# Scalable Service-Oriented Middleware over IP

Dai Yang

Seminar – Selected Topics:

Operating Systems and Distributed Systems

WiSe 2015/16

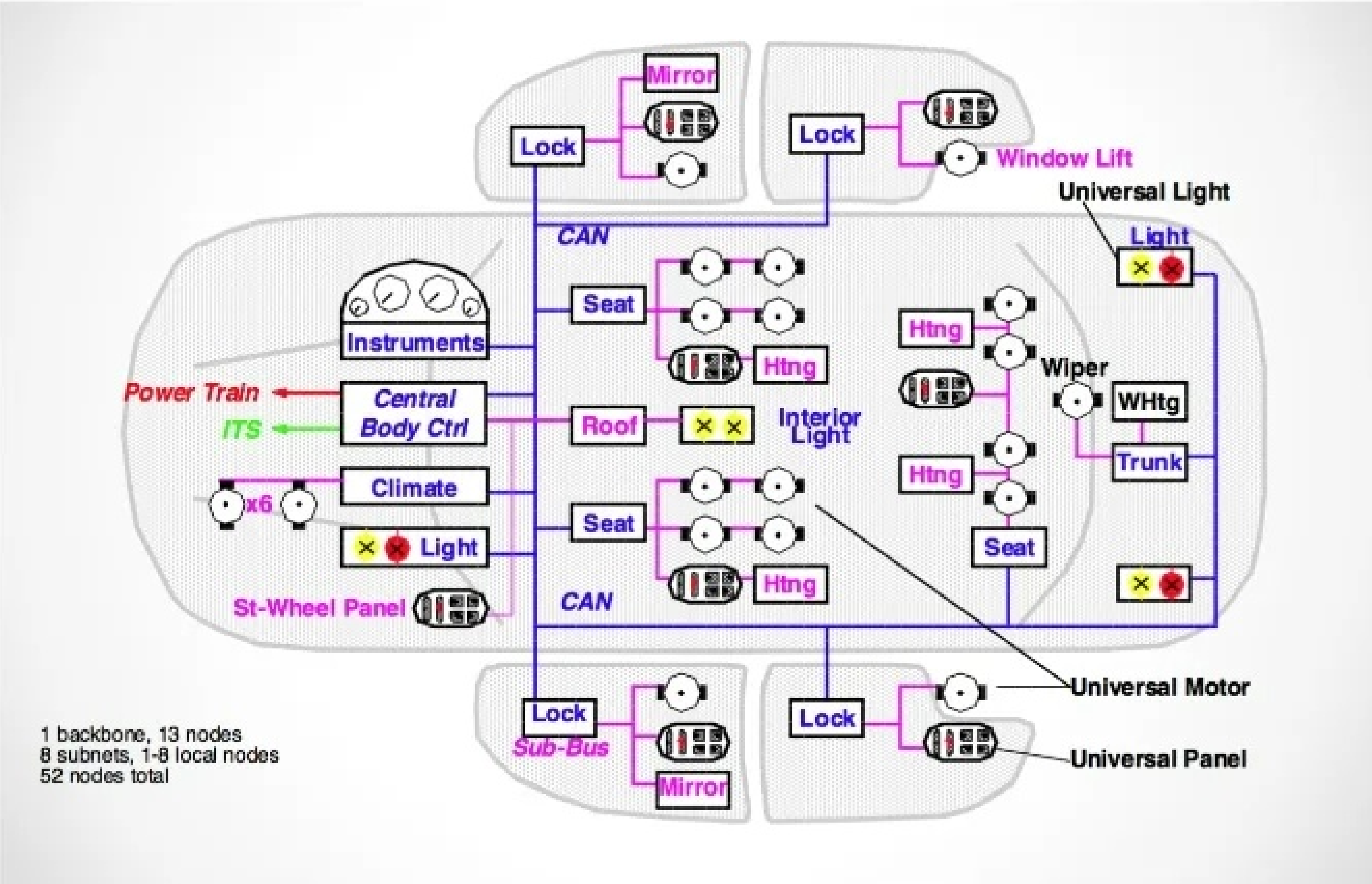
# Outline

- **Introduction**
- **Technological Background**
  - ISO/OSI Model
  - Ethernet and TCP/IP Stack
  - CAN Bus
  - Data Serialization
    - Extensible Markup Language
    - JavaScript Object Notation
    - Google Protocol Buffer
  - Automotive Open System Architecture
- **Scalable Service-Oriented Middleware over IP**
  - Description
  - Data Serialization
  - Service Orientation
  - Service Discovery
- **Evaluation and Future Work**
  - Evaluation
  - Future Work

# Motivation

## Automotive Onboard Network

Increasing Number of Nodes Onboard




Source:
تبعدمحمىحطا ,
**Embedded Systems in Automotive, Intel Coorperation,**
http://www.slideshare.net/ssuser92b33b/embedded-systems-in-automotive,
Last visit: 07.11.2015

# Scale of Electronic Control Units

- up to 100 ECUs
- up to 100 Mio. LoC (Lines of Codes)
- several in-vehicle Networks (CAN, FlexRay, MOST, etc.)
- Examples of ECUs:
  - ABS
  - ESP
  - Engine Control
  - Airbag
  - Navigation
  - Camera
  - Fuel Control
  - …

Source: Alternative Fuel ECU, Virtual R&D Ltd, http://www.virtrnd.com/alternative-fuel-electronic-control-unit/, Last visited: 07.11.2015

## Current and Target Designs

- many ECUs
  - all ASIC (Application Specific Integrated Circuit)
  - distributed
  - Car Model specific

- reduced number of ECUs
  - multipurpose
  - less-distributed or central
  - generic

- multiple Specific Networks
  - Low Data Rate
  - Cupper wired

- single onboard network
  - High Data Rate
  - optical wired

# Backgrounds

## ISO/IEC 7498-1

Open System Interconnection model (OSI Model)

- Application Layer (HTTP, Telnet, FTP, DNS, …)
- Presentation Layer (SSL, TLS, ASCII, MPEG, …)
- Session Layer (Sockets, NetBIOS, RPC, Named Pipes, …)
- Transport Layer (TCP, UDP, X. 224, …)
- Network Layer (IP, IPsec, ICMP, ARP, …)
- Data Link Layer (PPP, IEEE802.3, IEEE802.11, …)
- Physical Layer (G.703, RS-232, RS-422, CAN bus, …)

# Ethernet and TCP/IP Stack [0]

Comparison

**Ethernet**

- Family of Computer Networking Technologies for LAN
- IEEE 802.3
- Physical Layer
  - Optical Fiber
  - Twisted Pair
  - others (e.g. via MIL-DTL-38999)
- Link Layer
  - Carrier sense multiple access with collision detection (CSMA/CD)
- Up To 25 Gbit/s

**TCP/IP Stack**

- Internet Protocol Suite
- Layers:
  - Link Layer: MAC
  - Network Layer: IP
  - Transport Layer: TCP, UDP
  - Application Layer:
    - OSI Socket Layer
    - OSI Presentation Layer
    - OSI Application Layer

- Everything but the Application Layer also called “Ethernet” unofficially

# Ethernet and TCP/IP Stack [1]

## Different Connectors

RJ45 with Twisted Pair Cable [1]

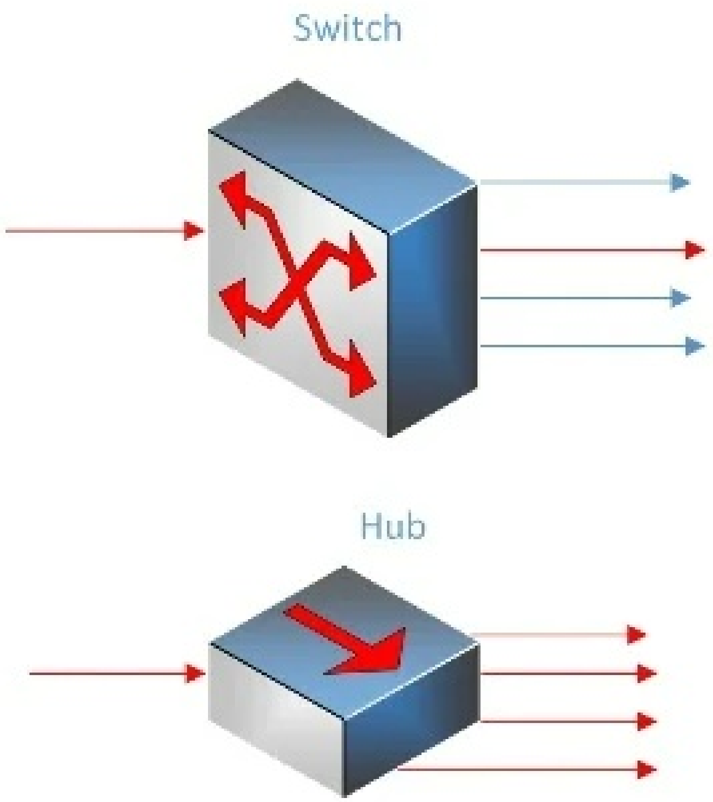

Optical Fiber with E-2000 Connector [2]

