Nizar Guidara

SOME/IP

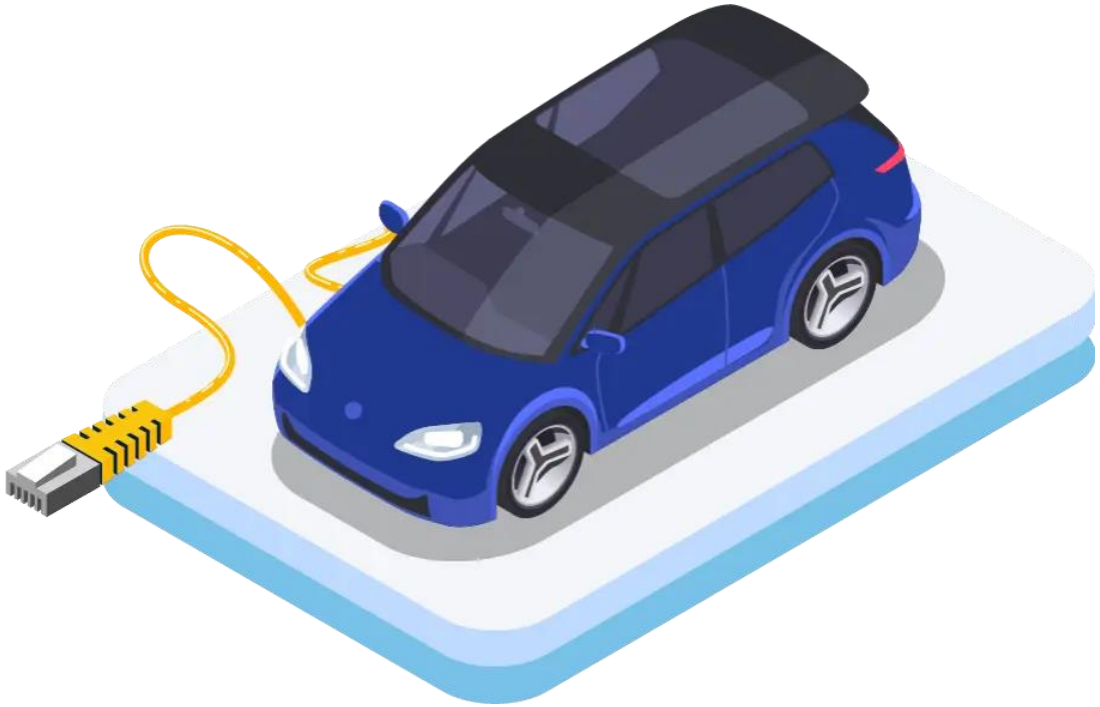
Scalable service-Oriented MiddlewarE over IP

10-01-2024

primatec engineering

TABLE OF CONTENT

INTRODUCTION

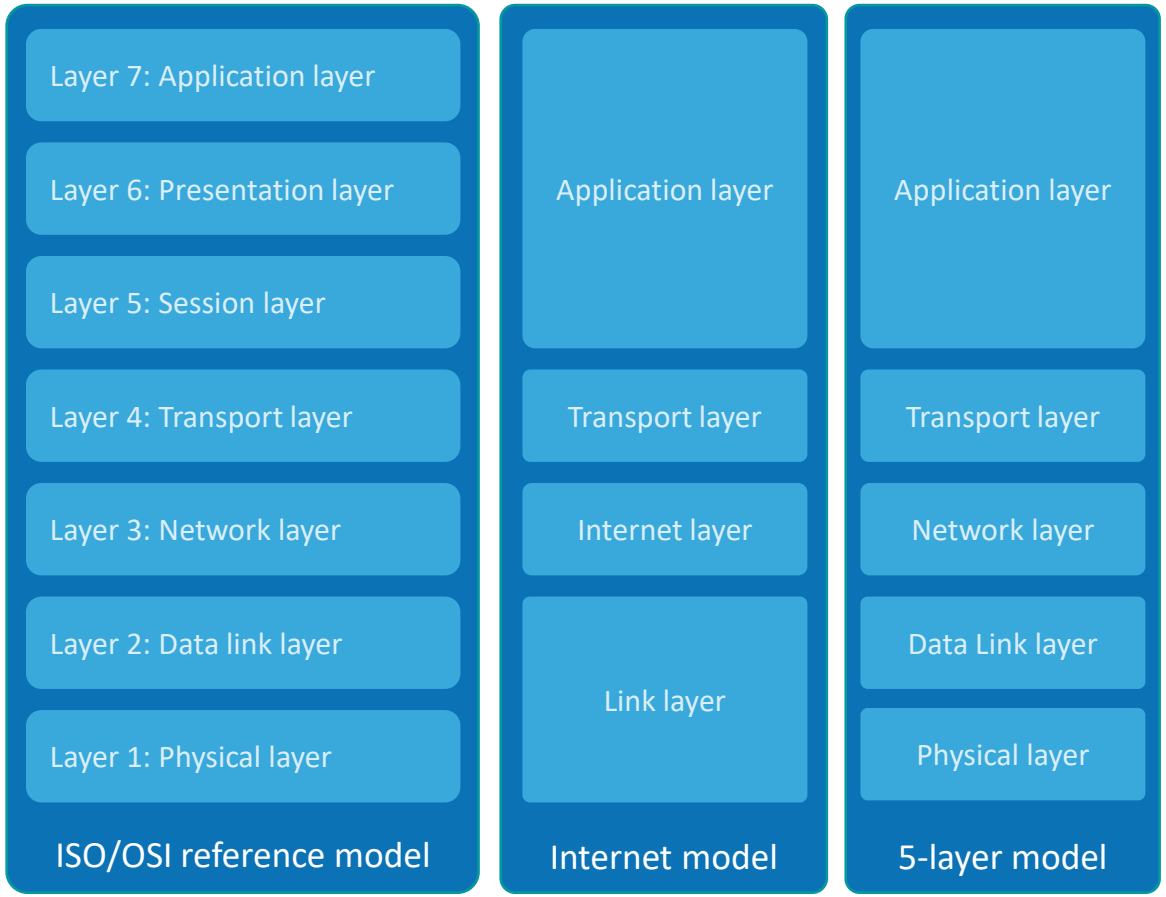


1

1- INTRODUCTION

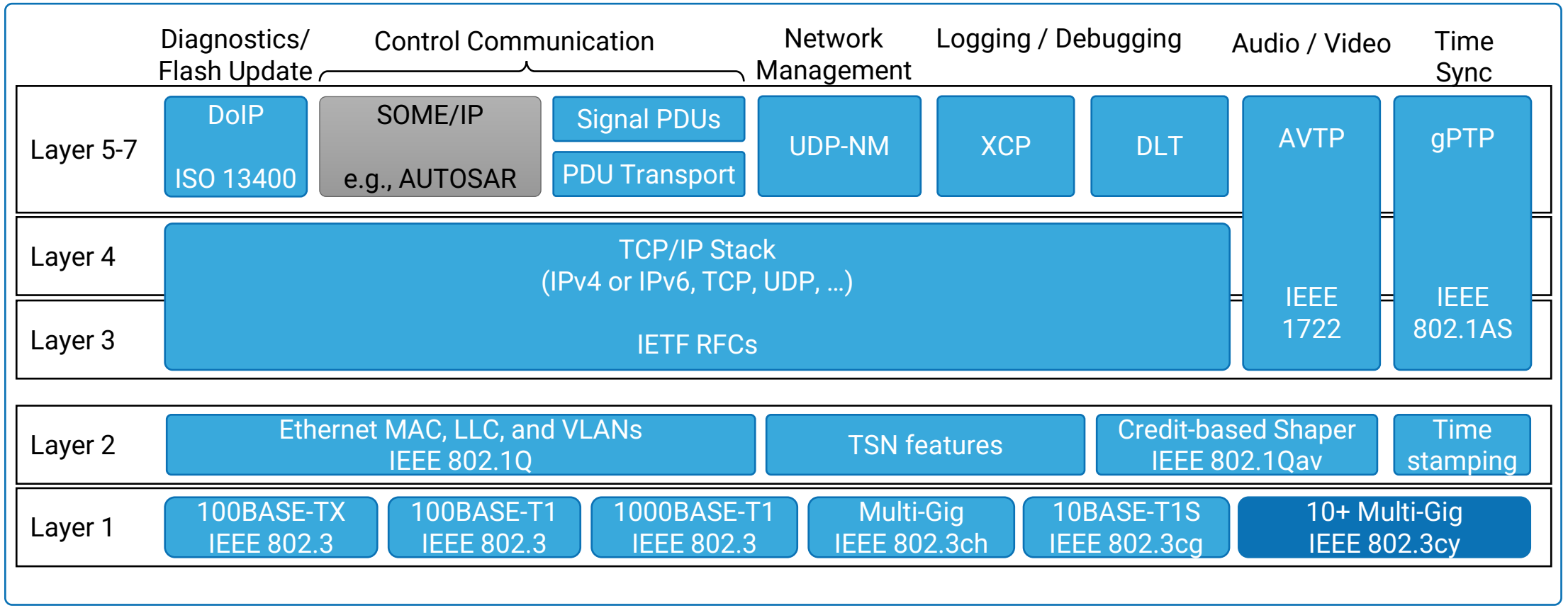
Protocol stack

- Protocol Model:
  - Structures into layers
  - Enables reuse and independent development
- ISO/OSI Basic Reference Model:
  - 7 layers
  - Top 3 layers, almost never differentiated
- Internet Model
  - 4 layers
  - Merges Data Link and Physical layer
- 5-Layer Model → Best compromise
  - Like ISO/OSI but merge the 3 top layers



1- INTRODUCTION

Protocol stack



Standardized | In standardization | Covered in this training

1- INTRODUCTION

Example

Layer

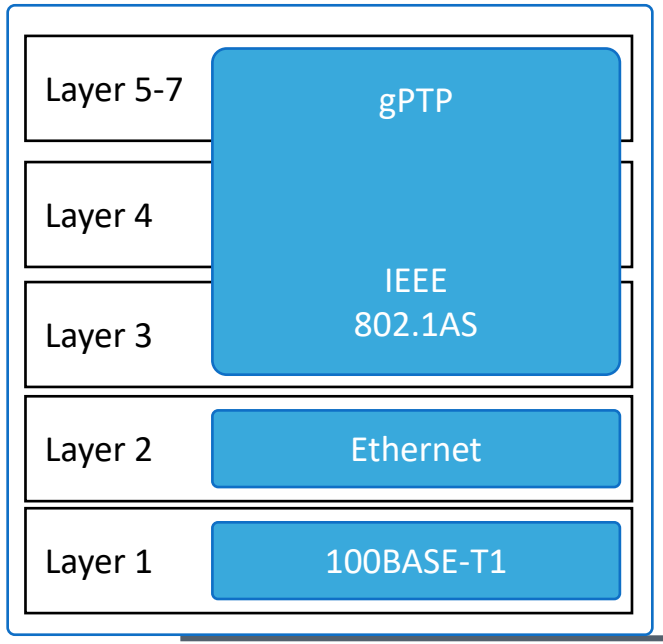
```
> Frame 32: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) on interface enp4s0f0, id 0
> Ethernet II, Src: 38:2a:19:00:43:9b, Dst: 01:80:c2:00:00:0e
> Precision Time Protocol (IEEE1588)
```

Metadata instead of layer 1

Layer 2

Layer 3 - 7

Layer



- Layer 1 is not recorded
  - Which Ethernet variant (e.g., 100BASE-T1) could be at best reconstructed thanks to the metadata
  - Recorded from layer 2 onwards
- Layer 2: Here “Ethernet II“, i.e., standard Ethernet
- Layer 3-7: Here gPTP, labeled as PTP
  - More details later

primatec engineering

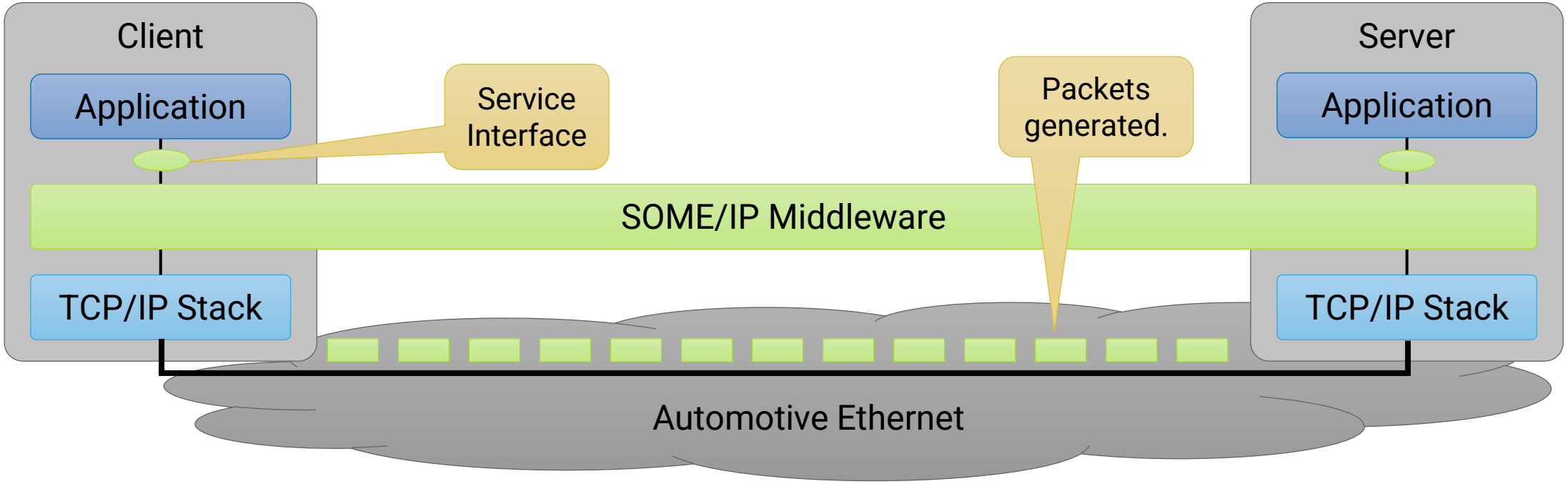
2

2- SOME/IP

- Overview
- Service interfaces
- Frame header
- Serialization

2- SOME/IP

Overview

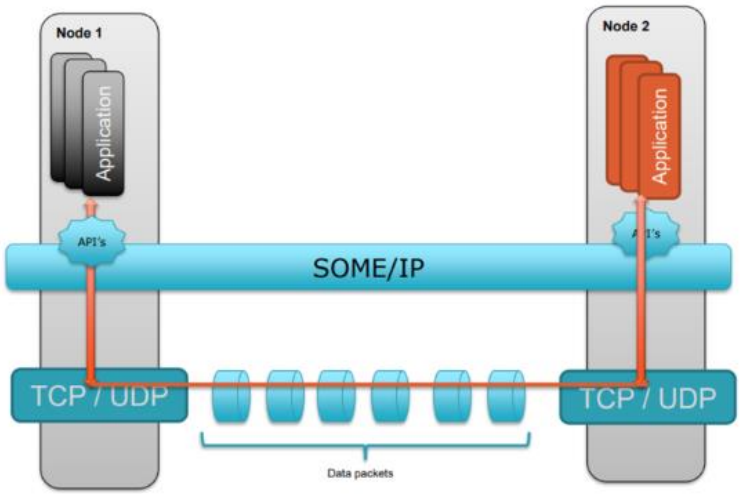


- Applications on two ECUs want to communicate using Ethernet and TCP/IP.
- The SOME/IP middleware translates local APIs (Service Interfaces) to packets.
- Applications do not need to worry about IPs, port numbers, and packet layouts.
- The Server “offers” a Service Instance of a Service Interface.

