Dr. Lars Völker, BMW Group.

COMMUNICATION PROTOCOLS FOR ETHERNET IN THE VEHICLE

AUTOMOTIVE BUS SYSTEMS AND ETHERNET
09 – 11 DECEMBER 2013, STUTTGART MARRIOTT HOTEL SINDELFINGEN

BMW
GROUP

AGENDA

- Use cases and requirements
- Example protocol stack
- Challenges on the basis of SOME/IP
  - (scalable Service-Oriented MiddlewarE over IP)
- Summary

USE CASES AND REQUIREMENTS

- Question: What do you want to do with Ethernet in the vehicle?

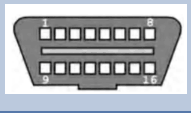
USE CASES AND REQUIREMENTS
SELECTED USE CASES FOR ETHERNET IN VEHICLES

Infotainment systems e.g. RSE connection or AV transport

Drivers Assistance e.g. surround view

Ethernet Domain Backbone

Diagnostics and Flash Update



USE CASES AND REQUIREMENTS
REQUIREMENTS FOR AUTOMOTIVE MIDDLEWARE

- Support CAN like communication

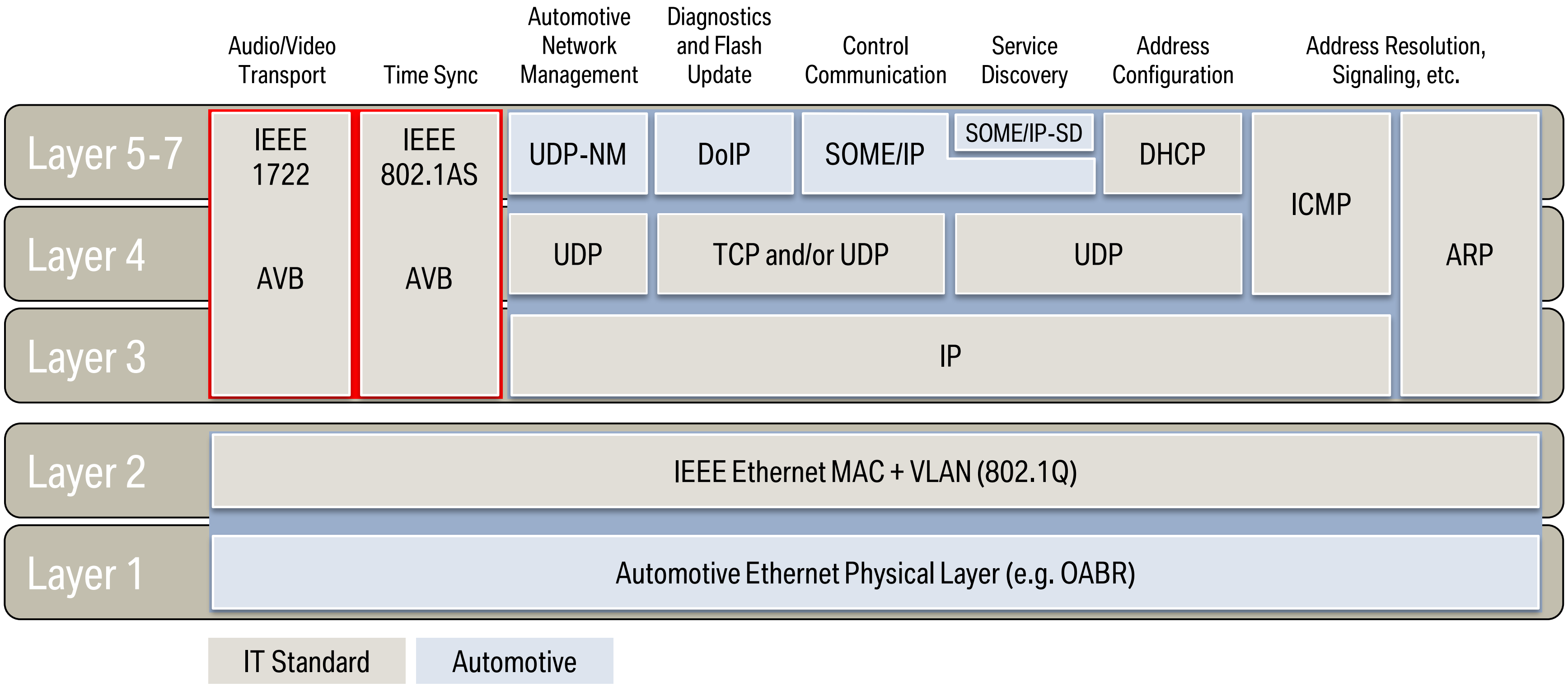
- Support MOST like control communication

- Shall efficiently support switched medium like Ethernet (not a bus!)
  - Support unicast communication
  - Limit multicast/broadcast to acceptable level

USE CASES AND REQUIREMENTS
WHAT HAVE WE DONE?

- With Ethernet, we tried to reuse as many protocols as possible:
  - Ethernet is a well proven technology
  - TCP/IP stack is proven solution, why develop a new one?
  - Parts of AVB fit out of the box, others need some adaption
- However:
  - Finding a suitable middleware solution to transport control data was not that easy!
- So SOME/IP was created!

