Vehicle Application Plugin Platform

Common Vehicle Interface Initiative

Robert Bosch GmbH
M/NET  Dr. Christian Kerstan
E/E Architecture Roadmap

**Trends for Future Mobility Systems**

**Vehicle Centralized E/E Architecture**
- Domain independent vehicle centralized approach with central vehicle brain(s) and neural network (zones): Logical centralization and physical distribution

**(Cross) Domain Centralized E/E Architecture**
- To handle complexity of increasing cross domain functions

**Distributed E/E Architecture**
- Mainly encapsulated E/E architecture structure

**Increasing number of vehicle functions in the cloud**
- Vehicle Cloud Computing

**Domain Fusion**
- Domain overlapping
  - “Cross Domain Control Units” / “Cross Domain Computer”

**Domain Centralization**
- Domain specific
  - “Domain Control Units” / “Domain Computer”

**Integration**
- Functional Integration
  - Each function has its ECU (“Function Specific Control Units”)

**Modular**
- Typ. state of the art automotive ECUs (function specific)
- Performance ECUs e.g. (Cross-)Domain Control Unit, (Cross-)Domain Computer, Vehicle Control Computer

**LEAGACY**

**FUTURE**

Optional ECUs (e.g. Central Gateway)
- Domain independent Zone ECUs
- Domain specific Zone ECUs (e.g. today’s Door ECU)

Sensors/Actuators
- ECU = Electronic Control Unit
Vehicle Application Plugin Platform
“Rework” Functional Architecture

```cpp
class Vehicle {
public:
    virtual int getCurrentSpeed() = 0;
    virtual void setSpeed(int speed) = 0;
    virtual bool getLock() = 0;
    ...;
};
```

Today

```cpp
// dedicated vehicle implementation
class OEM_A_Mod_B_VAR_42 : public Vehicle {
public:
    int getCurrentSpeed();
    ...;
};
```

Future

Cloud

Edge

Personal Micro Mobility

© Robert Bosch GmbH 2019. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.
Vehicle Application Plugin Platform
Decoupling of Development & Deployment Cycles

- Decoupling of implementation reduces effort and complexity
- Decoupling of deployment cycles allows fast updates for high level features and well-proven processes for embedded functionality
- Service development does not require knowledge of all future functionality
- New business models possible due to independent deployment
Example: AUTOSAR: Exchange type of Front Light

Halogen Head Light

- SwitchEvent
  - check_switch()
  - switch_event(event)
- LightRequest
  - switch_event(event)
  - request_light(type, mode)
- Front-Light Manager
  - request_light(type, mode)
  - set_keyposition()
  - set_light(type, mode)

Xenon Head Light

- Xenonlight
  - set_light(type, mode)
  - set_current(…)

Exchange SWC

- Reconfigure BSW
- Exchange Hardware

Source: based on AUTOSAR Guided Tour
Vehicle Application Plugin Platform
Example: AUTOSAR: Exchange type of Front Light

Set_Light(bool state)  setLight(enum state)

switchHeadLight(enum type, enum mode)

lightOn()  lightSwitchEvent(enum state)

SetLight(bool state)  Set_Beam(enum range)

OP_MOD_Light_Func2(enum param1)

g_DrvReqHB(enum state)

Remaining Challenge: NO Standardized Application Interface over OEMs / Project Borders
Vehicle Application Plugin Platform
VAPP Data Model – based on well defined standards
A standardized trailer hitch allows a variety of vehicle models to tow a variety of trailers.

By establishing a SW interface standard, VAPP wants to enable a variety of vehicle models to make use of a variety of high-level application software.
Vehicle Application Plugin Platform

Chances in Limiting Interface Variants

Reducing interface variants frees development capabilities for new features!
Thank you!

Dr. Christian Kerstan
Technical Strategy, Sector Electric & Electronic (M/NET)
Tel. +49 711 811-24406 | Mobile +49 173 7593331
christian.kerstan@de.bosch.com