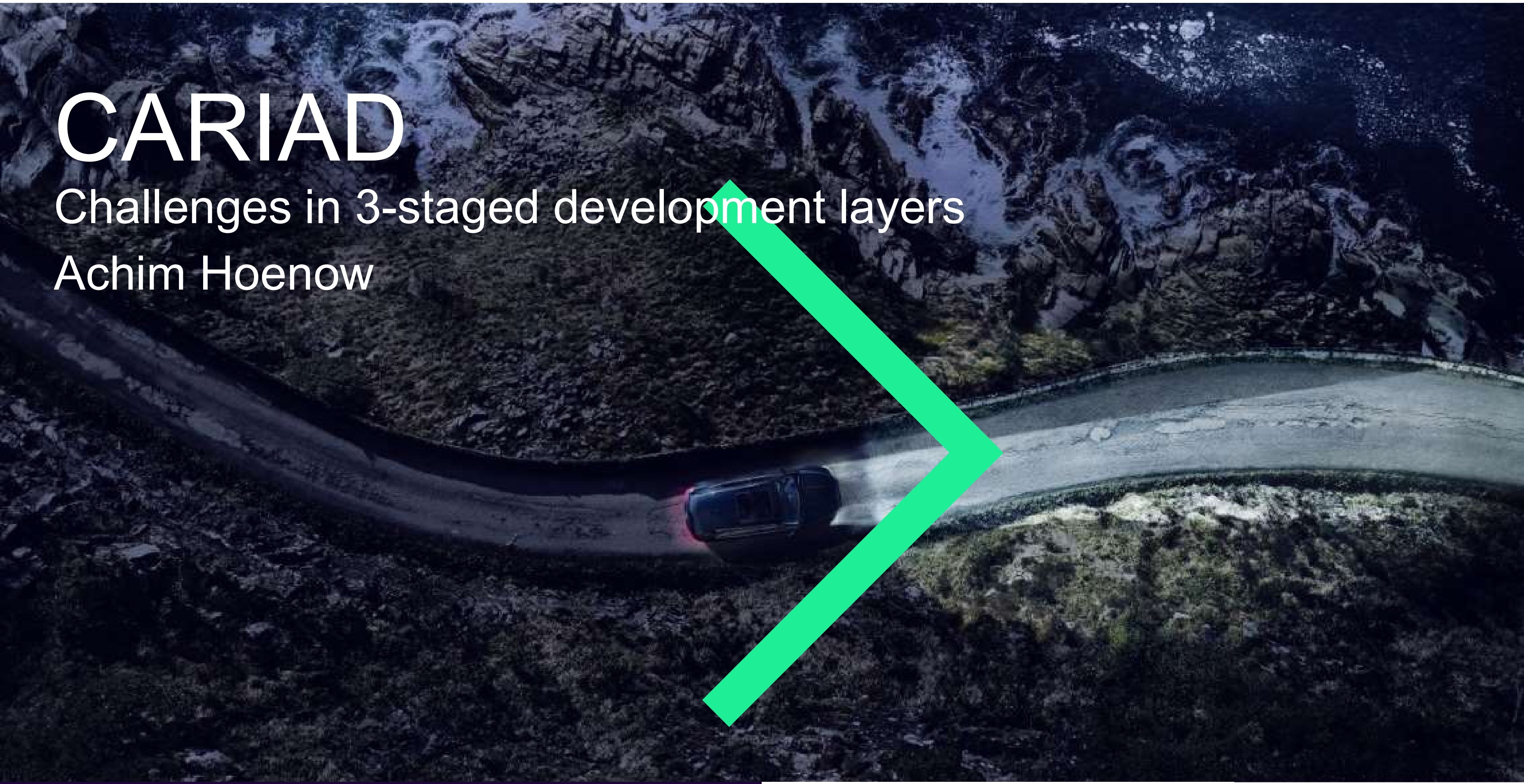


CARIAD

Challenges in 3-staged development layers

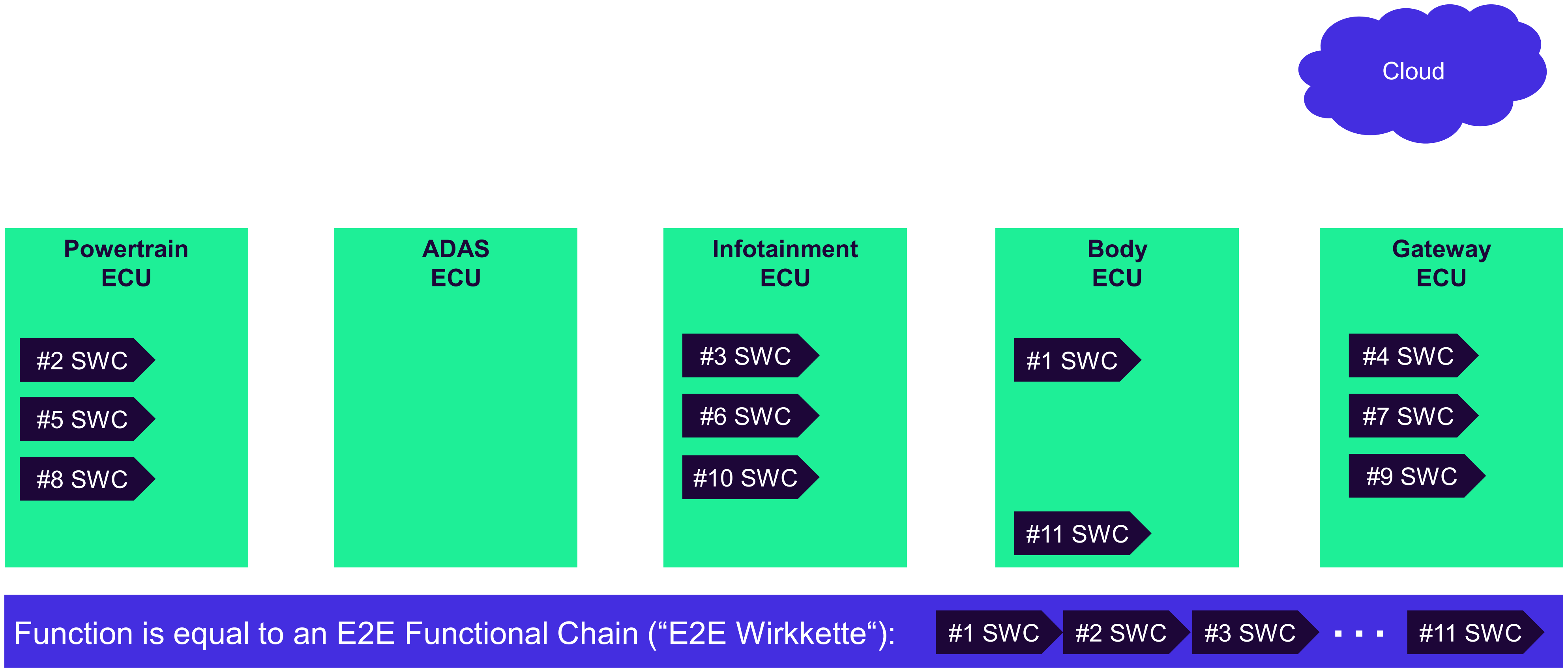
Achim Hoenow

We transform automotive mobility

CARIAD
A VOLKSWAGEN GROUP COMPANY

Motivation

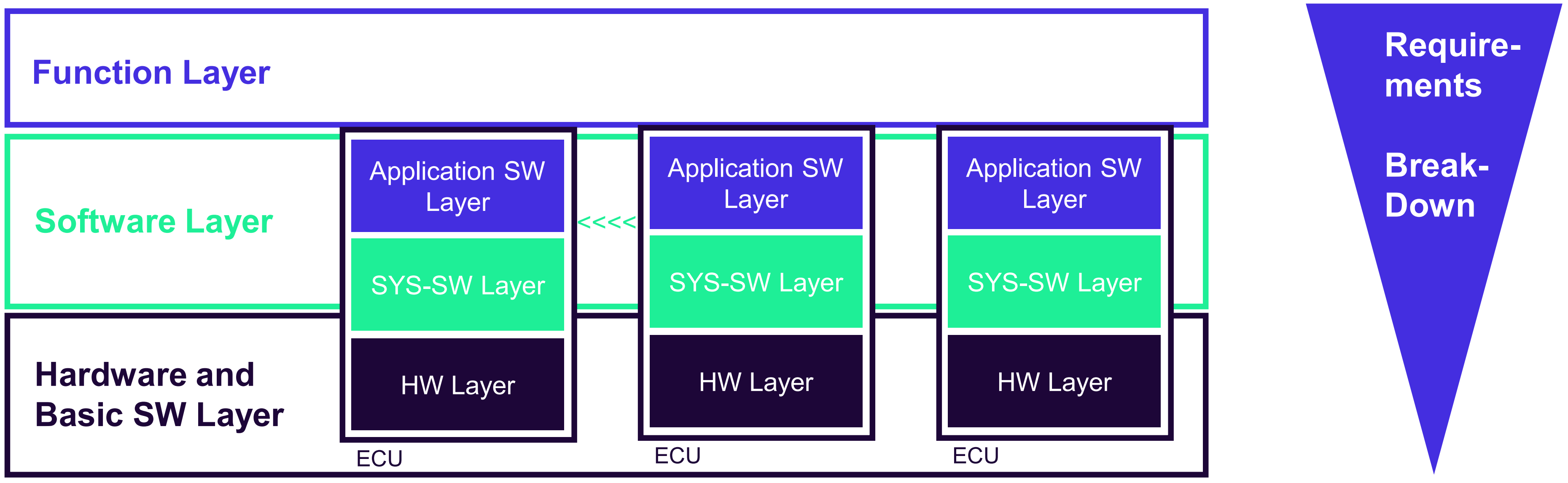