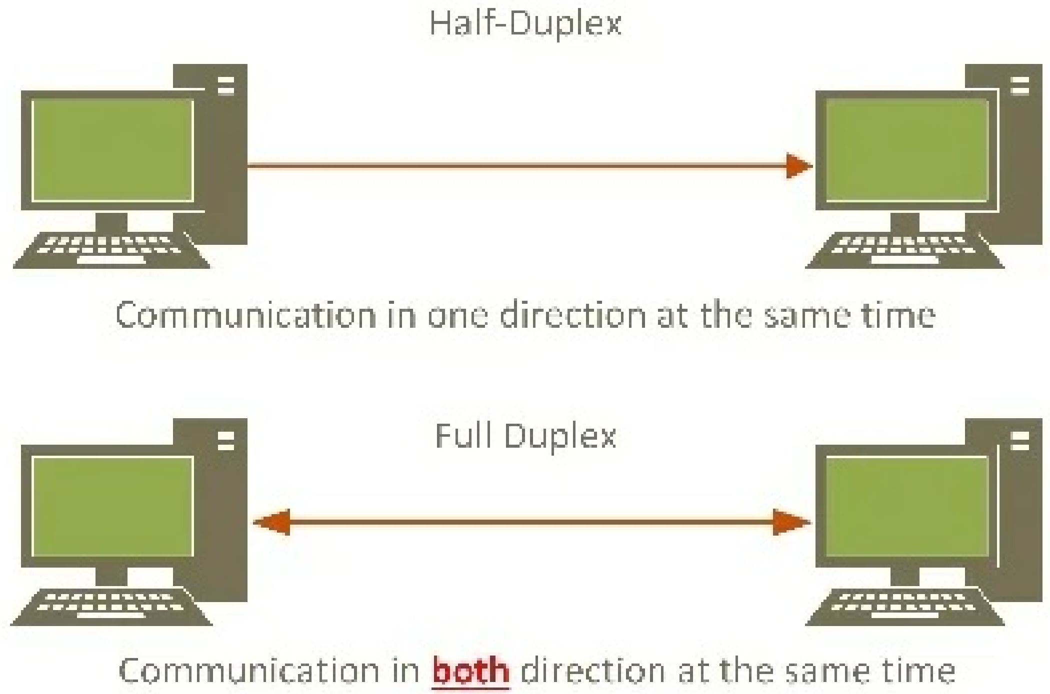
RJ45 to MIL-STD 38999 Connector [3]

Sources:
[1] Wikimedia Foundation, https://upload.wikimedia.org/wikipedia/commons/thumb/0/05/EthernetCableBlue2.jpg/800px-EthernetCableBlue2.jpg, last visited 15.11.2015
[2] UnternehmerTUM GmbH und Leibniz Rechenzentrum
[3] Airbus Defence and Space © 2015 COMPANY CONFIDENTIAL

# Ethernet and TCP/IP Stack [2]

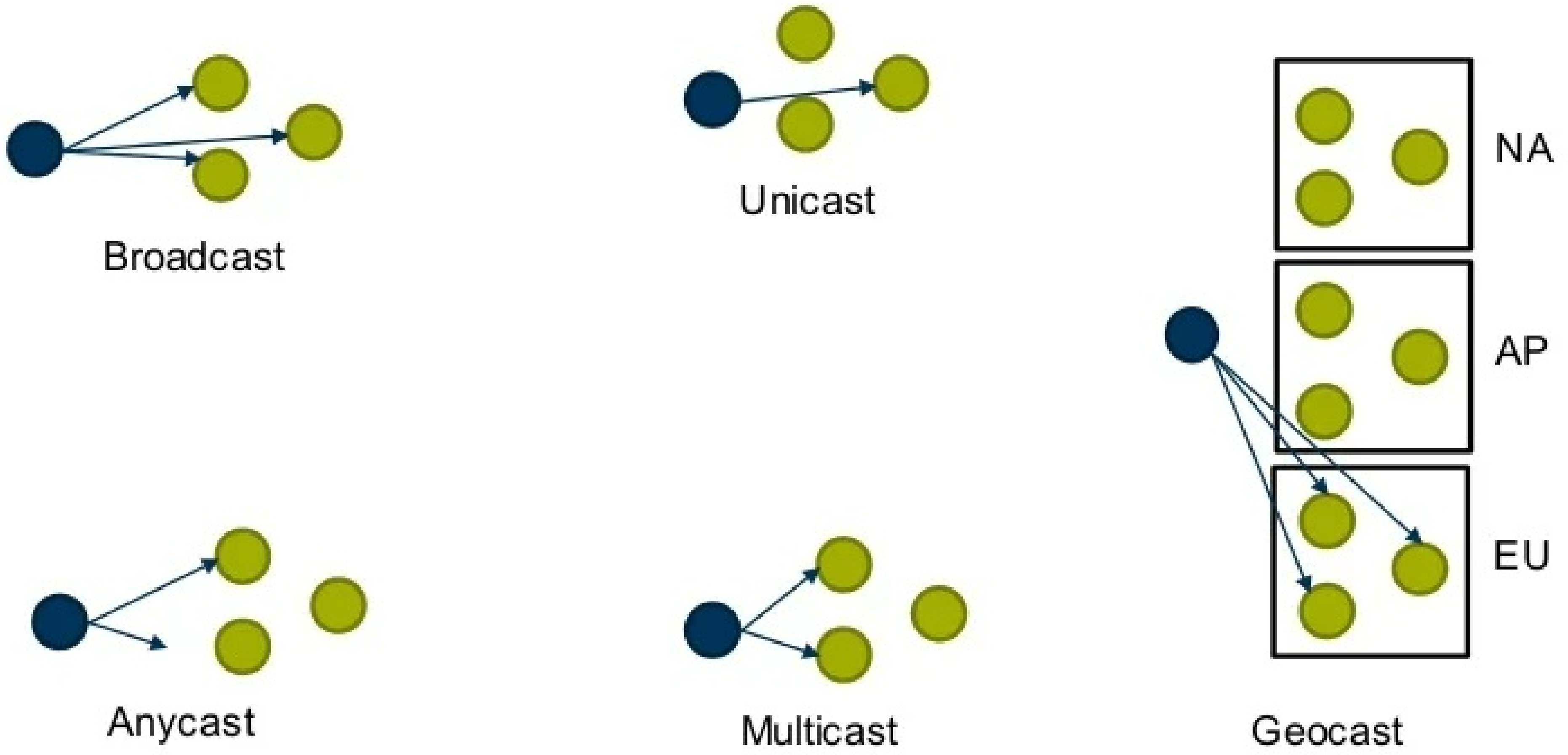
## Network Topology



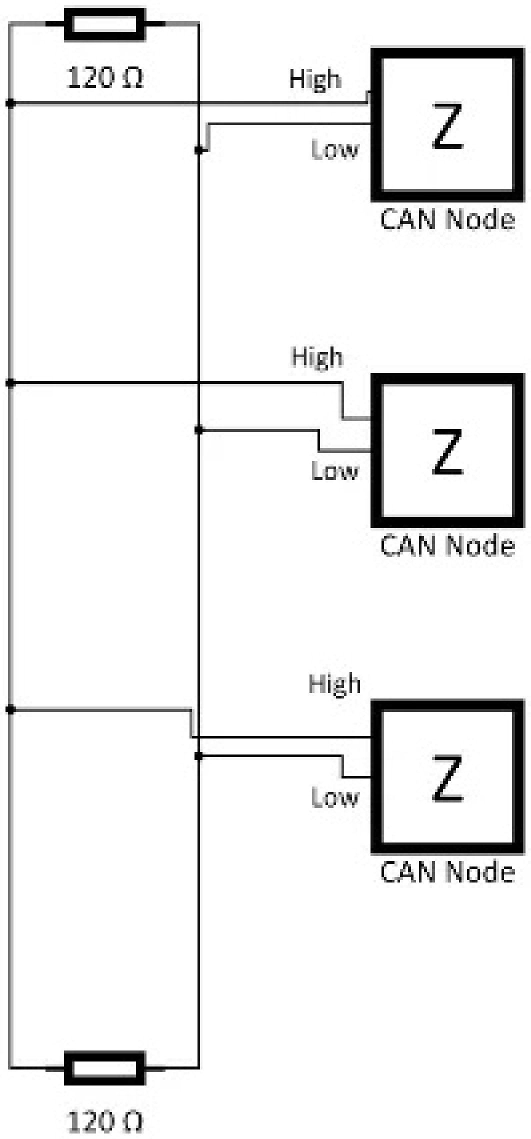
# Ethernet and TCP/IP Stack [3]

