INTERNATIONAL STANDARD

ISO 14229-5

Second edition
2022-04

Road vehicles — Unified diagnostic services (UDS) —

Part 5: Unified diagnostic services on Internet Protocol implementation (UDSonIP)

Véhicules routiers — Services de diagnostic unifiés (SDU) —

Partie 5: SDU sur l'implémentation du protocole internet (SDUsurIP)

Reference number
ISO 14229-5:2022(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword v
Introduction vi
1 Scope 1
2 Normative references 1
3 Terms and definitions 1
4 Symbols and abbreviated terms 1
4.1 Symbols 1
4.2 Abbreviated terms 2
5 Conventions 2
6 Service primitive interface definition 2
7 Technical requirements overview 3
8 Application layer 4
8.1 ISO 14229-1 service primitive parameters 4
8.2 A_Data.req, A_Data.ind, and A_Data.conf service interface 4
8.3 UDSonIP services overview 4
8.4 A_PDU definition 5
8.4.1 Generic DoIP header of A_PDU 5
8.4.2 A_PDU for UDS request and response message 6
8.4.3 A_PDU for UDS periodic response message 6
8.5 DiagnosticSessionControl service UDSonIP implementation requirements 7
8.5.1 General 7
8.5.2 TCP connection handling 7
8.5.3 TCP connection closing 7
8.6 ECUReset service UDSonIP implementation requirements 9
8.6.1 General 9
8.6.2 TCP connection handling 9
8.6.3 TCP connection closing 9
8.7 ReadDataByPeriodicIdentifier service UDSonIP implementation requirements 9
8.7.1 General 9
8.7.2 Service interface 9
8.7.3 Service primitive data types 11
8.7.4 Periodic response message A_PDU format 11
8.7.5 Periodic transmission response message handling 11
8.7.6 Periodic transmission message flow 12
8.8 ResponseOnEvent service UDSonIP implementation requirements 14
8.8.1 General 14
8.8.2 Activated storageState 14
8.9 Timing parameter definition 17
8.9.1 Request and response message timing parameter values 17
8.9.2 Unsolicited response messages 17
9 Presentation layer 17
10 Session layer 17
10.1 Service primitive parameter definition 17
10.2 S_Data.req, S_Data.ind, and S_Data.conf service interface 18
11 Transport layer 18
11.1 Service primitive parameter definition 18
11.2 T_Data.req, T_Data.ind, and T_Data.conf service interface 18
11.3 T_PDU definition 21
11.4 DoIP transport layer and network layer interface adaptation 22

11.4.1 Mapping of data link-independent service primitives onto IP data link-dependent service primitives 22
11.4.2 Mapping of T_PDU onto DoIP_PDU 22
12 Network layer 23
12.1 Service primitive parameter definition 23
12.2 DoIP_Data.req, DoIP_Data.ind, and DoIP_Data.conf service interface 23
12.3 Logical address information 23
12.4 DoIP_PDU definition 24
13 Data link layer 24
13.1 Service primitive parameter definition 24
13.2 L_Data.req, L_Data.ind, and L_Data.conf service interface 24
13.3 L_PDU definition 24
14 Physical layer 25
Bibliography 26

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 31, *Data communication*.

This second edition cancels and replaces the first edition (ISO 14229-5:2013), which has been technically revised.

The main changes are as follows:

— restructuration of the document;

— introduction of requirement numbers, names and definitions;

— technical content improvements based on implementation feedback from the automotive industry.

A list of all parts in the ISO 14229 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

The ISO 14229 series has been established in order to define common requirements for diagnostic systems, whatever the serial data link is.

To achieve this, the ISO 14229 series is based on the Open Systems Interconnection (OSI) Basic Reference Model in accordance with ISO/IEC 7498-1[1] and ISO/IEC 10731[2], which structures communication systems into seven layers. When mapped on this model, the services used by a diagnostic tester (client) and an electronic control unit (ECU, server) are structured into the following layers:

- application layer (layer 7) specified in ISO 14229-1 and ISO 14229-3 to ISO 14229-8;
- presentation layer (layer 6) specified in ISO 14229-1 and ISO 14229-3 to ISO 14229-8;
- session layer services (layer 5) specified in ISO 14229-2 and ISO 14229-3 to ISO 14229-8.