Motivation – Example Plug & Charge



Agenda

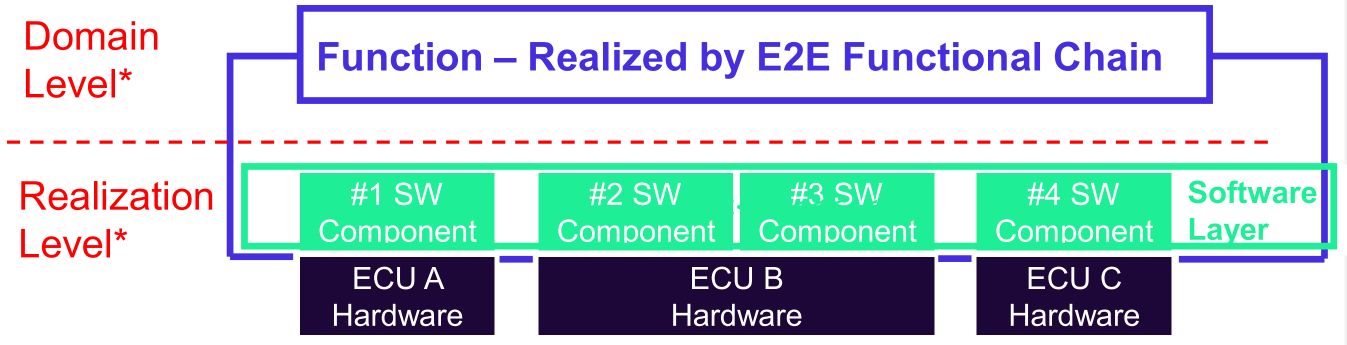
3-Staged Development Layers

3-Staged Development Layers



Car functions are the origin of development: Start of requirements breakdown – End with Function Release

3-Staged Development Layers – Single Function



Key facts of E2E Architecture

> 700 Functions
> 500 SW Components
~ 3-5 High Performance ECUs
> 50 Standard ECUs

Functional Chains contains between 1 and >20 Software Components

One Software Component can contribute to more than 200 different functions!