Basic Routing Schemes



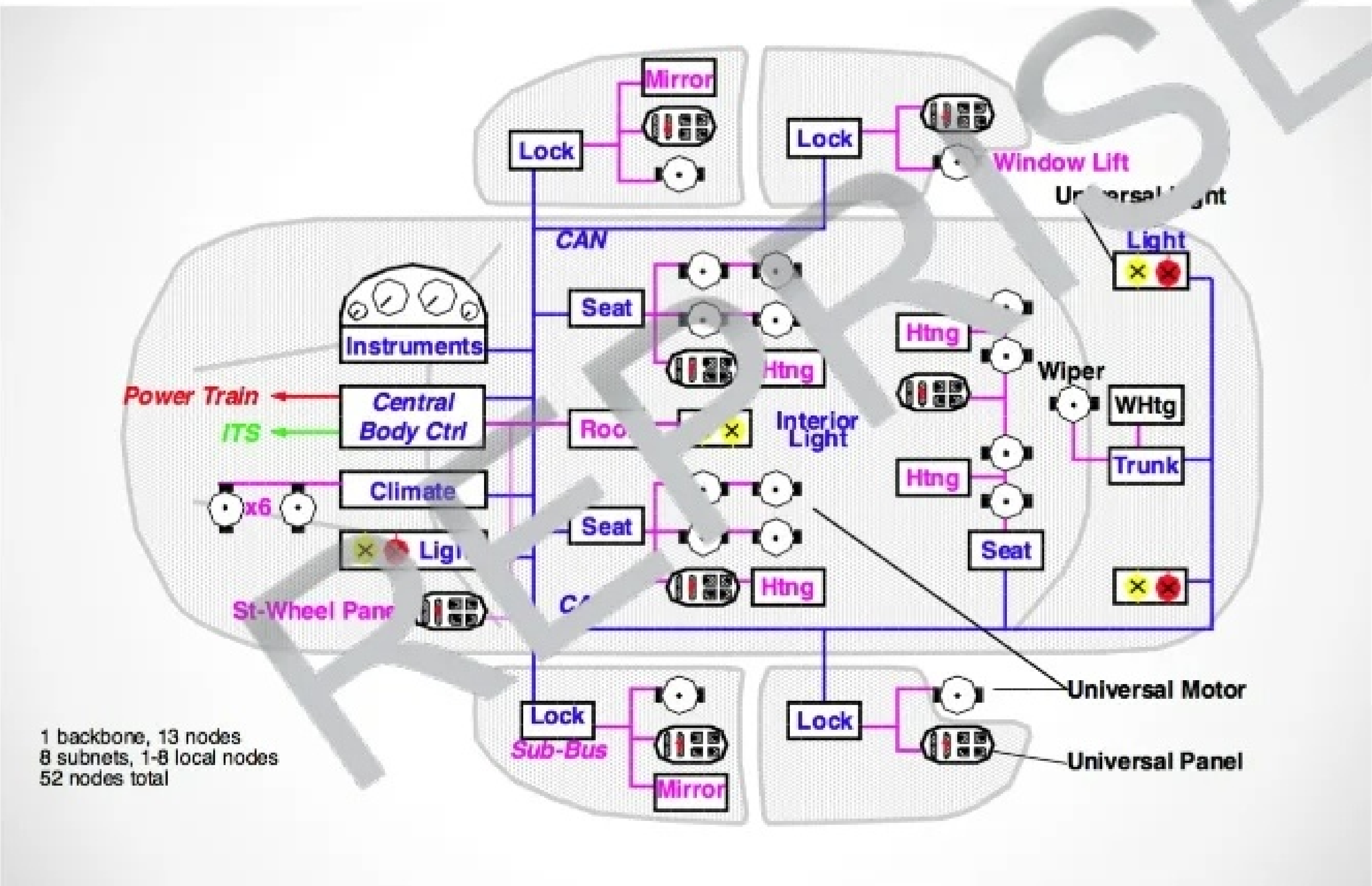
# Controller Area Network [0]

- Designed by Robert Bosch GmbH in 1983, released by SAE (Society of Automotive Engineers) in 1986
- ISO 11898 – 1
- Simple Circuit (picture right, High Speed CAN)
- A CAN Node contains:
  - Sensor / Actuator
  - CPU
  - CAN Controller (Data Link Layer)
  - CAN Transceiver (Physical/Electrical Layer)
- Field of Applications:
  - Automotive
  - Industrial
  - Entertainment
- Layers:
  - Application and Object Layer
  - Transfer and Physical Layer



# Automotive Onboard Network

## Increasing Number of Nodes Onboard

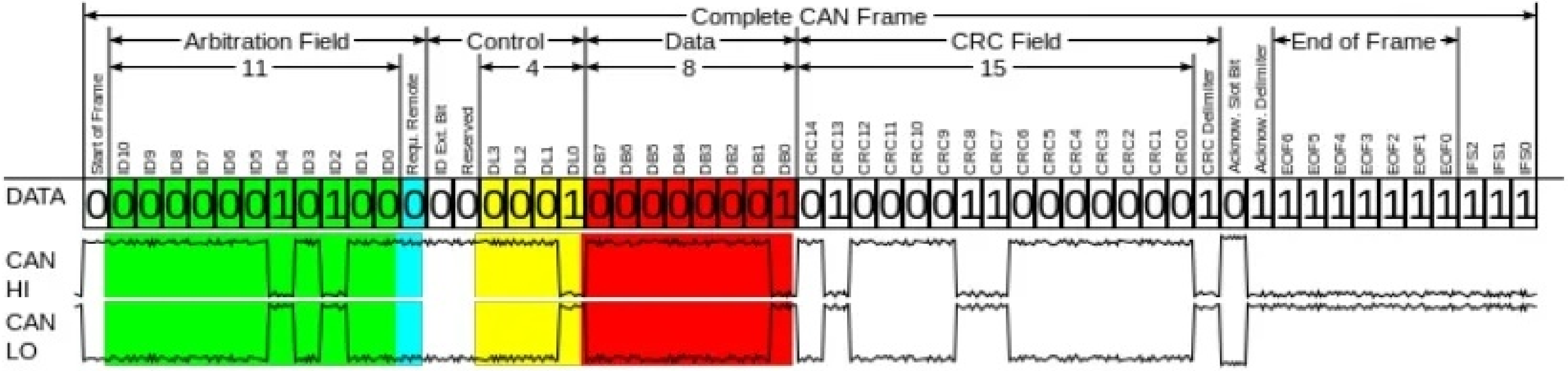





Source:
بوعدمحمیخا ,
**Embedded Systems in Automotive, Intel Coorperation,**
http://www.slideshare.net/ssuser92b33b/embedded-systems-in-automotive,
Last visit: 07.11.2015

# Controller Area Network [1]

CAN Message




Source: Wikimedia Foundation, https://commons.wikimedia.org/wiki/File:CAN-Bus-frame_in_base_format_without_stuffbits.svg, last visit: 07.11.2015

- Max. 8 bytes data per message
- Dominant Level = 0, Recessive Level = 1;
- Lower ID = Higher Sending Priority
- NRZ (Non-Return-to-Zero Code)
- Bitrate dependent to network length (40m ~ 1Mbit/s, 500m ~ 125kbit/s)
- Channel must be synchronized -> Bit Stuffing

- Green: ID
- Blue: Remote Transmission Request
- Yellow: Data Length
- Red: Data Field

# Data Serialization [0]

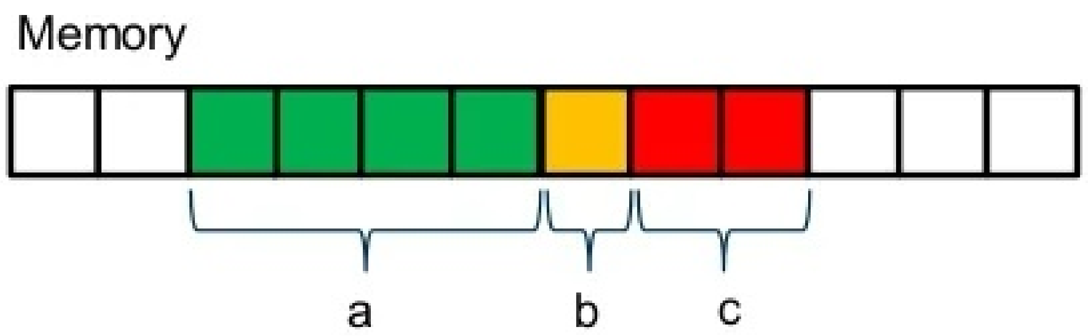
- Data Structures cannot be stored directly
- Data must be serialized before store/send
- easiest way: C

```c
#pragma (pack, 1)

typedef struct tag_foo{
int a;
unsigned short b;
char c[2];
}FOO;

#pragma (pop)

FOO myFoo;
…
```




Advantages:
- Compact
- Simple

Disadvantages:
- not human-friendly
- hard to debug

# Data Serialization [1]

## CAN

- restricted payload size
  - -> save storage
  - -> calculation algorithms

- Example
  - Torque of a Motor need to be transmitted in CAN in 12 Bit
  - Range of Values: 200 N·m – 820 N·m

  - Torque = ((CAN-Value) x 0.25) + 200 (N·m)
  - Resolution: 0.25 N·m

- Disadvantages:
  - Computational Effort
  - Resolution Restriction

# Data Serialization [2]

## Extensible Markup Language

- defined by W3C
- human-readable and machine-readable
- Unicode Support
- Application: Web Applications, Document Storage, etc.