EXAMPLE PROTOCOL STACK



Audio/Video Transport

Time Sync

Automotive Network Management

Diagnostics and Flash Update

Control Communication

Service Discovery

Address Configuration

Address Resolution, Signaling, etc.

Layer 5-7

Layer 4

Layer 3

Layer 2

Layer 1

IEEE 1722

AVB

IEEE 802.1AS

AVB

UDP-NM

DoIP

SOME/IP

SOME/IP-SD

DHCP

ICMP

ARP

UDP

TCP and/or UDP

UDP

IP

IEEE Ethernet MAC + VLAN (802.1Q)

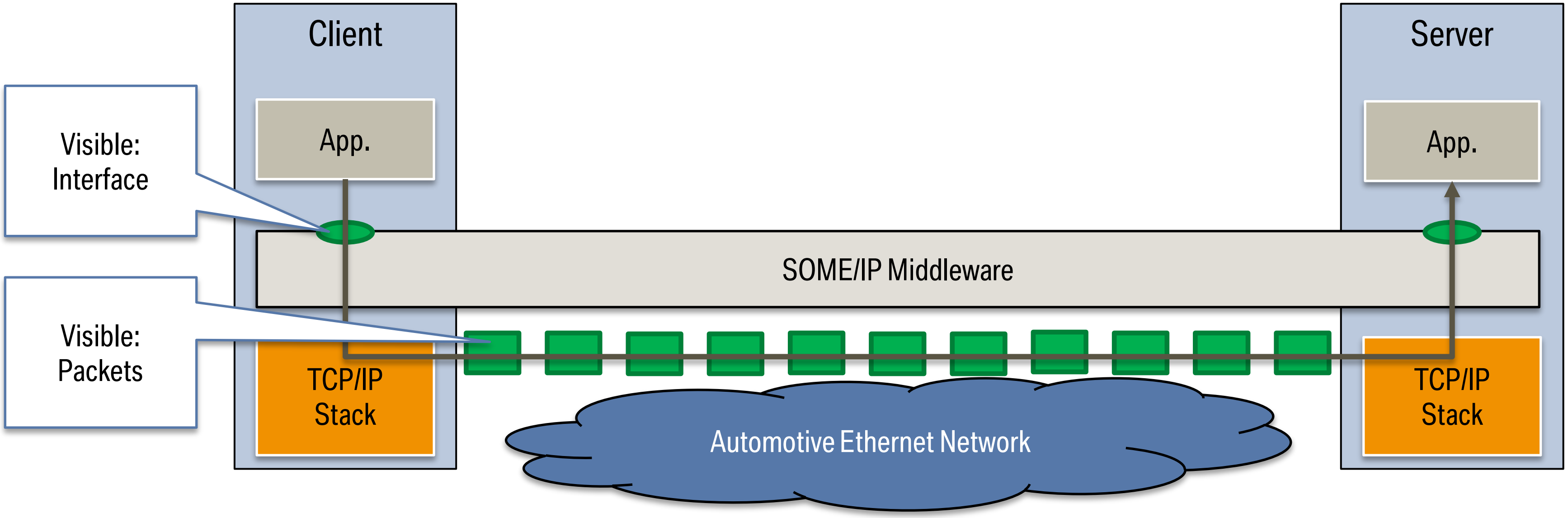
Automotive Ethernet Physical Layer (e.g. OABR)

IT Standard

Automotive

Most parts are reused but on Layer 1 and Layer 7 specific protocols are needed.

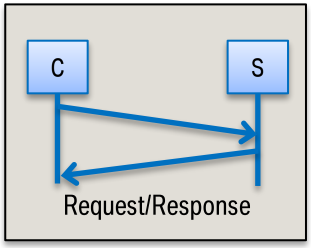
SOME/IP CONCEPTS
MIDDLEWARE

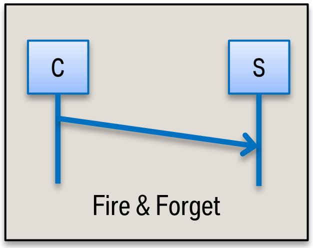


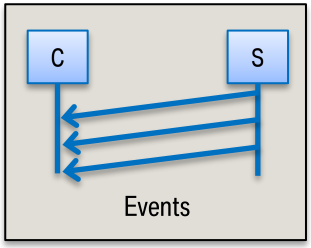
- **SOME/IP** allows applications to communicate.
- Packet formats are automatically determined by the specification of the Service.
- **Server** offers a **Service Instance** that implements the **Service Interface.**
- **Client** uses the **Service Instance** using **SOME/IP**.

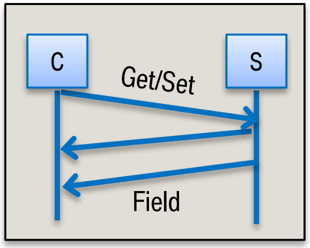
SOME/IP allows applications to communicate over Ethernet and TCP/IP.

SOME/IP CONCEPTS SERVICE









Client ≠ AUTOSAR Client/Server.