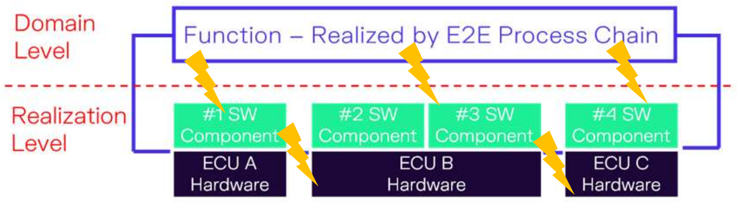
* Domain Driven Design [E. Evans]

E2E Functional Chain is the connecting link between Domain and Realization Level

Managing Complexity

Managing Complexity

Complexity is given on both levels “Domain Level” and “Realization Level” and between their interfaces



Examples for managing complexity on Domain Level (key driver: communications between parties):

- Partitioning of functions to groups of similar Knowledge Areas (e.g. ADAS, Infotainment, Body, Connect)
- Equal process steps, tools and communications (e.g. reporting) between Knowledge Areas

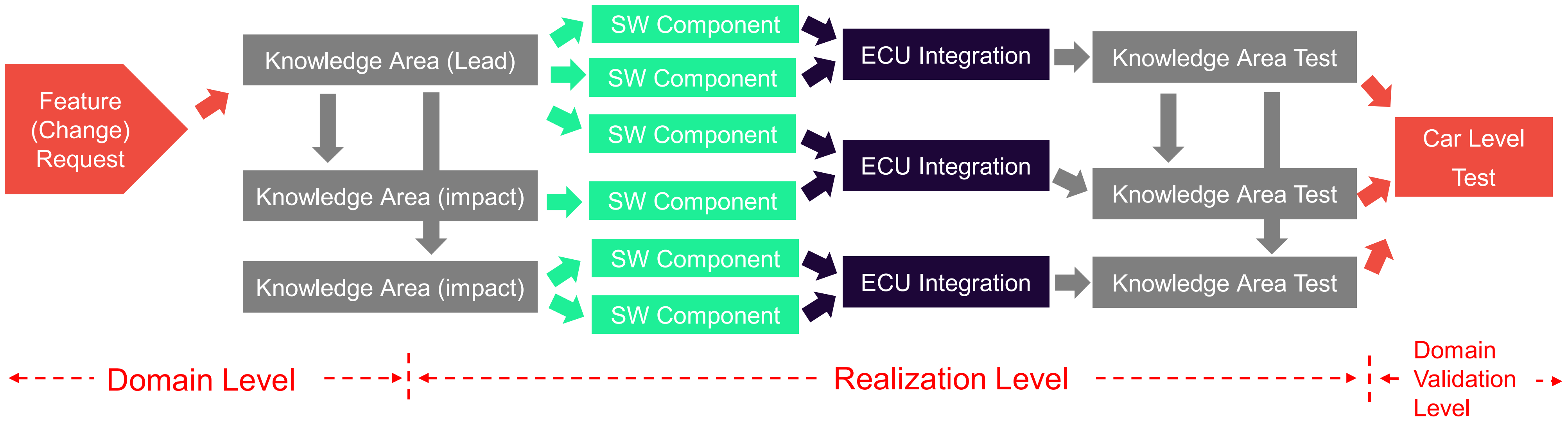
Examples for managing complexity on Realization Level (key driver: interfaces between elements):

- Completely documented and under change control E2E Functional Chain for realization, testing and error management
- Equal process steps, tools and control boards (e.g. Change Control Boards) on interfaces

Key drivers of complexity are the interfaces on human and technical level

Managing Complexity – Realization of function

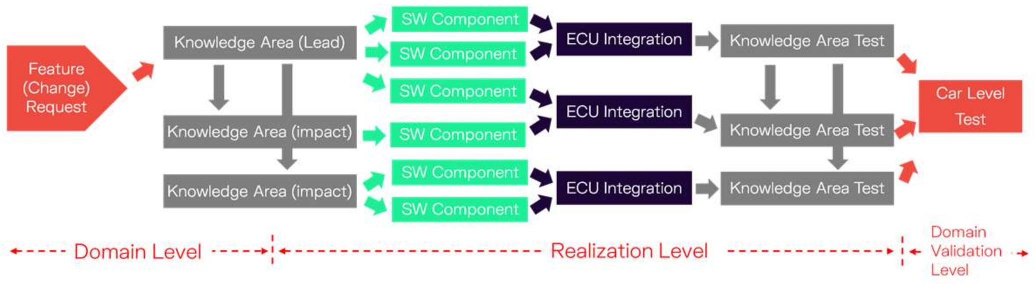
How to release a new function or a function change on domain and realization level?