Figure 1 illustrates the UDSonIP document and related documents according to the OSI model.

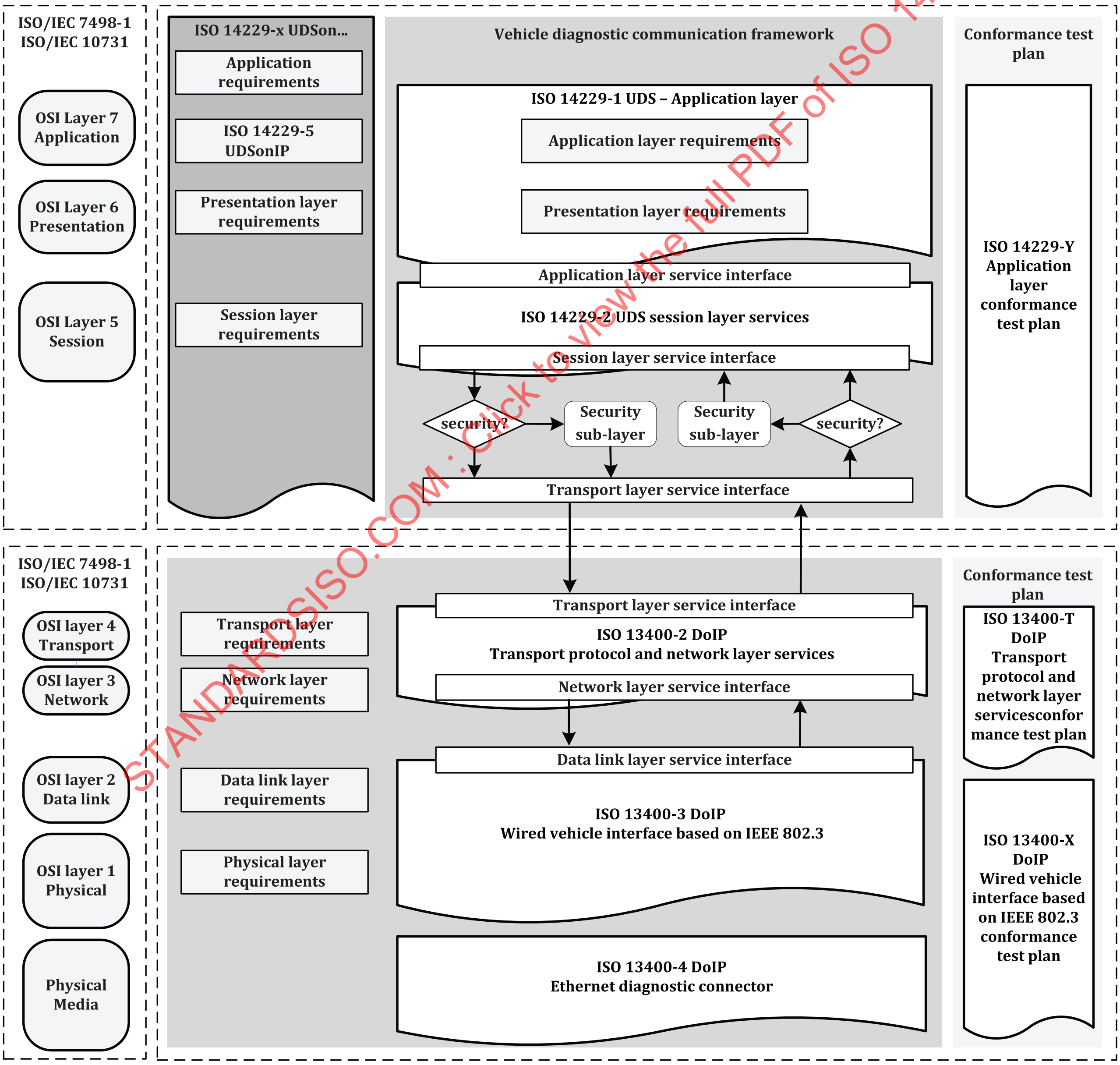

ISO/IEC 7498-1
ISO/IEC 10731
OSI Layer 7 Application
OSI Layer 6 Presentation
OSI Layer 5 Session
ISO 14229-x UDSon...
Application requirements
ISO 14229-5 UDSonIP
Presentation layer requirements
Session layer requirements
Vehicle diagnostic communication framework
ISO 14229-1 UDS – Application layer
Application layer requirements
Presentation layer requirements
Application layer service interface
ISO 14229-2 UDS session layer services
Session layer service interface
security?
Security sub-layer
Security sub-layer
security?
Transport layer service interface
Conformance test plan
ISO 14229-Y Application layer conformance test plan
ISO/IEC 7498-1
ISO/IEC 10731
OSI layer 4 Transport
OSI layer 3 Network
OSI layer 2 Data link
OSI layer 1 Physical
Physical Media
Transport layer requirements
Network layer requirements
Data link layer requirements
Physical layer requirements
Transport layer service interface
ISO 13400-2 DoIP
Transport protocol and network layer services
Network layer service interface
Data link layer service interface
ISO 13400-3 DoIP
Wired vehicle interface based on IEEE 802.3
ISO 13400-4 DoIP
Ethernet diagnostic connector
Conformance test plan
ISO 13400-T DoIP Transport protocol and network layer servicesconformance test plan
ISO 13400-X DoIP Wired vehicle interface based on IEEE 802.3 conformance test plan

Figure 1 — UDSonIP document reference according to OSI model

Road vehicles — Unified diagnostic services (UDS) —

Part 5: Unified diagnostic services on Internet Protocol implementation (UDSonIP)

1 Scope

This document specifies an application profile for the implementation of unified diagnostic services (UDS) Internet Protocol (IP) in road vehicles (UDSonIP).

UDSonIP references ISO 14229-1 and ISO 14229-2 and specifies implementation requirements of the diagnostic services to be used for diagnostic communication on Internet Protocol.