2- SOME/IP

Overview

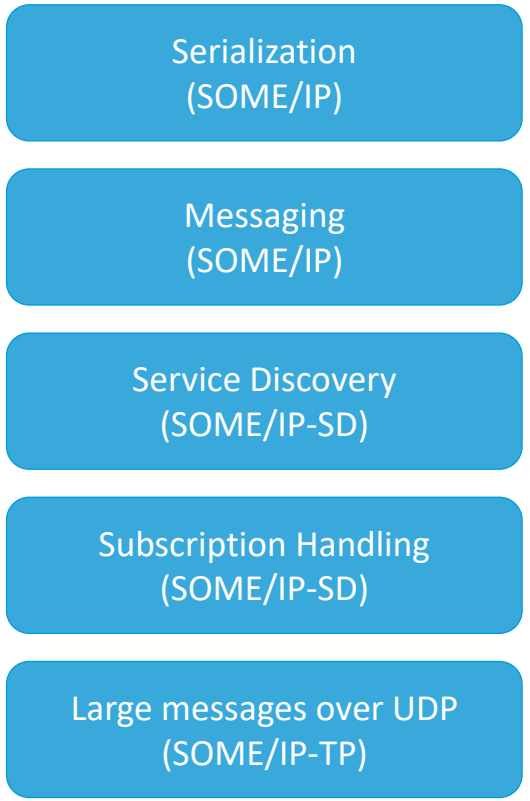
- Scalable service-Oriented MiddlewarE over IP (SOME/IP).
- Development started in 2011. First SOP in 2014 (model year 2015).
- AUTOSAR Classic starting in 4.0. For best results use 4.3 or newer.
- AUTOSAR Adaptive supported.
- Most used solution for Automotive Ethernet.
- Support by many commercial and open-source tools (e.g., Wireshark).



2- SOME/IP

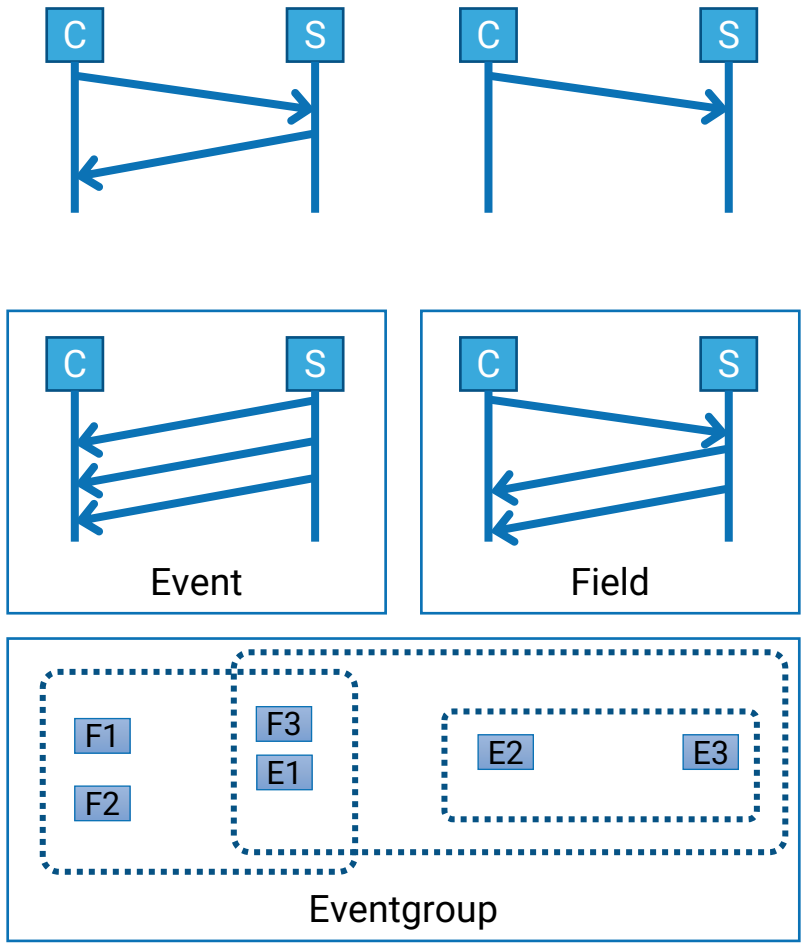
Overview

- Scalable service-oriented MiddlewarE over IP (SOME/IP).
- Highly efficient serialization.
- Different messaging paradigms, like RPC.
- Automatic configuration of TCP/IP stack and controlled “dynamic” configuration with Service Discovery.
- Supports highly scalable and efficient unicast communication using Subscription Handling.
- Adds support for large messages over UDP with more than 1500 Bytes using SOME/IP-TP.



2- SOME/IP

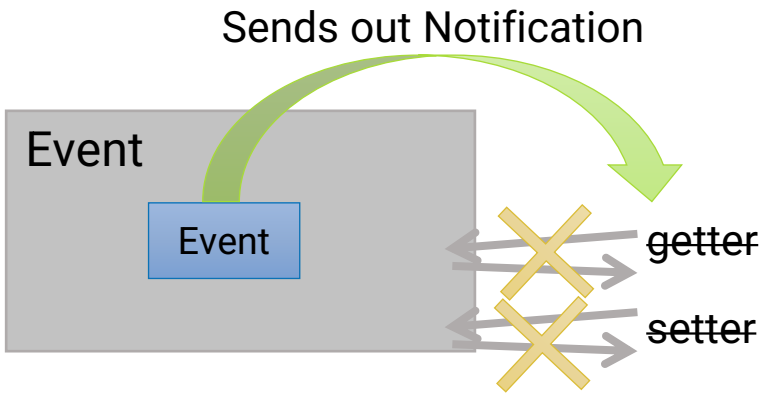
Service Interfaces

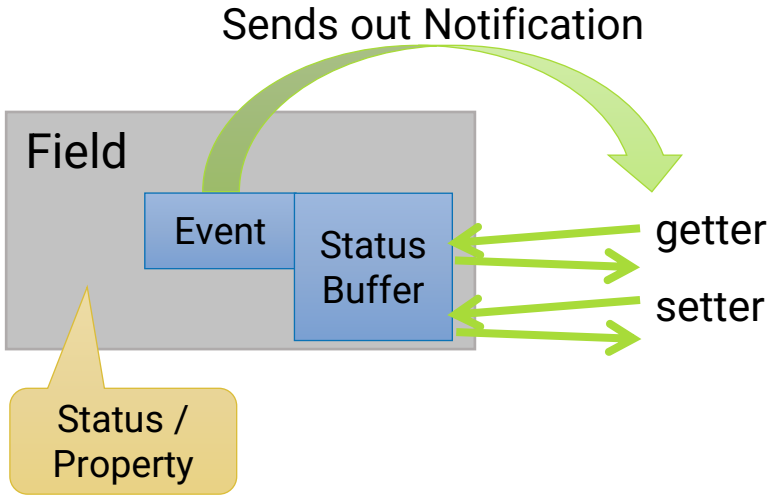


- Service Interfaces define the Service and may contain:
  - Request/Response Methods.
  - Fire & Forget Methods.
  - Events.
  - Fields.
  - Eventgroups (grouping of Events and Fields), required for SOME/IP-SD.

2- SOME/IP

Service Interfaces

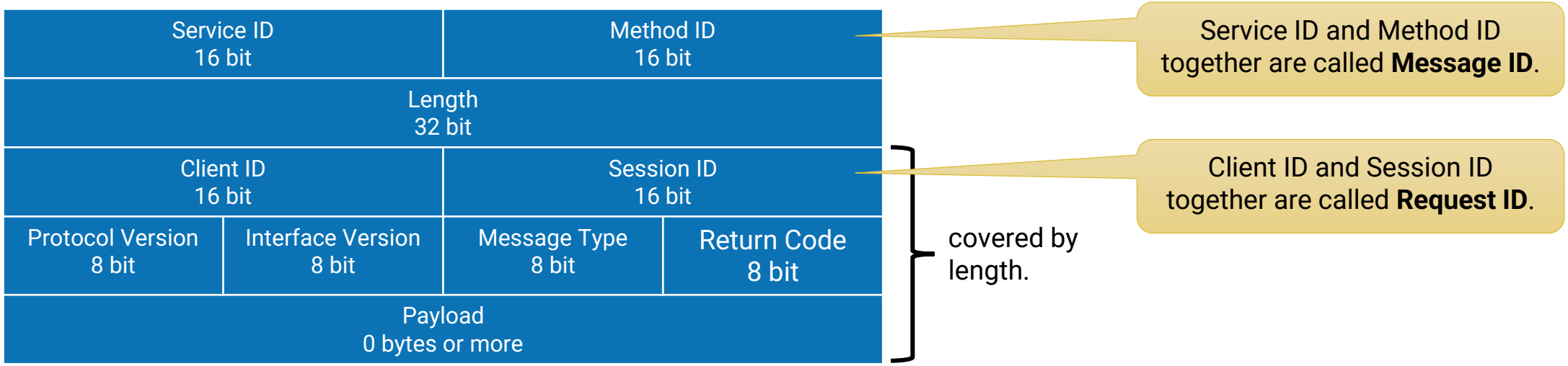




- Events:
  - Send out, when something happens.
  - Bad service design: Creating methods for reading or writing this value (getter, setters).
  - Do not send Initial Events on subscription.
- Fields:
  - Represent a Status or Property.
  - Initial Events on Subscription.
  - Notifier to send out event (optional).
  - Getter to get current value (optional).
  - Setter to set current value (optional).

2- SOME/IP

Frame header



- Service ID uniquely identifies the service and the Method ID, the method or event.
- Length covers all bytes after itself. The smallest valid value is 8.
- Client ID and Session ID are discussed later.
- Protocol Version is the SOME/IP version (0x01).
- Interface Version is the Major version of the Service Interface.
- Message Type and Return Code are discussed later.

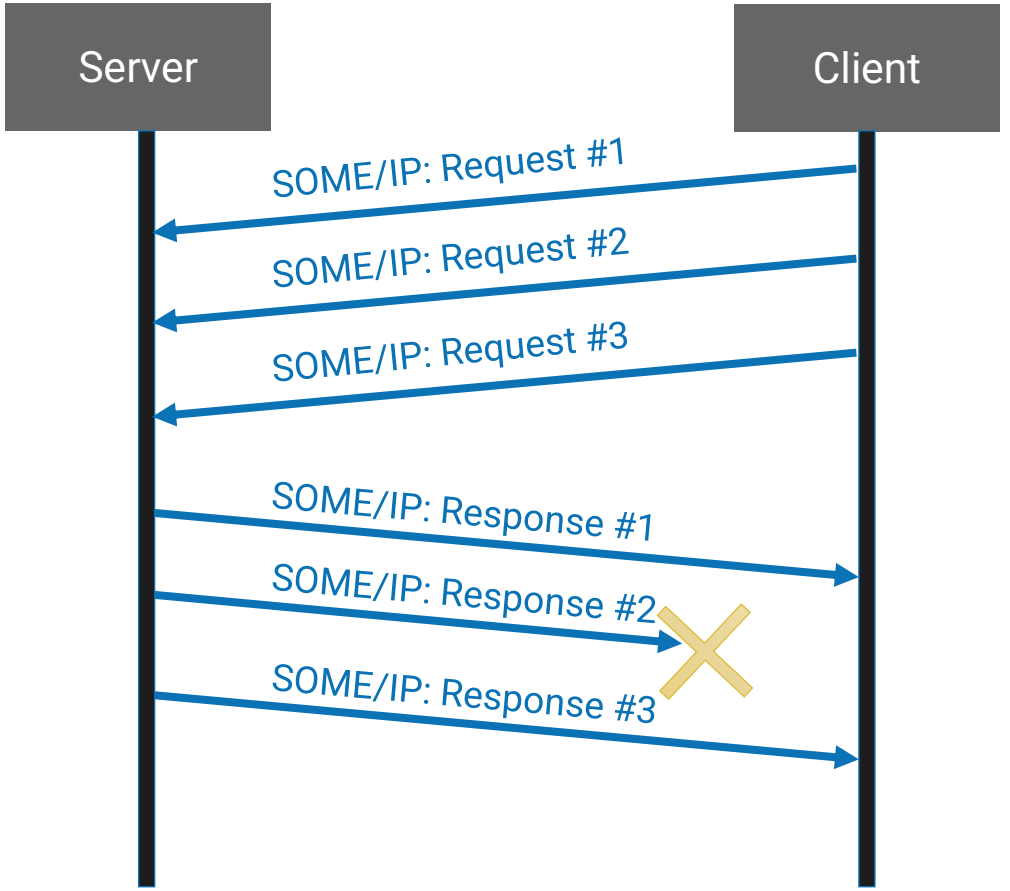
2- SOME/IP

Frame header – Request ID

- Session ID numbers the requests to match response to request.
- Needed for parallel calls and loss.

- Client ID can be used by the client to document, which internal component sent this call.
- Allows for optimizations when identifying target for response.

- The Server copies Client ID and Session ID for request to response.



2- SOME/IP

Frame header – Message Types

- Message Types (for non-TP messages):

ID	Name	Description
0x00	REQUEST	A request of a request/response method.
0x01	REQUEST_NO_RETURN	A fire&forget request.
0x02	NOTIFICATION	An event or notification of a field.
0x40	REQUEST_ACK	Ack for REQUEST (Reserved)
0x41	REQUEST_NO_RETURN ACK	Ack for REQUEST_NO_RETURN (Reserved)
0x42	NOTIFICATION_ACK	Ack for NOTIFICATION (Reserved)
0x80	RESPONSE	A response of a request/response method.
0x81	ERROR	An exception/error response for a request/response method.
0xC0	RESPONSE_ACK	Ack for RESPONSE (Reserved)
0xC1	ERROR_ACK	Ack for ERROR (Reserved)

SOME/IP-TP has influence on these (covered later).

ACK Message Types not used.