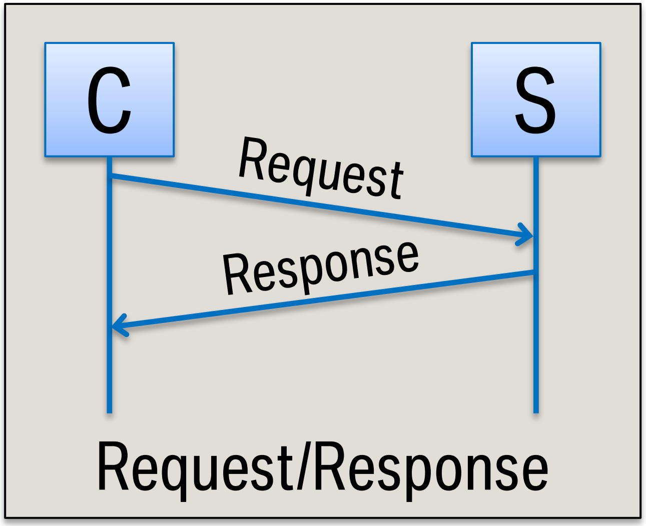
Services

A Service is defined by its Service Interface. This is comparable to a MOST Functional Block (FBlock) and may include:

- **Methods:**
  - With response (Request/Response).
  - Without response (Fire&Forget).
- **Events:** Message from Server to Client when something happens.
- **Fields:** Getter/Setter/Notifier of a property/status.
- **Eventgroups:** A logical group of Events and Fields used for publish/subscribe handling.

A Service is an interface that contains Methods, Fields, and Events.

SOME/IP CONCEPTS
REQUEST WITH RESPONSE METHOD



Services: Request/Response Methods

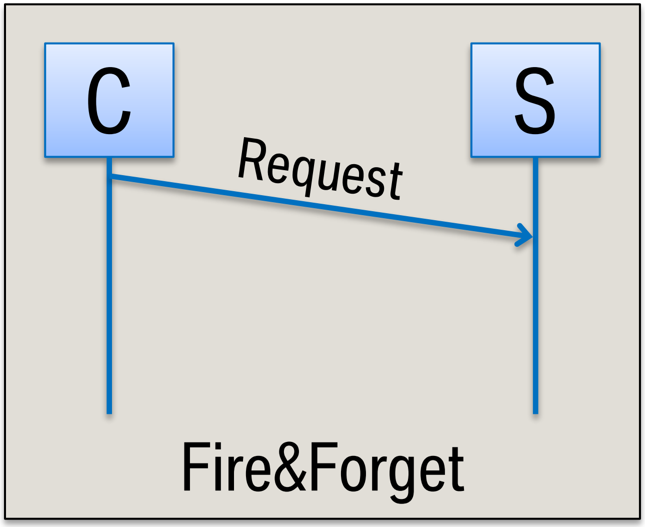
Request – a message from client to server calling a method.

Response – a message from server to client transporting the results of the method invocation.

Request/Response – a method call with Request and Response messages.

Request/Response methods allow calls with answers.

SOME/IP CONCEPTS
FIRE&FORGET METHOD



Services: Fire&Forget Methods

Request – a message from client to server calling a method.

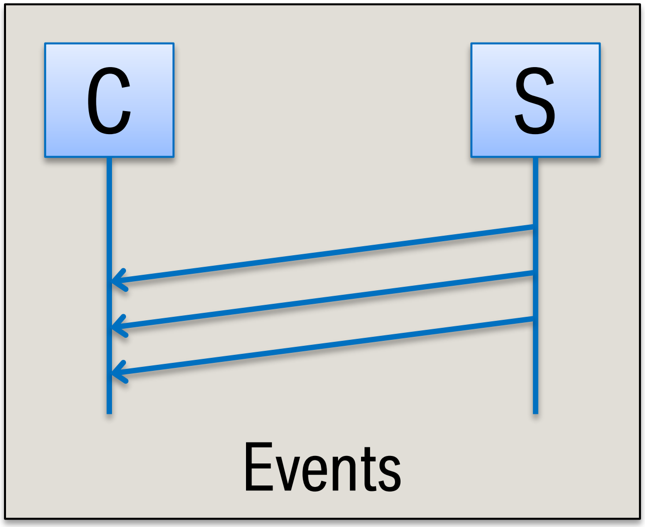
Fire&Forget – a method invocation with just a Request message.

Message Type „REQUEST_NO_RETURN“.

Does not support answers and errors.

Fire&Forget methods do not have answer messages.

SOME/IP CONCEPTS EVENTS



Services: Events

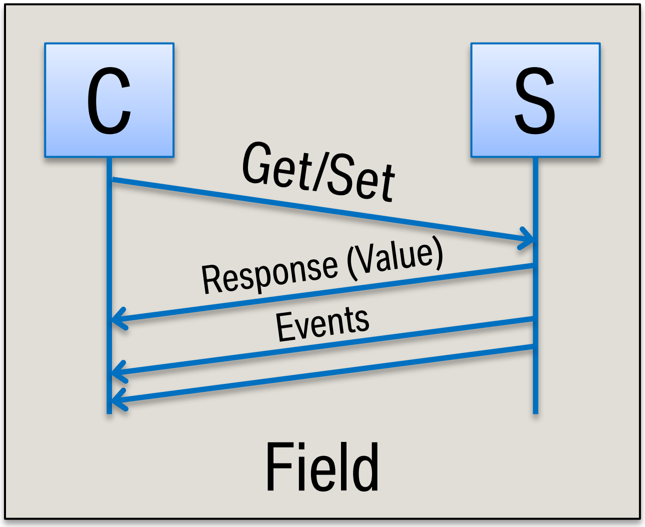
Event – a Fire&Forget callback, that is sent out by the Server (e.g. cyclically or on change).