This document includes

— additional requirements specific to the implementation of UDS on the Ethernet network, and

— specific restrictions in the implementation of UDS on the Ethernet network.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 13400-2, *Road vehicles — Diagnostic communication over Internet Protocol (DoIP) — Part 2: Transport protocol and network layer services*

ISO 13400-3, *Road vehicles — Diagnostic communication over Internet Protocol (DoIP) — Part 3: Wired vehicle interface based on IEEE 802.3*

ISO 14229-1, *Road vehicles — Unified diagnostic services (UDS) — Part 1: Application layer*

ISO 14229-2, *Road vehicles — Unified diagnostic services (UDS) — Part 2: Session layer services*

IETF RFC 793:1981, *Transmission Control Protocol — DARPA Internet Program — Protocol Specification*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 14229-1 and ISO 14229-2 apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at https://www.electropedia.org/

4 Symbols and abbreviated terms

4.1 Symbols

—	empty table cell or feature undefined
t	time
$t_{\mathtt{P\_Client}}$	client application layer timer
$t_{\mathtt{P2\_Server}}$	server application layer timer
$t_{\mathtt{S3\_Client}}$	client session layer timer
$t_{\mathtt{S3\_Server}}$	server session layer timer
$t_{\mathtt{S3\_Server\_Reload}}$	server session layer timeout value-reload
$t_{\mathtt{P6\_DoIP\_Client}}$	client application layer timeout value for DoIP

4.2 Abbreviated terms

DID	data identifier
DoIP	diagnostic communication over Internet Protocol
DoIP_AI	DoIP address information
DoIP_SA	DoIP source address
DoIP_TA	DoIP target address
DoIP_TAtype	DoIP target address type
GH_PT	generic header payload type
GH_PL	generic header payload length
IP	Internet Protocol
OSI	Open System Interconnection
pDID	periodic data identifier
UDS	unified diagnostic services
VM	vehicle manufacturer