Disadvantage:

- high syntax overhead

- Example:

```
#pragma (pack, 1)

typedef struct tag_foo{
int a;
unsigned short b;
char c[2];
}FOO;

#pragma (pop)

FOO myFoo;
…
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<myFoo>
	<a type="int">1337</a>
	<b type="unsigned short">10</b>
	<c type="char">
		<0>26</0>
		<1>27</1>
	</c>
</myFoo>
```

# Data Serialization [3]

## JavaScript Object Notation

- defined by Douglas Crockford
- human-readable and machine-readable
- valid JavaScript document
- Applications: Web Applications

- Example

```
#pragma (pack, 1)

typedef struct tag_foo{
int a;
unsigned short b;
char c[2];
}FOO;

#pragma (pop)

FOO myFoo;
…
```

```
{
    "a" : 1337
    "b" : 10
    "c" : [
            {0:26},
            {1:27}
    ]
}
```

Disadvantage:

- (still) high syntax overhead

# Data Serialization [4]

## Google Protocol Buffer

- developed by Google in 2008
- designed for data storage or data transfer
- user defined Protocol Definition files (.proto) -> precompiled to libraries (e.g. .h/.c, .java)
- can only be de-serialized if protocol definition presents
- Supports enumerations and classes

- Example Protocol Definition:

```
#pragma (pack, 1)

typedef struct tag_foo{
int a;
unsigned short b;
char c[2];
}FOO;

#pragma (pop)

FOO myFoo;
…
```

```
Message myFoo {
        required int32 a  = 1;
        optional uint8 b  = 2;
        repeated int8 c = 3;
}
```

# AUTOSAR [0]

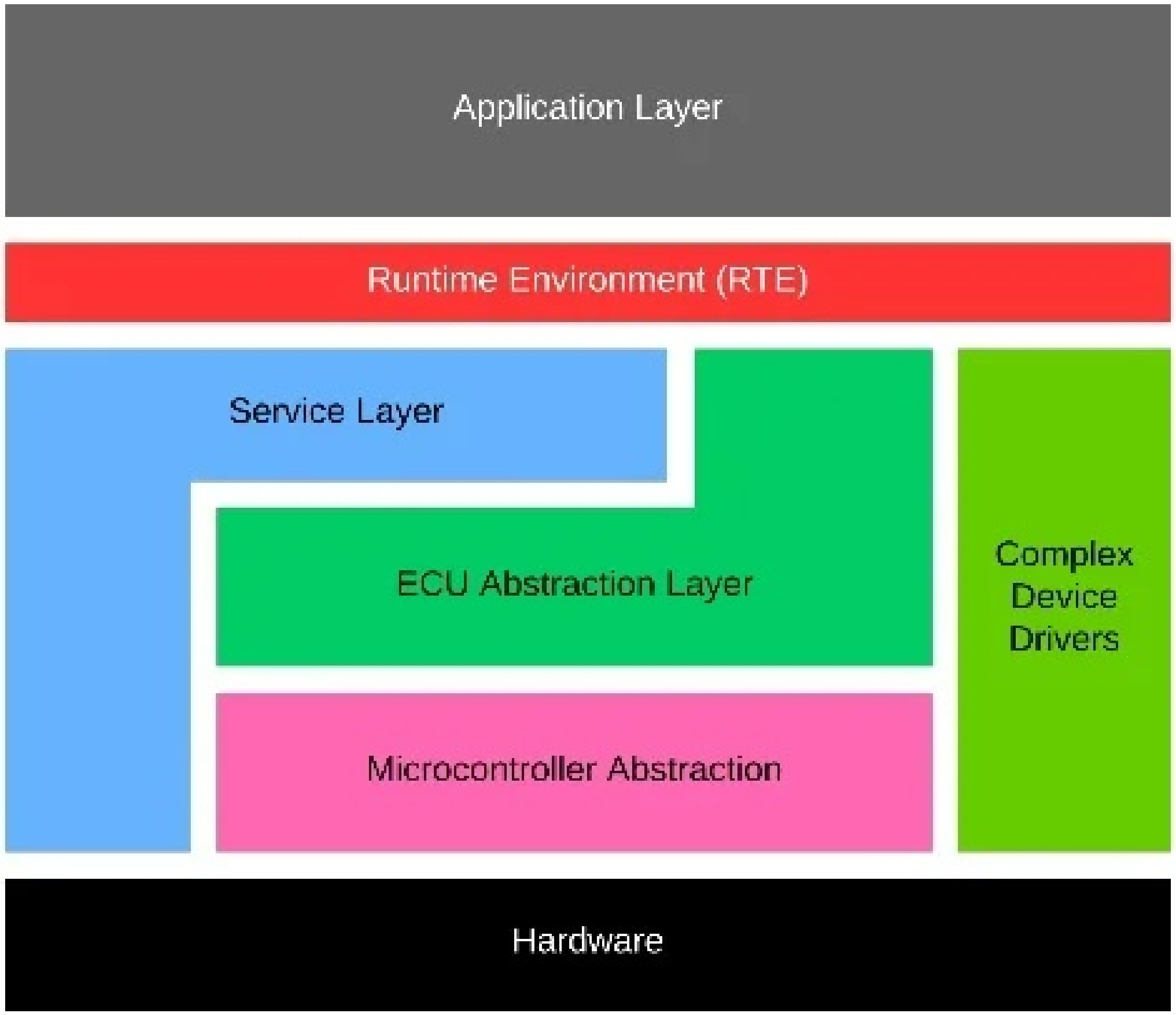
AUTomotive Open System ARchitecture

- Cooperation of Development
- Standardization of automotive Software
- Automobile Manufactures, suppliers, tool developers and OEMs
- Goals:
  - Implementation and standardization of basic system functions
  - Scalability
  - Integration between suppliers
  - Consideration of availability and safety
  - Redundancy
  - Maintainability
  - Increase the use of standard-software
  - Software upgrade over whole PLC (Product Life Cycle)
- Supports TCP/IP Suite since ver. 4.1

Source: www.autosar.org, last visit: 08.11.2015

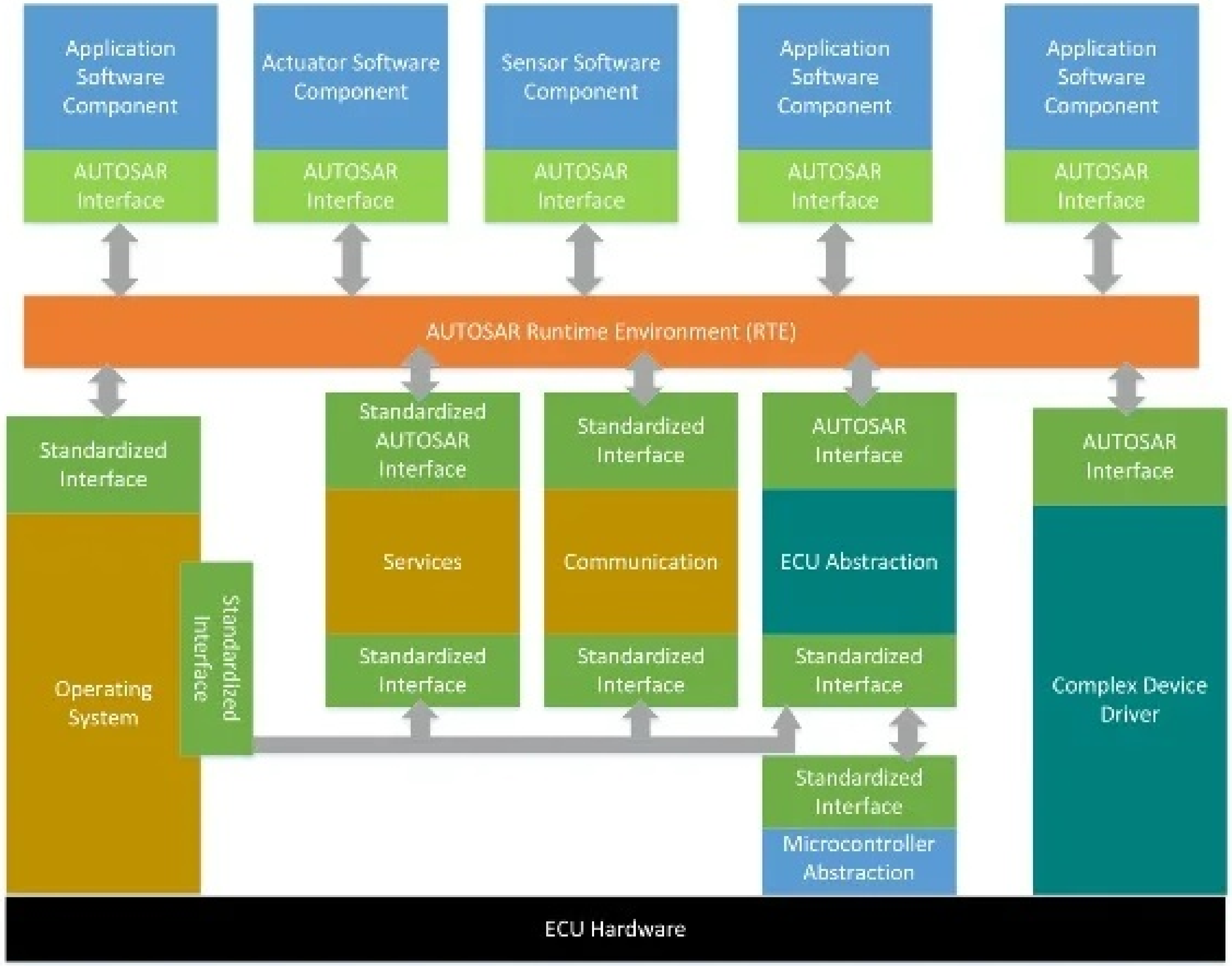
## AUTOSAR [1]

