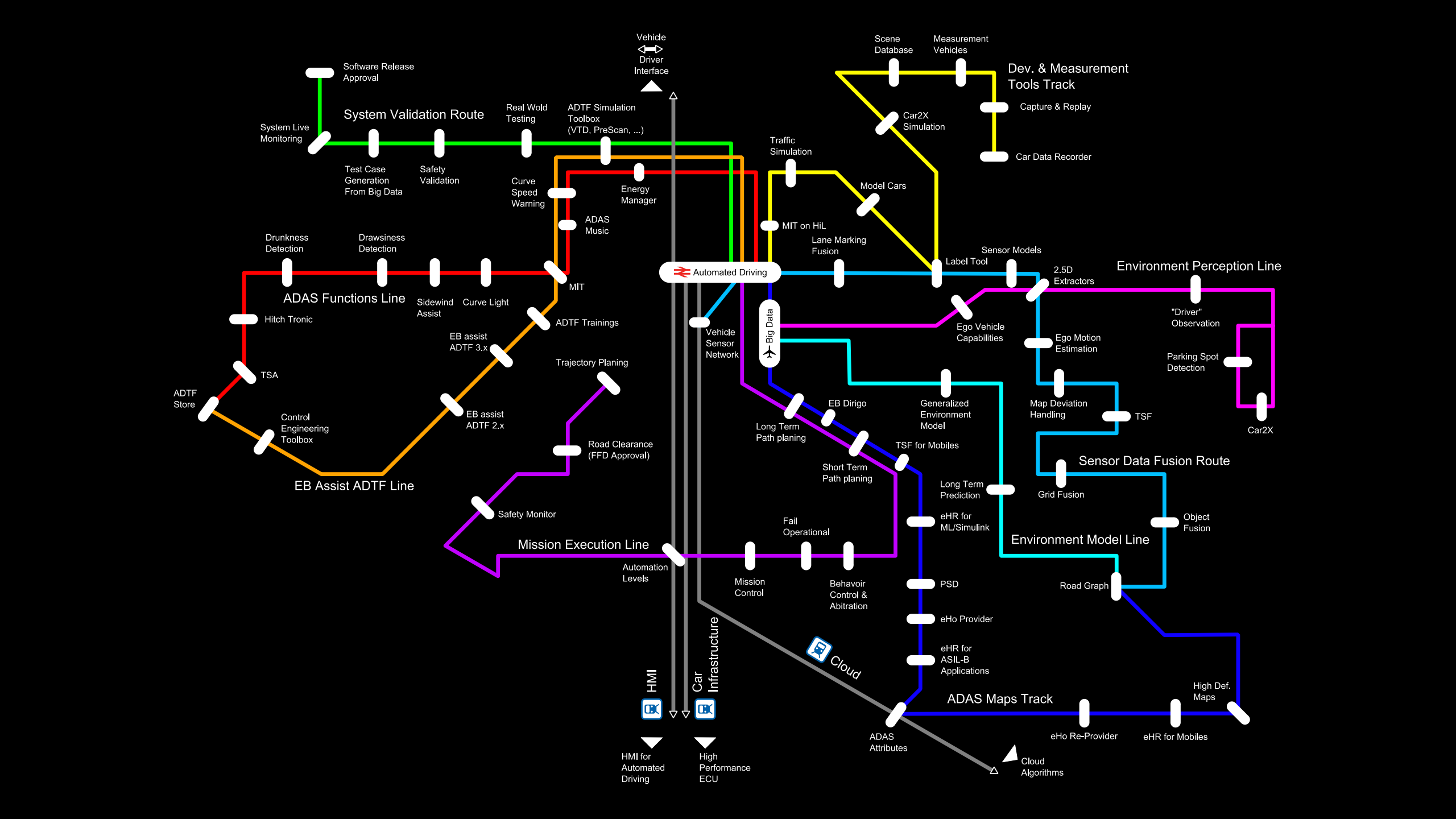
Developing software architectures for autonomous driving vehicles

Sebastian Ohl - Senior Expert Driver Assistance, Erlangen, Germany

April 6th, 2016

Elektrobit


Vehicle
Driver
Interface
Software Release
Approval
System Validation Route
System Live
Monitoring
Test Case
Generation
From Big Data
Safety
Validation
Real Wold
Testing
ADTF Simulation
Toolbox
(VTD, PreScan, ...)
Curve
Speed
Warning
Energy
Manager
ADAS
Music
Drunkness
Detection
Drawsiness
Detection
ADAS Functions Line
Sidewind
Assist
Curve Light
MIT
Hitch Tronic
ADTF Trainings
EB assist
ADTF 3.x
TSA
ADTF
Store
Control
Engineering
Toolbox
EB assist
ADTF 2.x
EB Assist ADTF Line
Trajectory Planing
Road Clearance
(FFD Approval)
Safety Monitor
Mission Execution Line
Automation
Levels
Mission
Control
Fail
Operational
Behavoir
Control &
Abitration
Automated Driving
Vehicle
Sensor
Network
Big Data
Traffic
Simulation
MIT on HiL
Lane Marking
Fusion
Model Cars
Scene
Database
Measurement
Vehicles
Dev. & Measurement
Tools Track
Capture & Replay
Car2X
Simulation
Car Data Recorder
Sensor Models
Label Tool
2.5D
Extractors
Environment Perception Line
"Driver"
Observation
Ego Vehicle
Capabilities
Ego Motion
Estimation
Parking Spot
Detection
Car2X
EB Dirigo
Long Term
Path planing
Short Term
Path planing
TSF for Mobiles
Generalized
Environment
Model
Map Deviation
Handling
TSF
Sensor Data Fusion Route
Grid Fusion
Long Term
Prediction
eHR for
ML/Simulink
Object
Fusion
Environment Model Line
Road Graph
PSD
eHo Provider
eHR for
ASIL-B
Applications
Cloud
HMI
Car
Infrastructure
ADAS Maps Track
High Def.
Maps
ADAS
Attributes
eHo Re-Provider
eHR for Mobiles
HMI for
Automated
Driving
High
Performance
ECU
Cloud
Algorithms

Challenges

Developing a DAS system from scratch is hard, but the core tasks are often similar:

- Fuse, improve, and cross-check different types of sensor data
- Derive scene elements from the fused results (driving corridor, obstacles)
- Plan a full trajectory or parts of it
- Execute the trajectory by controlling the vehicle‘s motion components
- Qualified and experienced personal is rare

Challenges

Step from development to embedded is a challenge:

- Computing performance may drop significantly
- Software components on ECU and a Car-PC differ (up to complete reimplementation)
- Automotive standards need to be taken into account (e.g. AUTOSAR, UDS, SomeIP, …)
- Safety standards may break your architecture
- „Hacked“ code from pre-development may be system depended
- Supervision of execution flow and algorithmic results necessary

How to Jump Start Your Development?