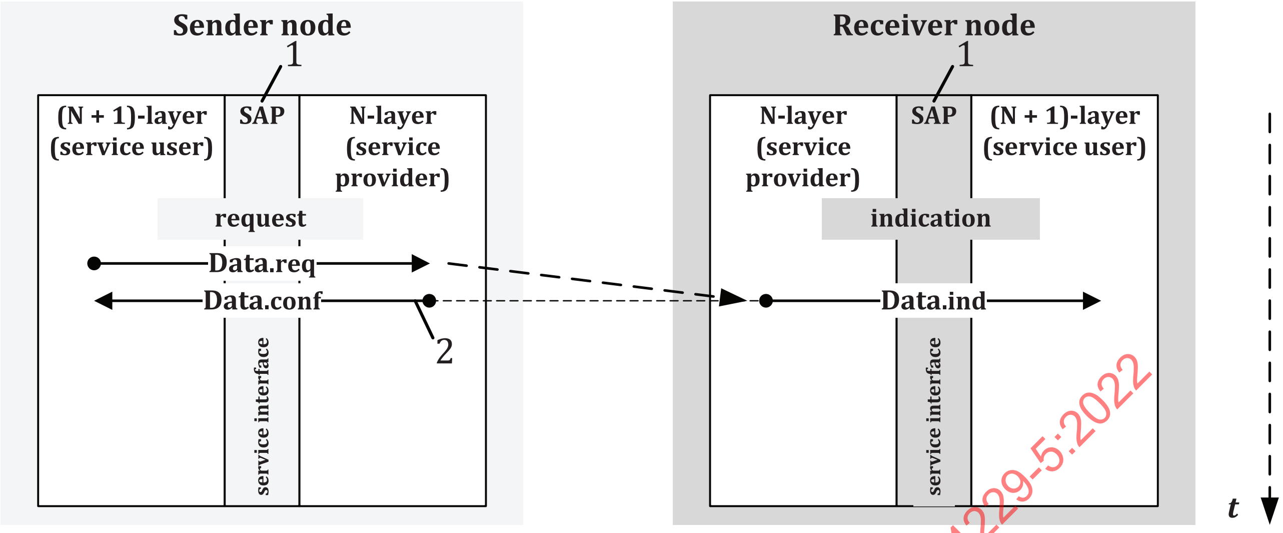
5 Conventions

This document is based on OSI service conventions as specified in ISO/IEC 10731[2].

6 Service primitive interface definition

The service interface defines the service and parameter mapping from the application layer to the session layer.

Figure 2 shows the Data.req (request), Data.ind (indication), and Data.conf (confirmation) service interface.



Key

1 service access point between application and application layer

2 read back from N-layer service provider

t time

Figure 2 — Data.req, Data.ind, and Data.conf service interface

7 Technical requirements overview

Table 1 provides an overview on the technical requirements and their associated requirement number.

Table 1 — Technical requirements overview

OSI#.REQ#	Technical requirement title
7	**Application layer**
7.1	ISO 14229-1 service primitive parameters
7.2	A_Data.req, A_Data.ind, and A_Data.conf service interface
7.3	UDSonIP-specific requirements
7.4	No UDSonIP-specific requirements
7.5	Generic DoIP header of A_PDU
7.6	A_PDU for UDS request and response message
7.7	A_PDU for UDS periodic response message
7.8	DiagnosticSessionControl – TCP connection handling
7.9	DiagnosticSessionControl – TCP connection closing
7.10	ECUReset – TCP connection handling
7.11	ECUReset – TCP connection closing
7.12	ReadDataByPeriodicIdentifier – A_Data.req
7.13	ReadDataByPeriodicIdentifier – A_Data.ind
7.14	ReadDataByPeriodicIdentifier – A_Data.conf
7.15	ReadDataByPeriodicIdentifier – Service primitive parameters
7.16	ReadDataByPeriodicIdentifier – Periodic response A_PDU format
7.17	ReadDataByPeriodicIdentifier – Periodic transmission response message handling
7.18	ReadDataByPeriodicIdentifier – Periodic transmission response message server restrictions

Table 1 *(continued)*

OSI#.REQ#	Technical requirement title
7.19	Timing parameter definition
7.20	Unsolicited response messages
6	**Presentation layer**
	No requirement statement in this document
5	**Session layer**
5.1	Service primitive parameter definition
5.2	S_Data.req, S_Data.ind, and S_Data.conf service interface
4	**Transport layer**
4.1	Service primitive parameter definition
4.2	T_Data.req, T_Data.ind, and T_Data.conf service interface
4.3	Mapping of data link-independent service primitives onto IP data link-dependent service primitives
4.4	Mapping of T_PDU onto DoIP_PDU
3	**Network layer**
3.1	Service primitive parameter definition
3.2	DoIP_Data.req, DoIP_Data.ind, and DoIP_Data.conf service interface
3.3	Logical address information
2	**Data link layer**
2.1	Service primitive parameter definition
2.2	L_Data.req, L_Data.ind, and L_Data.conf service interface
1	**Physical layer**
	No requirement statement in this document

8 Application layer

8.1 ISO 14229-1 service primitive parameters

This document is part of the ISO 14229 series and therefore, the service primitive parameter implementation follows the ISO 14229-1 specification.