2- SOME/IP

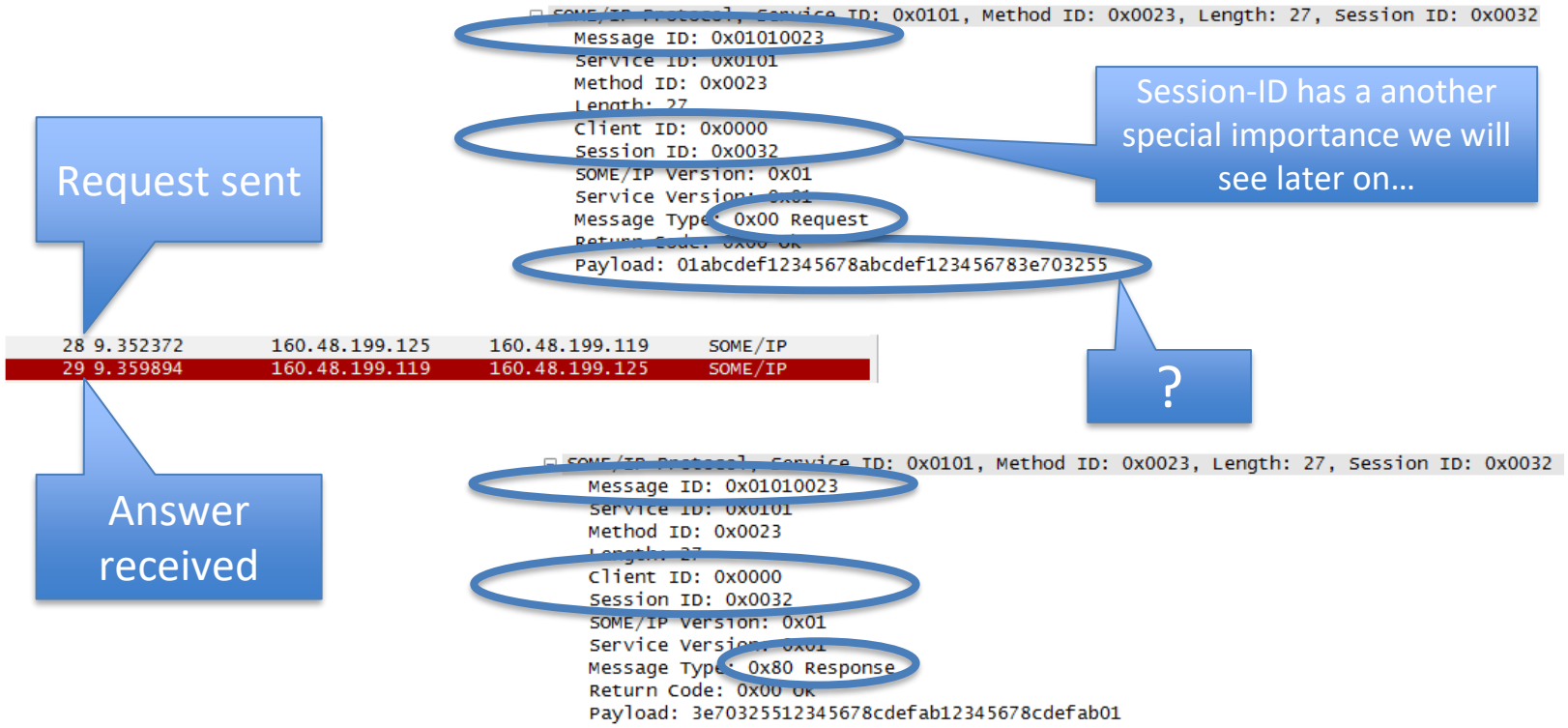
Frame header – Return Codes

ID	Name	Description
0x00	E_OK	No error occurred.
0x01	E_NOT_OK	An unspecified error occurred.
0x02	E_UNKNOWN_SERVICE	The Service ID is unknown. (*)
0x03	E_UNKNOWN_METHOD	The requested Method ID is unknown. (*)
0x04	E_NOT_READY	Application not ready. (*)
0x05	E_NOT_REACHABLE	Peer cannot be reached. Internal error.
0x06	E_TIMEOUT	A timeout occurred. Internal error.
0x07	E_WRONG_PROTOCOL	Version of SOME/IP not supported.
0x08	E_WRONG_INTERFACE	Major version of interface not supported.
0x09	E_MALFORMED_MESSAGE	Deserialization error.
0x0a	E_WRONG_MESSAGE_TYPE	Unexpected message type received.
0x0b – 0x1f	RESERVED	Reserved for generic SOME/IP errors.
0x20 – 0x5f	RESERVED	Reserved for service specific errors.

- (*) not supported in AUTOSAR classic due to architecture limitation.
- Most generic SOME/IP errors are not really needed in production software.

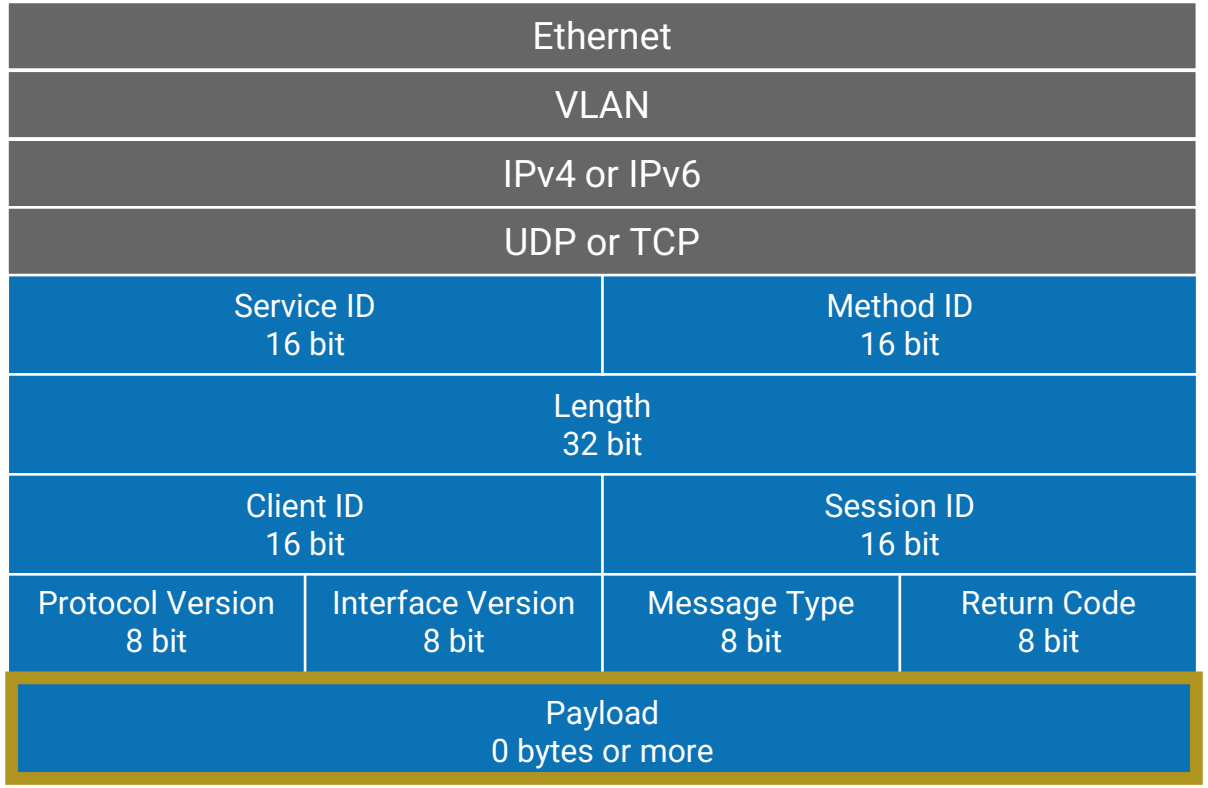
2- SOME/IP

Frame header – Example Request/Response



2- SOME/IP

Serialization



Maximum Payload sizes of SOME/IP:

- UDP: 1400-1456 bytes (depending on protocols used).
- TCP: up to 4GB (AUTOSAR 4.2).
- UDP: with SOME/IP-TP up to 4GB (AUTOSAR 4.3).

Volkswagen Group specific:

- Payload (UDP) up to 1452 Bytes.

2- SOME/IP

Serialization

- Basic datatypes – see table.
- Bitfields – pack multiple Booleans into an uint8/16/32.
- Enums – named values of uints.
- Strings – dynamic or static length string of ascii, utf-8, or utf-16 chars.
- Structs – combination of multiple parameters.
- Arrays – repetition of parameters.
- Unions – dynamic container to select datatype at runtime.
- All length and type fields in Big Endian.

Type	Description	Size [bit]	Remark
boolean	True/False value	8	0: False, 1: True.
uint8	unsigned Integer	8	
uint16	unsigned Integer	16	
uint32	unsigned Integer	32	
uint64	unsigned Integer	64	
sint8	signed Integer	8	
sint16	signed Integer	16	
sint32	signed Integer	32	
sint64	signed Integer	64	
float32	floating point number	32	IEEE 754
float64	floating point number	64	IEEE 754

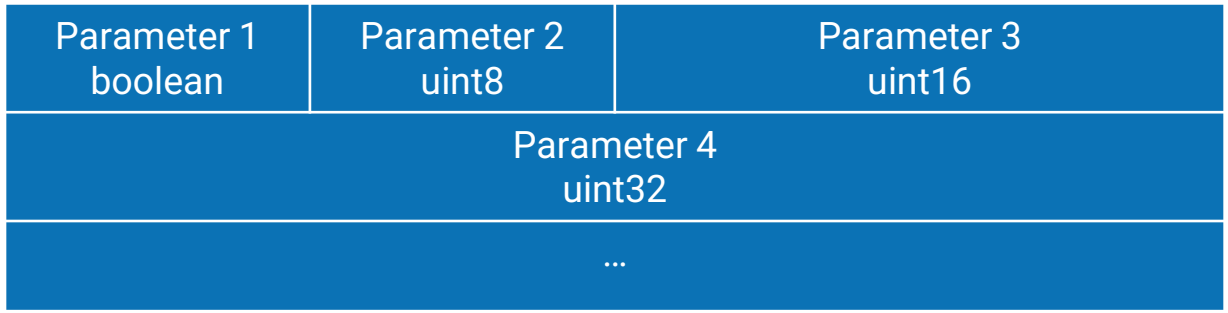
Volkswagen Group specific:

- Only 8-bit Bitfields supported.
- Strings only with utf-8 supported.
- Strings may not have zero-terminator or BOM.
- Unions are not supported.

2- SOME/IP

Serialization (Regular)

```
SOME/IP Protocol, Service ID: 0x0101, Method ID: 0x0023, Length: 27, Session ID: 0x0032
    Message ID: 0x01010023
    Service ID: 0x0101
    Method ID: 0x0023
    Length: 27
    Client ID: 0x0000
    Session ID: 0x0032
    SOME/IP Version: 0x01
    Service Version: 0x01
    Message Type: 0x00 Request
    Return Code: 0x00 ok
    Payload: 01abcdef12345678abcdef123456783e703255
```

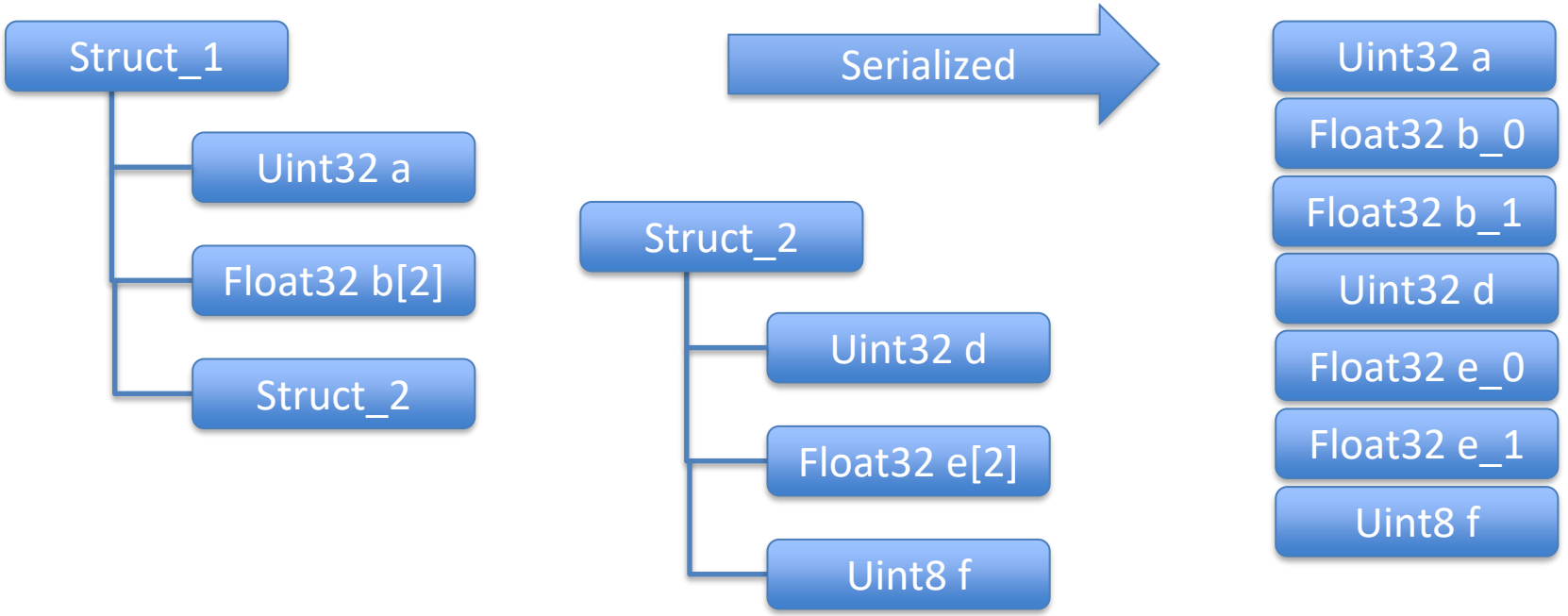


- Regular SOME/IP serialization writes parameters close to in-memory representation into packets.
- This is very fast and efficient. Benefit for high-speed communications.

2- SOME/IP

Serialization – Structs (Regular)

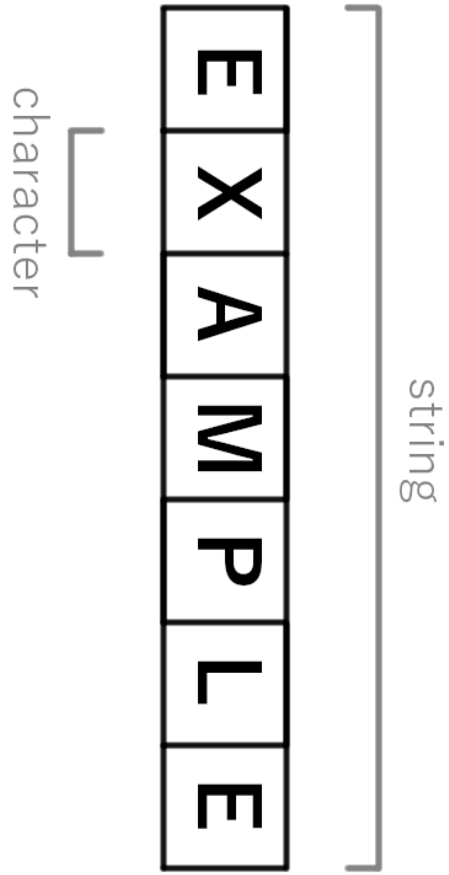
- Basic datatypes are sequentially ordered in the payload as they would in memory.
- A length field of 8, 16 or 32 bits can be added in front of the struct.



2- SOME/IP

Serialization - Strings (fixed length) (Regular)

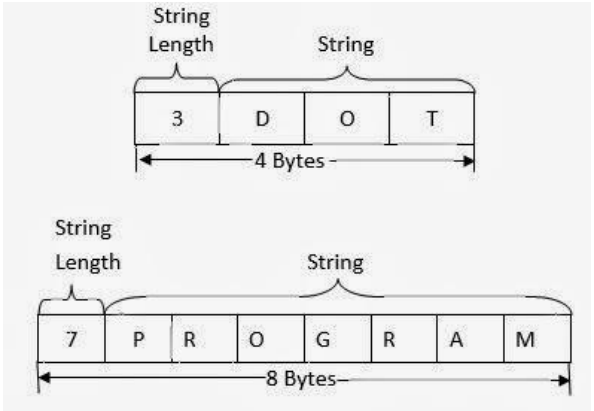
- Encoded using Unicode and terminated with a “\0” character.
- Length, including termination char, is specified by the interface definition.
- The Unicode encodings include UTF-8, UTF-16BE and UTF-16LE.
  - UTF-16 use two termination chars and must not have odd length, if not, the last byte (not termination) is ignored.
- All strings start with a Byte Order Mark (BOM).
  - It is validated by the Interface Specification.