Sent from Server to Client.

Similar to regular CAN messages.

Events are simple messages from Server to Client.

SOME/IP CONCEPTS FIELDS



Services: Fields

Field – represents a remote accessible property that includes Getter/Setter and/or Notification.

Getter – Method to read field value.

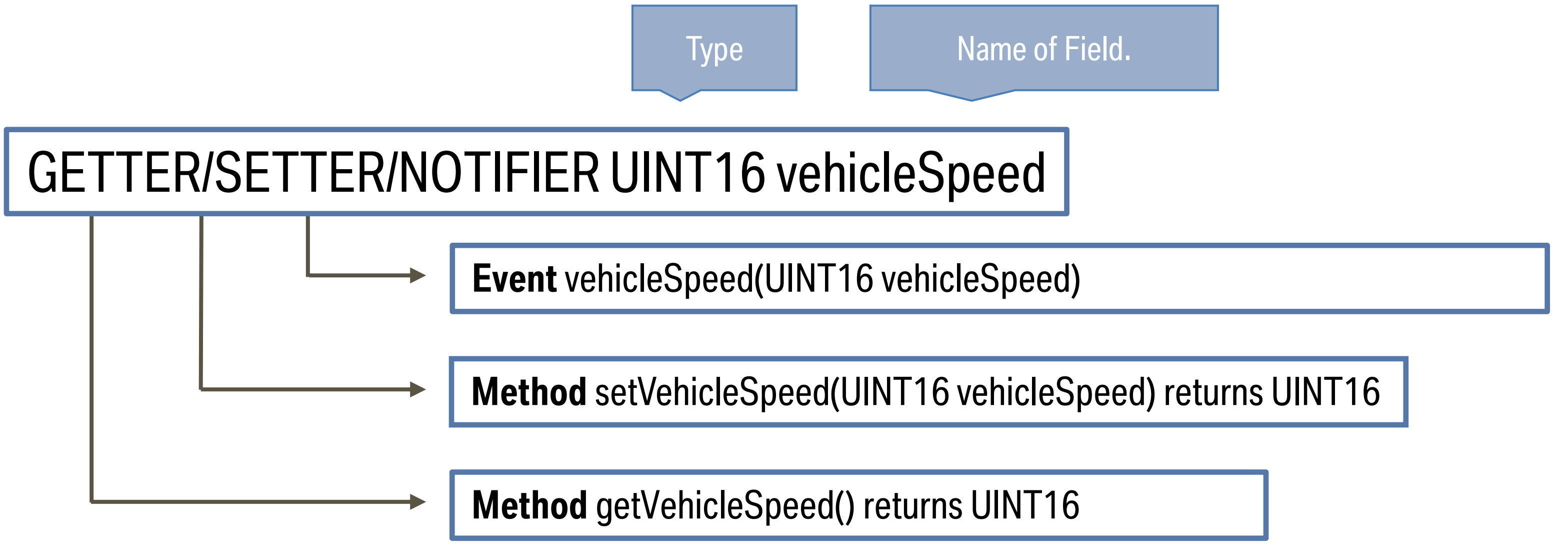
Setter – Method to set field value.

Notification (sends out Events with new values on change of field value).

Similar to a property on MOST.

Fields are properties that may include a Getter, a Setter, and a Notification.

SOME/IP CONCEPTS
FIELDS (2)



Type

Name of Field.

GETTER/SETTER/NOTIFIER UINT16 vehicleSpeed

Event vehicleSpeed(UINT16 vehicleSpeed)