Function realization is vice versa to historical ECU development: SW before HW

Managing Complexity – Realization of functions

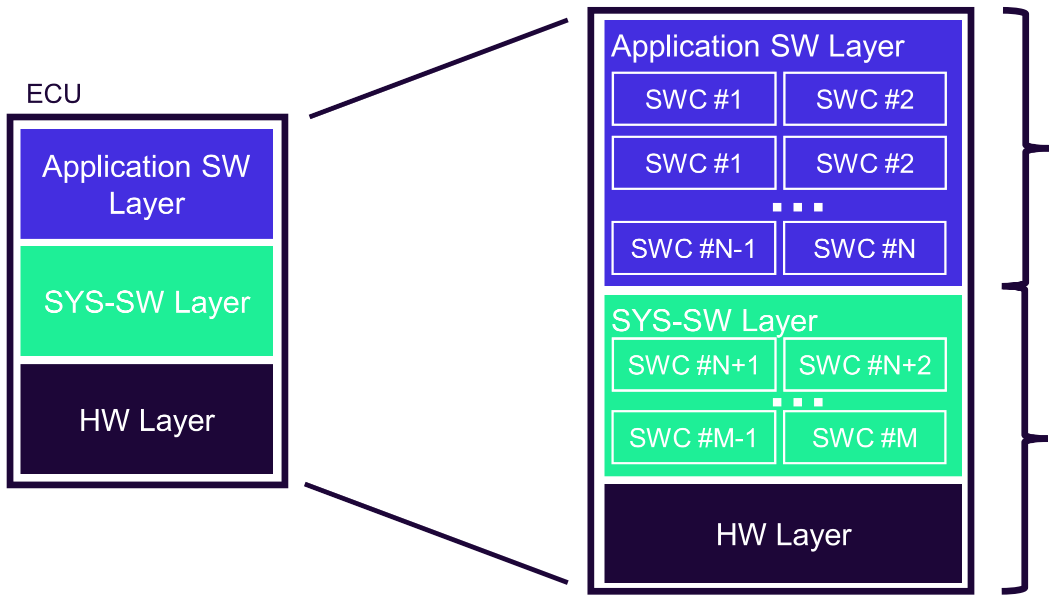
Circumstances in function realization model



- Feature (Change) Request is assigned to one leading Knowledge Area for realization and responsibility
- Feature (Change) Request is assigned in a 1:N dependencies to SW components
- Knowledge Area can be responsible for 1 to M SW components and for 0 to N ECUs
- Test criteria needs to be defined on SW Component (e.g. SIL), ECU (e.g. HIL), Knowledge Area (e.g. HIL) and Car Level.
  Final “release” – feature testing can only be done on complete Car Level
- Cross functionality (safety, security, diagnosis, Update/OTA) is realized by and associated to functions

Function realization is abstract below car level

Managing Complexity – ECU Level



SW Components of Application SW Layer are delivered by different suppliers including OEM itself. Expertise of supplier is key driver. "Proven in use" as well as new developed SW Components will be integrated

Hardware on SYS-SW Layer often realized by one TIER1 supplier. Series production of ECU executed by same supplier (industrialization)

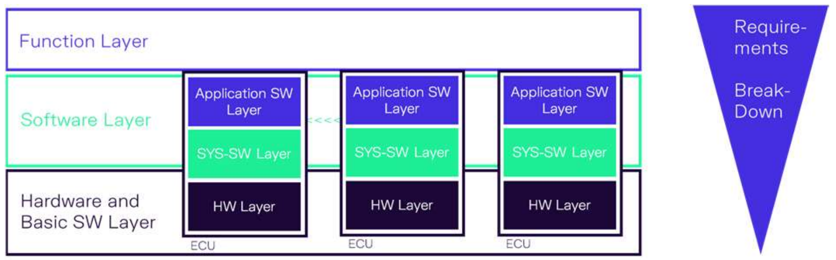
Complexity of SW Integration on ECU level is dependent to number of suppliers and its contracts with OEM. Test on single ECU level can only be done for function fragments and basic software of SYS-SW Layer