Sharing SWCs
Algorithmic Cores
SWCs for HAD
Safety Consulting
Tools
Test Automation
Well defined interfaces
Training
AUTOSAR
Software
Scalable Reference Architecture
High Performance ECUs
Integrated Simulation Tools
Development Process

Approach: HAD – from development to embedded

Development

Series production

Rapid **prototyping**

Rapid **embedding**

Mass production

Application layer architecture and modules

ADAS development framework

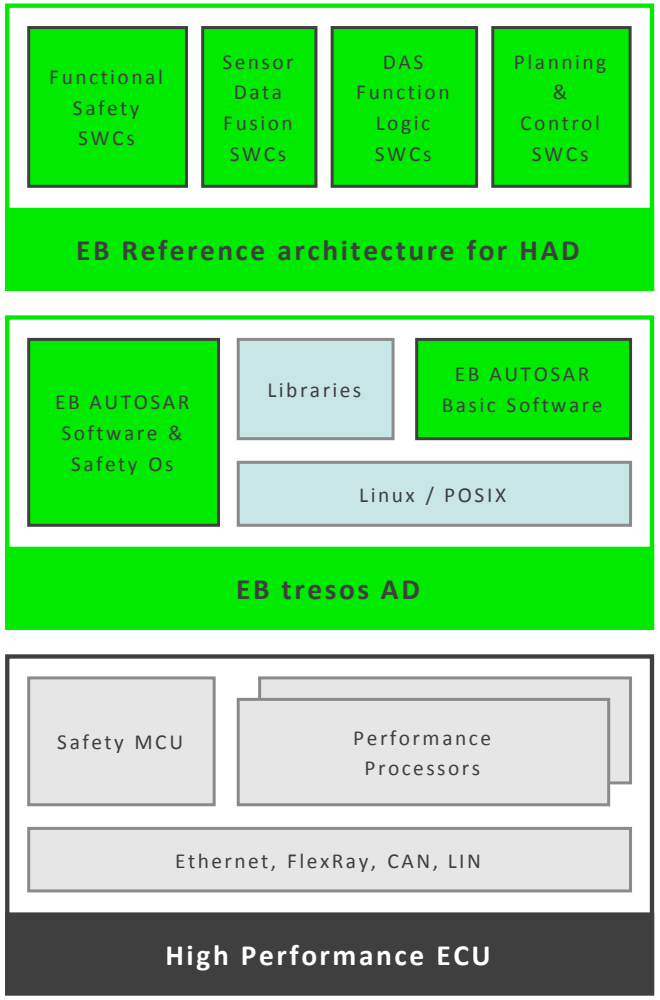
AUTOSAR basic software

Car-PC

Development ECU

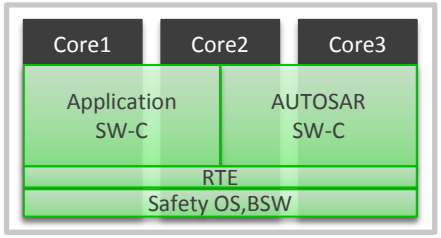
Target ECU

Approach: HAD – from development to embedded



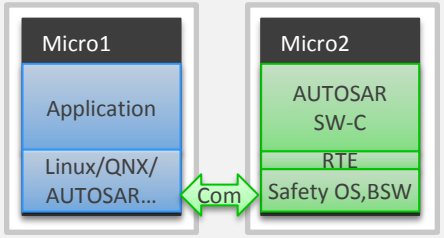
- **EB Reference architecture for HAD**
  - Application-Layer Architecture and SWCs for Automated Driving Development
  - Rich Abstraction for DAS tasks like fusion, planning, control, multi-function coordination, and functional safety
- **EB tresos AD**
  - Full AUTOSAR BSW and Safety products on Safety µC,
  - RTE on Linux on High-performance µC
  - OS-to-OS communication via Ethernet + SPI
- **Hardware**
  - NVIDIA DRIVE PX for prototyping or similar
  - Conti CDCU or similar for production
- **Scope**
  - Generic platform for ADAS/HAD Applications
  - Safety & Security solutions for High-Performance Systems

Different Hardware Architectures



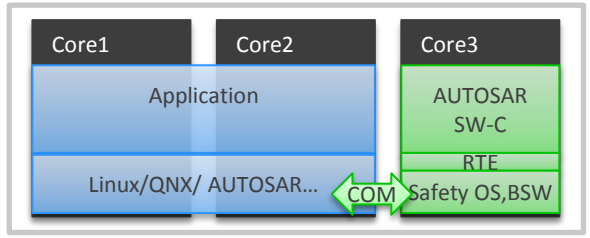
Full AUTOSAR

for
Safety or
Performance



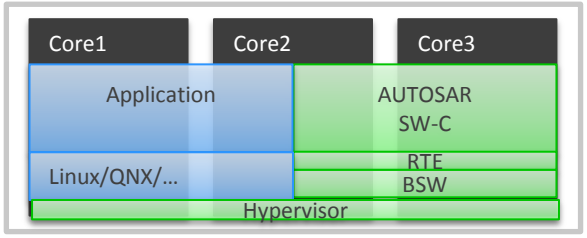
Microcontroller Partitioning

for
Safety and
Performance
EB tresos AD for NVIDIA DRIVE PX



Core Partitioning

for
Safety and
Performance optimized



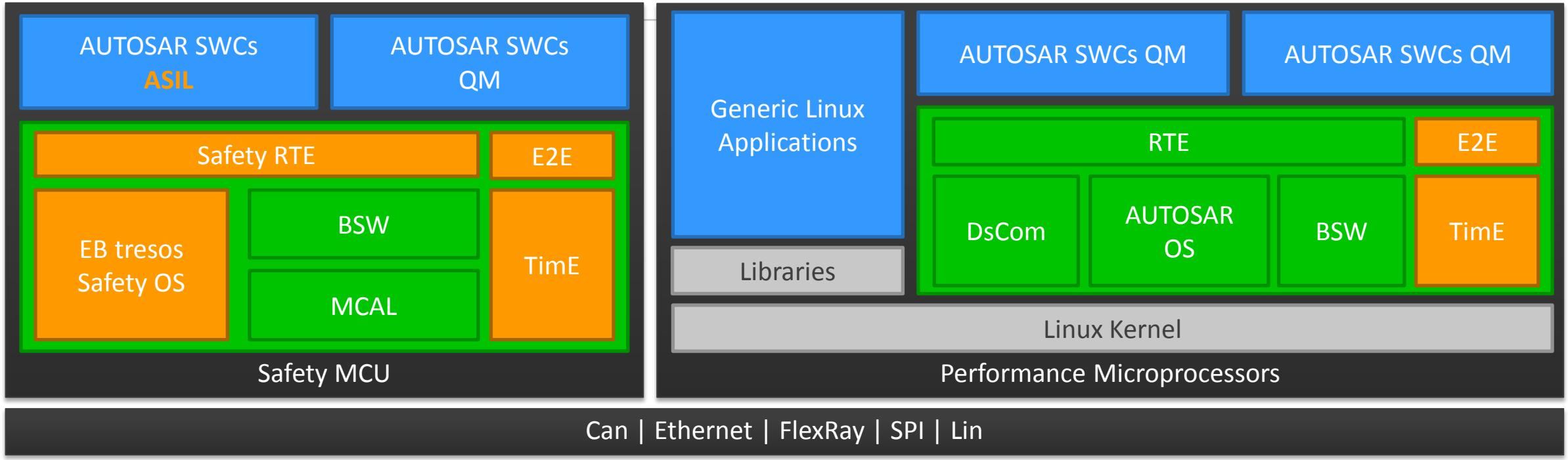
Hypervisor

for
Security
Architecture

- Software Architectures define the next generation of microcontroller architectures
- AUTOSAR is part of each architecture as a common standard for
  - Basic Software, Safety and Security in ECUs

