HAD – open reference architecture for highly automated driving as base for efficient testing

Jürgen Ludwig

09. November 2016
1. Motivation
An ACC Developer‘s World in 2002

<table>
<thead>
<tr>
<th>Radar</th>
<th>Steering Wheel Angle</th>
<th>Motor Speed Control</th>
<th>Head Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>target vehicle candidates, their velocity / acceleration</td>
<td>target vehicle selection</td>
<td>ego vehicle speed control</td>
<td>system activation, status communication</td>
</tr>
</tbody>
</table>

Items to specify: 4
Driving Function Growth

- 1999: Mercedes S-Class Distronic
- 2002: VW Phaeton ACC
- 2002: Mercedes S-Class Distronic plus
- 2005: Audi Q7 ACC plus / AEB
- 2010: Audi A8 GPS-guided ACC
- 2013: Audi Q7 Traffic Jam Assistant
- 2013: Mercedes S-Class Dist.+ / Steering Assistant
- 2015: Audi Q7 Traffic Jam Assistant
A Driverless Car Developer’s World in 2016

<table>
<thead>
<tr>
<th>Sensors</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radar</td>
<td>Moving objects, Trajectory planning &amp; control</td>
</tr>
<tr>
<td>Camera</td>
<td></td>
</tr>
<tr>
<td>LIDAR</td>
<td>Static obstacles, Lanes, Signs, Vehicle ego motion</td>
</tr>
<tr>
<td>Sonar</td>
<td></td>
</tr>
<tr>
<td>Steering Wheel Sensors</td>
<td></td>
</tr>
<tr>
<td>Wheel Speeds</td>
<td></td>
</tr>
<tr>
<td>IMU / Gyro</td>
<td></td>
</tr>
<tr>
<td>GPS</td>
<td></td>
</tr>
<tr>
<td>Global Position</td>
<td></td>
</tr>
<tr>
<td>Maps</td>
<td></td>
</tr>
<tr>
<td>Motor Speed Control</td>
<td></td>
</tr>
<tr>
<td>Regenerative Braking</td>
<td></td>
</tr>
<tr>
<td>Brake Pressure</td>
<td></td>
</tr>
<tr>
<td>Steering Angle Control</td>
<td></td>
</tr>
</tbody>
</table>

**Items to specify:** 24
And all needs to be tested!

Items to specify: 24
What Does Help Reducing Complexity?

Interactions: $n \times m$ for $n$ sensors, $m$ functions
So What Does Help?

Interactions: $n + m + k$
for $n$ sensors, $m$ functions, $k$ abstraction components
Sensor Data Fusion:
From echos to objects and free space

Automated Driving

Automated Emergency Brake

Automated Valet Parking

HMI Management:
From buttons and LEDs to user interactions

Motion Management:
From brake pressure to trajectory control
From Exponentiality to Linearity

Well-defined interfaces

Automated Driving

Automated Emergency Brake

Automated Valet Parking

Well-defined interfaces

Well-defined interfaces

Well-defined interfaces

HMI Manager

Head Unit

Side Tasks

Steering Angle Control

Brake Pressure

Regenerative Braking

Motor Speed Control

Trajectory planning & control

Lateral Control

Longitudinal Control

Motion Manager
Best Practice Leverage: Project, Product Family, Industry

**Project-Level Architecture Best Practices:**
Reduce complexity

**Product Family Architecture Best Practices:**
Re-use specified, developed, industrialized and tested components

**Industry-Wide Architecture Best Practices**
Peer review of functionalities, safety and security mechanisms
Lower entry into HAD development
No vendor lock-in
2. Implementation
Open Architecture for Automated Driving

**EB robinos**

- Software modules
  - for prototyping in EB Assist ADTF
  - for rapid embedding on AUTOSAR
  - for production on vehicle ECU
- Developed, tested, verified according to **functional safety standards**

**Open robinos**

- Specifies a **reference platform** for automated driving up to Level 5 (SAE)
- **Freely available** and licensed as Creative Commons
PC Based Quick Start

**EB robinos**

**EB Assist ADTF**

**Rapid prototyping**

C, C++, Model based

**Desktop PC / Car PC**

PC based development environment for

- rapid prototyping
- simulation
- test drive recording
- analysis and visualization

Implement complete automated driving functions in test cars

- Valet parking
- Highway driving
- Others

[www.try-eb-robinos.com](http://www.try-eb-robinos.com)
Transfer onto Embedded Hardware / AUTOSAR

Rapid prototyping
C, C++, Model based
Desktop PC / Car PC

Embedded prototyping
C, C++
Evaluation Hardware

EB robinos
EB Assist ADTF
EB tresos solution for AD

Step by step transfer
- same source
- optimize module by module
- re-use of
  - visualization
  - simulation
  - recorded test data
Integration into Domain Control ECU

- **EB robinos**
- **EB Assist ADTF**
- **EB tresos solution for AD**

**Rapid prototyping**
- C, C++, Model based
  - Desktop PC / Car PC

**Embedded prototyping**
- C, C++
  - Evaluation Hardware

**Automotive grade software**
  - Domain Control ECU

© Elektrobit (EB) 2016
EB robinos – Modular Framework for Automated Driving
Example Application: Automated Valet Parking
Example Application: Automated Valet Parking
Example Application: Automated Valet Parking
Example Application: Automated Valet Parking
Industry wide Best Practice – Growing testing eco system

**Industry-Wide Architecture Best Practices**
- Peer review of functionalities, safety and security mechanisms
- Lower entry into HAD development
- No vendor lock-in
Automated driving systems are complex

Efficient collaboration requires a standard architecture

Open robinos ist the first step

open-robinos.com
Thank you.