ECU realization is different to "One Supplier ECU" development: No unique responsibility and testability

Quality Management

Quality Management

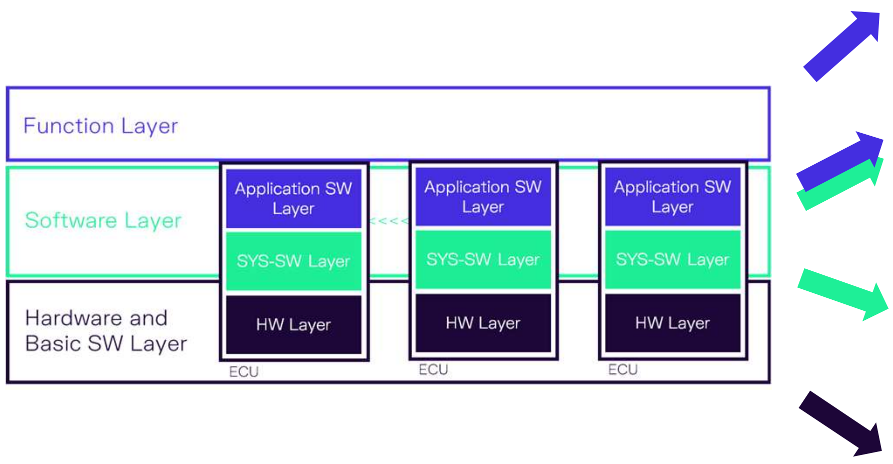
Prerequisites and Circumstances



- No explicit standards existing for the presented 3 staged layer model
- VDA 2 to be used for releasing ECU HW and SW - VDA 2 is entry point for quality management
    - but ECU responsibility and testability is not given for standalone ECUs with software components
- Releasing verification and validation tests are done on car level based on functions
    - Cross functionality (safety, security, diagnosis, network, Update/OTA) is associated to functions
- E2E Functional Chain is the connecting link between Function and Software/Hardware Layer
- Key drivers of complexity are the interfaces on human and technical level (see above)
    - Interfaces needs to be in focus of an effective quality management

Challenges for Quality Management to adapt existing standards for all three levels

Quality Management – Maturity of elements



Function Maturity:
Maturity Grade for every function including cross functionality (safety, security, diagnosis, network, OTA) according a life cycle from idea to release

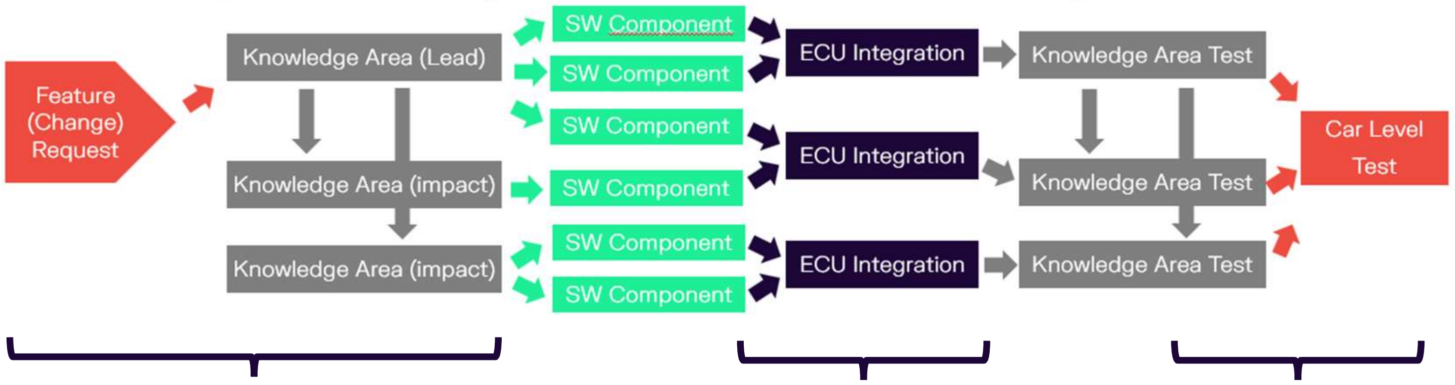
Function-Software Maturity:
Maturity Grade for E2E functional chain according (e.g. SW Maturity of SW components of functions) life cycle from idea to release