Method setVehicleSpeed(UINT16 vehicleSpeed) returns UINT16

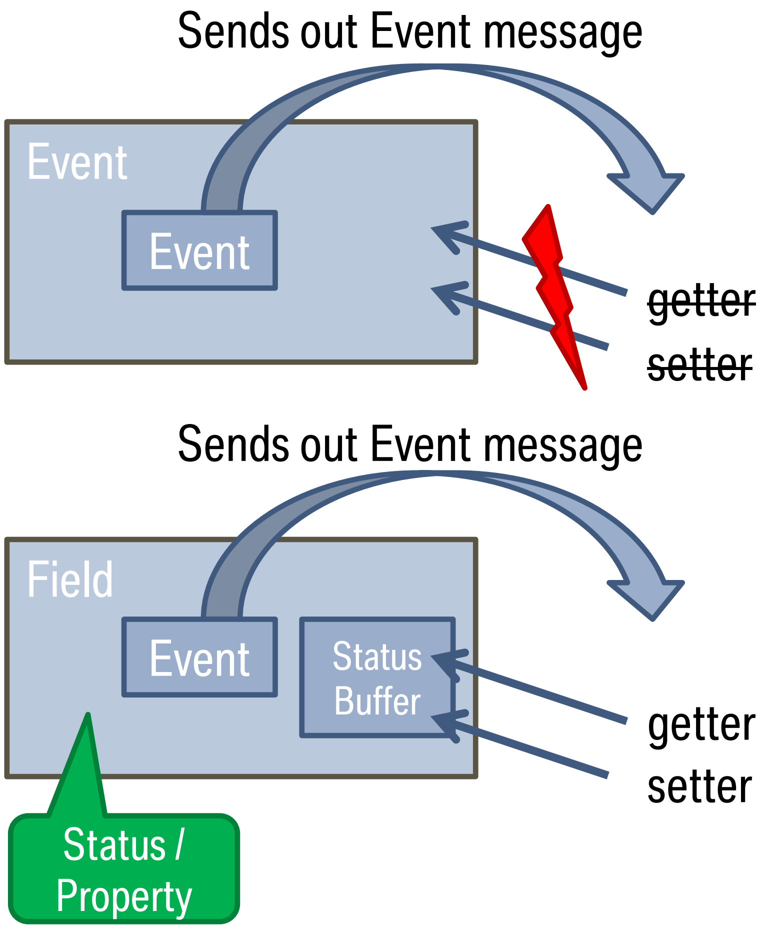
Method getVehicleSpeed() returns UINT16

Field consists of:

- 0..1 Getter → Request/Response
- 0..1 Setter → Request/Response
- 0..1 Notifier → Events

Field consists of Getter, Setter, and Notifier.

SOME/IP CONCEPTS
EVENTS VS. FIELDS

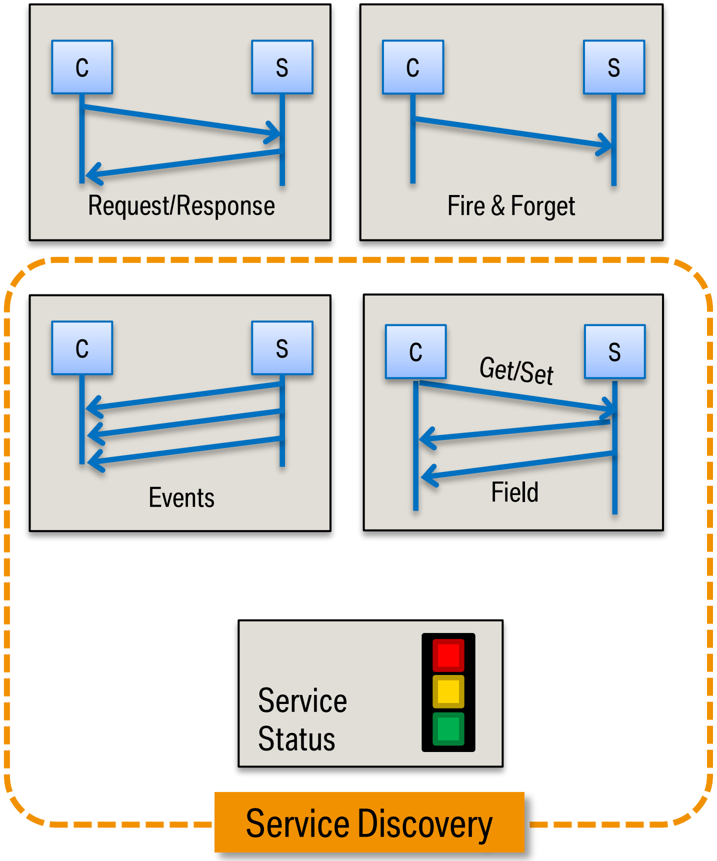


- Event is only when something happens.
- Events do not have initial values.
- The lifetime of an Event is not defined.

- Status based elements shall be modeled as Field.
- Event messages of Event and Field are identical.
- Difference:
  Initial Events only exist for Fields.

Use Events for time limited observations, Fields for status like data.

SOME/IP CONCEPTS
SERVICE DISCOVERY



Service Discovery

Service Discovery is used to explicitly signal:

- **Status of Service Instances (available or not)**
  - **as well as how to reach the service**
- **Publish/Subscribe**
  - **Which Events/Fields does a client need?**

This state is transported using cyclic messages carrying entries:

- **Service: Find, Offer, and StopOffer**
- **Eventgroup: Subscribe, StopSubscribe, SubscribeAck, and SubscribeNack**

Service Discovery transports status explicitly

Service Discovery handles Service State as well as Publish/Subscribe.