Elektrobit

EB tresos AD for NVIDIA DRIVE PX



EB tresos Safety Products

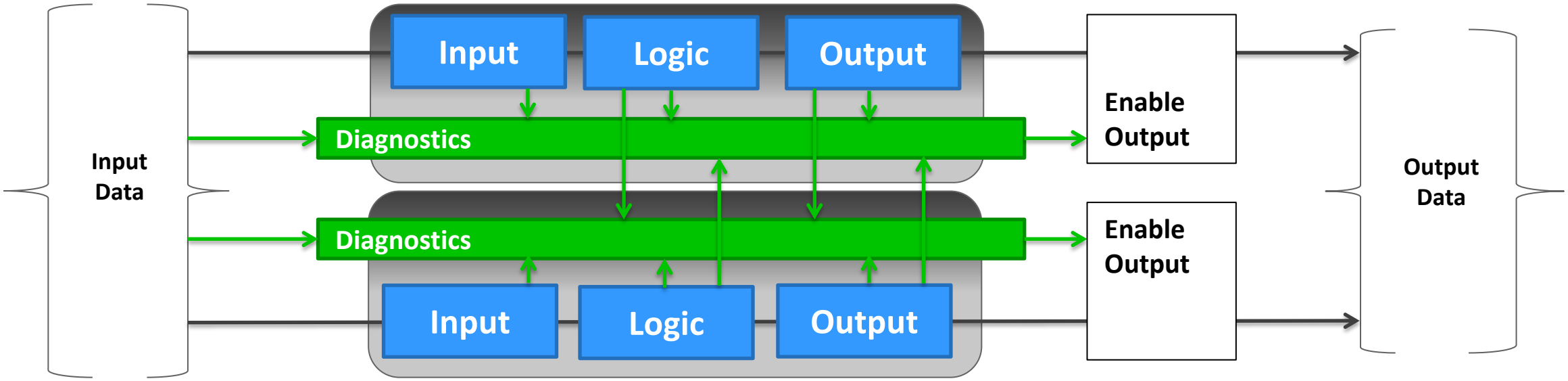
EB tresos Autocore

Application Software

Architecture

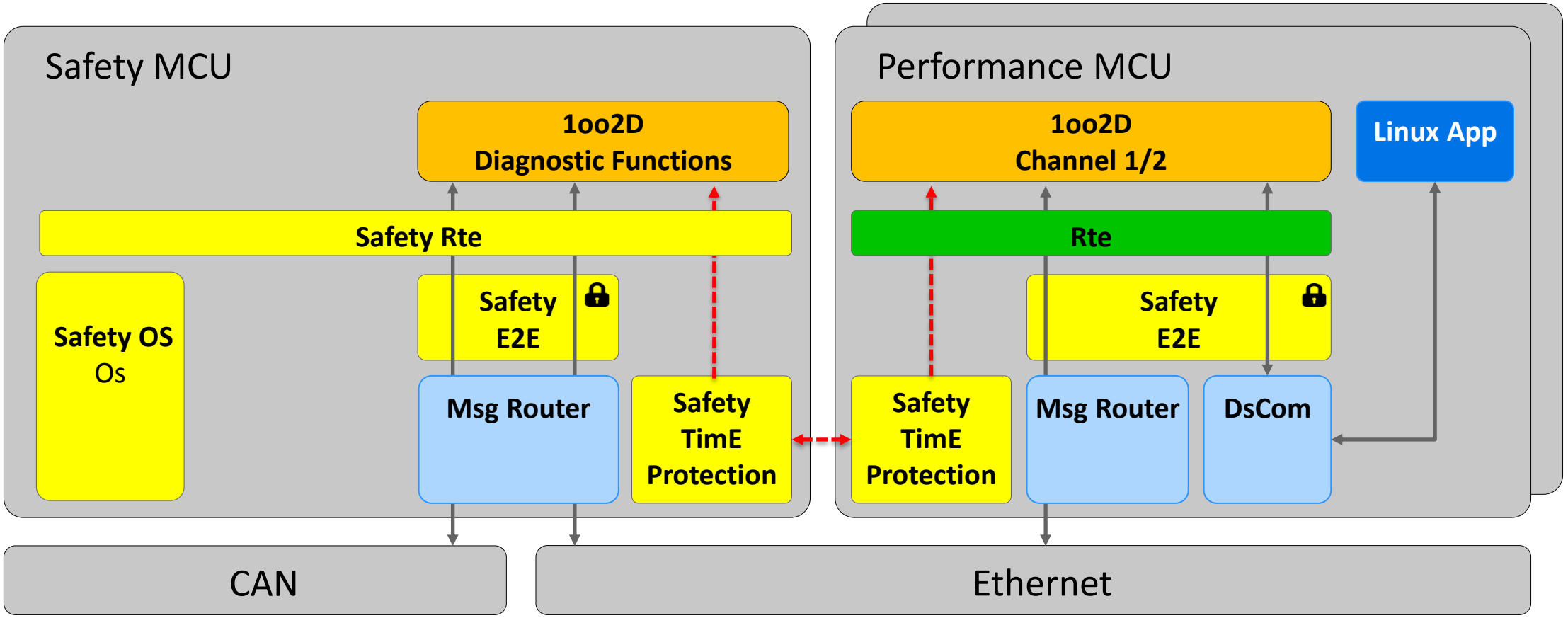
- AUTOSAR SWCs & Linux Applications
- Functional Safety & Security concept
- Linux Libraries like OpenCV / CUDA
- On Board Communication concept
- AUTOSAR Tool Environment