Software Maturity:
Maturity Grade for every software component and aggregated software components on ECU level. VDA2 is reference (product/process quality)

Hardware Maturity:
Maturity Grade for every ECU based on standards, e.g. VDA2, VDA6.3, APQP

Maturity of elements of 3 layer development model is key for quality control and releasing E2E System

Quality Management – Maturity of Interfaces



CCB with quality gate for every feature (change) request.
Severity of errors in domain specification are spread 1:N to SW components

ECU Integration quality gate:
Check of test vectors of all integrated SW component.
Execution of OEM defined tests as outcoming inspection

Car level Integration quality gate:
Incoming inspection of all ECUs based on test vector results for product and production.
Smoke tests of test vectors of all Knowledge Areas as first level test

Managing quality by mastering the complexity of the model

Quality Management – Limitations

- Existing standards are often based on “functionality on one ECU”
  Standards needs to be adapted and reworked for three staged layers systems

- Automotive SPICE is a very well-established process quality management measurement method on SW development level but has gaps on system level (e.g. SYS.3 System Architecture) and needs to be adapted on function level

- Many software metrics are historically defined for hand coded C software on Classical Autosar systems
  - Today’s high-performance ECUs use modern computer languages (e.g. JAVA, C#) running on modern operating systems (e.g. Linux, Android) with dynamic memory usage instead of statical memory allocation
  - Code generation is used for many software applications
  - Metris needs to be adapted, newly defined or set to obsolete. Identification of effective metrics is key! For less effective metrics cost-benefit ratio needs to be analyzed!

Historical quality assurance metrics and standards needs to be analyzed and adapted