2- SOME/IP

Serialization - Strings (dynamic length) (Regular)

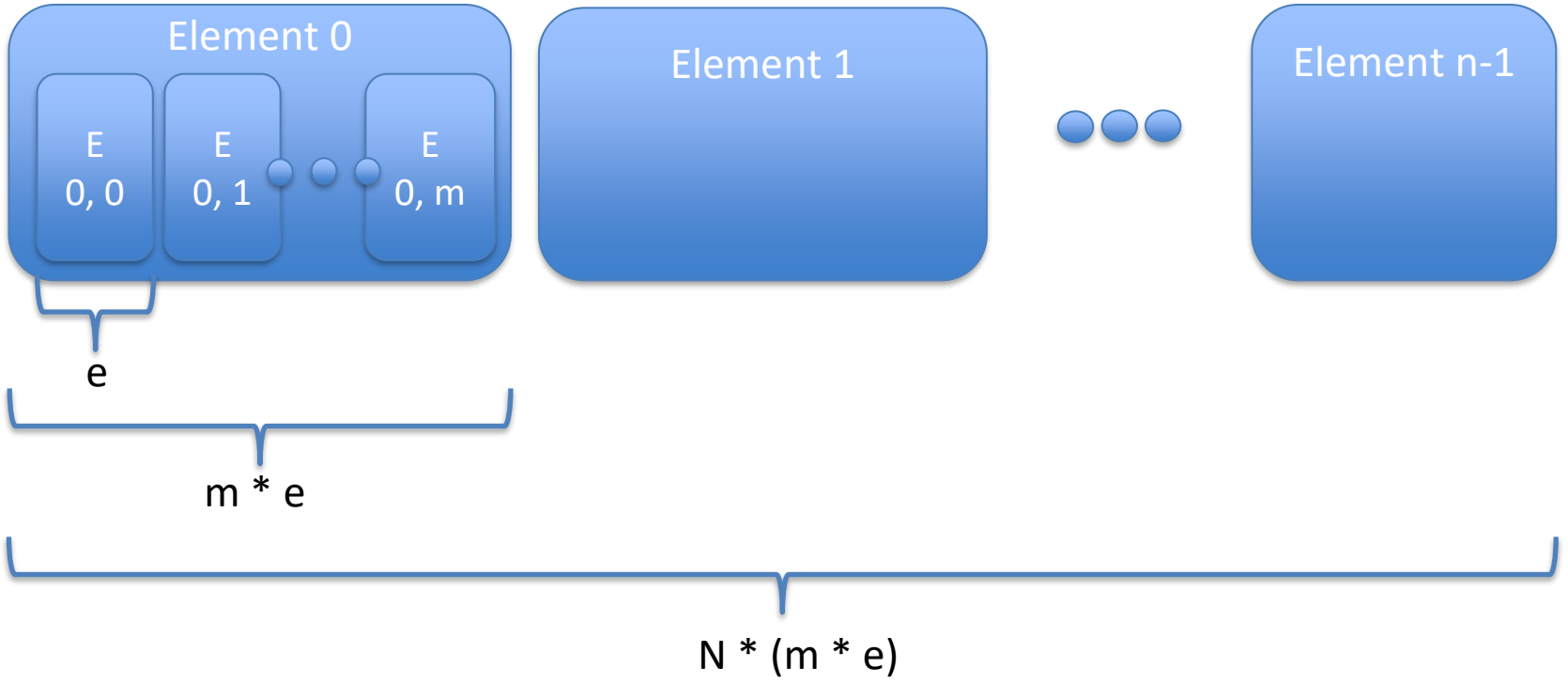
- They always start with a length field measured in bytes.
  - The length field is 8, 16 or 32 bit long (32 if not specified otherwise by the Interface).
  - The length field itself is not part of the value it holds.
  - A fixed string can be seen as a dynamic string with length field 0.
- If the next data element in the payload has a defined alignment, the string is extended with an “\0” termination char.



2- SOME/IP

Serialization - Multidimensional arrays (Array of Arrays) (Regular)

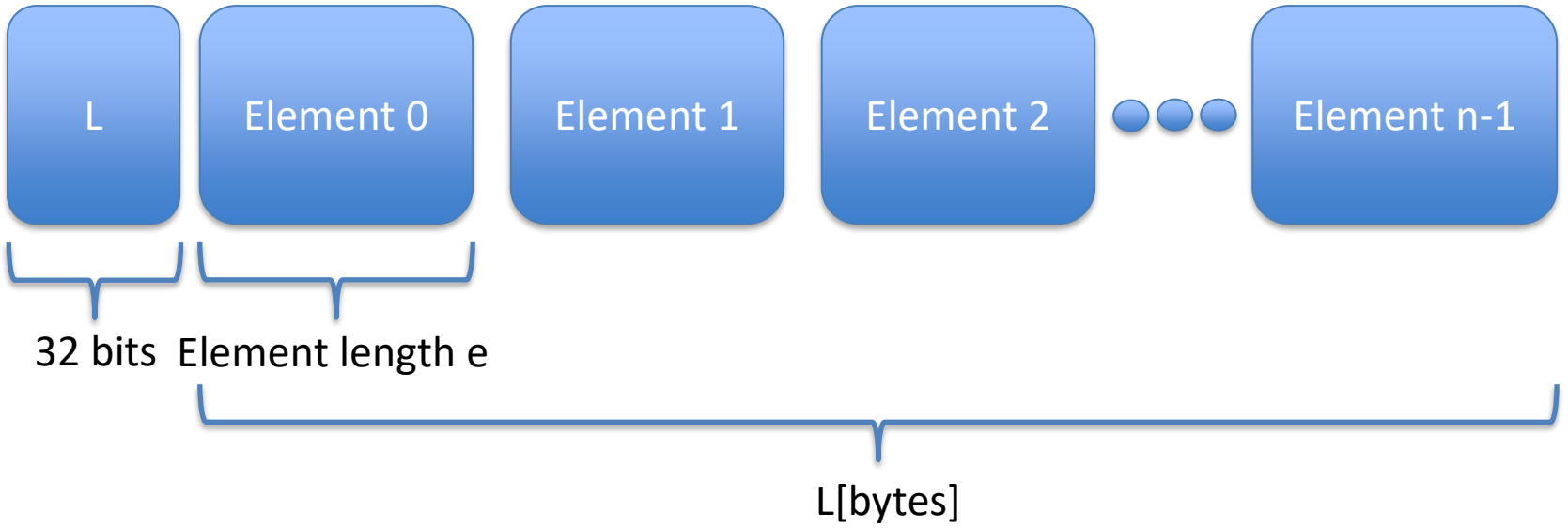
- Arrays can also follow a more C++ like style.



2- SOME/IP

Serialization - Dynamic length arrays (Regular)

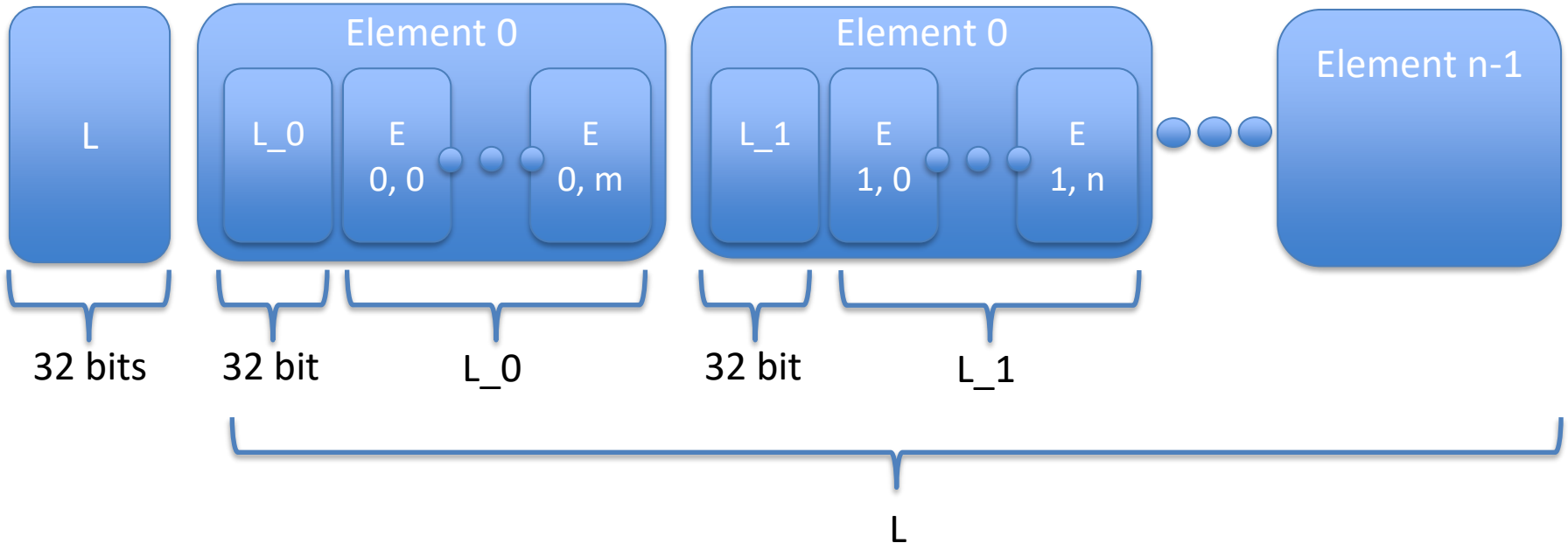
- They extend fixed length arrays by adding a length field with length 0, 8, 16 or 32 bits.
  - If the length field has length 0, it is considered a fixed length array.



2- SOME/IP

Serialization - Multidimensional arrays (dynamic length) (Regular)

- The number of elements cannot be calculated as easy as with static length elements
  - Elements must be parsed sequentially.
- In multidimensional arrays multiple length fields are needed.



2- SOME/IP

Serialization - Enumerations and Unions (Regular)

- **Enumeration**: The Interface definition defines a series of unsigned integers to enumerate possible values.
  - Usually one value of the enumeration is transported and the peer interprets it's significance based on the Interface definition.

```
enum Level {
  LOW = 25,
  MEDIUM = 50,
  HIGH = 75
};
```

- **Union (or Variant)**: Can contain different types of elements

Length field [32 bit]
Type field [32 bit]
Data including padding [sizeof(padding) = length - sizeof(data)]

Table 4.8: Default Serialization of Unions

Length = 4 Bytes			
Type = 1			
uint8	Padding 0x00	Padding 0x00	Padding 0x00

Table 4.9: Example: uint8

Length = 4 Bytes		
Type = 2		
uint16	Padding 0x00	Padding 0x00

Table 4.10: Example: uint16

2- SOME/IP

Serialization – TLV (VW/CARIAD)

- VW Group defined an “extension” to SOME/IP serialization.
- The aim is to add more flexibility and compatibility to serialization:
  - Allows to remove parameters later.
  - Allows to reorder parameters.
  - Allows to add parameter at arbitrary position.

ADD PARAMETER

- Cost for this:
  - Trading in efficiency and high-performance of serialization for flexibility.
  - Forward compatibility more expensive or not possible anymore.

2- SOME/IP

Serialization – TLV (VW/CARIAD)

- “TLV” adds a so-called Tag.
  - Reserved bit (R) = 0.
  - WireType describes the size of data.
    - 0 - 3: static length
    - 4: dynamic length with preconfigured length field size.
    - 5 - 7: dynamic length with encoded length field size.
  - ID identifies the parameter.
  - But: parameter type is not part of Tag.
- Applicable to parameters and struct members.

WireType	Data
0	8bit data
1	16bit data
2	32bit data
3	64bit data
4	length, preconfigured
5	length, 8bit
6	length, 16bit
7	length, 32bit

R	WireType 3 bit	ID 12 bit	Length (opt) 0, 8, 16, 32bit	Data

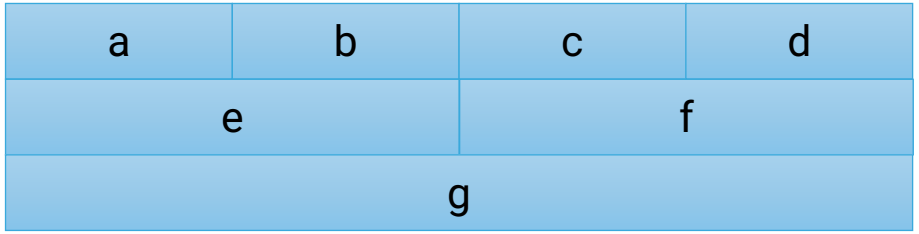
0	“1”	0x123	uint16

2- SOME/IP

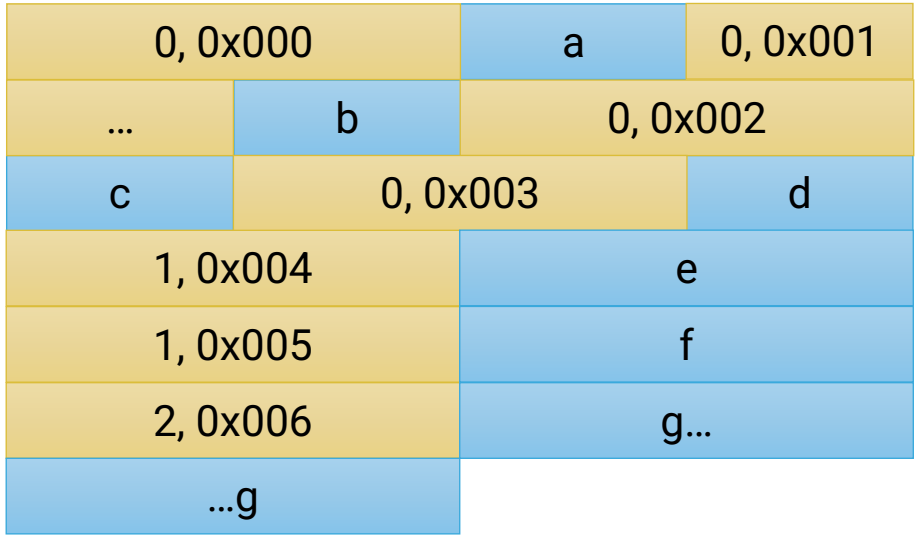
Serialization – TLV (VW/CARIAD)

- Example payload: 4x uint8, 2x uint16, 1x uint32.