Design Architecture



# AUTOSAR [2]

Example of one AUTOSAR Configuration

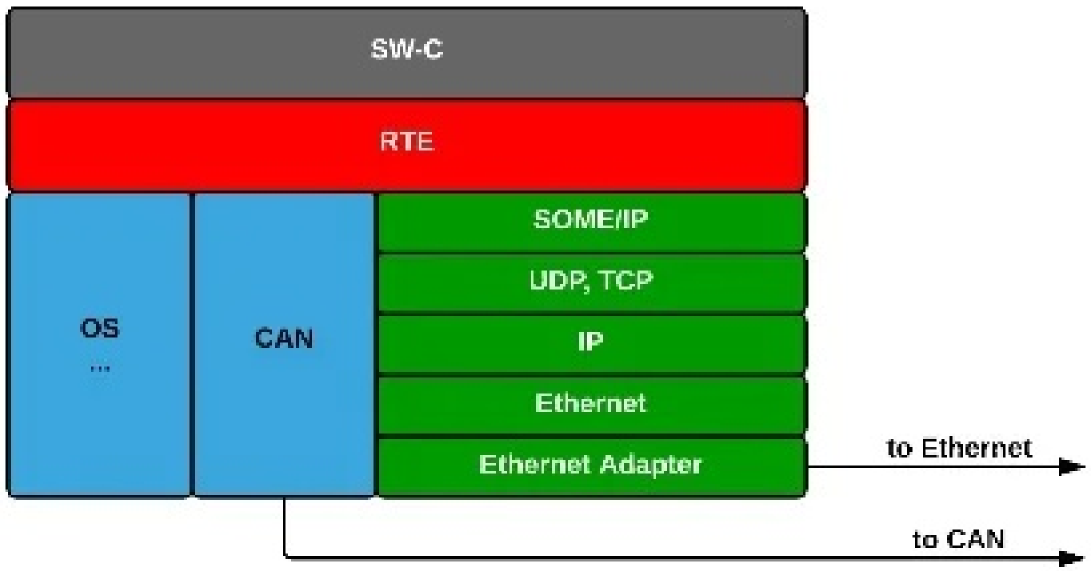


# SOME/IP

# SOME/IP [0]

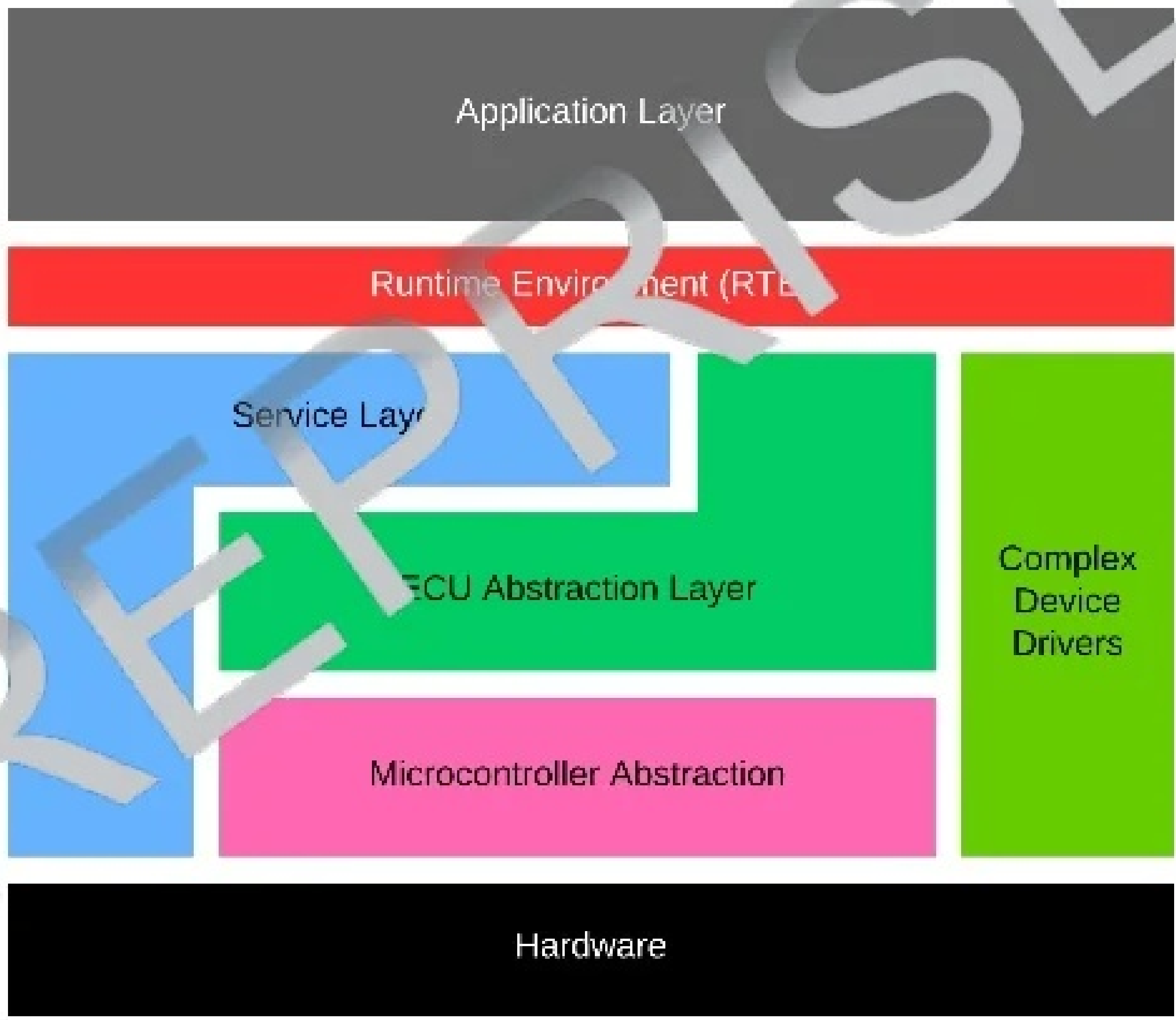
## Scalable Service-Oriented Middleware over IP

- Designed by BMW Group in 2011
- Targeted at the increasing pressure on onboard network of automobiles
- Designed as communication system in:
  - Infotainment
  - Driver Assistance System
- Compatible to AUTOSAR
- Decoupling of SW-Unit from Communication-Unit
- Based on the TCP/IP Protocol Suite



# AUTOSAR [1]

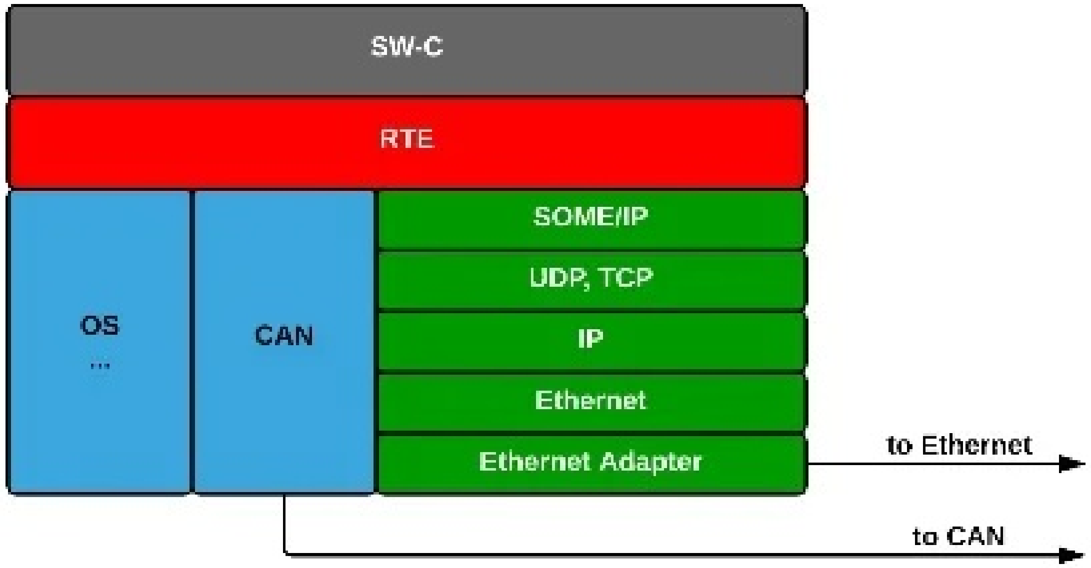
## Design Architecture



## SOME/IP [0]

Scalable Service-Oriented Middleware over IP

- Designed by BMW Group in 2011
- Targeted at the increasing pressure on onboard network of automobiles
- Designed as communication system in:
  - Infotainment
  - Driver Assistance System
- Compatible to AUTOSAR
- Decoupling of SW-Unit from Communication-Unit
- Based on the TCP/IP Protocol Suite