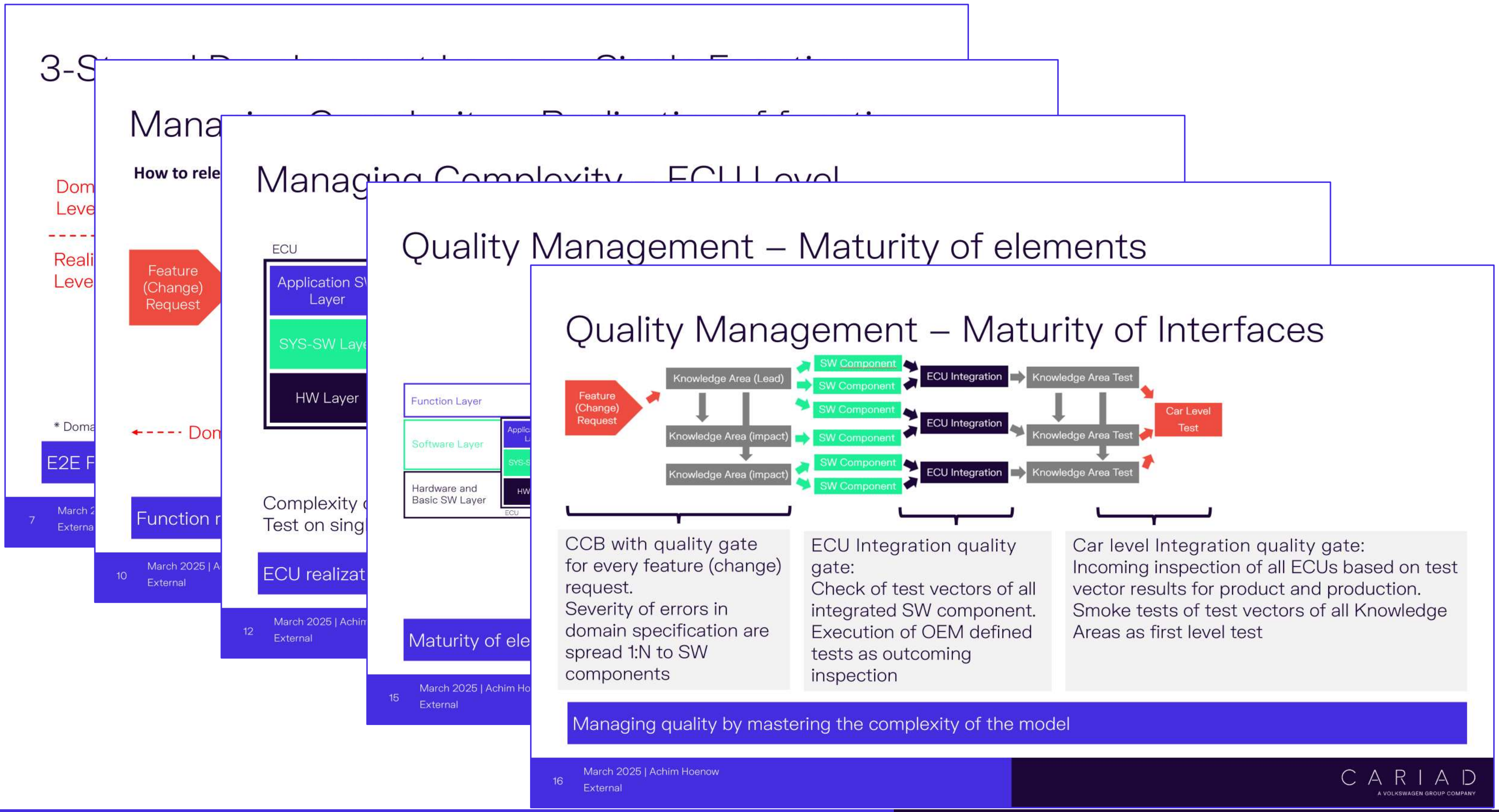
Outlook & Summary

Outlook

Further complexity drivers are known and needs to take into coordination for an E2E architectur

- Updates (Digital lifecycle management)
- Operating Systems
- Backwards compatibility
- Carry Over Parts (cross different E2E architectures)
- HW Revisions
- Different Tool Chains (e.g., update tool chains of different brands)
- Fulfillment proof of „State of the Art“
- etc.

Summary



3-S

Mana

How to rele

Managing Complexity – ECU Level

Dom Leve

Reali Leve

Feature (Change) Request

ECU

Application S Layer

SYS-SW Laye

HW Layer

* Doma

E2E F

Don

Function r

Complexity Test on sing

ECU realizat

Quality Management – Maturity of elements

Function Layer

Software Layer

Hardware and Basic SW Layer

Maturity of ele

Quality Management – Maturity of Interfaces

CCB with quality gate for every feature (change) request.
Severity of errors in domain specification are spread 1:N to SW components

ECU Integration quality gate:
Check of test vectors of all integrated SW component.
Execution of OEM defined tests as outcoming inspection

Car level Integration quality gate:
Incoming inspection of all ECUs based on test vector results for product and production.
Smoke tests of test vectors of all Knowledge Areas as first level test

Managing quality by mastering the complexity of the model

7 March 2 Externa

10 March 2025 | A External

12 March 2025 | Achim External

15 March 2025 | Achim Ho External

16 March 2025 | Achim Hoenow External

CARIAD A VOLKSWAGEN GROUP COMPANY

Thank you!

We transform automotive mobility

CARIAD
A VOLKSWAGEN GROUP COMPANY

Abstract

In the past, an electronic control unit (ECU) realized one or more functions with very little interoperability towards other ECUs . An OEM usually outsourced the ECU development to exactly one supplier with full warranty and liability.
Today, in addition to classic ECUs, vehicles have high-performance computers that together implement complex functions, such as Plug and Charge, driver assistance, connect and infotainment functions.

A complex function is realized by a chain of software components on a number of ECUs, which leads to three development layers:
i) Functions ii) ECU/Hardware with basic software and iii) Software Components

The development challenges arise from the high complexity that results from multi-matrix organizations which is needed to develop the so called End-to-End (E2E) Architecture.
Which proven and new quality methods can be used to master the challenges to release the E2E Architecture according standards? Which methods are efficient or less valuable or even obsolete?