SOME/IP



SOME/IP-TLV



- SOME/IP: 12 bytes → ”TLV”: 26 bytes.
- Massive overhead for short datatypes (typical for control loops).
- Overhead for receiver even worse because order unknown.

primatec engineering

3

SOME/IP-SD

- Overview
- Header
- Entries
- Endpoint options
- State machines and timings
- Reboot detection
- Session handling

3- SOME/IP-SD

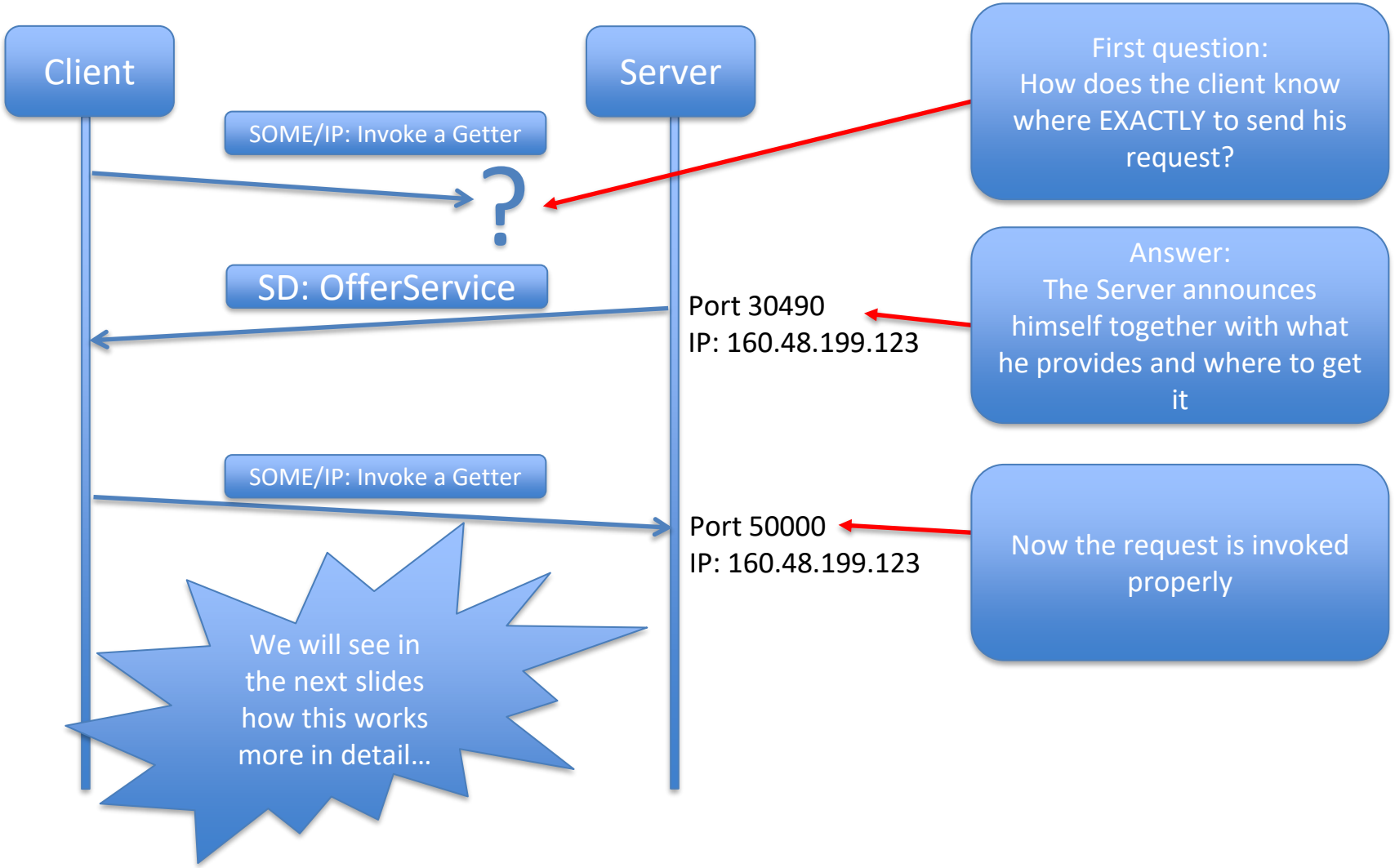
Overview

- To invoke a Service from a counterpart, one has to know that it exists.
  - This is done using Service Discovery (SD).
  - SD allows to announce a Service one offers (provided) and to ask for the one needed (consumed).
  - It makes possible to receive Events and to notify the operative status of the device.
- The location of a certain Service Instance is already know inside the car but not it's status, that's where SD helps out.

SOME/IP allows to transport Request/Response messages and SD to publish and subscribe the corresponding Services.

3- SOME/IP-SD

Overview



3- SOME/IP-SD

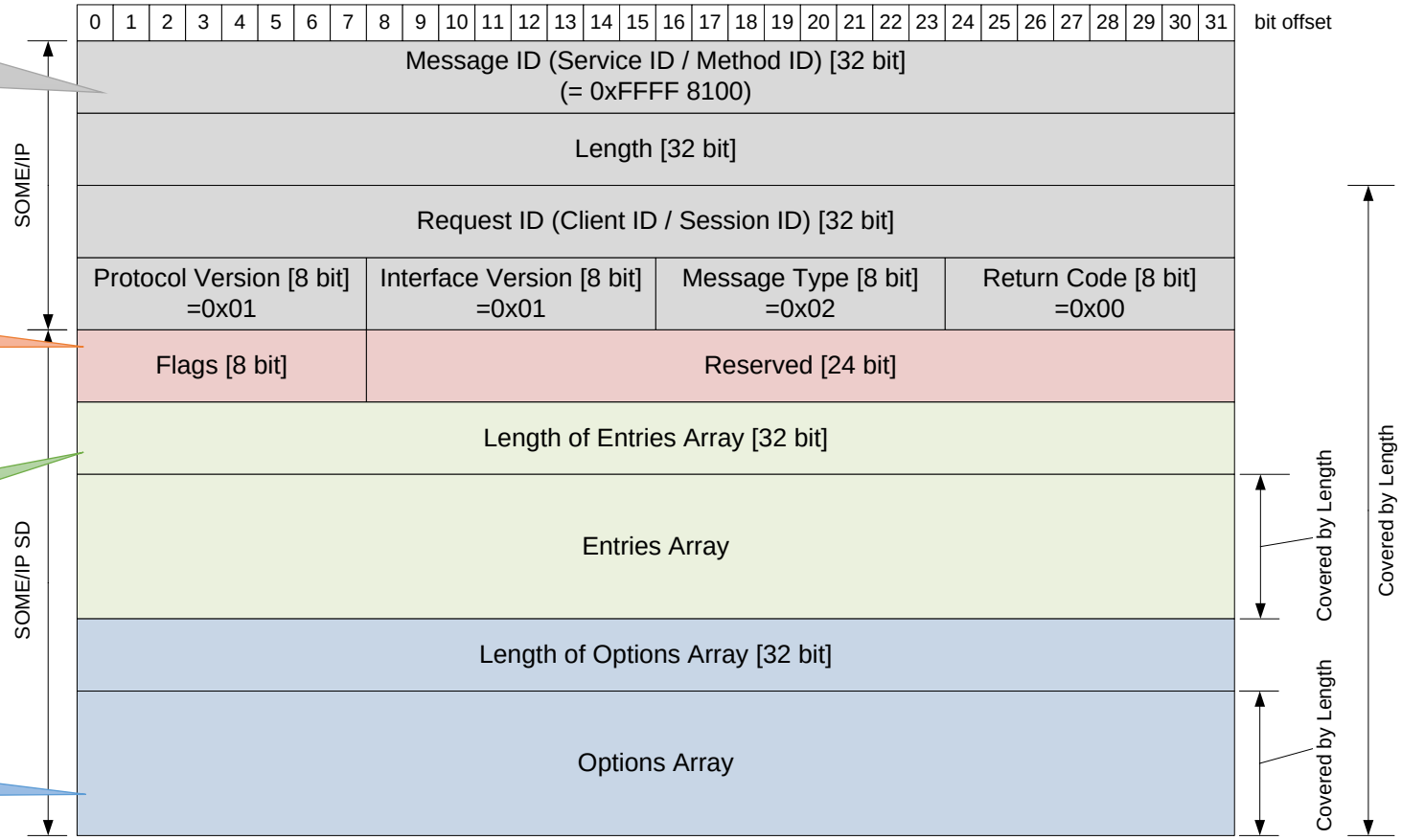
Header

SOME/IP Header:
Service ID: 0xFFFF
Method ID: 0x8100
Interf. Version: 0x01
MsgType: Notification

SOME/IP-SD Header

SOME/IP-SD Entries

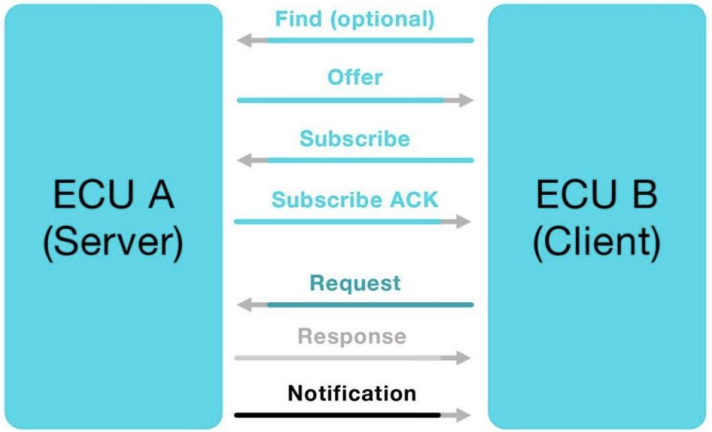
SOME/IP-SD Options



3- SOME/IP-SD

Entries

- “Offer/Stop Offer” Service:
  - SD messages where a server (ECU offering the service) announces that a service is available/no longer available.
- Find Service:
  - A client uses this message to request the availability of a specific service
- “Subscribe/Stop subscribe” Eventgroup:
  - Used by a client ECU to subscribe/unsubscribe to/from event groups that a service offers
- Subscribe Eventgroup Ack/Nack
  - The server sends this message to positively/negatively acknowledge the subscription request from a client



3- SOME/IP-SD
Entries

FindService, OfferService, StopOfferService

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	bit offset
Type								Index 1st options								Index 2nd options								# of opt 1				# of opt 2				
Service ID																Instance ID																
Major Version								TTL																								
Minor Version																																

SubscribeEventgroup, StopSubscribeEventgroup, SubscribeEventgroupAck, SubscribeEventgroupNack

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	bit offset
Type								Index 1st options								Index 2nd options								# of opt 1				# of opt 2				
Service ID																Instance ID																
Major Version								TTL																								
Reserved (0x000)																Counter								Eventgroup ID								

3- SOME/IP-SD
Entries

- Entries are based on their ID and TTL.
  - ID 0x00 – 0x03 for Service Instances and 0x04 – 0x07 for Eventgroups.
  - Other IDs not used today.

Entries for Services Instances:

Type	TTL > 0	TTL = 0
0x00	FindService (multicast)	reserved
0x01	OfferService (unicast/multicast)	StopOfferService (multicast)
0x02	reserved	reserved
0x03	reserved	reserved

Entries for Eventgroups:

Type	TTL > 0	TTL = 0
0x04	reserved	reserved
0x05	reserved	reserved
0x06	SubscribeEventgroup (unicast)	StopSubscribeEventgroup (unicast)
0x07	SubscribeEventgroupAck (unicast)	SubscribeEventgroupNack (unicast)

3- SOME/IP-SD

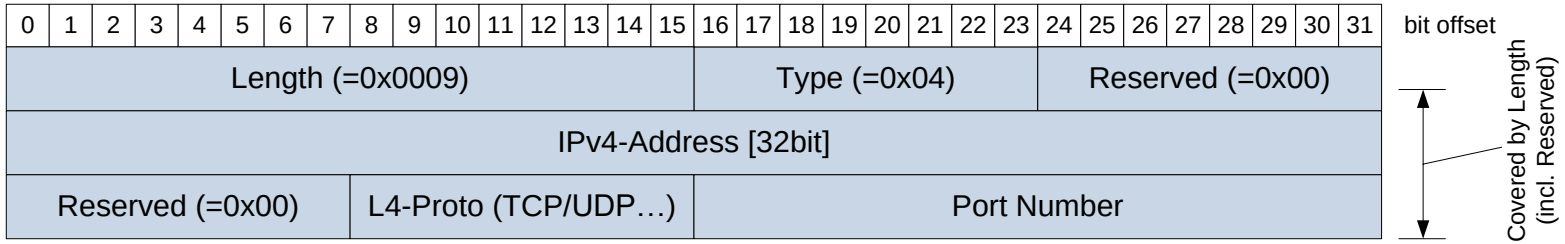
Endpoint options

- Nearly all entries need a set of parameters to make sense.
- Those are called Endpoint options and are referenced by each entry separately:
  - Unicast UDP/TCP: Indicates from where (IP and Port) a unicast Event or Field will be sent. Method invocations are also aimed to this destination.
    - A client will use this endpoint option to tell the server where the Events/Fields have to be sent to.
  - Multicast UDP: To which IP and Port will the Server send its multicast Events and Fields.
- Options are referenced by an entry the following way:
  - **Index 1st options** is the index in the options array of the first option to take.
  - **# of opts 1** tells us how many options to read from this point.
  - Same for Index **2nd options** and **# of opts 2**.

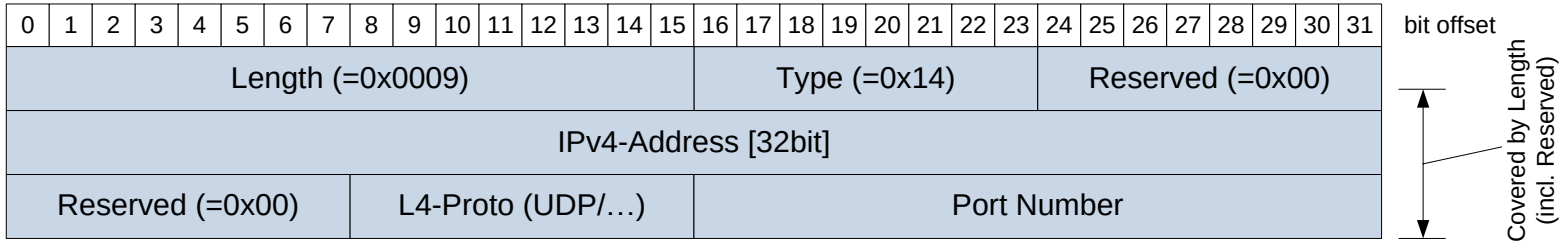
3- SOME/IP-SD

Endpoint options – IPv4

IPv4 Endpoint Option (option type 0x04):



IPv4 Multicast Option (option type 0x14):

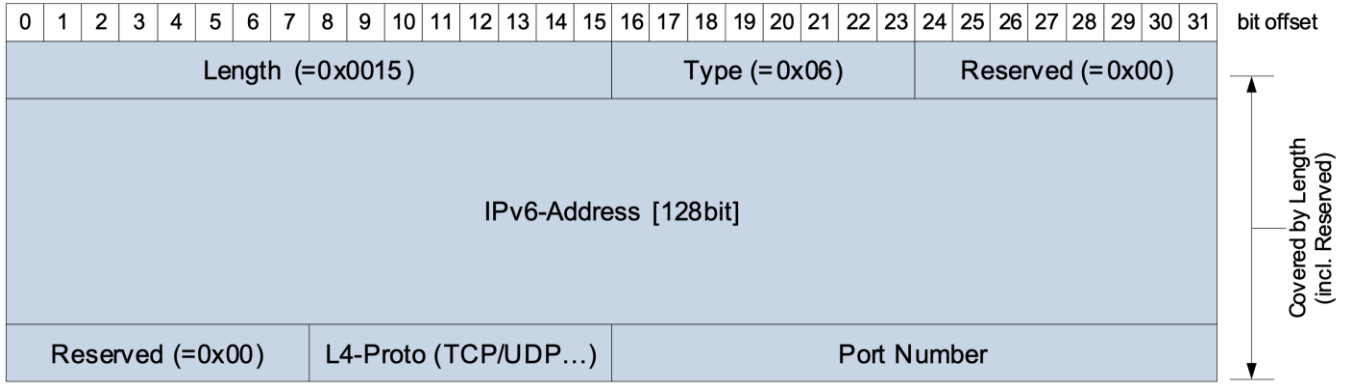


- Endpoint Options for IPv4.
- 0x04 for Unicast.
- 0x14 for Multicast.
- 0x24 for SD endpoint. (not relevant).