REQ	**7.1 UDSonIP – ISO 14229-1 service primitive parameters**
The service primitive parameter shall be implemented as specified in ISO 14229-1.	

8.2 A_Data.req, A_Data.ind, and A_Data.conf service interface

This document is part of the ISO 14229 series and therefore, the service interface implementation follows the ISO 14229-1 specification.

REQ	**7.2 UDSonIP – `A_Data.req`, `A_Data.ind`, and `A_Data.conf` service interface**
The `A_Data.req`, `A_Data.ind`, and `A_Data.conf` service interface shall be implemented as specified in ISO 14229-1.	

8.3 UDSonIP services overview

The purpose of Table 2 is to reference all ISO 14229-1 and ISO 14229-2 services as they are applicable for an implementation in this document. Table 2 contains the UDSonIP diagnostic services. Certain UDSonIP applications can restrict the number of useable services and can categorize them in application areas/diagnostic sessions (default session, programming session, etc.).

REQ	7.3 UDSonIP – UDSonIP-specific requirements
Services that are marked “UDSonIP-specific requirements” shall be implemented as specified in the referenced subclause number in accordance with Table 2 "Reference" column.	

REQ	7.4 UDSonIP – No UDSonIP-specific requirements
Services specified in Table 2 that are marked “No UDSonIP-specific requirements” shall be implemented as specified in ISO 14229-1 and ISO 14229-2 with no additional restrictions.	

Table 2 — Overview of applicable ISO 14229-1-defined services

Functional unit name	Diagnostic service name	Comment	Reference
Diagnostic and communication management	DiagnosticSessionControl	UDSonP-specific requirements	see 8.5
	ECUReset	UDSonP-specific requirements	see 8.6
	SecurityAccess	No UDSonP-specific requirements	—
	CommunicationControl	No UDSonP-specific requirements	—
	TesterPresent	No UDSonP-specific requirements	—
	Authentication	No UDSonP-specific requirements	—
	SecuredDataTransmission	No UDSonP-specific requirements	—
	ControlDTCSetting	No UDSonP-specific requirements	—
	ResponseOnEvent	UDSonP-specific requirements	see 8.8
	LinkControl	No UDSonP-specific requirements	—
Data transmission	ReadDataByIdentifier	No UDSonP-specific requirements	—
	ReadMemoryByAddress	No UDSonP-specific requirements	—
	ReadScalingDataByIdentifier	No UDSonP-specific requirements	—
	ReadDataByPeriodicIdentifier	UDSonP-specific requirements	see 8.7
	DynamicallyDefineDataIdentifier	No UDSonP-specific requirements	—
	WriteDataByIdentifier	No UDSonP-specific requirements	—
	WriteMemoryByAddress	No UDSonP-specific requirements	—
Stored data transmission	ReadDTCInformation	No UDSonP-specific requirements	—
	ClearDiagnosticInformation	No UDSonP-specific requirements	—
Input/output control	InputOutputControlByIdentifier	No UDSonP-specific requirements	—
Remote activation of routine	RoutineControl	No UDSonP-specific requirements	—
Upload/ download	RequestDownload	No UDSonP-specific requirements	—
	RequestUpload	No UDSonP-specific requirements	—
	TransferData	No UDSonP-specific requirements	—
	RequestTransferExit	No UDSonP-specific requirements	—
	RequestFileTransfer	No UDSonP-specific requirements	—

8.4 A_PDU definition

8.4.1 Generic DoIP header of A_PDU

REQ	7.5 UDSonIP – Generic DoIP header of A_PDU
The A_PDU definition includes the generic DoIP header which shall be followed according to ISO 13400-2.	

8.4.2 A_PDU for UDS request and response message

The A_PDU of UDSonIP implements an ISO 13400-2 generic DoIP header and payload for UDS request and response messages.

