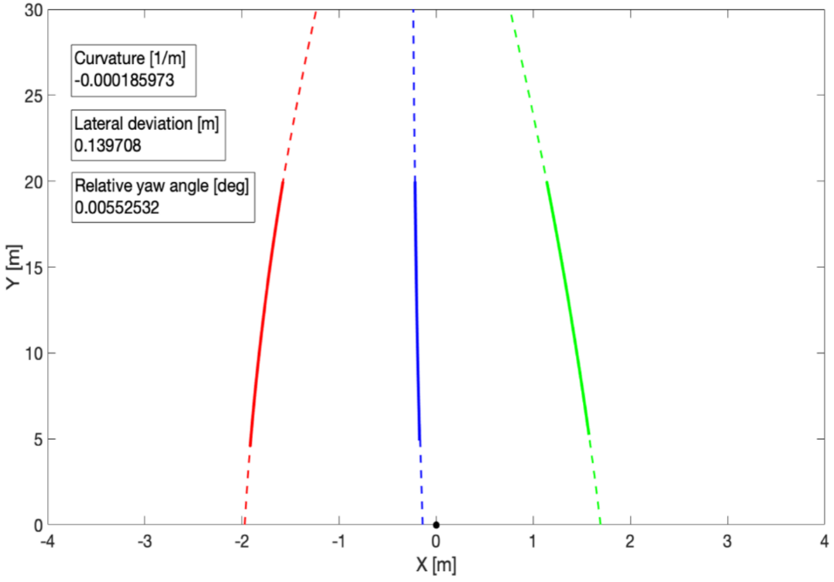
Lane detection and Reference trajectory generation

Stereo camera is implemented in Simulink using the Vision Detection Generator block from Automated Driving Toolbox. Which generate vision detections from simulated scenarios.

- Receives images acquired by the simulated stereo camera, with a frame rate equal to 10 Hz.
- Provides the equation of the left and right lane boundaries of the current lane in the current field-of-view of the stereo camera.

Computation of the reference trajectory. i.e. Center line of the lane. Vehicle model dynamic parameters are computed geometrically after the reconstruction of the lane:

- **Lateral deviation**: the distance of the center of gravity of the vehicle from the center line of the lane.
- **Relative yaw angle**: the orientation error of the vehicle with respect to the road.
- The absolute **curvature** $|k|$ of the center line at a point.

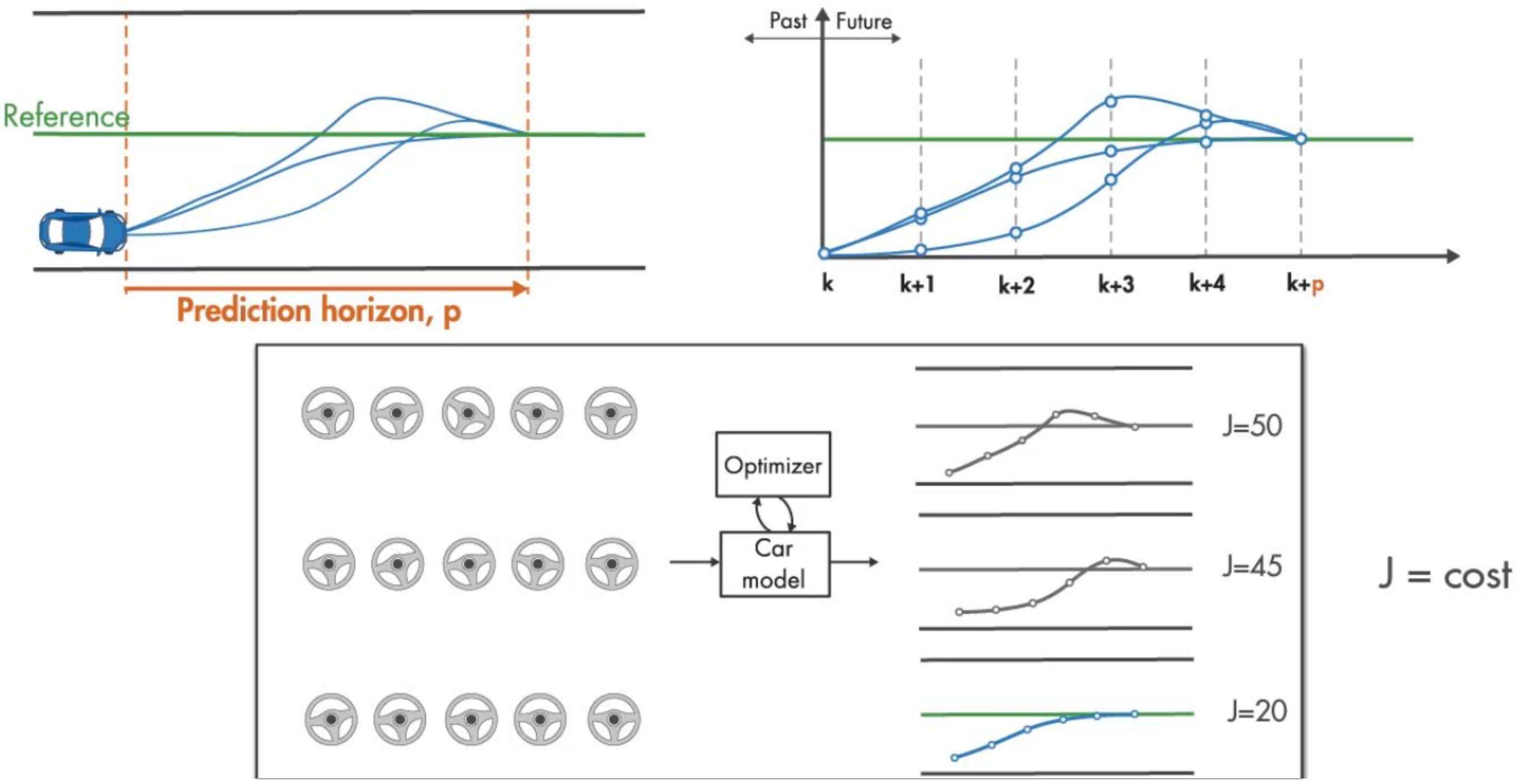


Source: MATLAB

Co-funded by the Erasmus+ Programme of the European Union

Model Predictive Control

- Ability to work with constraints both on the states and the control signals.
- Its preview capability and possibility to work with MIMO system.