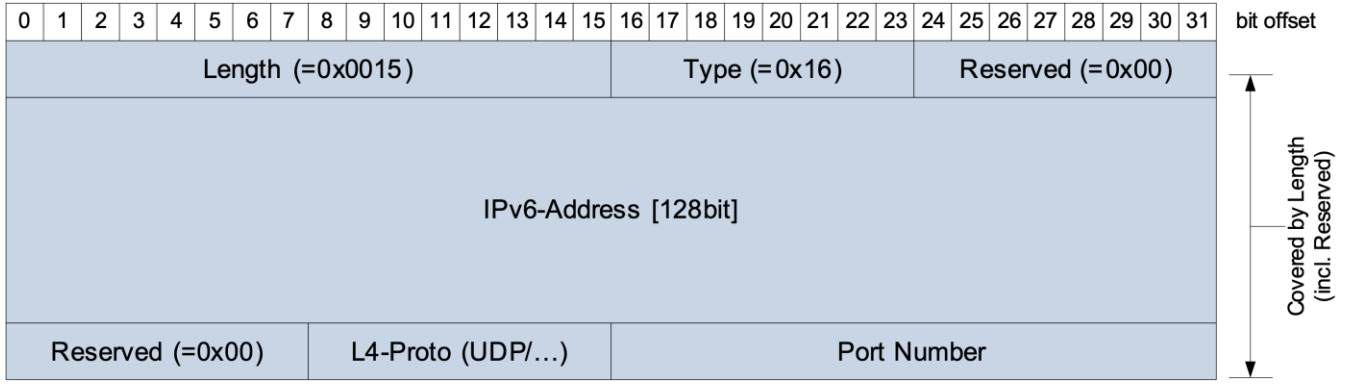
3- SOME/IP-SD

Endpoint options – IPv6

IPv6 Endpoint Option (option type 0x06):



IPv6 Multicast Option (option type 0x16):



- Endpoint Options for IPv6.

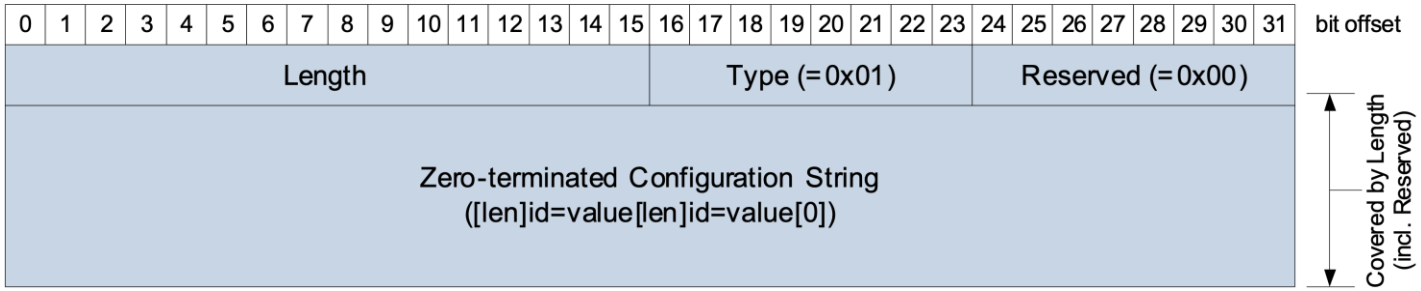
- 0x06 for Unicast.
- 0x16 for Multicast.
- 0x26 for SD endpoint. (not relevant).

3- SOME/IP-SD

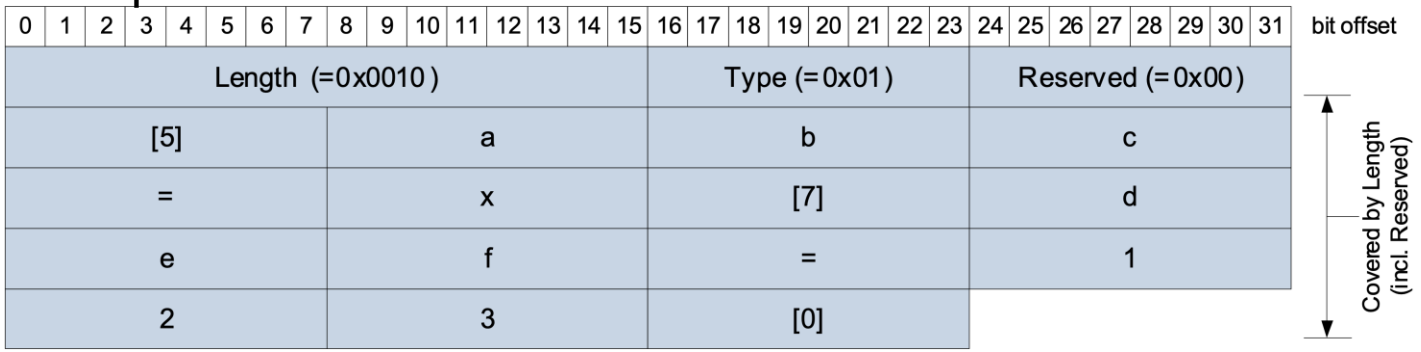
Endpoint options – Configuration Option (ViWi)

VW

Configuration Option (option type 0x01):



Example:



- Configuration Option
- Allows to transport additional config data as key value strings.
- Intended usage:
  - Offering non-SOME/IP services (e.g., ViWi)
  - “otherserv=viwi”

3- SOME/IP-SD

Endpoint options – Configuration Option (ViWi)

VW

```
› Flags: 0xe0, Reboot Flag, Unicast Flag, Explicit Initial Events Flag
  Reserved: 0x000000
  Length of Entries Array: 16
› Entries Array
  Length of Options Array: 101
⌄ Options Array
  ⌄ 0: IPv6 Endpoint Option (TCP)
      Length: 21
      Type: 6
      Reserved: 00
      IPv6 Address: ICAS3-IVI_Info. (fd53:7cb8:383:3::108)
      Reserved 2: 00
      Protocol: 6 (TCP)
      Port: 49585
  ⌄ 1: Configuration Option
      Length: 74
      Type: 1
      Reserved: 00
      Configuration String: \020category=webApps\fl6proto=viwi\021otherserv=browser\ttxtvers=1\rversion=0.2.0
```

Intended usage:

- Offering non-SOME/IP services
- “otherserv=viwi”

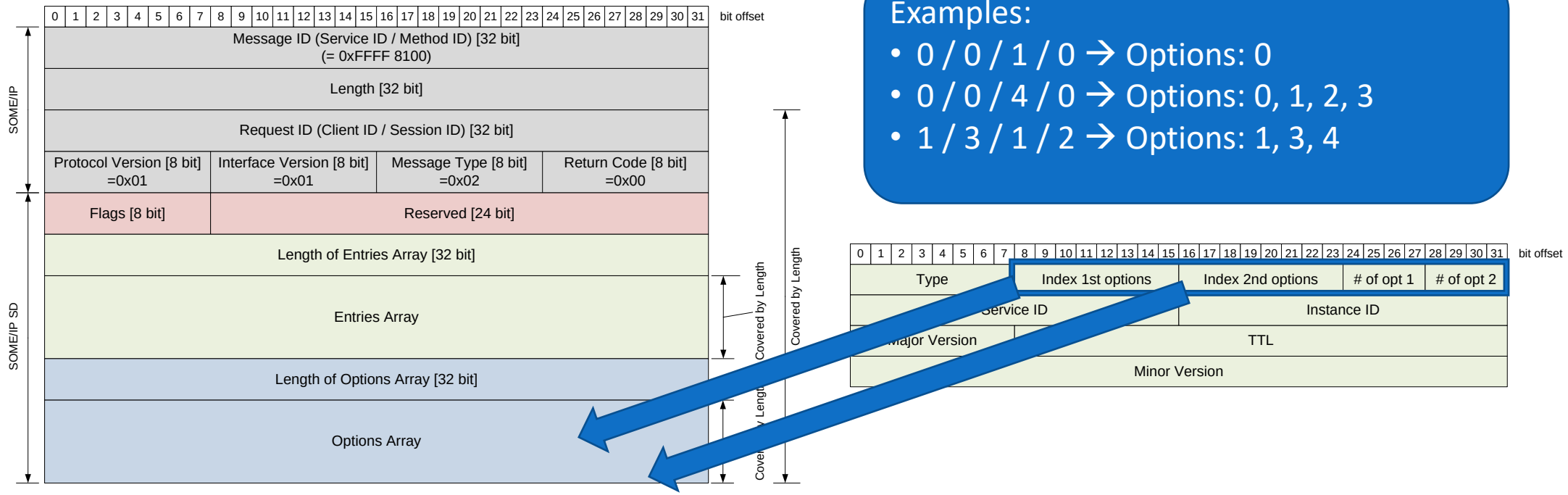
ViWi:

- category=webApps
- l6proto=viwi
- otherserv=browser
- txtversion=1
- version=0.2.0

3- SOME/IP-SD

Endpoint options – Referencing

Entries reference relevant options:



Examples:

- 0 / 0 / 1 / 0 → Options: 0
- 0 / 0 / 4 / 0 → Options: 0, 1, 2, 3
- 1 / 3 / 1 / 2 → Options: 1, 3, 4

- Options are numbered: 0, 1, 2, 3, …
- “Index” determines start position, “Number of Options” the number of options referenced.
- Two option runs (Index / number of options) can be used.

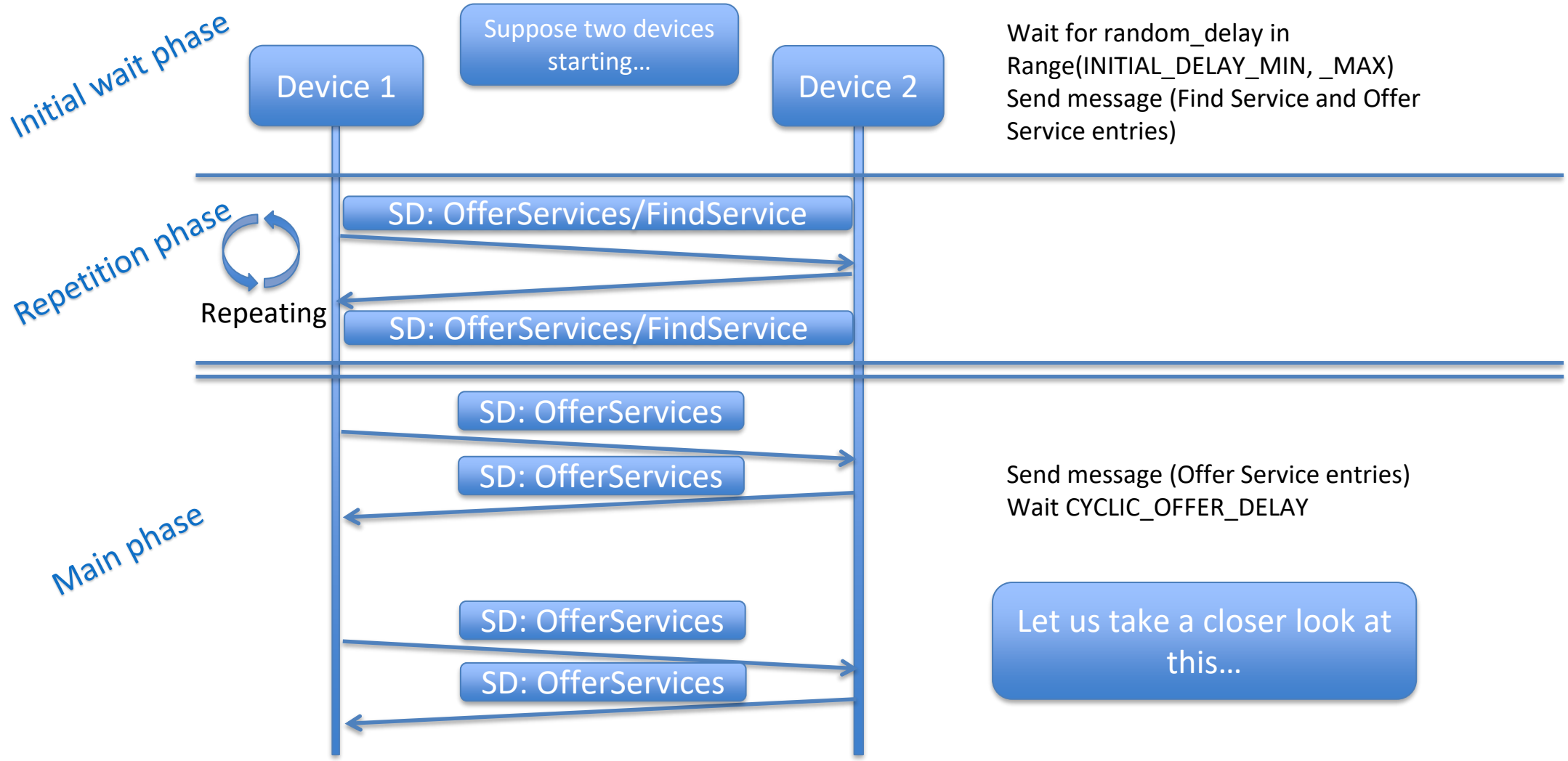
3- SOME/IP-SD

Endpoint options – Referencing

- OfferService references:
  - 0..1 IP Endpoint Options for UDP.
  - 0..1 IP Endpoint Options for TCP.
  - 0..1 Configuration Options (ViWi).
- SubscribeEventgroup references:
  - 0..1 IP Endpoint Options for UDP.
  - 0..1 IP Endpoint Options for TCP.
- SubscribeEventgroupAck references:
  - 0..1 IP Multicast Options for UDP.

3- SOME/IP-SD

State machines and timings - Phases

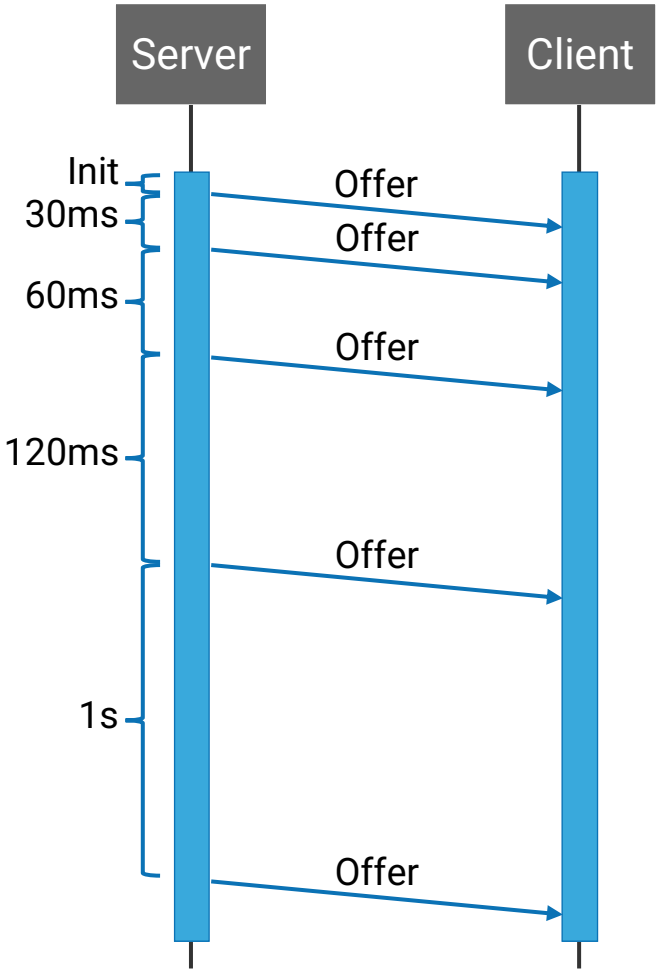


3- SOME/IP-SD

State machines and timings - Server

- SOME/IP-SD defines three phases:
  - Initial wait phase:
    - Allows to system to get ready.
    - Collect some services to better aggregate messages.
  - Repetition phase:
    - Used for fast synchronization.
    - Ensures a fast system startup, if a message gets lost.
  - Main phase:
    - Stabilizes the system and keeps it alive.

