DNA for Autonomous Driving
Porting Robot Architectures to AUTOSAR

Dr.-Ing. Björn Giesler

Apr 14, 2016
### An ACC Developer’s World in 2002.

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

Items to specify: 4.
A Driverless Car Developer’s World in 2016.

Items to specify: 24.
How Many Interactions Can We Expect For $n$ Components?

**Best case:**
Every specification gets written or checked only once

$$\frac{n(n-1)}{2}$$

24 components: 276 interactions

**Worst case:**
All specifications must be re-examined after one is changed

$$2^{n-1} - 1$$

24 components: 8.4 million interactions
Does A Centralized ECU Help?

Items to specify: 24.
A Centralized ECU Makes The Problem Worse.

Items to specify: 25.
So What Does Help?

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
From Exponentiality to Linearity

Sensor Data Fusion:
From echos to objects and free space

Automated Driving

Automated Emergency Brake

Automated Valet Parking

Well-defined interfaces

HMI Management:
From buttons and LEDs to user interactions

Motion Management:
From brake pressure to trajectory control

HMI Manager

Sensor Data Fusion:
Vehicle ego motion

Motion Management:
Steering Angle Control
Brake Pressure
Regenerative Braking
Motor Speed Control

HMI Manager

Well-defined interfaces

Trajectory planning & control
Lateral Control
Longitudinal Control
Motion Manager
Can we make complicated behavior even simpler?

- Drive in your own lane!
- Change lanes if you must!
- Don’t crash! (for heaven's sake!)
- Come to a safe stop if necessary!

Continue on, or switch lanes?
Swerve, get back in lane, or brake?
Is everything still OK?
Robot Architectures

- Complex behavior from simple, **stand-alone building blocks**
- Abstract away function behavior behind **well-defined interfaces**
- **Develop once, use many times**

Goal: Make the individual function behavior as simple as possible!

Subsumption Architecture
R. Brooks, MIT, 1986

Behavioral Networks
J. Albiez / R. Dillmann, KIT, 2006
Robot Architectures for Highly Automated Driving
Architecture Is Key To Managing Complexity.

Clear component responsibilities: functionality, safety, security.
Well-defined interfaces and data flow.
Well-defined interactions and redundancies.
Separation of hardware and functionality.

Best Practices – Standards.
### Best Practice Leverage: Project, Product Family, Industry

<table>
<thead>
<tr>
<th>Level</th>
<th>Best Practices</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Project-Level</strong></td>
<td>Reduce complexity</td>
</tr>
<tr>
<td><strong>Product Family</strong></td>
<td>Re-use specified, developed, industrialized and tested components</td>
</tr>
<tr>
<td><strong>Industry-Wide</strong></td>
<td>Peer review of functionalities, safety and security mechanisms</td>
</tr>
<tr>
<td></td>
<td>Lower entry into HAD development</td>
</tr>
<tr>
<td></td>
<td>No vendor lock-in</td>
</tr>
</tbody>
</table>
Conclusion

The problem is not difficulty but complexity.

Good functional architectures help solve it.

We should talk about software, not hardware architectures.

We should move towards best practices across the whole industry.

Less work, less cost, more competitiveness and better functions.
Thank you.

www.elektrobit.com
bjoern.giesler@elektrobit.com