MPC Controller

Source: MATLAB

The main parameters for the performance of the MPC are:

- Prediction Horizon (N)
- Control Horizon (H)
- Sampling time (T_s)

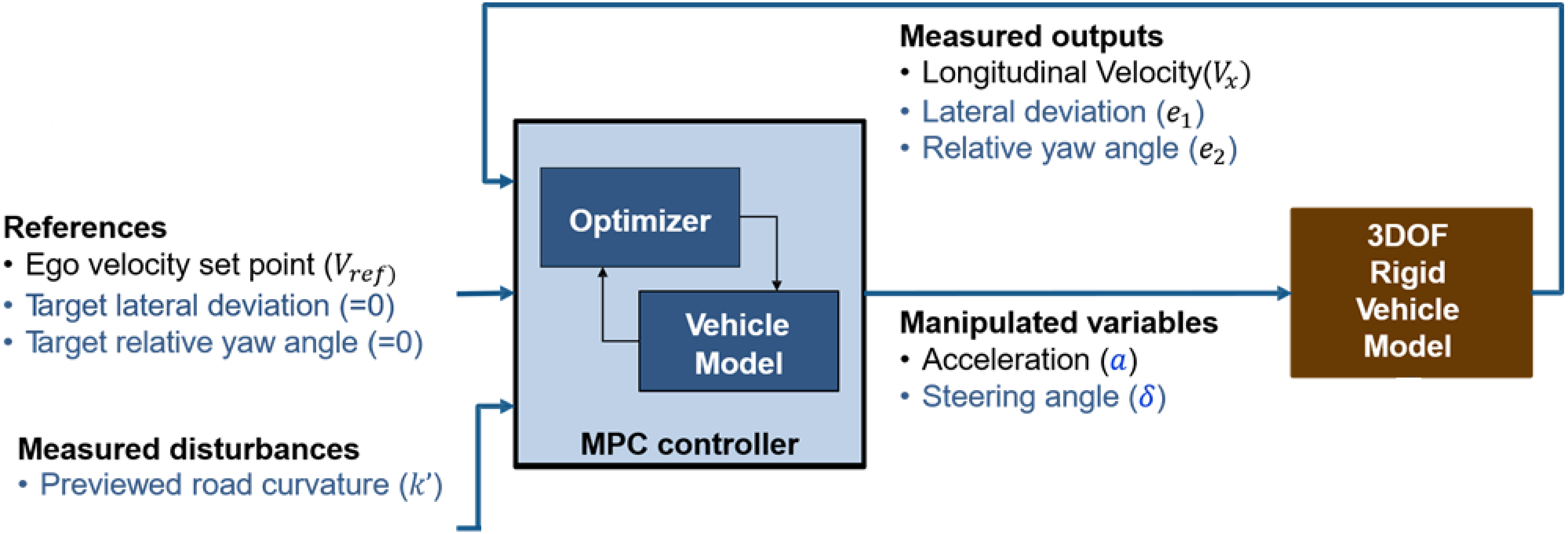
Here, we set N = 20 steps and H = 5 steps.

- **Cost function:**

$$\min J = \sum_{j=1}^{N} \left\| y_p(k+i) - y_{ref}(k+i) \right\| Q_y + \sum_{j=0}^{H} \left\| \Delta u(k+i) \right\| R_u$$

Co-funded by the
Erasmus+ Programme
of the European Union

Model Predictive Control: Problem formulation



While, the state vector for reference is given by:

$$[V_{ref} \quad 0 \quad 0]^T$$

The state vector of the feedback states to the MPC is given by:

$$[V_x \quad e1 \quad e2]^T$$

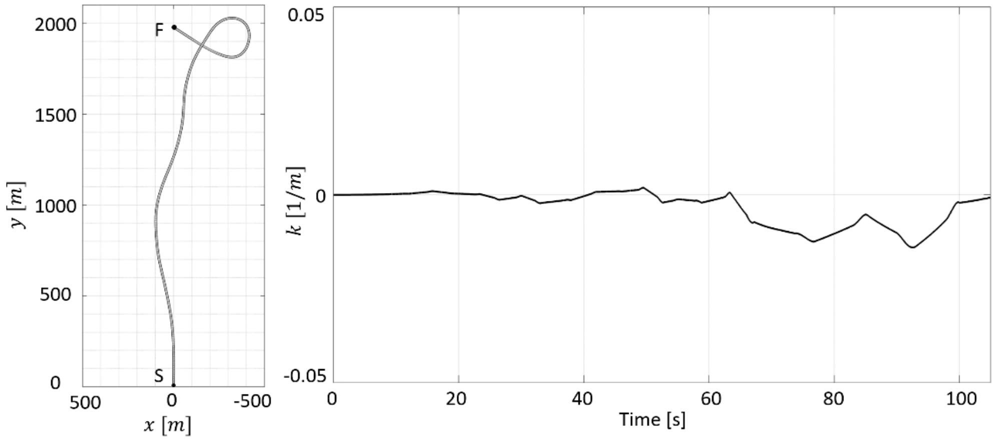
The control output of the MPC are given by:

$$[a \quad \text{?}]^T$$

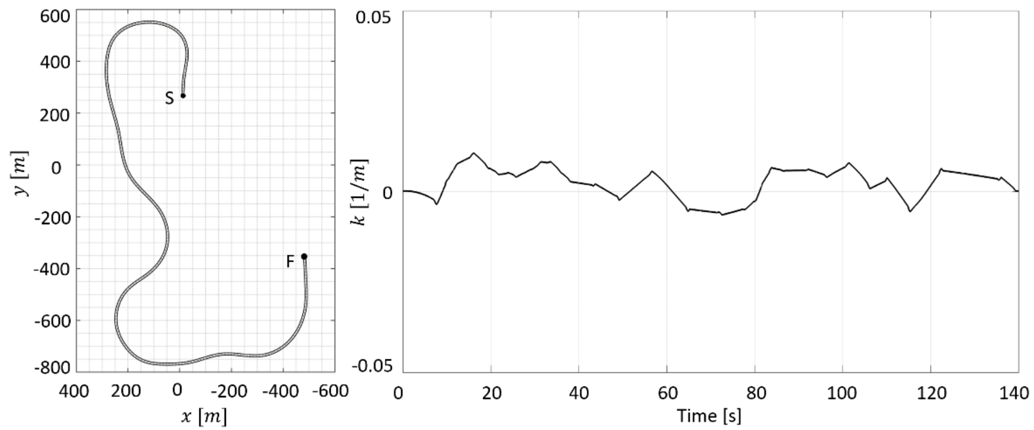
Co-funded by the Erasmus+ Programme of the European Union

Driving scenarios

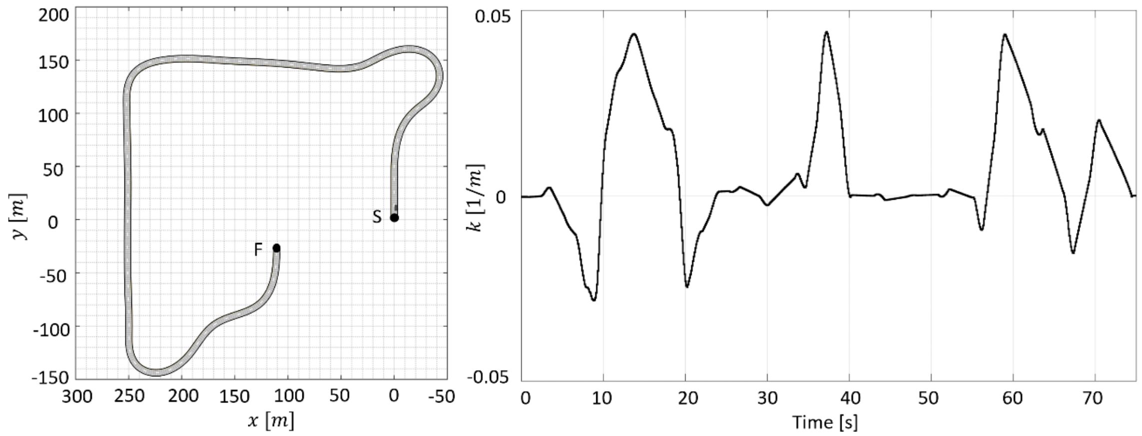
Scenario 1. Highway driving



Scenario 2. Inter Urban driving:

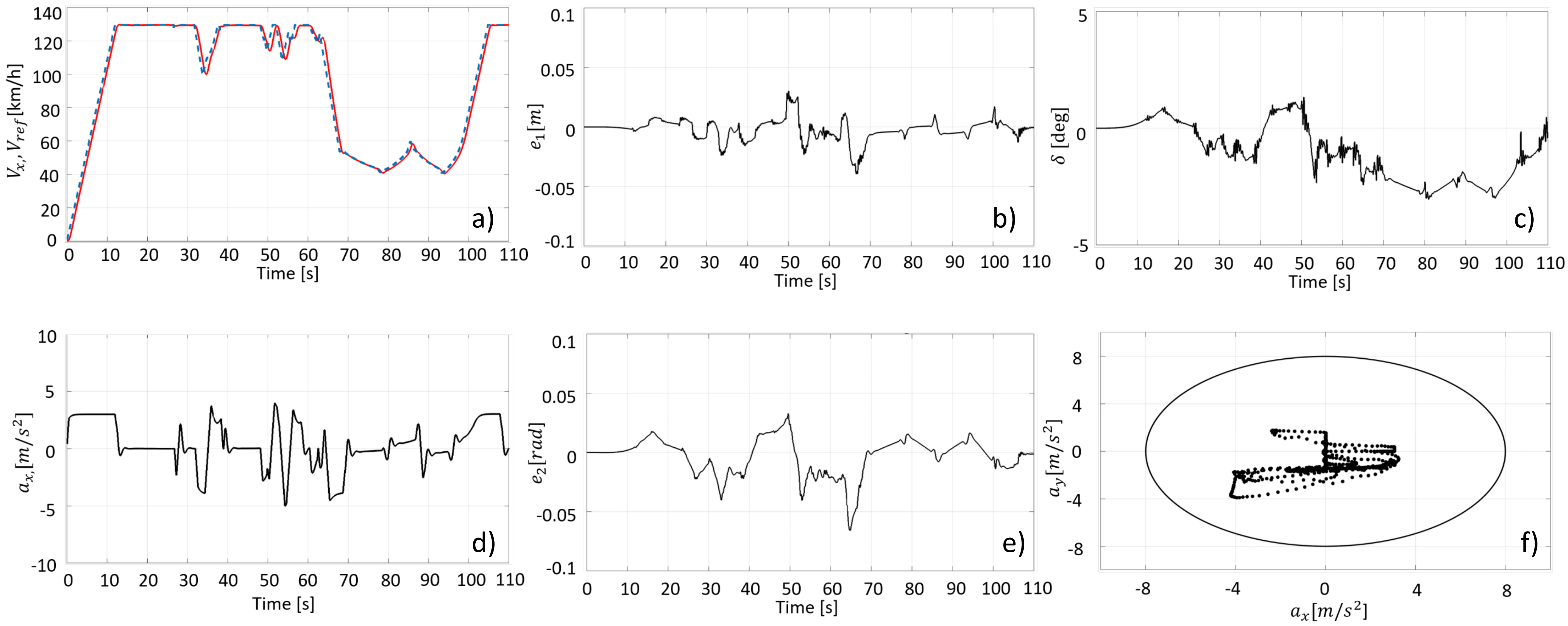


Scenario 3. Urban driving



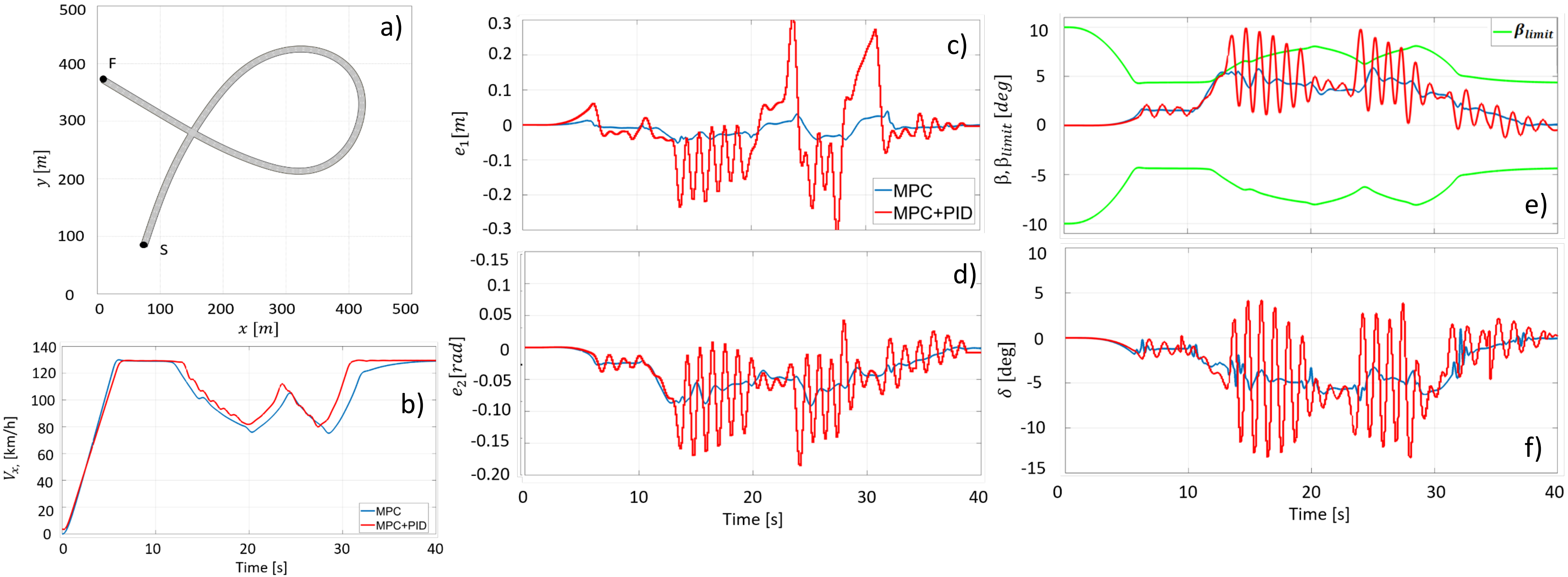
Co-funded by the Erasmus+ Programme of the European Union

Results and discussion: Highway



Co-funded by the
Erasmus+ Programme
of the European Union

Results



a) Highway exit driving scenario

b) Vehicle's longitudinal speed (V_x)

c) Lateral deviation (e_1)

d) Relative yaw angle (e_2)

e) Vehicle side slip angle β with βlimit

f) Front wheels steering angle command δ

Co-funded by the
Erasmus+ Programme
of the European Union

Overview of charging stations for electric vehicles

- People are worried about how far they can travel in electric vehicles (EVs), before their batteries run out of energy. As a matter of fact, most production EVs can only go few hundreds of kilometers with a single charge.

- Most of the existing charging stations are placed at home. Therefore, people who live in shared housing or use street parking will experience troubles in charging.

- Improving the infra structure and providing more public charging stations on highways and in cities can be a solution.

Co-funded by the Erasmus+ Programme of the European Union

Types of charging systems

Conductive Charging (Plug-in charging) systems

- Common charging technique
- Used in residential areas
- Requiring a connector between the Electric power source and the vehicle battery

Inductive Charging (Wireless charging) systems

- More recent charging technique, related to the electromagnetic induction
- Wireless charging coupled with magnetic resonance to transfer power
- Requiring transmitting and receiving infrastructures

Battery Switching systems

- Depleted batteries are switched with a fully charged battery pack in the switching station

Co-funded by the Erasmus+ Programme of the European Union

Conductive Plug-in: AC and DC charging

- **AC charging** – electricity is delivered with alternating current (AC), the Electric Vehicle (EV) requires direct current (DC), thus a rectifier is needed between the grid and the battery. This AC/DC conversion is performed by the EV on-board charger.

- **DC charging –** in DC fast charging stations, electricity from the grid is delivered to the vehicle without the need of a rectifier.

Co-funded by the
Erasmus+ Programme
of the European Union

Plug-in charging stations

- There are different levels of plug-in charging stations:

- **Level 1, 120 Volt Charging**

It uses a plug to connect to the on-board charger and a standard household outlet. This setup provides between 3-10 kilometers per hour of charge.

- **Level 2, 220/240 Volt Charging**

It provides power at 220 V or 240 V and up to 30 A. Drivers can add about 30 kilometers of range in an hour of charging at home or at a public station.

- **Level 3, 480 Volt Charging (DC Fast Charging)**

In this case, the charger is a gas pump-sized machine. All fast chargers deliver about 80% charge in a very short time.