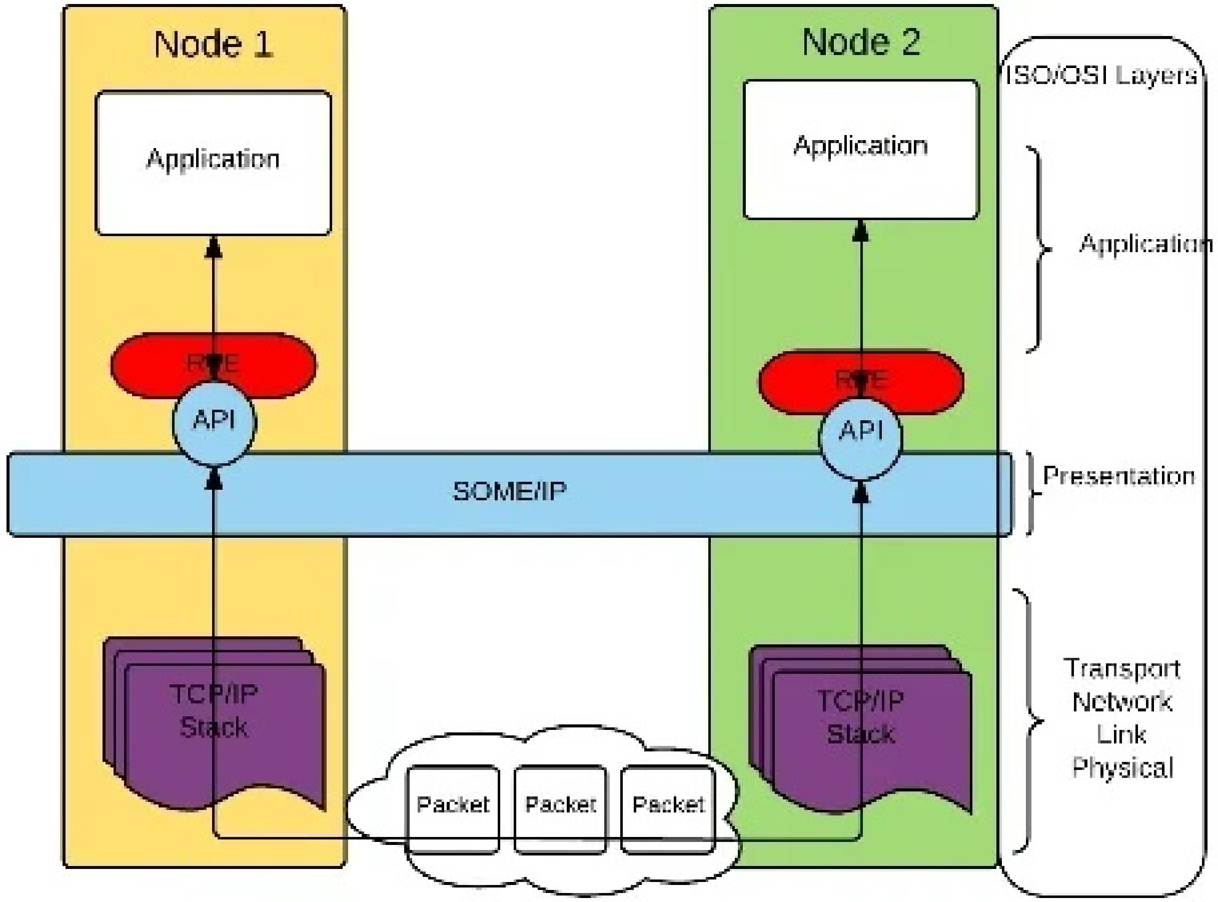


# SOME/IP [1]

## Architecture

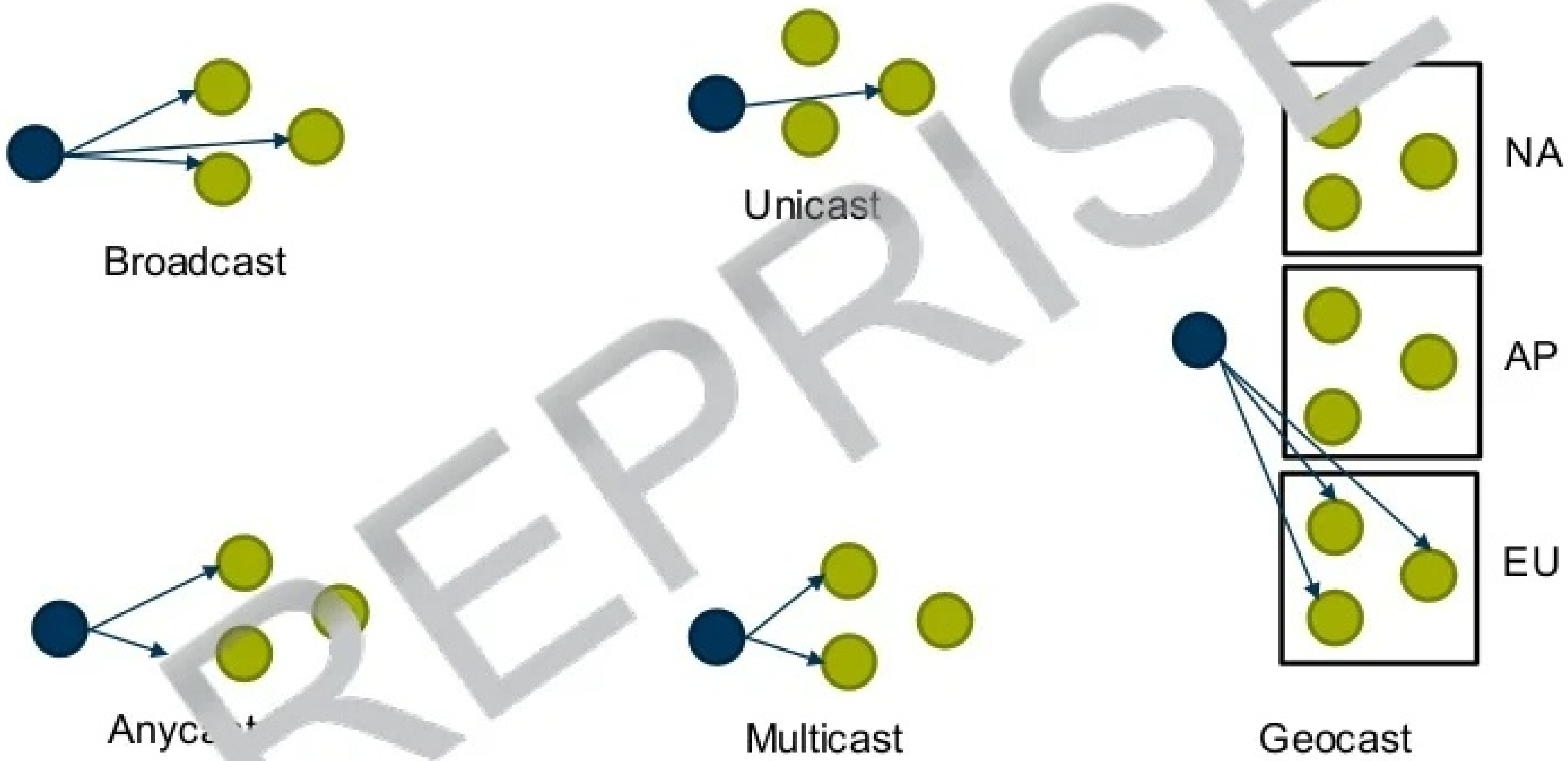
- Physical, Data Link and Network:
  - Full-duplex, Full-switched IEEE802.3 compatible network up to 1Gbps
  - Unicast

  -> Better Link utilization
- Transport Layer:
  - TCP, for reliable communication
  - UDP, for lightweight communication
- Session and Presentation Layer:
  - SOME/IP-SD
  - Communication for Application via API



# Ethernet and TCP/IP Stack [1]

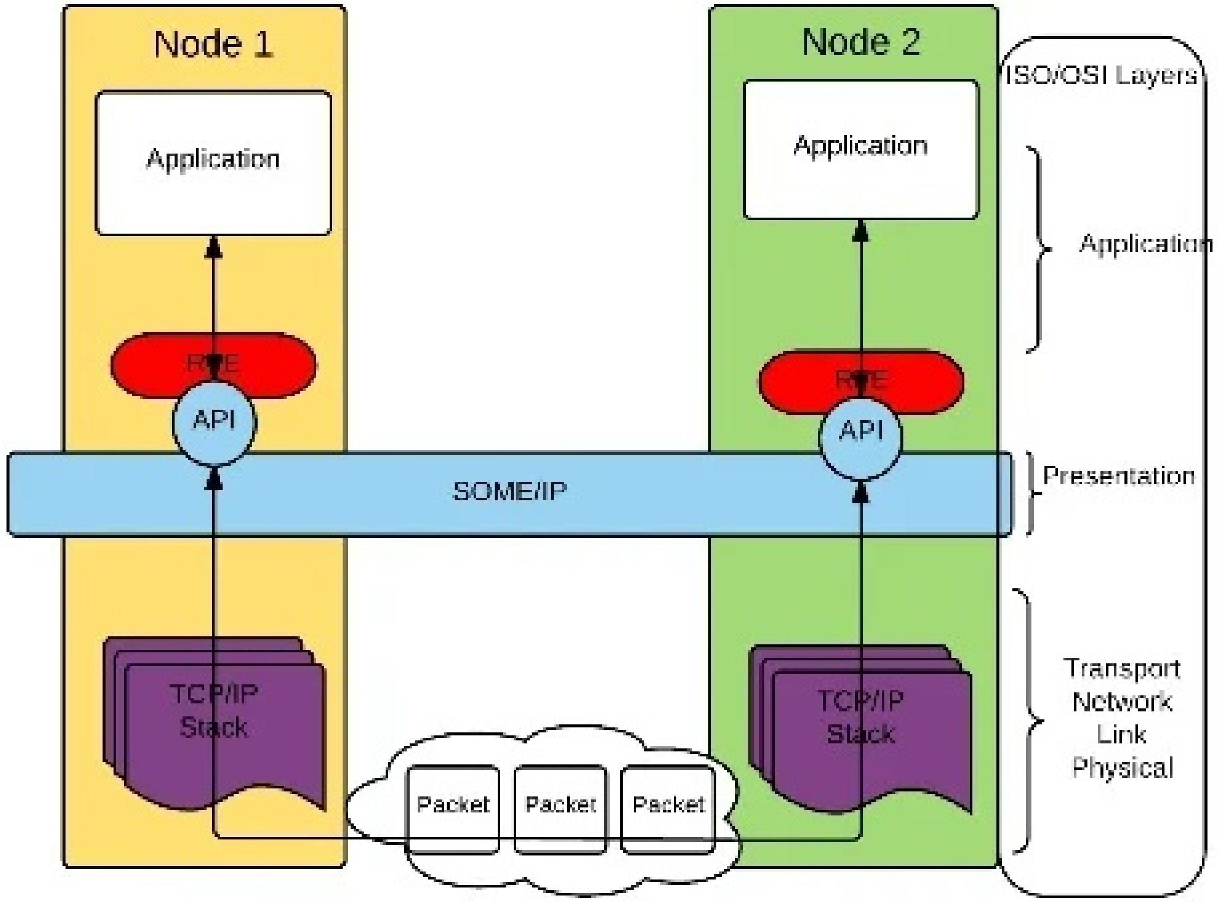
## Basic Routing Schemes



# SOME/IP [1]

## Architecture

- Physical, Data Link and Network:
  - Full-duplex, Full-switched IEEE802.3 compatible network up to 1Gbps
  - Unicast
  