- Offer announces Service Instances.
  - This includes IP address + Port number.



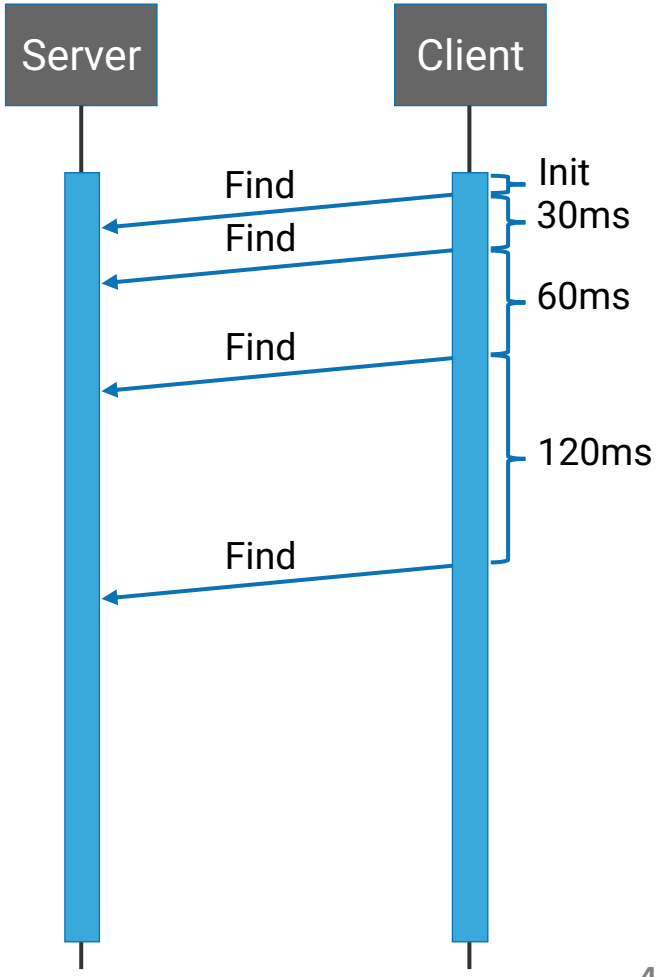
Example timings

3- SOME/IP-SD

State machines and timings - Client

- Clients are like Servers:
  - Initial wait phase:
    - Allows to system to get ready.
    - Collect some services to better aggregate messages.
  - Repetition phase:
    - Used for fast synchronization.
    - Ensures a fast system startup, if a message gets lost.
  - Main phase:
    - No Finds in Main Phase!

- Find looks for Service Instance.
- Find stops, when offer is received.



Example timings

3- SOME/IP-SD

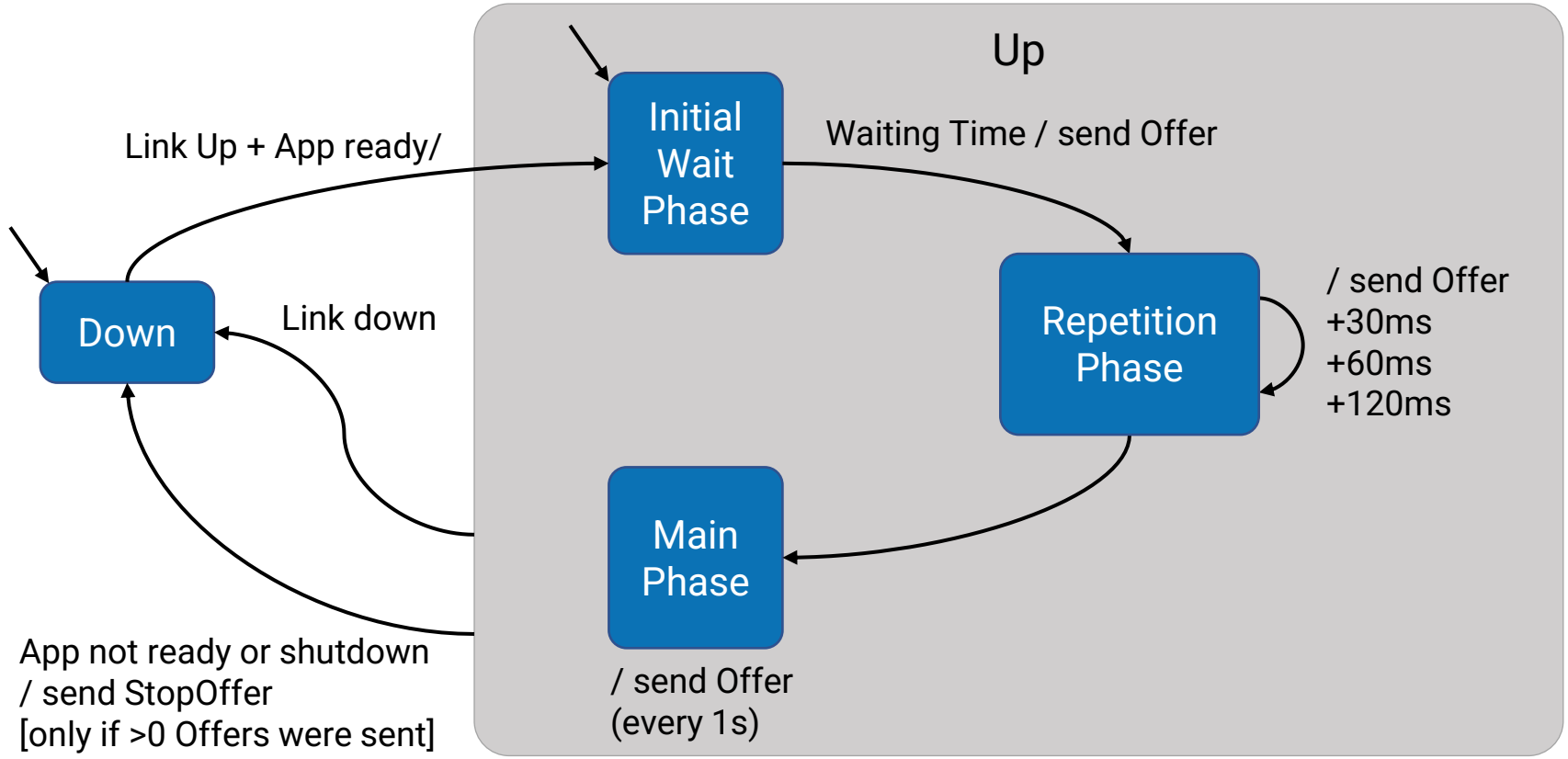
State machines and timings - Timings

	Example (AUTOSAR)	Example PPE	Comments
Initial Delay	10ms – 10ms	0ms – 0ms	This parameter keeps back service offers to pack more entries together.
Repetitions Base Delay	30ms	100ms	Fast startup to make startup more robust. Loss of the first offer results in this delay.
Repetitions Max	3	3	Number of repetitions while doubling delay.
Offer Cyclic Delay	1s	1s	Keeping system alive with cyclic offer.
TTL	3s	3s	How fast to detect problem.
Request-Response-Delay	10ms – 10ms	0ms – 0ms	This parameter keeps back subscribes to pack more entries together.

3- SOME/IP-SD

State machines and timings – Server state machine

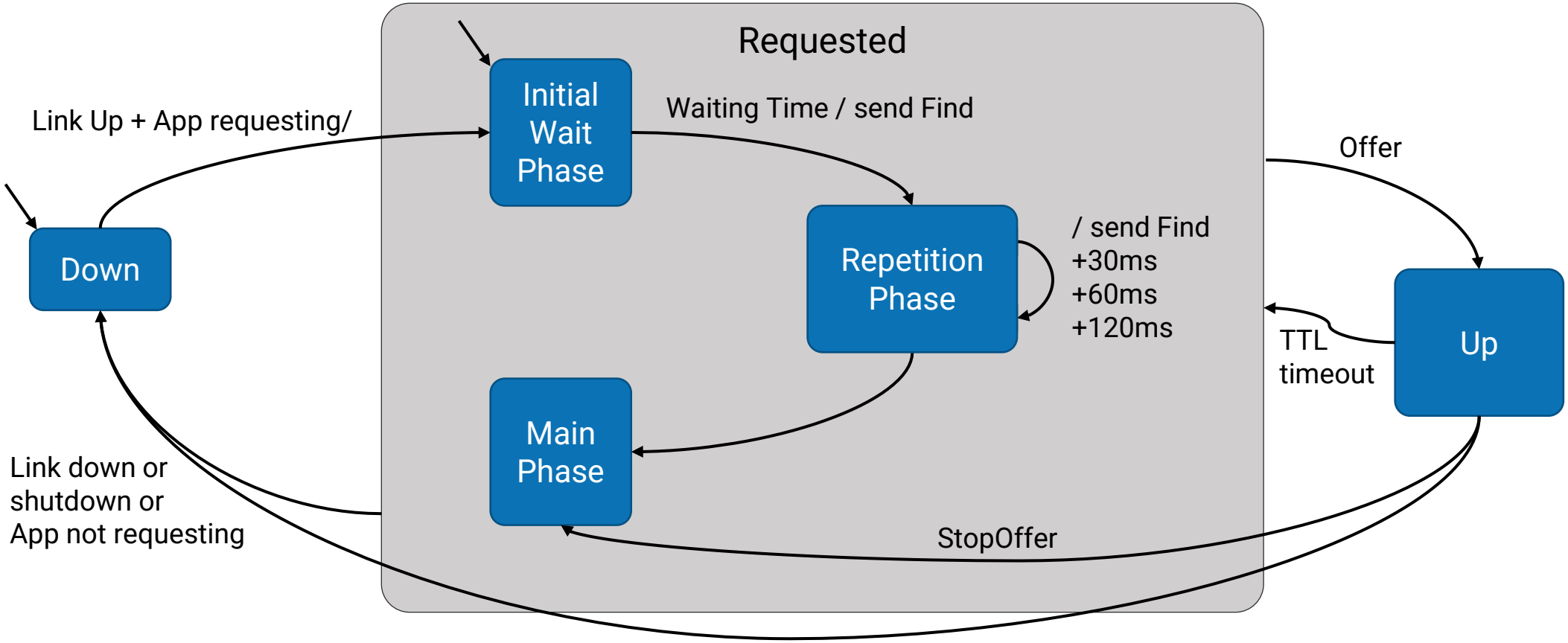
- Server State Machine for Service Instance (simplified).
  - (more detailed discussion on interaction with NM later).



3- SOME/IP-SD

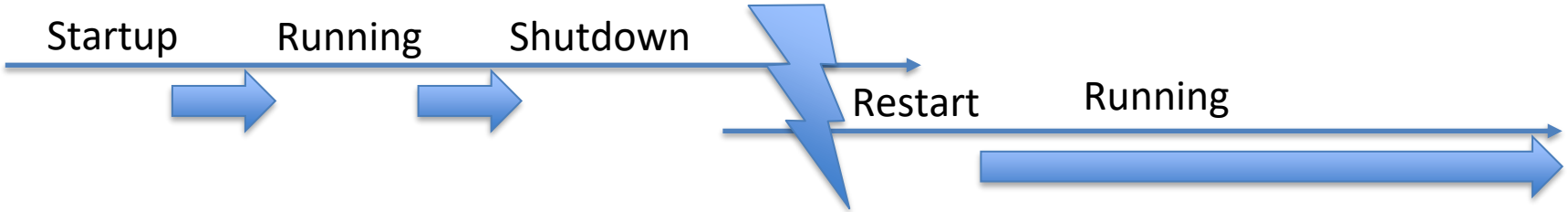
State machines and timings – Client state machine

- Client State Machine for Service Instance (simplified).
  - (more detailed discussion on interaction with NM later).



3- SOME/IP-SD
Reboot detection

- Reinitialization of a component, either Application or ECU.
  - Is not the same as a Shutdown (a sequence to bring the ECU from the running to a low power or power state).
  - Makes the device restart: Interruption of the shutdown to bring the device back into running state.



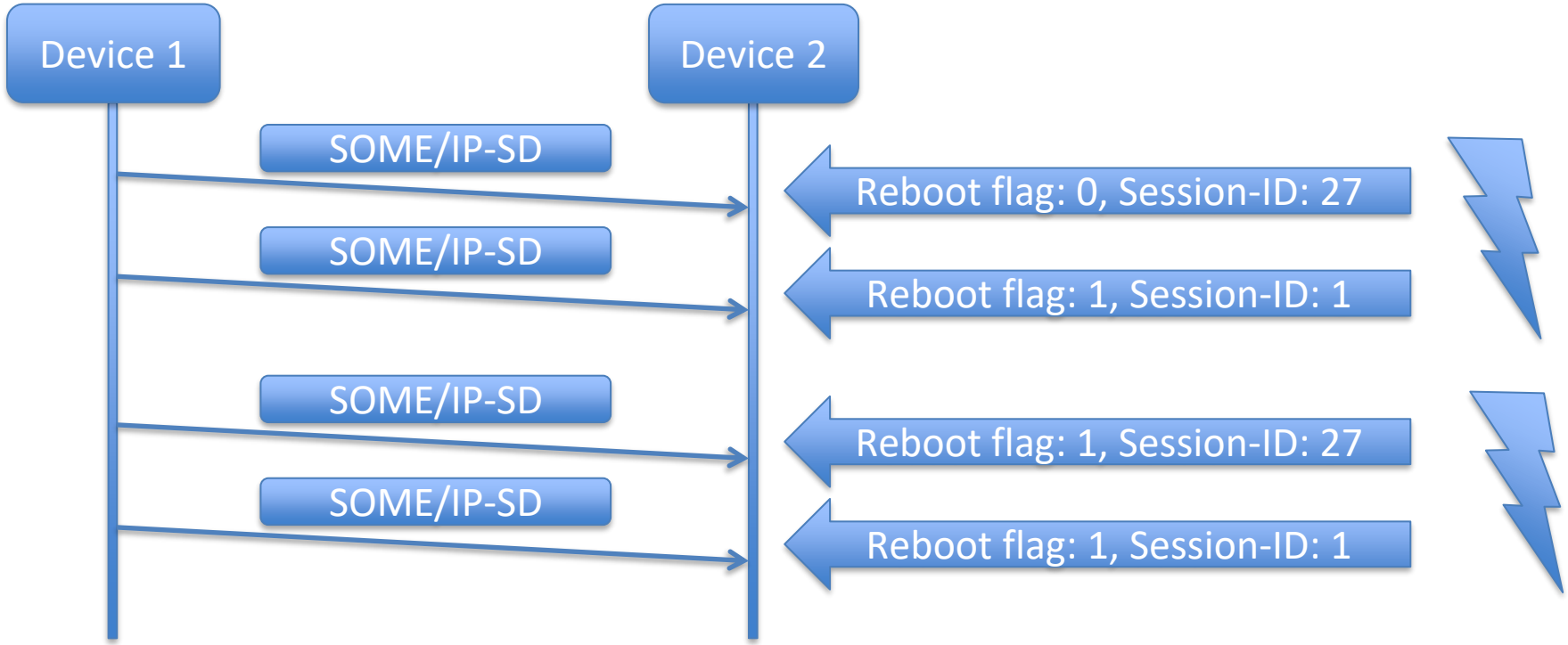
- Generally a reboot only occurs due to a failure or error.