Co-funded by the
Erasmus+ Programme
of the European Union

Wireless inductive charging

- The technology depends on the same principle of **electromagnetic induction** that enables a transformer to change the voltage of an alternating current. This current flows through one coil of wire, creating a magnetic field whose polarity reverses with each cycle and inducing a corresponding alternating field in a secondary coil.

- If the two coils are separated by air, current flowing through the first coil will create a magnetic field, which will still be picked up by the second coil. The greater the air gap, the less efficient the transfer of power will be. The resulting current in the second coil can charge the vehicle's battery.

- The system can be used also in motion enabling the future application of electric charging lane on the highways.

Co-funded by the Erasmus+ Programme of the European Union

SOC estimation techniques

UNITED

Direct methods

- Coulomb counting
- Open circuit voltage measurement
- Discharge test
- Impedence spectroscopy

Indirect Methods

Model based

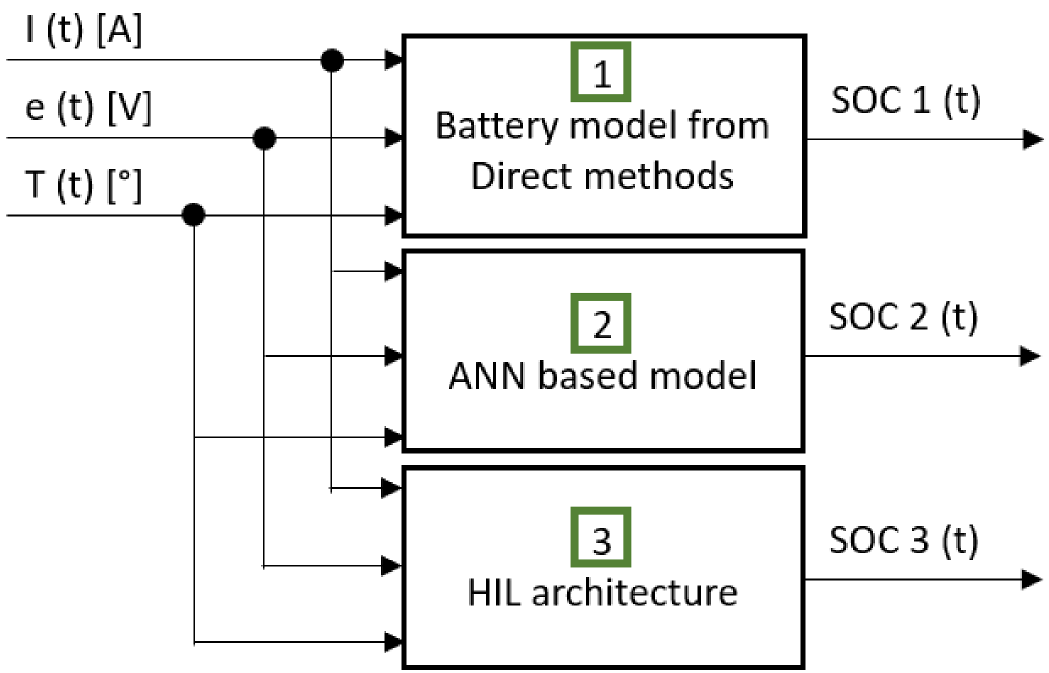
- Kalman Filter (KF)
- Extended Kalman Filter (EKF)
- Smooth Variable Structure Filter (SVSF)

Adaptive techniques

- Artificial Neural Network (ANN)
- Fuzzy Logic (FL)

Co-funded by the
Erasmus+ Programme
of the European Union

Test setup



The battery model refers to a battery pack (nominal voltage: 48V, nominal capacity: 60 Ah), composed by 156 LiPo cells (configuration: 13p12s)

❑ 1 – Battery model from Direct methods

- high computational cost
- not applicable for on-board SOC estimation
- it emulates battery behaviour
- it generates training datasets for **Model 2**
- SOC 1 is the reference

❑ 2 – ANN based model

- it performs the SOC estimation by simulation
- ANN is trained with data provided by **Model 1**
- SOC 2 is the estimate

❑ 3 – HIL architecture

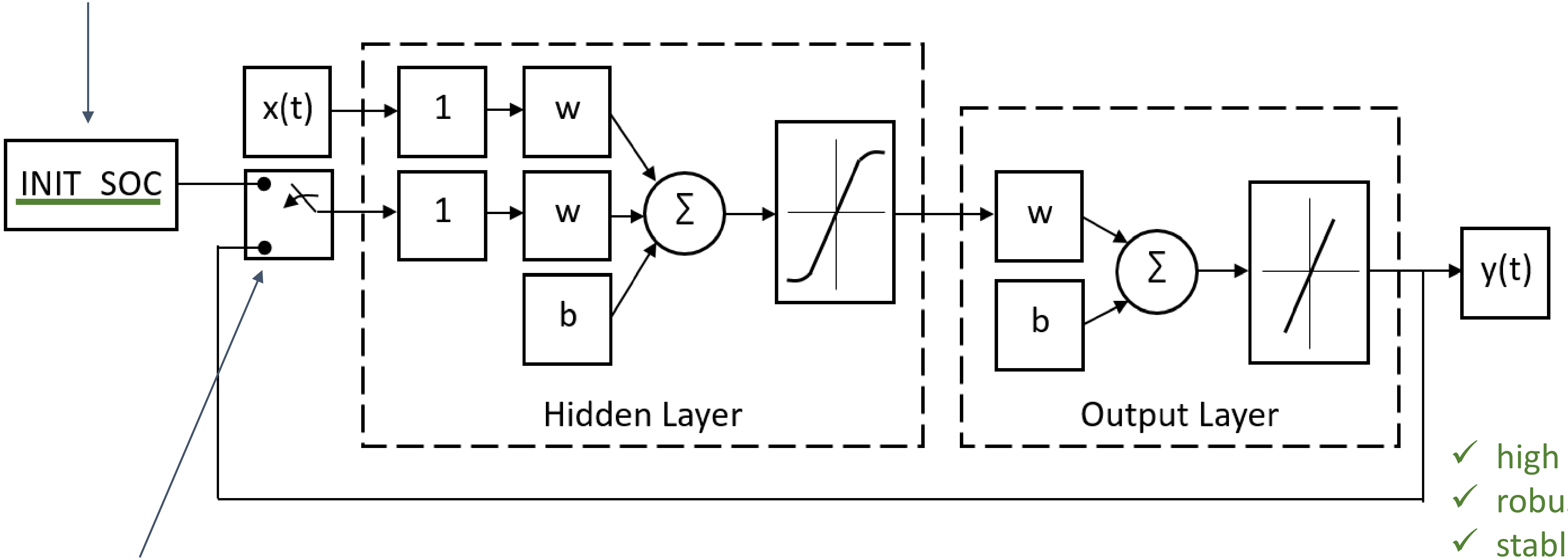
- it is the experimental validation of **Model 2**
- Model 2 is built on a **hardware** board
- SOC 3 is the estimate