  -> Better Link utilization
- Transport Layer:
  - TCP, for reliable communication
  - UDP, for lightweight communication
- Session and Presentation Layer:
  - SOME/IP-SD
  - Communication for Application via API



# SOME/IP [2]

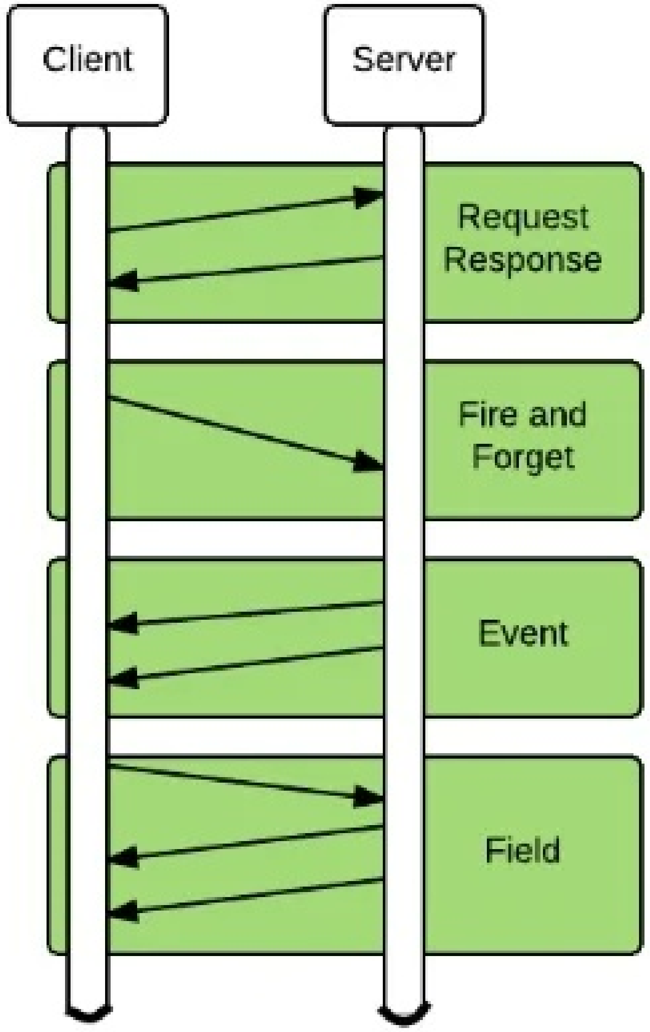
## Basic Data Types and Data Serialization

- Basic Data Types:
  - Unsigned Integer: 8/16/32/64 bit
  - Signed Integer: 8/16/32/64 bit
  - Floating Point Numbers: 32/64 bit
  - Enumerations
  - Booleans
  - Bit fields
  - Structures
  - Unions
  - (Multidimensional-) Arrays
- API with Serialization Functionalities
  - C – Style, “as-is”
  - No data conversion
  - No calculation
  - Large data structures are segmented via TCP or IP Segmentation

# SOME/IP [3]

## Service Orientation

- Client Server Architecture of SW-Cs
- Communication via **Messages**
- Types of services:
  - Request and Response
  - Fire and Forget
  - Event
  - Field
  - Eventgroup



# SOME/IP [4]

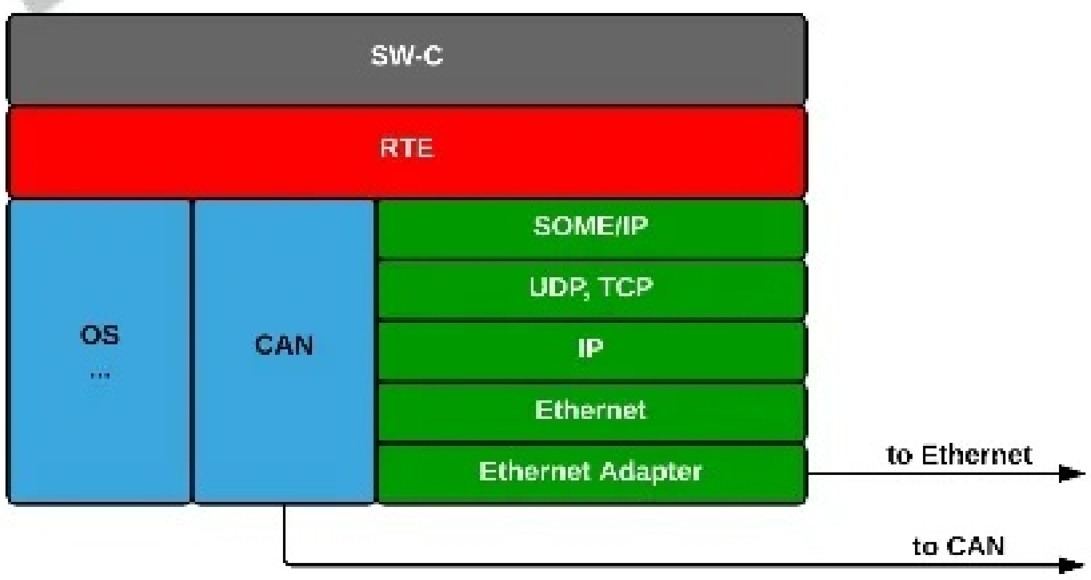
## Message Format

<table>
<tr><td></td><td colspan="32">Offset</td></tr>
<tr><td>Bytes</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td><td>28</td><td>29</td><td>30</td><td>31</td></tr>
<tr><td>0<br>...<br>20</td><td colspan="32">Ethernet Header (0-23 Bytes)</td></tr>
<tr><td>24<br>...<br>40</td><td colspan="32">IP Header (24-43 Bytes)</td></tr>
<tr><td>44<br>...<br>60</td><td colspan="32">TCP Header (44-63 Bytes)</td></tr>
<tr><td>64</td><td colspan="16">Service ID</td><td colspan="16">Method ID</td></tr>
<tr><td>68</td><td colspan="32">Length</td></tr>
<tr><td>72</td><td colspan="16">Client ID</td><td colspan="16">Session ID</td></tr>
<tr><td>76</td><td colspan="8">Protocol Version</td><td colspan="8">Interface Version</td><td colspan="8">Message Type</td><td colspan="8">Return Code</td></tr>
<tr><td>80<br>...</td><td colspan="32">Payload (Parameters, etc.)</td></tr>
</table>

# SOME/IP [0]

Scalable Service-Oriented Middleware over IP

- Designed by BMW Group in 2011
- Targeted at the increasing pressure on onboard network of automobiles
- Designed as communication system in:
  - Infotainment
  - Driver Assistance System
- Compatible to AUTOSAR
- Decoupling of SW-Unit from Communication-Unit
- Based on the TCP/IP Protocol Suite



# SOME/IP [4]

## Message Format

<table>
<tr><td></td><td colspan="32">Offset</td></tr>
<tr><td>Bytes</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td><td>28</td><td>29</td><td>30</td><td>31</td></tr>
<tr><td>0<br>...<br>20</td><td colspan="32">Ethernet Header (0-23 Bytes)</td></tr>
<tr><td>24<br>...<br>40</td><td colspan="32">IP Header (24-43 Bytes)</td></tr>
<tr><td>44<br>...<br>60</td><td colspan="32">TCP Header (44-63 Bytes)</td></tr>
<tr><td>64</td><td colspan="16">Service ID</td><td colspan="16">Method ID</td></tr>
<tr><td>68</td><td colspan="32">Length</td></tr>
<tr><td>72</td><td colspan="16">Client ID</td><td colspan="16">Session ID</td></tr>
<tr><td>76</td><td colspan="8">Protocol Version</td><td colspan="8">Interface Version</td><td colspan="8">Message Type</td><td colspan="8">Return Code</td></tr>
<tr><td>80<br>...</td><td colspan="32">Payload (Parameters, etc.)</td></tr>
</table>