Example: 1oo2D System on NVIDIA DRIVE PX

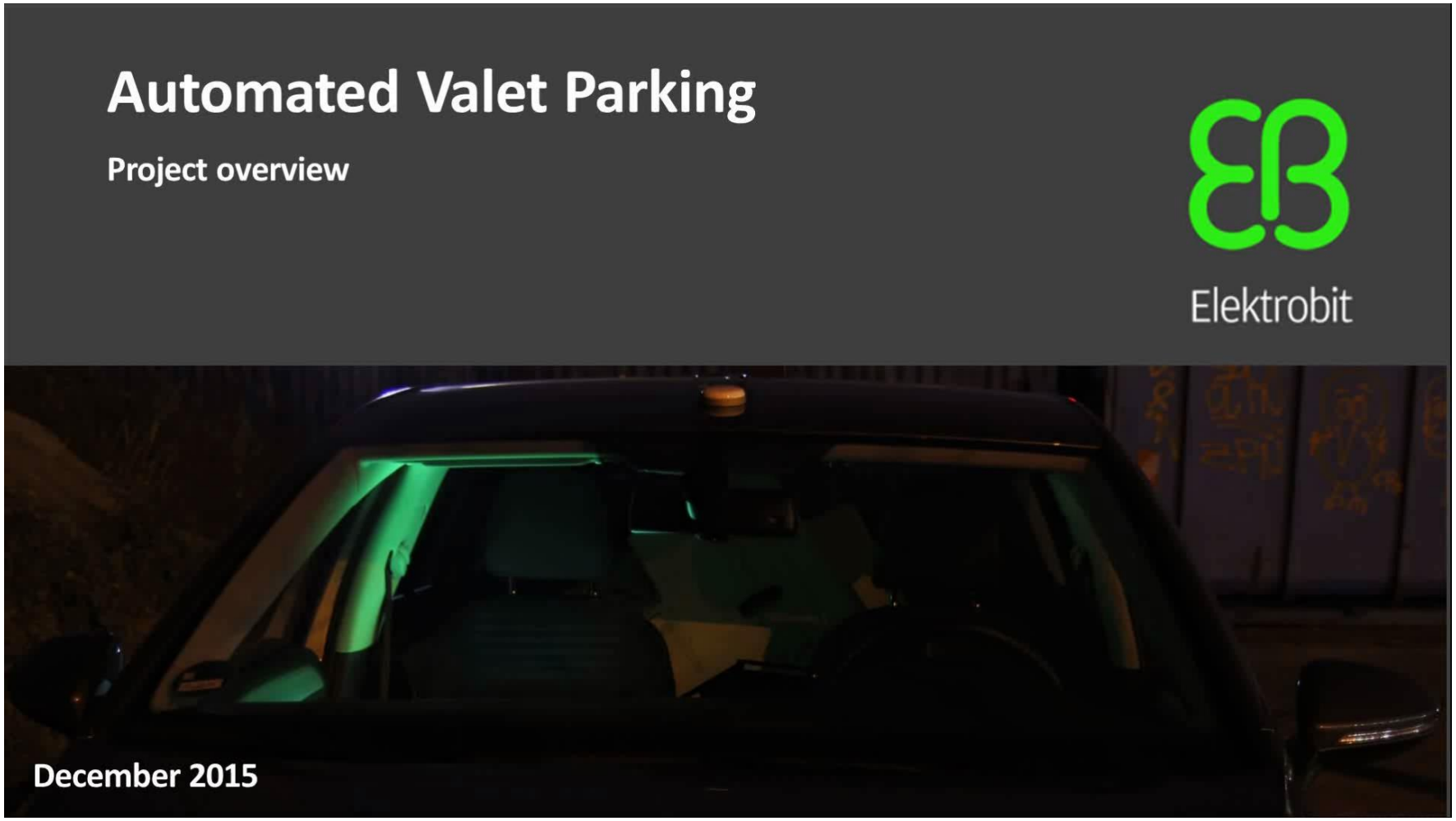


- If one channel fails, the system continues to operate with only one channel for a limited amount of time (see e.g. chemical process industry).
- Controlling this limited time and matching it with the system behavior is crucial.
- **Precondition**: high diagnostic coverage required for each channel.

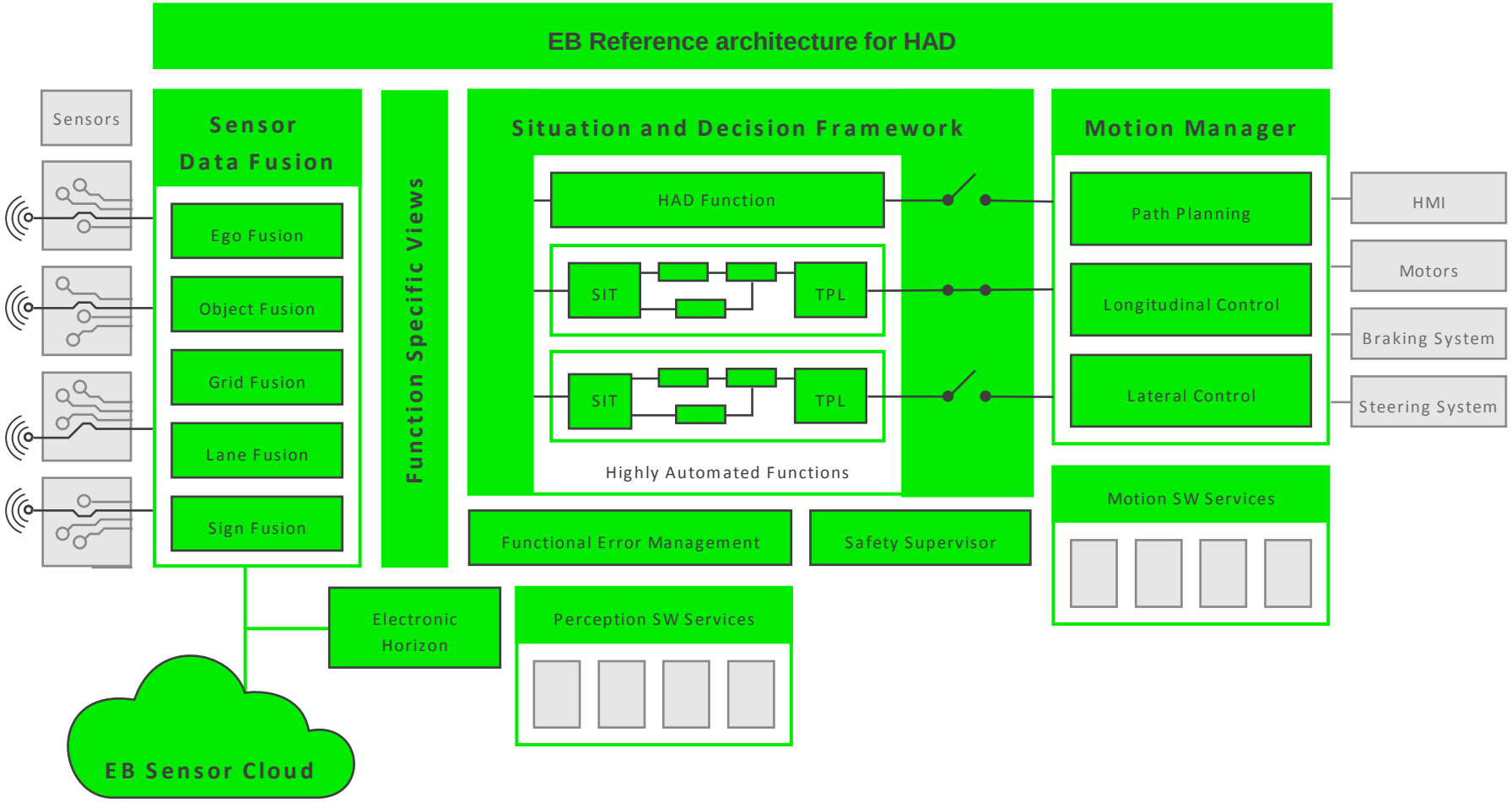
Example: 1oo2D System on NVIDIA DRIVE PX