estimation goal: SOC 1 = SOC 2 = SOC 3

Co-funded by the Erasmus+ Programme of the European Union

❑ 5. Closed-loop NARX ANN with a known initial SOC state

- it is needed for an accurate estimate
- it is stored in the BMS Flash memory

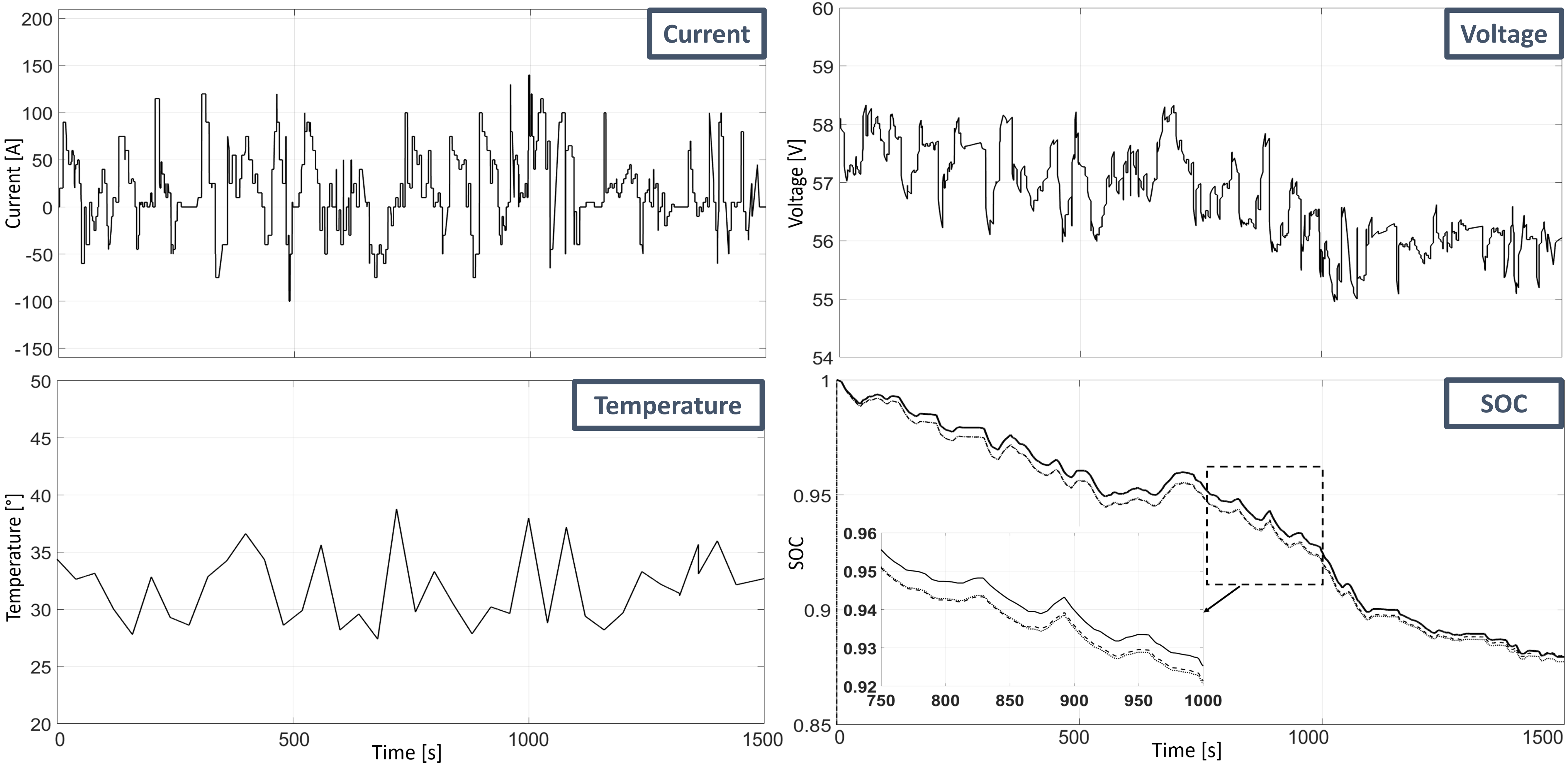


- the switch changes the input to the fed-back estimated SOC value (after 1 second)

✓ high accuracy
✓ robust
✓ stable
✓ fast training phase
✓ light computational cost