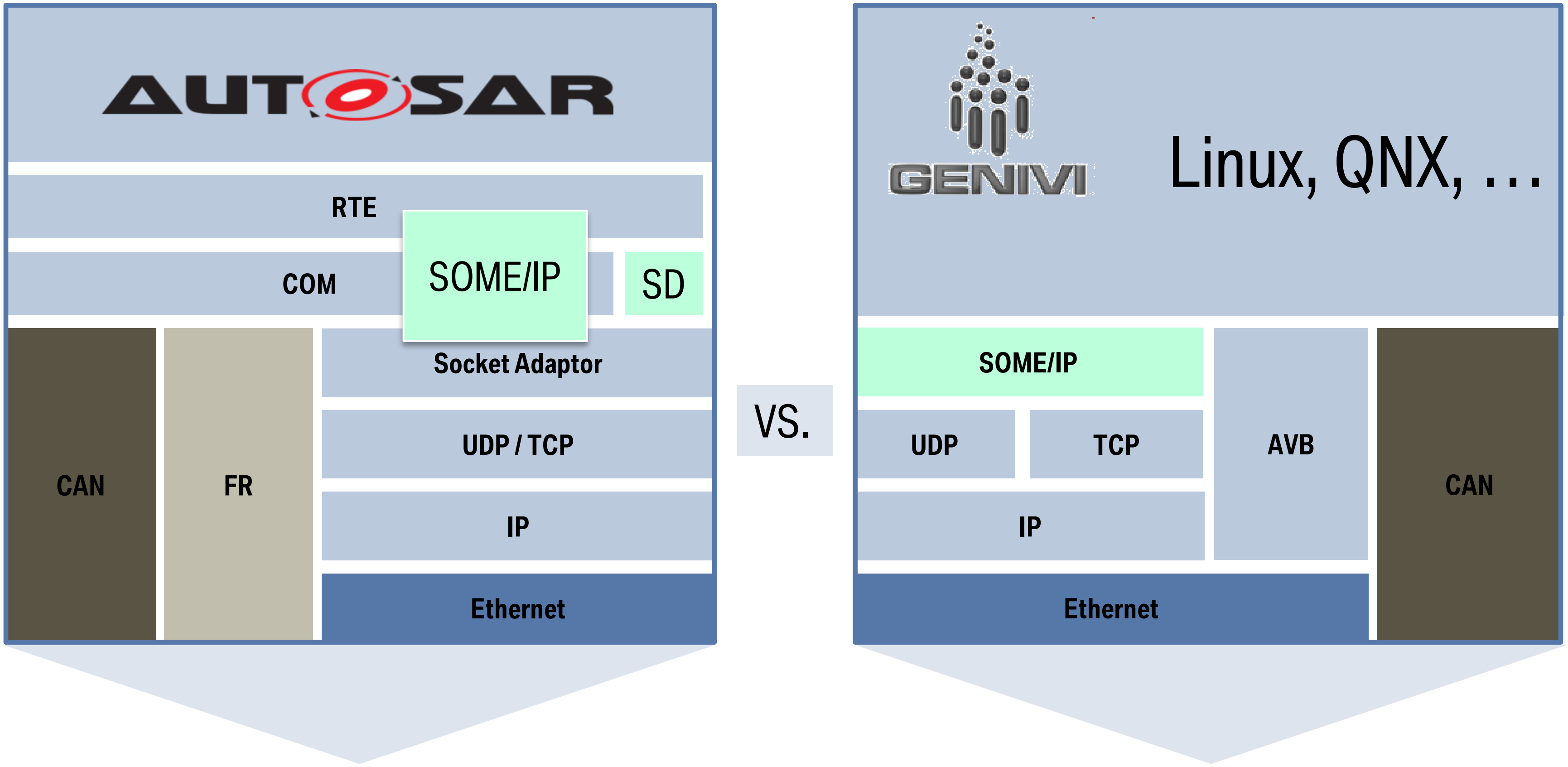
CHALLENGES
EXAMPLE SOME/IP

- Architecture for different systems (e.g. AUTOSAR and GENIVI)
  - AUTOSAR is based on CAN and FlexRay messages, Ethernet is more dynamic
  - Ethernet is common to Linux (e.g. GENIVI)
  - What's a good compromise for a protocol?

- Agile process (specification and implementation in parallel)
  - More innovation in less time
  - AUTOSAR process stressed