# SOME/IP-SD [0]

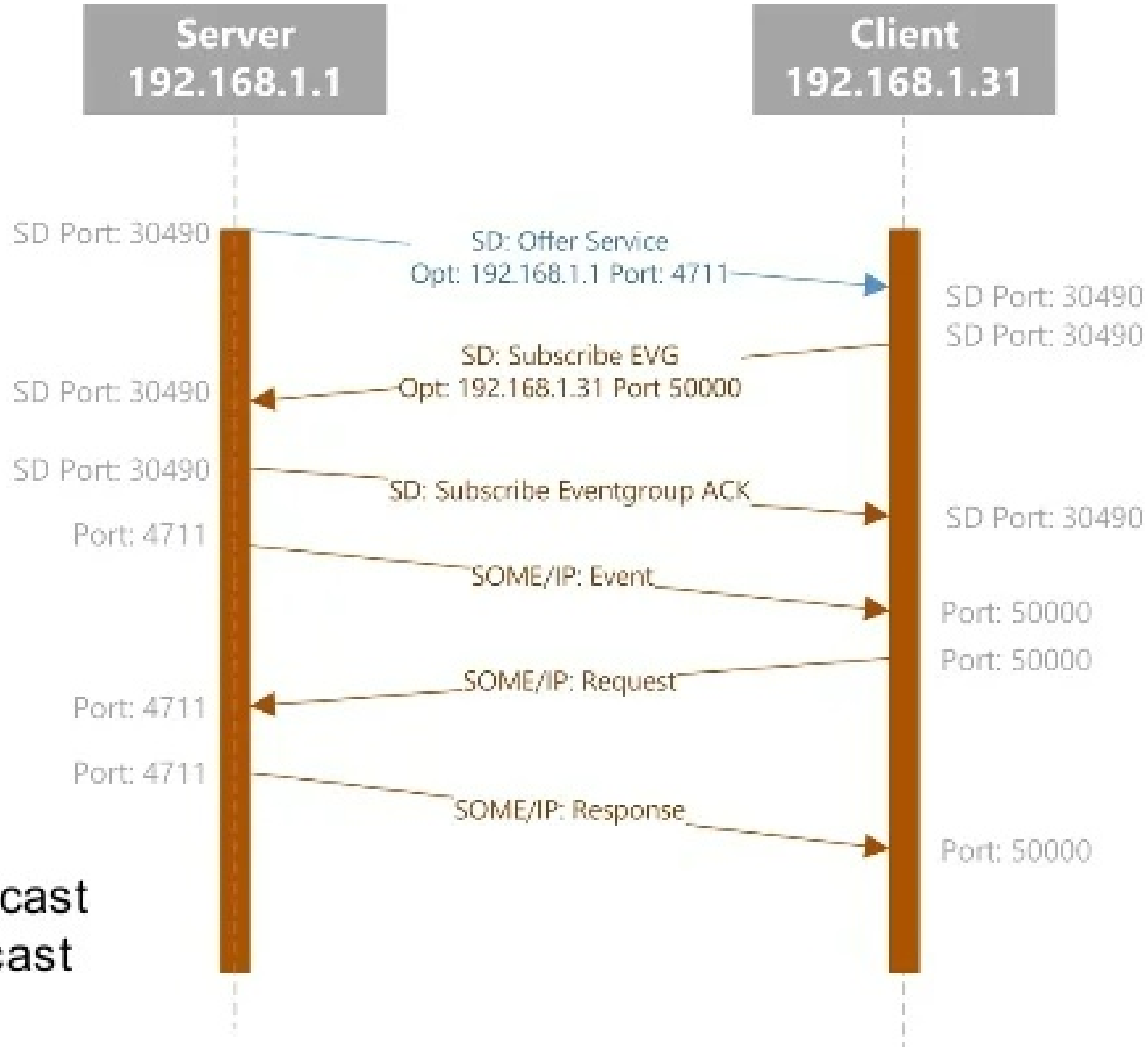
SOME/IP Service Discovery

- Registration between Service and Client
  - IP Address
  - Ports
  - Service ID
  - Event Subscription
  - Disconnect detection
  - active or passive service publishing
- Grant for the flexibility
  - Decouple Server from fixed IP
  - Plug and Play for new Server/Client
- Keep broadcast message as less as possible

# SOME/IP-SD [1]

SOME/IP-SD-Entries

- Find Service (passive Server)
- Offer Service (active Server)
- Stop Offer Service
- Subscribe Eventgroup
- Stop Subscribe Eventgroup
- Subscribe Eventgroup ACK
- Subscribe Eventgroup NAK




Blue: Broadcast
Brown: Unicast

# Evaluation and Future Work

# Evaluation [0]

## Is SOME/IP suitable for automotive Engineering?

Main Advantages of SOME/IP

- Coexistence with existing system
  -> No functional loss
- High Data Rate and Unicast
  -> Increased data transfer amount
- High Service Availability
- Dynamic IP Addressing
  -> Gain in Maintainability and Flexibility

Possible Issues of SOME/IP

- Computational Overhead due to complex architecture
- Increased Storage Requirement (Buffers, etc.)
- Less predictable computation time
- Single Point of Failure (e.g. Switch malfunction)

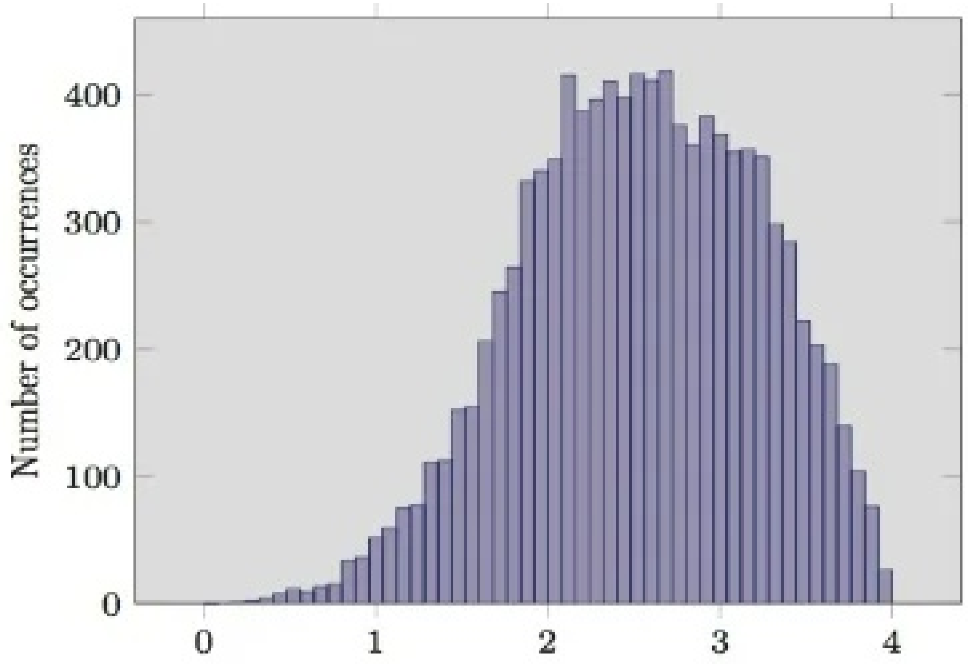
- Good for Infotainment and Driver Assistance System with high requirements on Data rate
- Less suitable for Safety Critical System
- Less suitable for Hard Real Time Systems (e.g. brakes)

## Evaluation [1]

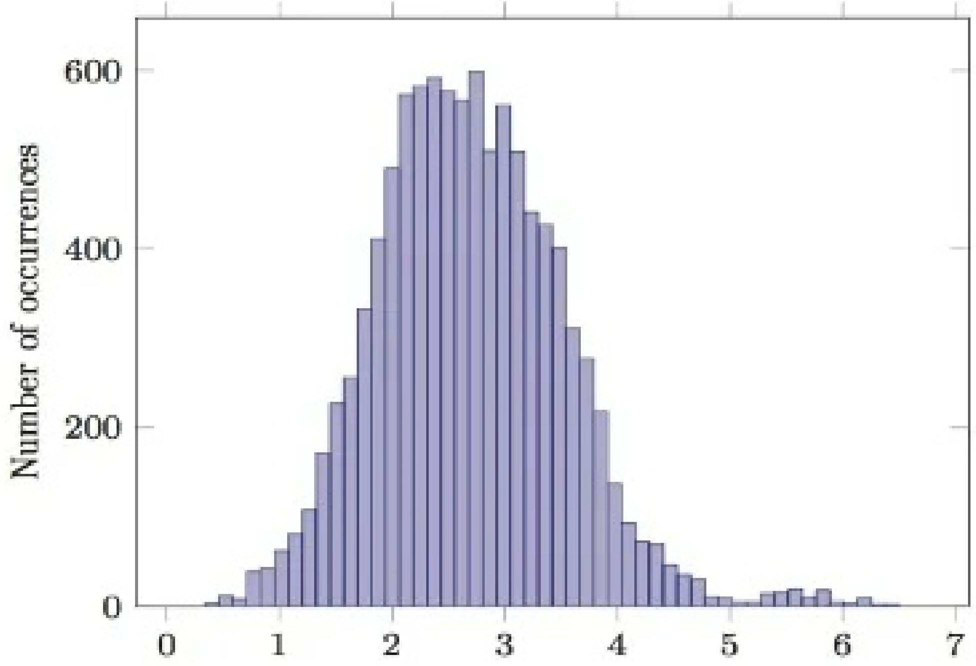
Are we expecting latency due to SOME/IP-SD?

Time, until server and client gets ready:

(1) Server in Offer Mode, Client in Listen Mode




(2) Server in Silent Mode, Client in Discover Mode




Both case:

- average round about 3ms startup time
- Worst case time is in scenario (1) better

Source: Seyler, J. R., Streichert, T., Glaß, M., Navet, N., and Teich, J. Formal analysis of the startup delay of some/ip service discovery. In Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition (2015), EDA Consortium, pp. 49–54.

## Summary and Future Work

- What we know:
  - SOME/IP supports a high data rate
  - SOME/IP has low transportation overhead
  - The initiation time of SOME/IP is notably short

    -> SOME/IP is suitable for driver assistance and infotainment systems

    -> but still to complex for hard real time systems like Motor control or brakes
  - Simple, better Maintainability, and Modular design

- Next Steps:
  - Case Studies
  - Real vehicle data
  - Network latency
  - Fault behavior
  - Real-time analysis

**Thank you.**