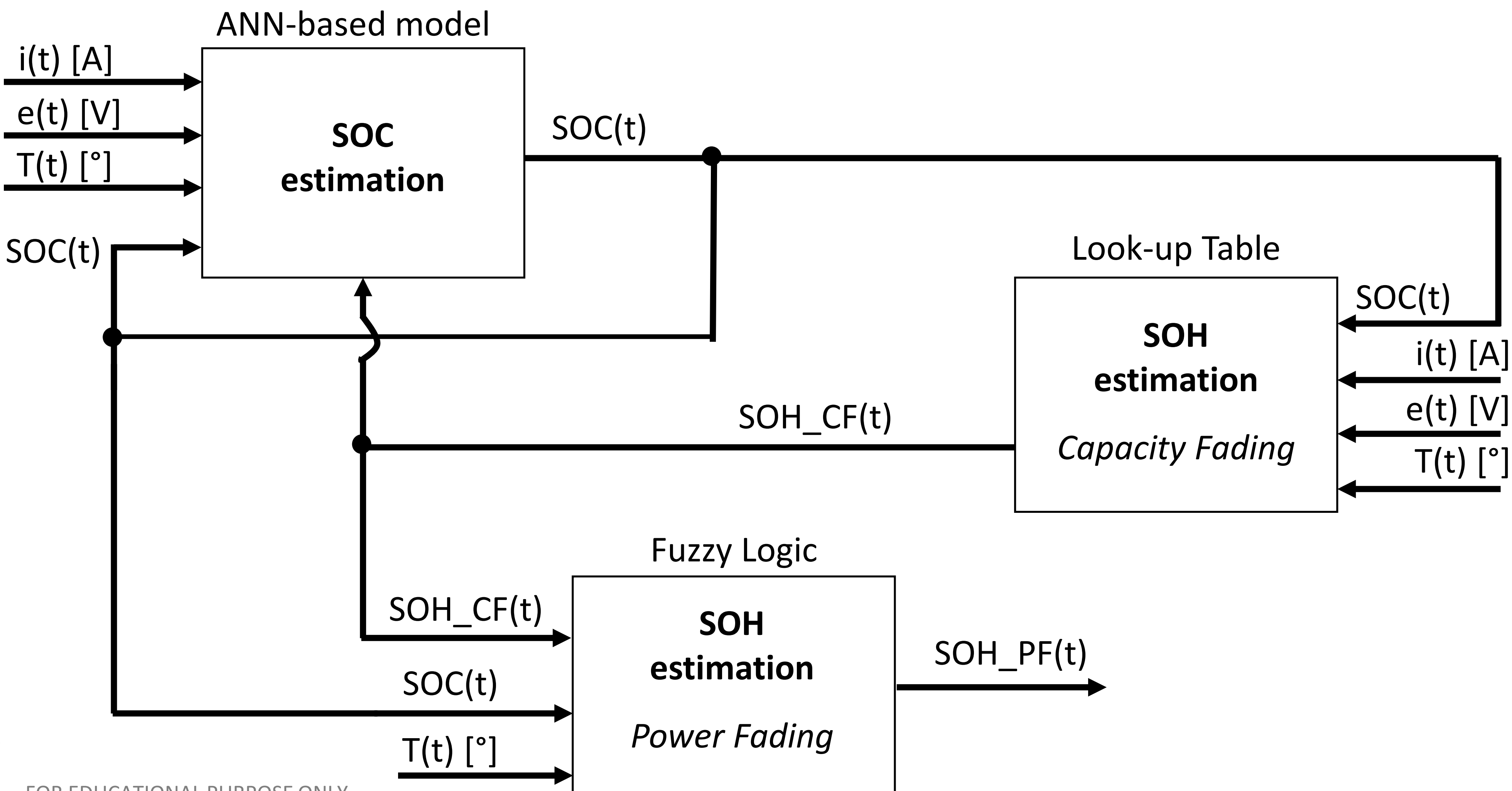
Co-funded by the Erasmus+ Programme of the European Union

Validation on real cases – (1/2) Moderate driving style



Co-funded by the
Erasmus+ Programme
of the European Union

SOH analysis



Co-funded by the Erasmus+ Programme of the European Union

Overview of connected vehicles

- A **connected vehicle** is a car that is equipped with Internet access, and usually also with is wireless local area network (LAN). This allows the car to share internet access and data with other devices both inside and outside the vehicle. For safety-critical applications, connected vehicles will also be connected using dedicated short-range communications (DSRC) radios, with a very low latency.

- General Motors was the first automaker to bring the first connected car features to market with OnStar in 1996. The primary purpose was safety and to get emergency help to a vehicle when there was an accident. OnStar only worked with voice but when cellular systems added data, the system was able to send the GPS location to the call center.

- In 2014, Audi was the first automaker to offer 4G LTE Wi-Fi Hotspots access in vehicles.

- In the future, the Internet of Things (IoT) will be used to provide mobile services in the car with high-speed internet connection to enable real time traffic control, interaction with the car manufacturer service for remote diagnostics and improved company logistics automation Moreover, the internet will be used for information exchange between the cars for better route selection and accident reports.

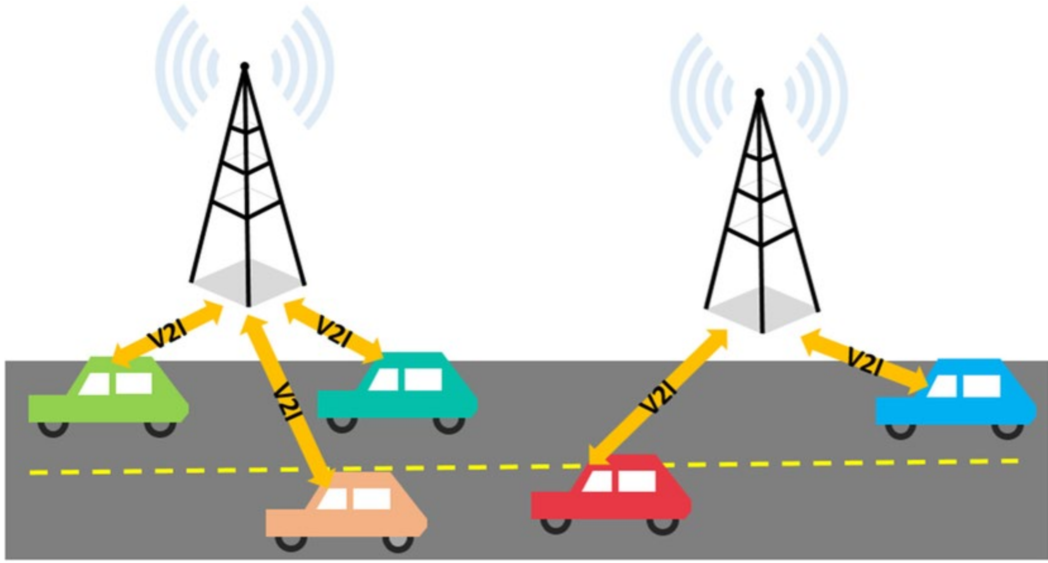
Co-funded by the Erasmus+ Programme of the European Union