- Testing a complex protocol stack

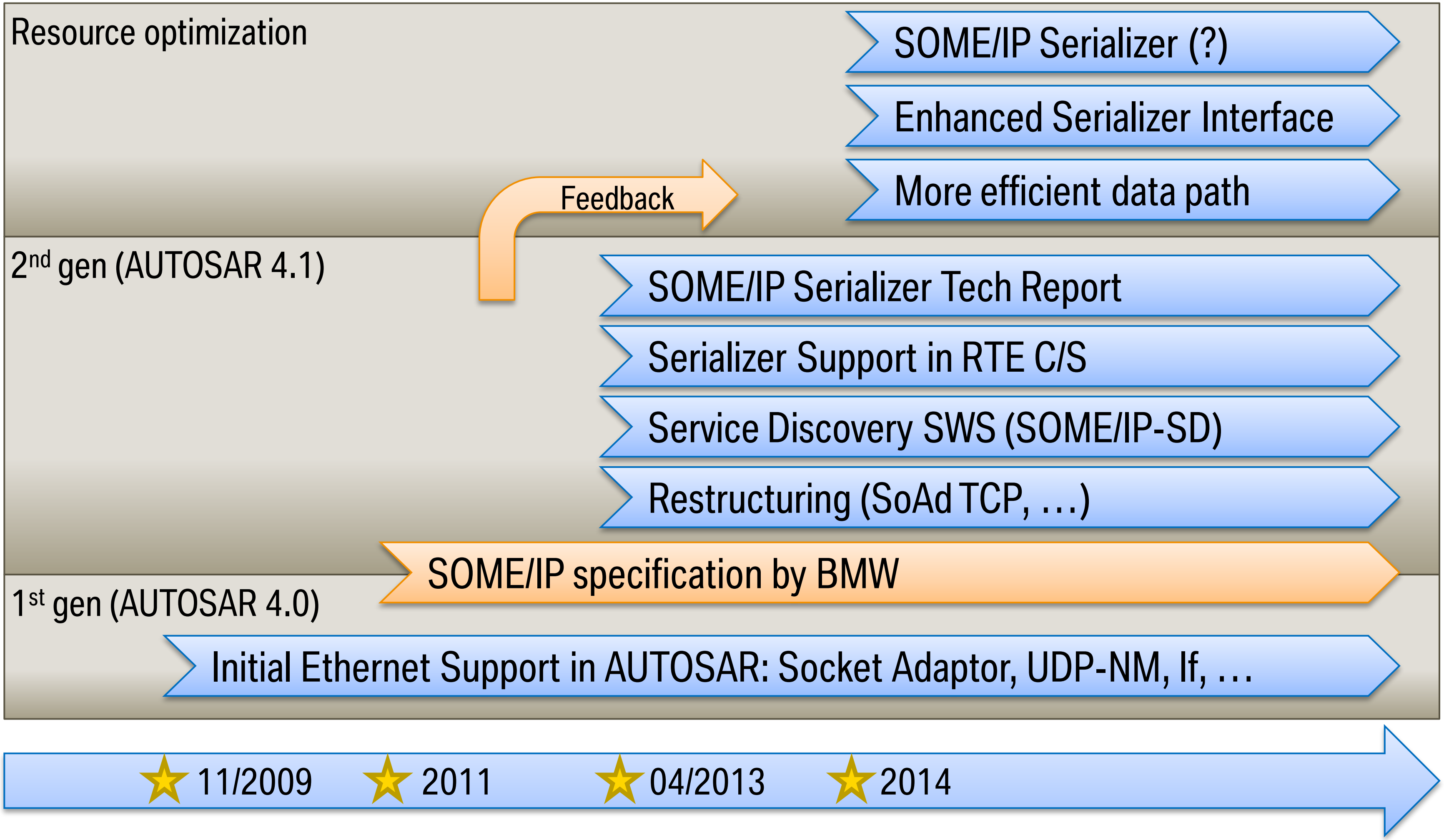
CHALLENGES AUTOSAR AND GENIVI



Socket Adaptor, COM and RTE for SOME/IP. SD has own module.

SOME/IP and SOME/IP-SD are implemented using library.

CHALLENGES
AUTOSAR ROADMAP



Resource optimization

SOME/IP Serializer (?)

Enhanced Serializer Interface

More efficient data path

Feedback

2nd gen (AUTOSAR 4.1)

SOME/IP Serializer Tech Report

Serializer Support in RTE C/S

Service Discovery SWS (SOME/IP-SD)

Restructuring (SoAd TCP, …)

SOME/IP specification by BMW

1st gen (AUTOSAR 4.0)

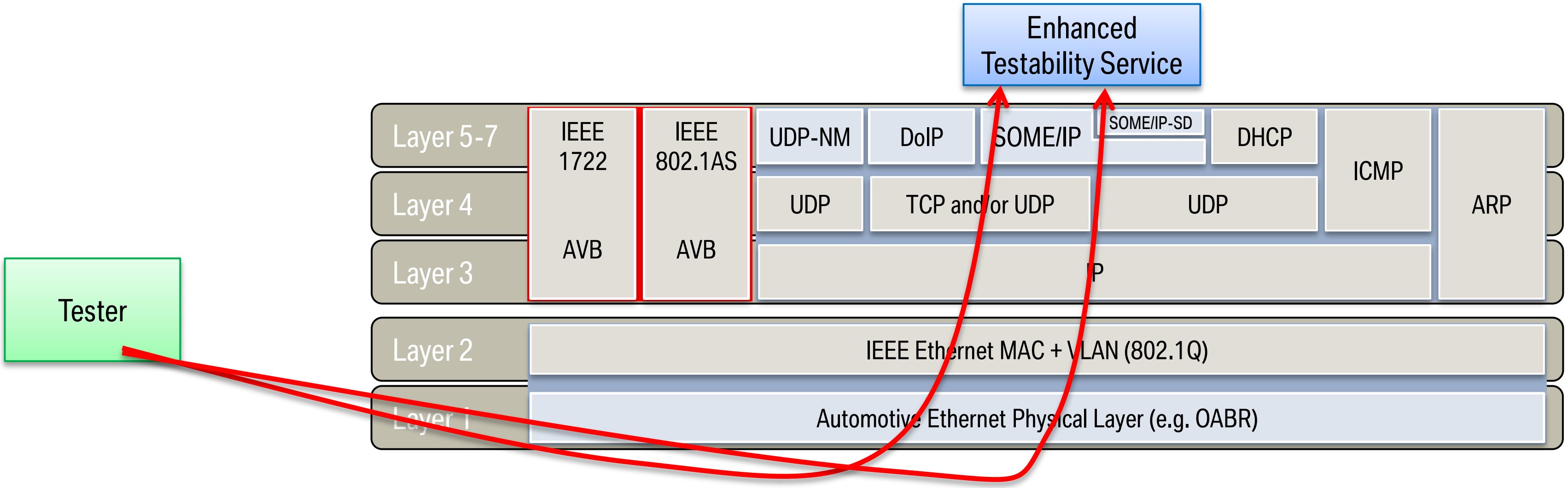
Initial Ethernet Support in AUTOSAR: Socket Adaptor, UDP-NM, If, …