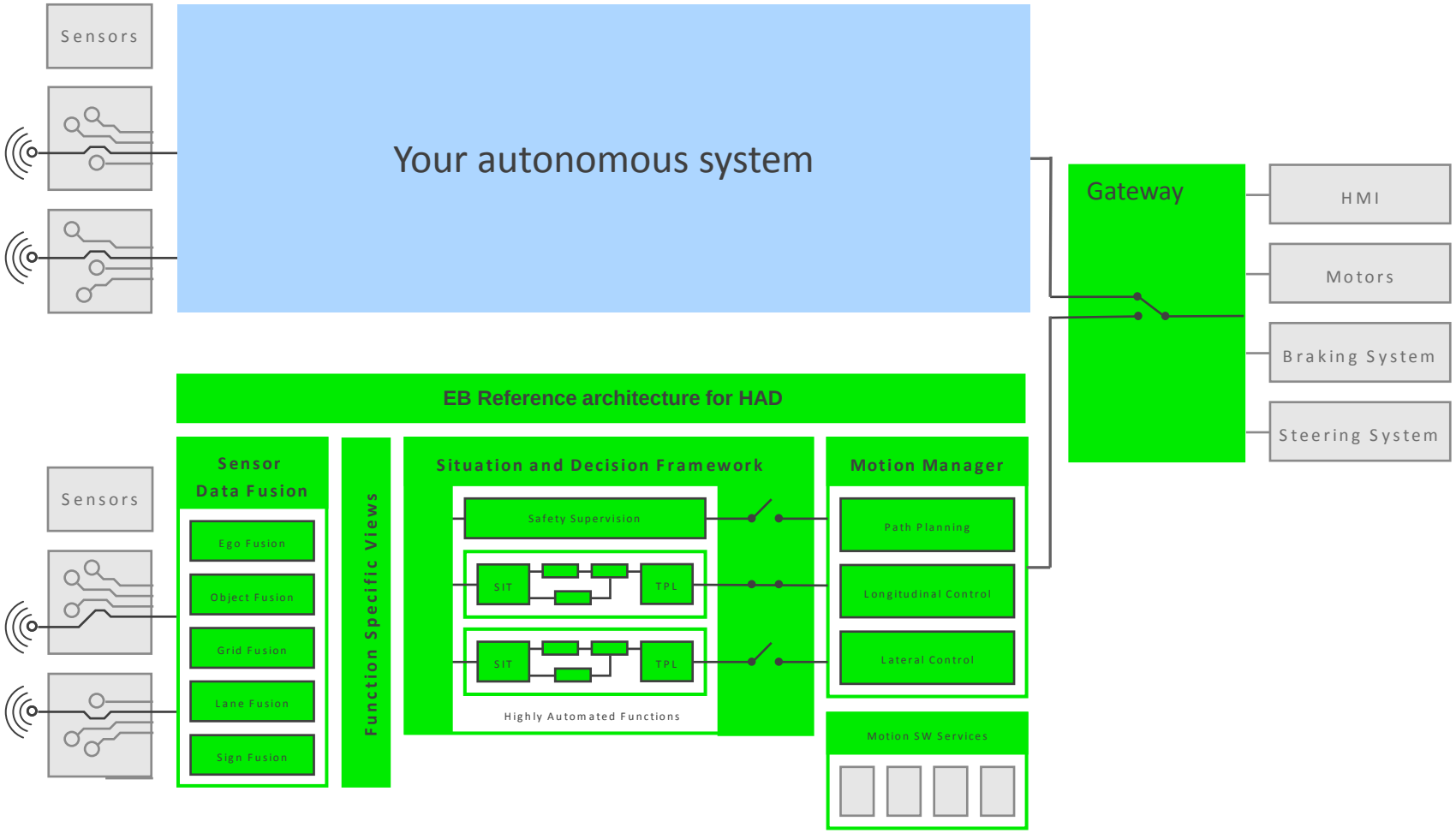
Example Application: Automated Valet Parking by Elektrobit



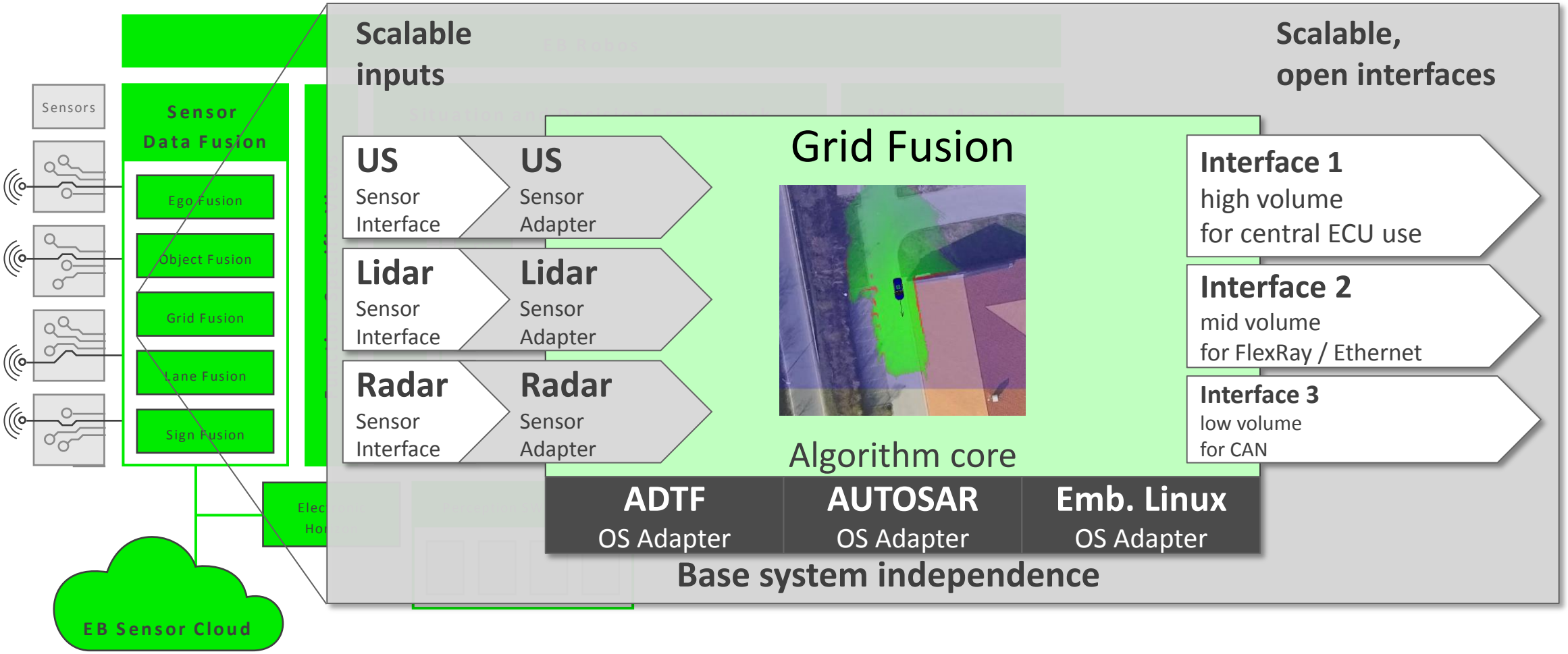
Reference architecture for highly automated driving functions