Types of vehicles connectivity

- **Vehicle-to-Vehicle (V2V) Connectivity**
  It provides an increased safety and interaction since vehicles can communicate with each other and exchange information about dangerous situations or road conditions, such as ice, accidents, etc.

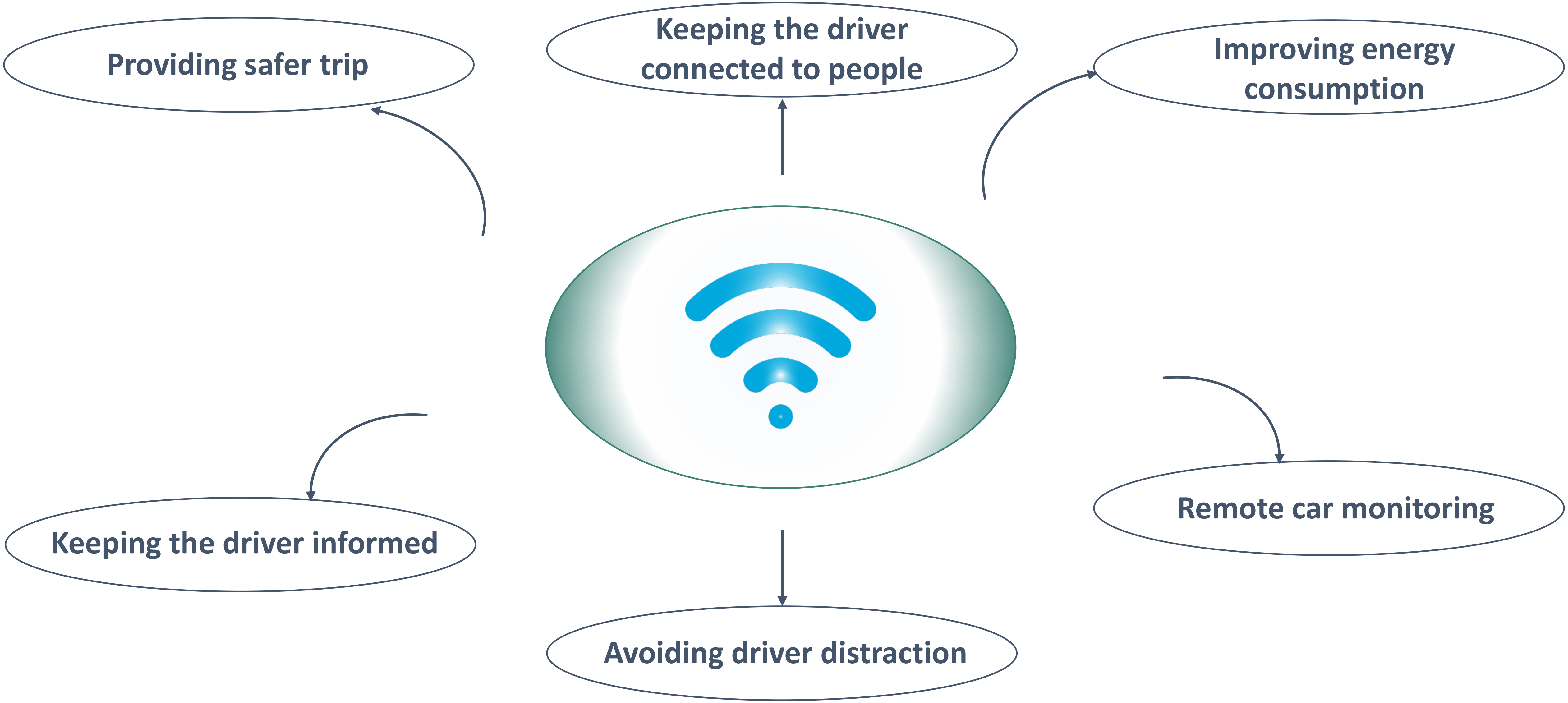
- **Vehicle-to-Infrastructure (V2I) Connectivity**
  It allows communication with any Internet-capable device that provides many tasks: traffic management, paying tolls, web services, remote diagnosis, connection to personal devices within car, etc.

- **Vehicle-to-Everything (V2X) Connectivity**
  This technology interconnects all types of vehicles and infrastructure systems with another. This connectivity includes cars, highways, ships, trains,airplanes, etc.



Co-funded by the
Erasmus+ Programme
of the European Union

Connected services



Co-funded by the Erasmus+ Programme of the European Union

Future challenges

- ❑ **Human Machine Interface (HMI) Design**
  - Still non-intuitive systems
  - Artificial Intelligence interfaces
  - Better customization
  - Larger usability
  - Improved gestures feedback

- ❑ **Convergence of multiple technologies into Standards**
  - Currently there are proprietary technologies only
  - Possible future custom integrations

- ❑ **Big Data and Cyber-security**
  - Artificial Intelligence services
  - Optimized car experience
  - Need for data privacy
  - Need for vehicles cyber-security

Co-funded by the
Erasmus+ Programme
of the European Union

UNITED

Engineering Knowledge Transfer Units to Increase Student´s Employability and Regional Development

https://www.facebook.com/unitederasmus/

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
598710-EPP-1-2018-1-AT-EPPKA2-CBHE-JP