3- SOME/IP-SD
Reboot detection

- A reboot is detected by comparing each incoming SOME/IP(-SD) message with it’s predecessor.
- Reboot flag and Session-ID of each is taken into account.
- Conditions checked:
  - old.reboot == 0 and new.reboot == 1.
  - old.reboot == 1 and new.reboot == 1 and old.session_id >= new.session_id.

- SOME/IP-SD implementations have to reliably detect the reboots of their peer.
- When the system detects the reboot of a peer, it shall update its state accordingly. Services and Subscriptions shall be expired if they are not updated again.

3- SOME/IP-SD
Reboot detection

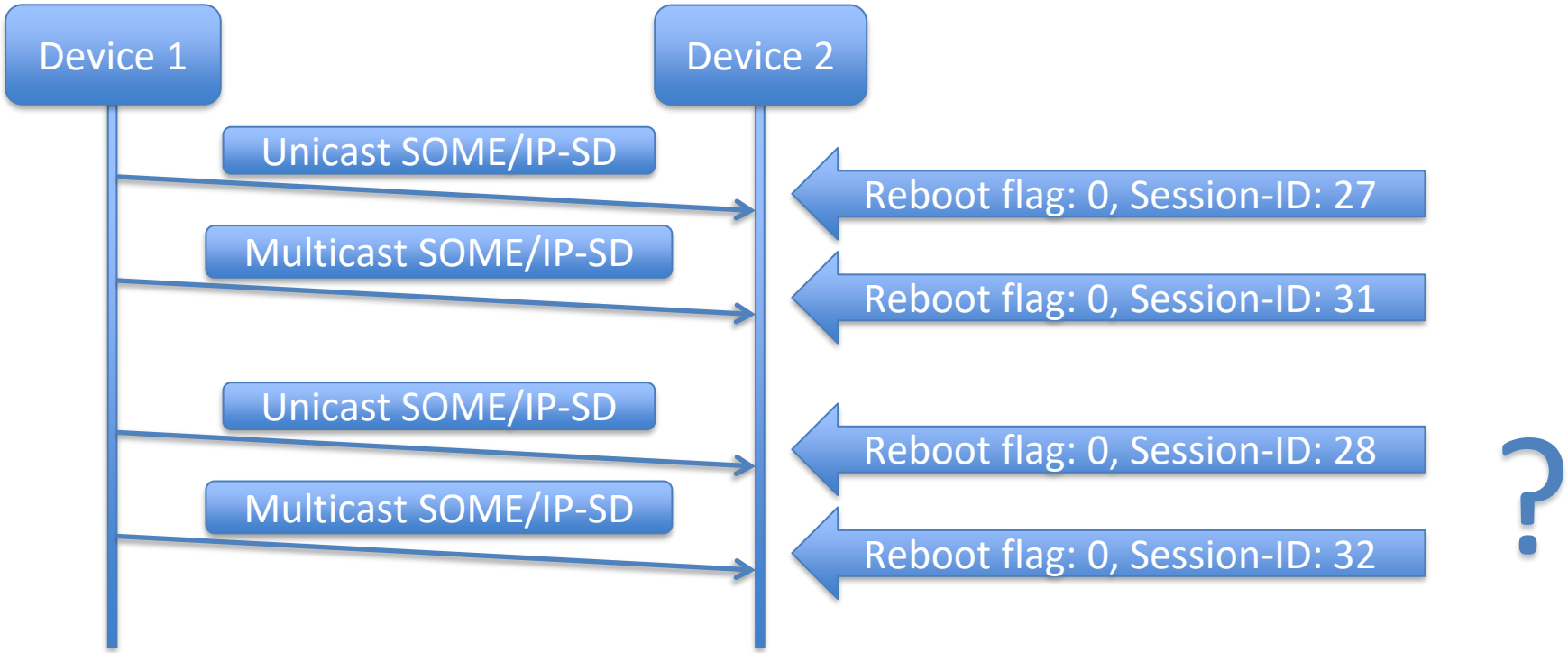


- The Reboot Flag of the SOME/IP-SD Header shall be set to one for all messages after reboot.

3- SOME/IP-SD
Session handling

- With each SOME/IP-SD message sent, the Session-ID is incremented by one.
- In case it reaches 0xFFFF it has to wrap around and start with 1 again but never have value 0.
  - In case of wraparound, the reboot flag is set to 0 again.
- Each device has to store and treat separately the Session-ID for each communication peer and also differentiate if the communication is unicast or multicast.
  - Each unicast communication has a different Session-ID.
  - Multicast communication has it’s own Session-ID.

3- SOME/IP-SD
Session handling



primatec engineering

SOME/IP-TP

- Motivation
- Header
- Facts and limitations

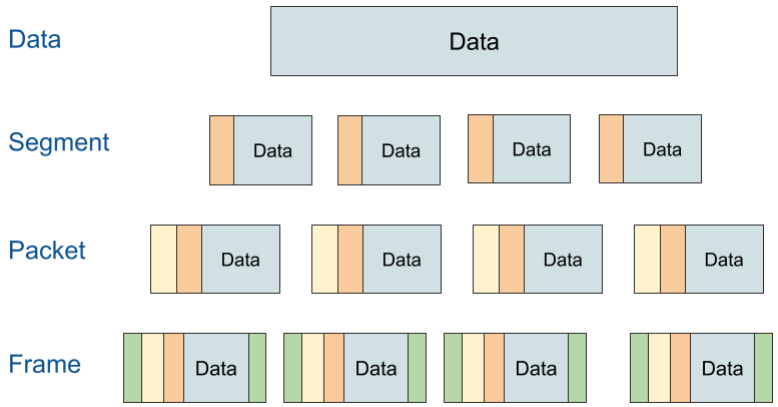
4

4- SOME/IP-TP
Motivation

- In vehicles exist use cases that need large payload and safety.
  - e.g., Radar and Lidar data.
  - TCP cannot be used due to timings (safety relevant messages).
  - Workaround “static splitting” creates “ugly” service interfaces.

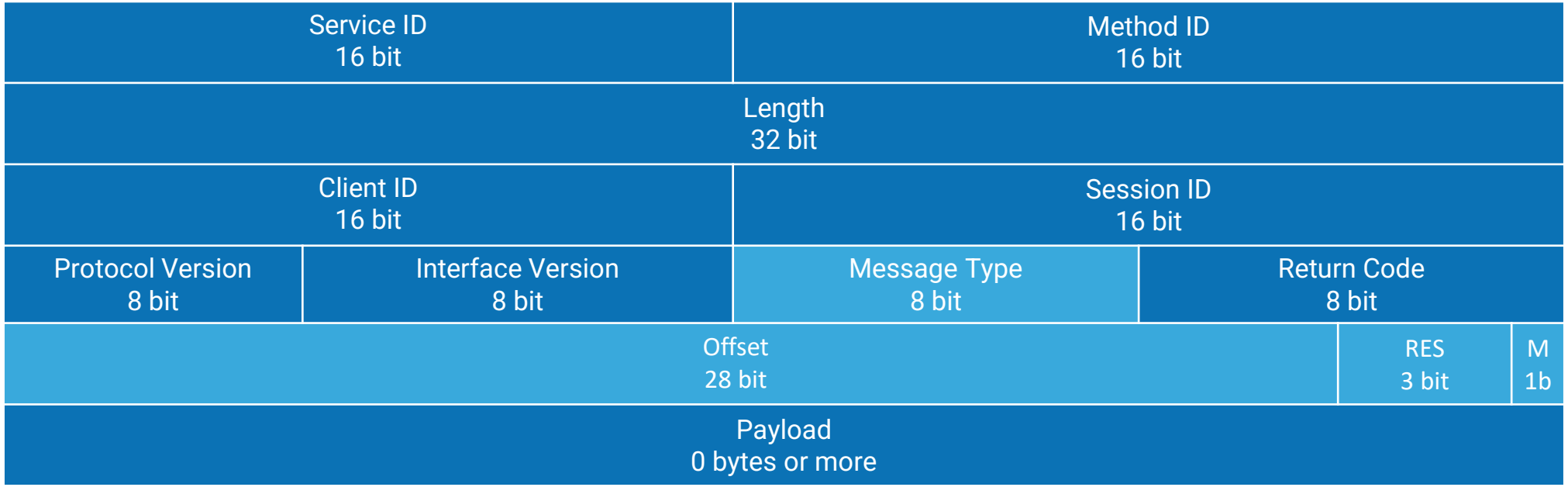
- SOME/IP-TP introduces a possibility to split SOME/IP messages over UDP in smaller “messages”.
  - This allows up to 4GB large messages, if buffers permit.
  - Buffers are assigned to Message ID. → More robust buffer management.

- Similar but better than IP fragmentation.
  - IP Fragmentation: up to 64k and no assigned buffers.



4- SOME/IP-TP
Header

- SOME/IP-TP adds a new header line by adjusting the message type.
  - Message type TP-Flag: 0x20 (third most significant bit). 1: TP header on.
  - Header line adds:
    - Offset: Starting position of segment (32-bit number with the last 4 bits set to zero).
    - 3 reserved bits: set to 0 and ignore.
    - “More Segments” flag: 1 (more segments following), 0 (this is the last segment).



4- SOME/IP-TP
Header - Example

Service ID = 0x0101		Method ID = 0x1234.	
Length = 8 + 3883			
Client ID = 0x0000		Session ID = 0x0001	
Protocol Version = 0x01	Interface Version 0x01	Message Type **0x00**	Return Code 0x00
Payload = [0..3882] (=3883 bytes)			

- 3883 Bytes segmented into 3 messages.
- These are transmitted over UDP.
- Last segment: “More segments” = 0.
- Length off segment depends on limits.
- Message type here Request (0x00):

0x00	REQUEST

→

0x20	REQUEST + TP

Service ID = 0x0101		Method ID = 0x1234.	
Length = 8 + 4 + 1392 = 1404			
Client ID = 0x0000		Session ID = 0x0001	
Protocol Version 0x01	Interface Version 0x01	Message Type **0x20**	Return Code 0x00
Offset **0**		RES 0 0 0	M **1**
Payload [0..1391] (1392 Bytes)			

Service ID = 0x0101		Method ID = 0x1234.	
Length = 8 + 4 + 1392 = 1404			
Client ID = 0x0000		Session ID = 0x0001	
Protocol Version 0x01	Interface Version 0x01	Message Type **0x20**	Return Code 0x00
Offset **1392**		RES 0 0 0	M **1**
Payload [1392..2783] (1392 Bytes)			

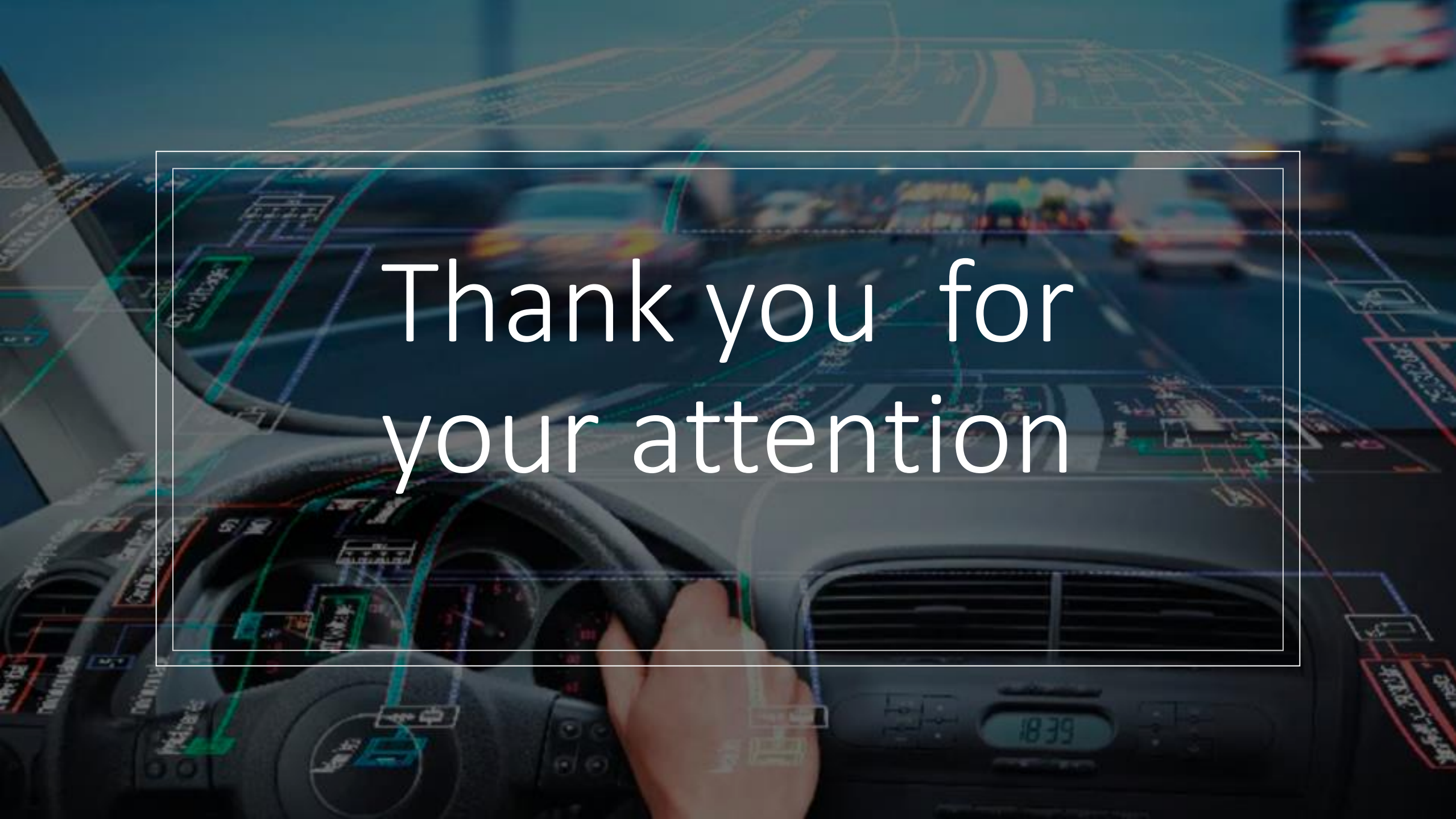
Service ID = 0x0101		Method ID = 0x1234.	
Length = 8 + 4 + 1099 = 1111			
Client ID = 0x0000		Session ID = 0x0001	
Protocol Version 0x01	Interface Version 0x01	Message Type **0x20**	Return Code 0x00
Offset **2784**		RES 0 0 0	M **0**
Payload [2784..3882]			

4- SOME/IP-TP
Facts and limitations

- Segment payload must be multiple of 16 Bytes (all but last segment).
- Lost segments lead to lost messages. No retransmission.
- Reordering of segments is not supported. Drop! (AUTOSAR classic).
- SOME/IP-TP currently not supported for Fields since ”Initial Events' would lead to overload situation.

- SOME/IP-TP requires E2E profile 7 when payload > 4096 Bytes.
  - (32bit length, 32bit Counter, 64bit CRC, 32bit Data ID).
  - Profile 7m for methods.

- Warning: SOME/IP-TP can lead to strong bursts. Shaping needed!


Thank you for
your attention