Reference architecture for HAD: safety path for your system

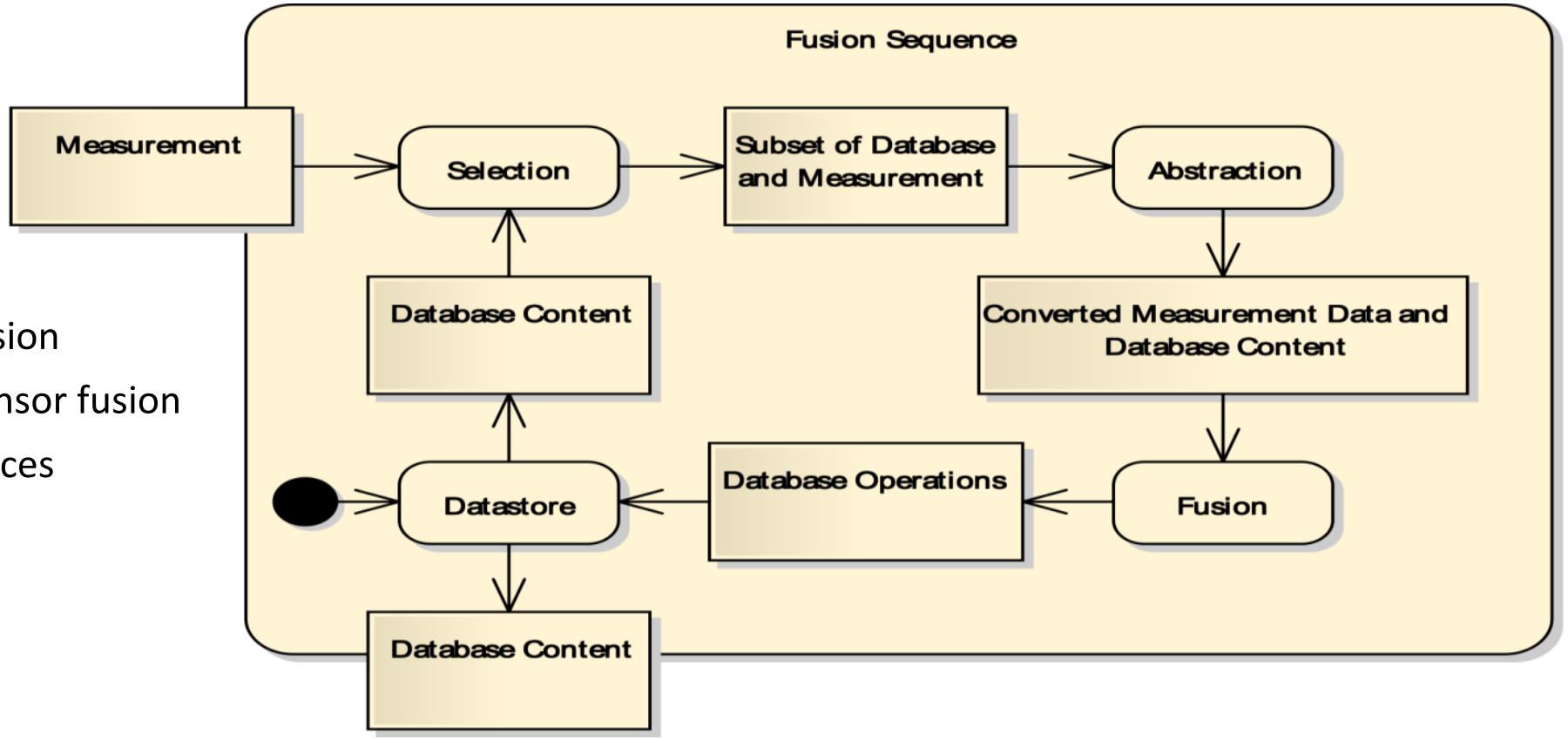


Reference architecture for HAD: Structure of a module



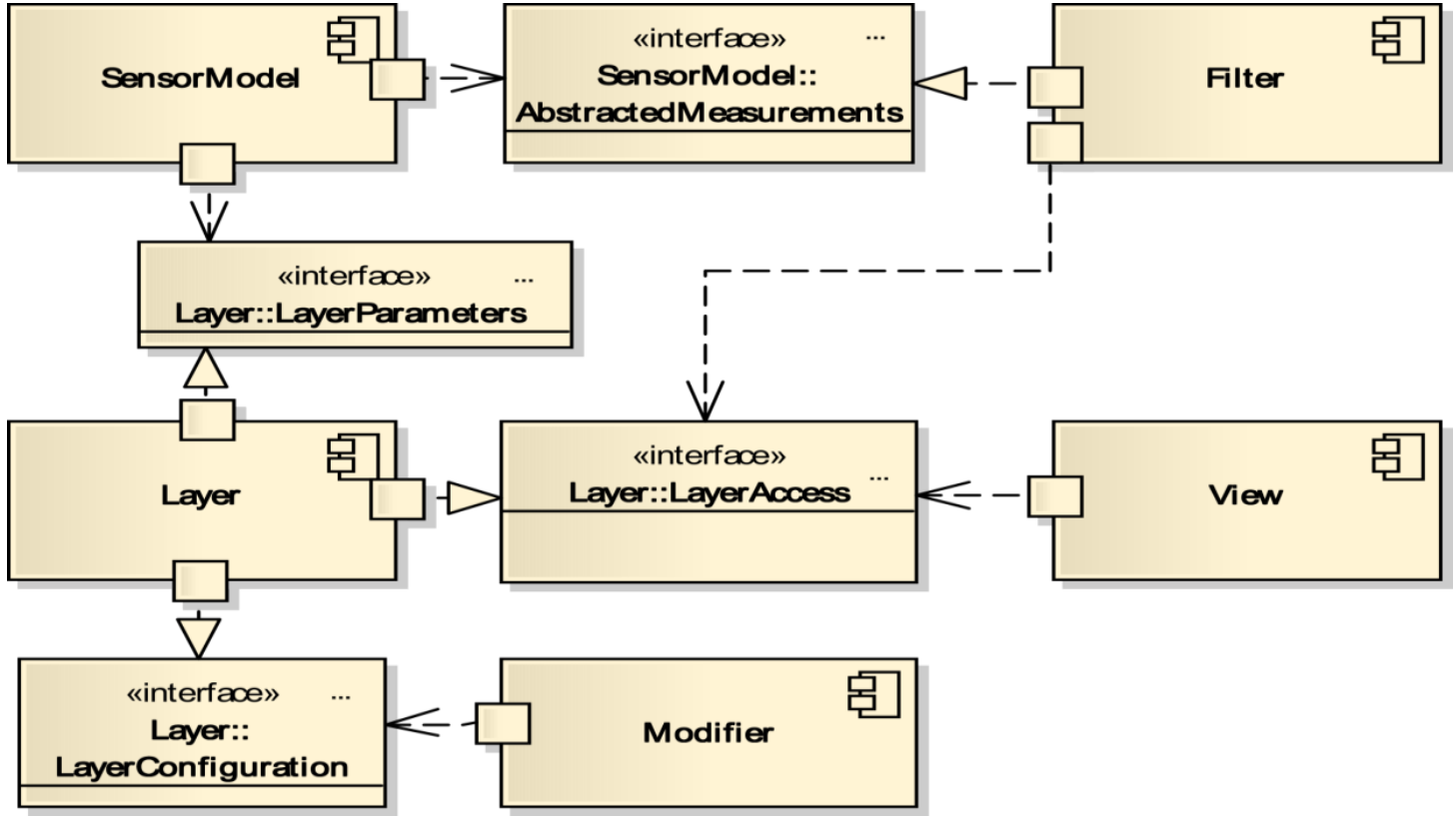
Logical fusion architecture

- Separation into four basic elements
  - Selection
  - Abstraction
  - Fusion
  - Datastore
- Mapping to
  - Grid based sensor fusion
  - Object hypothesis sensor fusion by specialized instances