★ 11/2009 ★ 2011 ★ 04/2013 ★ 2014

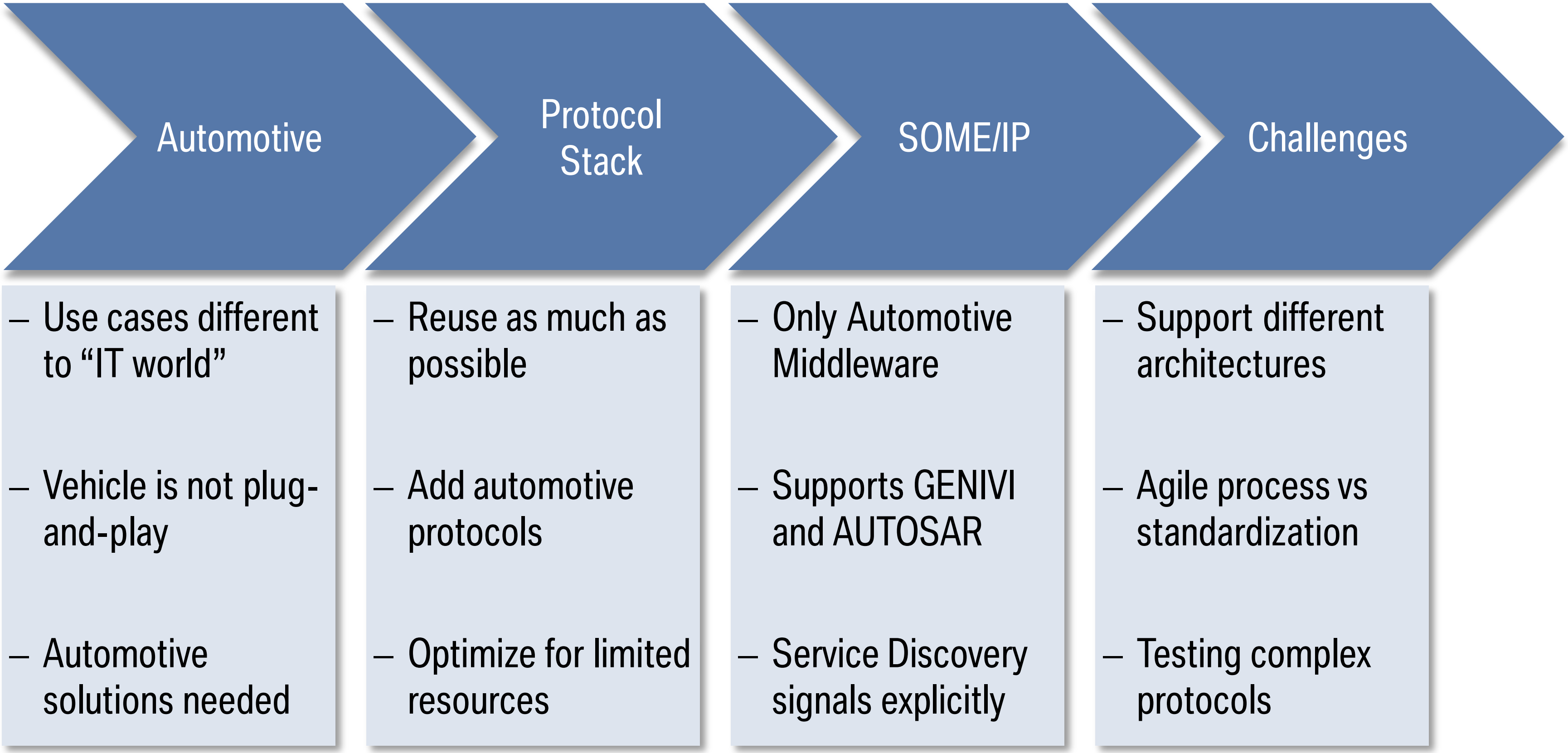
Fast pace roadmap reflected in AUTOSAR standardization.

CHALLENGES
TESTING SOME/IP AND SOME/IP-SD

- Protocol stack testing often requires support for the testing
  - The test application is called Enhanced Testability Service (ETS)
  - Different methods, events, and fields are included
  - Standardization of ETS is in discussion



SUMMARY



Automotive

- Use cases different to “IT world”
- Vehicle is not plug-and-play
- Automotive solutions needed

Protocol Stack

- Reuse as much as possible
- Add automotive protocols
- Optimize for limited resources

SOME/IP

- Only Automotive Middleware
- Supports GENIVI and AUTOSAR
- Service Discovery signals explicitly

Challenges

- Support different architectures
- Agile process vs standardization
- Testing complex protocols

Ethernet-based in-vehicle communication comes with solvable challenges!