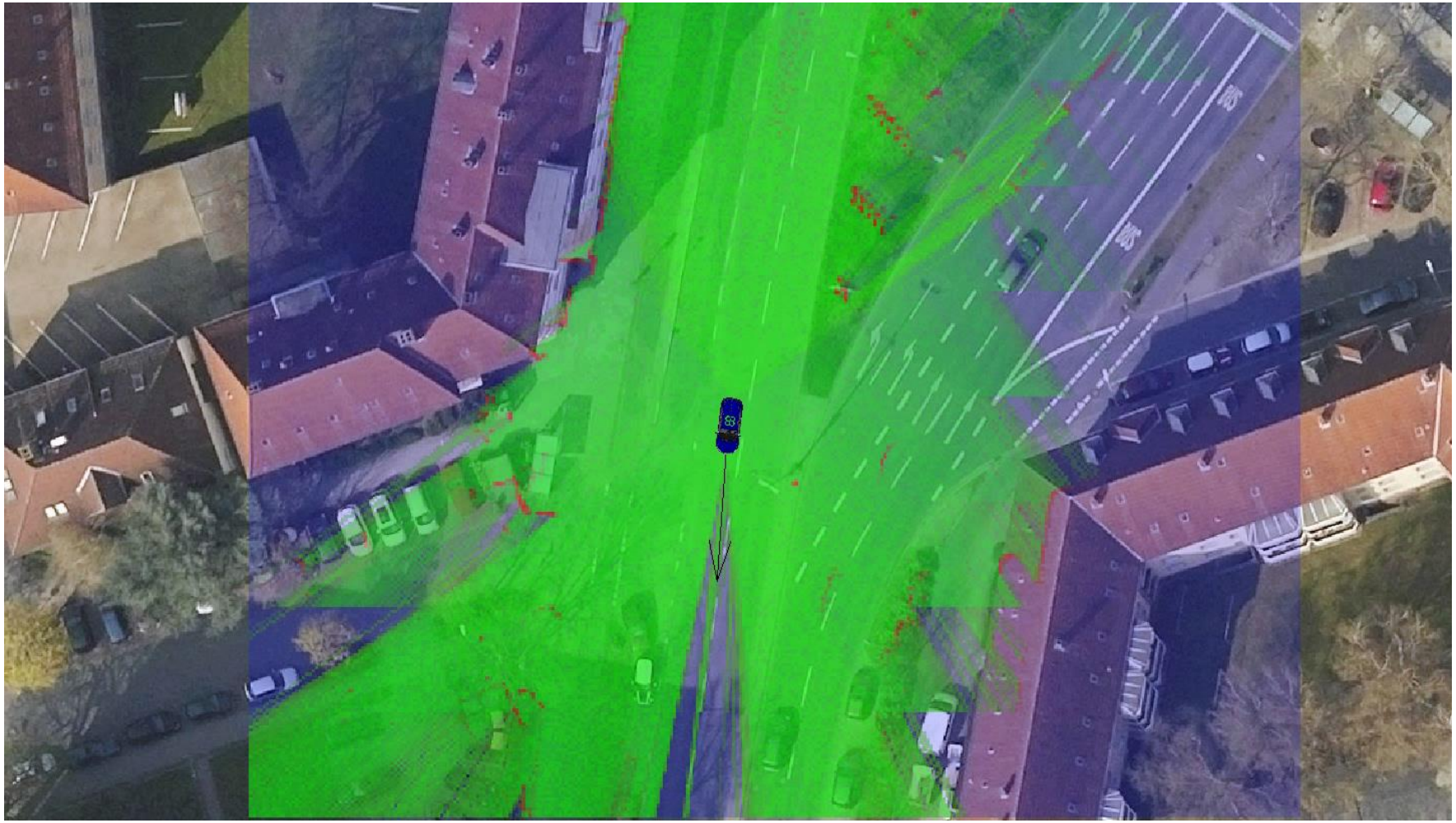


Occupancy grid

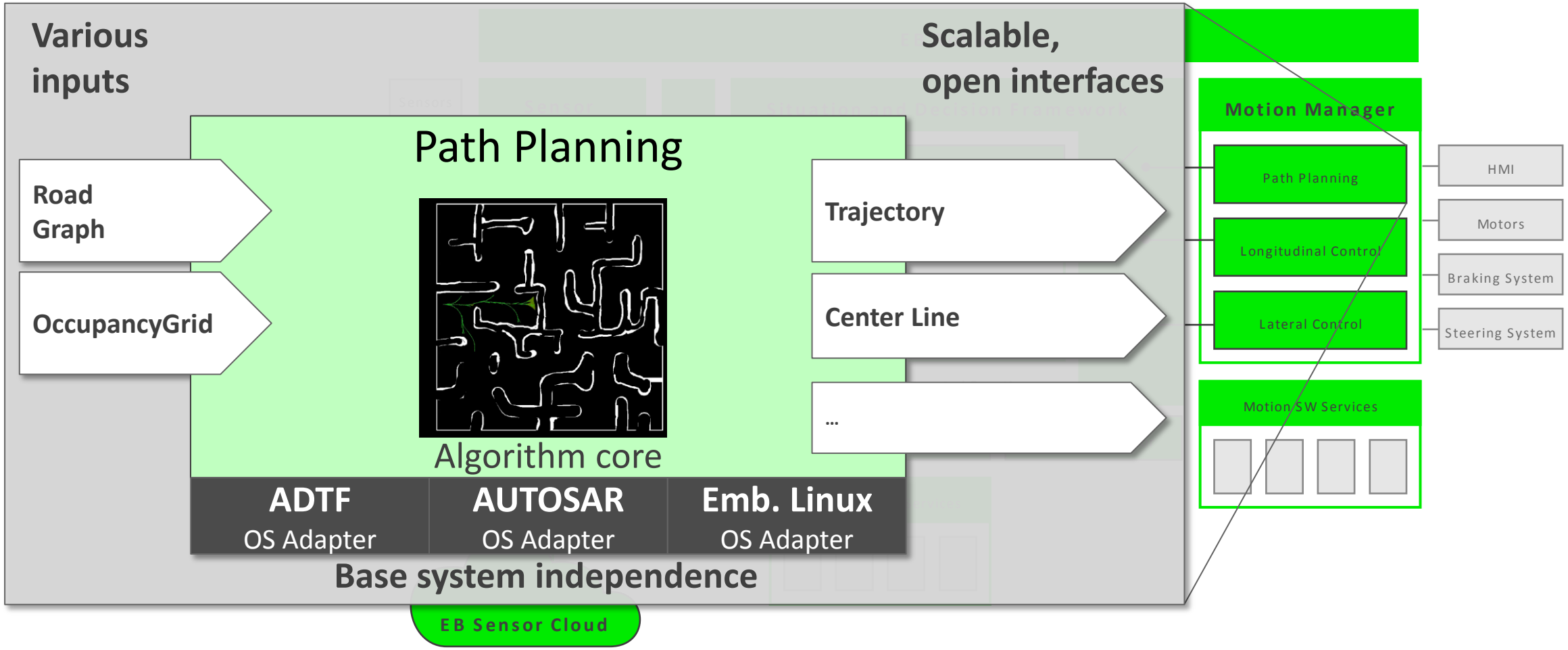
- Hierarchical sensor fusions by using multiple layers
- Mapping of the logical architecture to the core elements:
  - SensorModel
  - Filter
  - Layer
  - View
  - Modifier



Occupancy grid in action

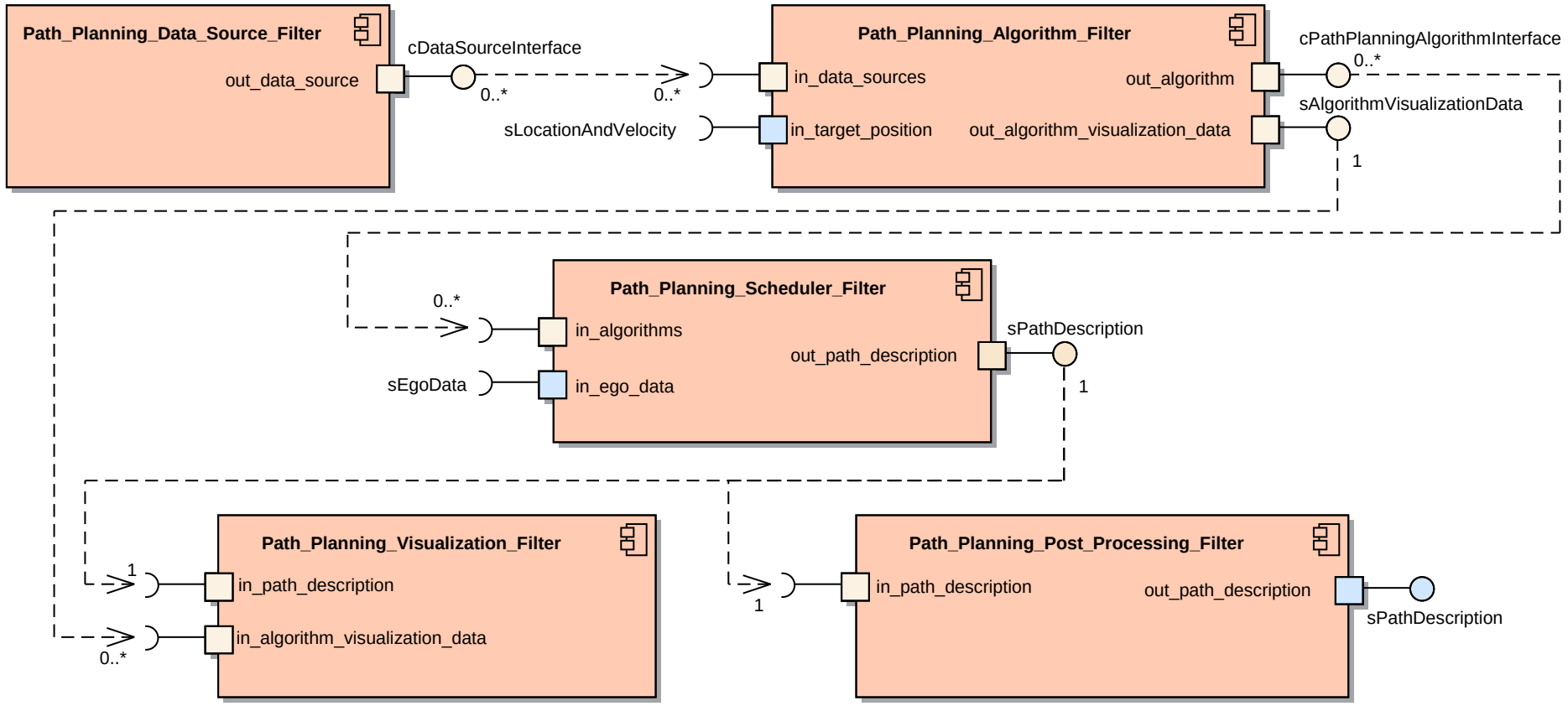


Reference architecture for HAD: Structure of a module



Path planning

- Separating the path planning process into several models:
  - Data Source
  - Algorithm
  - Conductor
  - Visualization
  - Post Processing



Developing software architectures for autonomous driving vehicles

Sebastian Ohl - Senior Expert Driver Assistance, Erlangen, Germany

Elektrobit

automotive.elektrobit.com
Sebastian.Ohl@elektrobit.com

Contact us!

Industry is facing the transition from assisting to fully automated driving functions

New software/hardware architectures are needed to make this transition a success

New systems are very complex, computationally intensive, and related to safety/security

EB's Platform for Autonomous Driving is one solution to face these upcoming challenges