<table>
<tr><th>REQ</th><th>7.6 UDSonIP – A_PDU for UDS request and response message</th></tr>
<tr><td colspan="2">Figure 3 specifies the UDSonIP A_PDU for UDS request and response message in accordance with ISO 13400-2, which shall be followed for all UDSonIP messages.</td></tr>
</table>

Figure 3 shows the A_PDU which consists of:

- protocol version;
- payload type: 8001_{16}, diagnostic message (request/response);
- payload length;
- logical source and target address;
- ISO 14229-1 service identifier;
- ISO 14229-1 data.

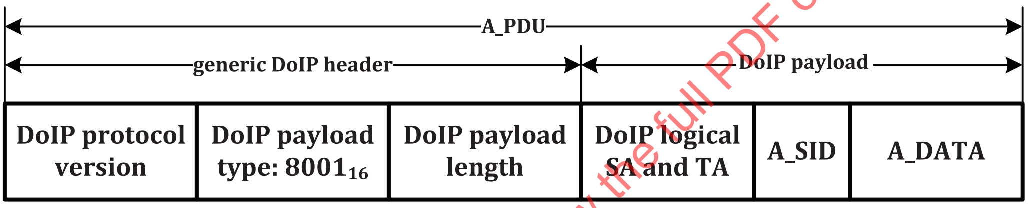


Figure 3 — A_PDU for UDS request and response message

8.4.3 A_PDU for UDS periodic response message

Periodic response messages are differentiated from non-periodic response messages with a specific DoIP payload type.

<table>
<tr><th>REQ</th><th>7.7 UDSonIP – A_PDU for UDS periodic response message</th></tr>
<tr><td colspan="2">Figure 4 specifies the UDSonIP A_PDU for UDS periodic response message in accordance with ISO 13400-2, which shall be followed for all UDSonIP messages.</td></tr>
</table>

Figure 4 shows the A_PDU which consists of:

- protocol version;
- payload type: 8004_{16}, diagnostic message (periodic response);
- payload length;
- logical source and target address;
- ISO 14229-1 periodic data identifier;
- ISO 14229-1 periodic data.

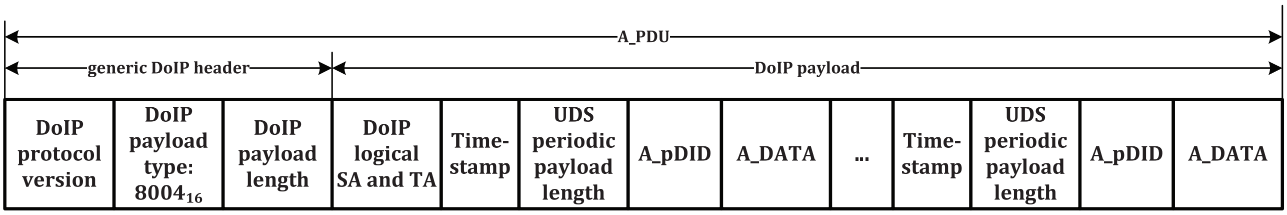


Figure 4 — A_PDU for UDS periodic response message

8.5 DiagnosticSessionControl service UDSonIP implementation requirements

8.5.1 General

This document specifies the specific UDSonIP DiagnosticSessionControl service implementation requirements and restrictions.

8.5.2 TCP connection handling

This requirement specifies the UDSonIP DiagnosticSessionControl service TCP handling if a connection is disconnected.

REQ	7.8 UDSonIP – DiagnosticSessionControl – TCP connection handling
If a TCP connection is disconnected due to a session change, a new TCP connection and routing activation as specified in ISO 13400-2 shall be performed before diagnostic communication is continued (see Figure 4).	

8.5.3 TCP connection closing

This requirement specifies the UDSonIP DiagnosticSessionControl service TCP connection closing.

REQ	7.9 UDSonIP – DiagnosticSessionControl – TCP connection closing
If the TCP connection is disconnected due to a session change, the server shall initiate the closing of the TCP connection as specified in IETF RFC 793:1981, 3.5 after sending a DiagnosticSessionControl positive response message and prior to the execution of the DiagnosticSessionControl service (see Figure 5).	

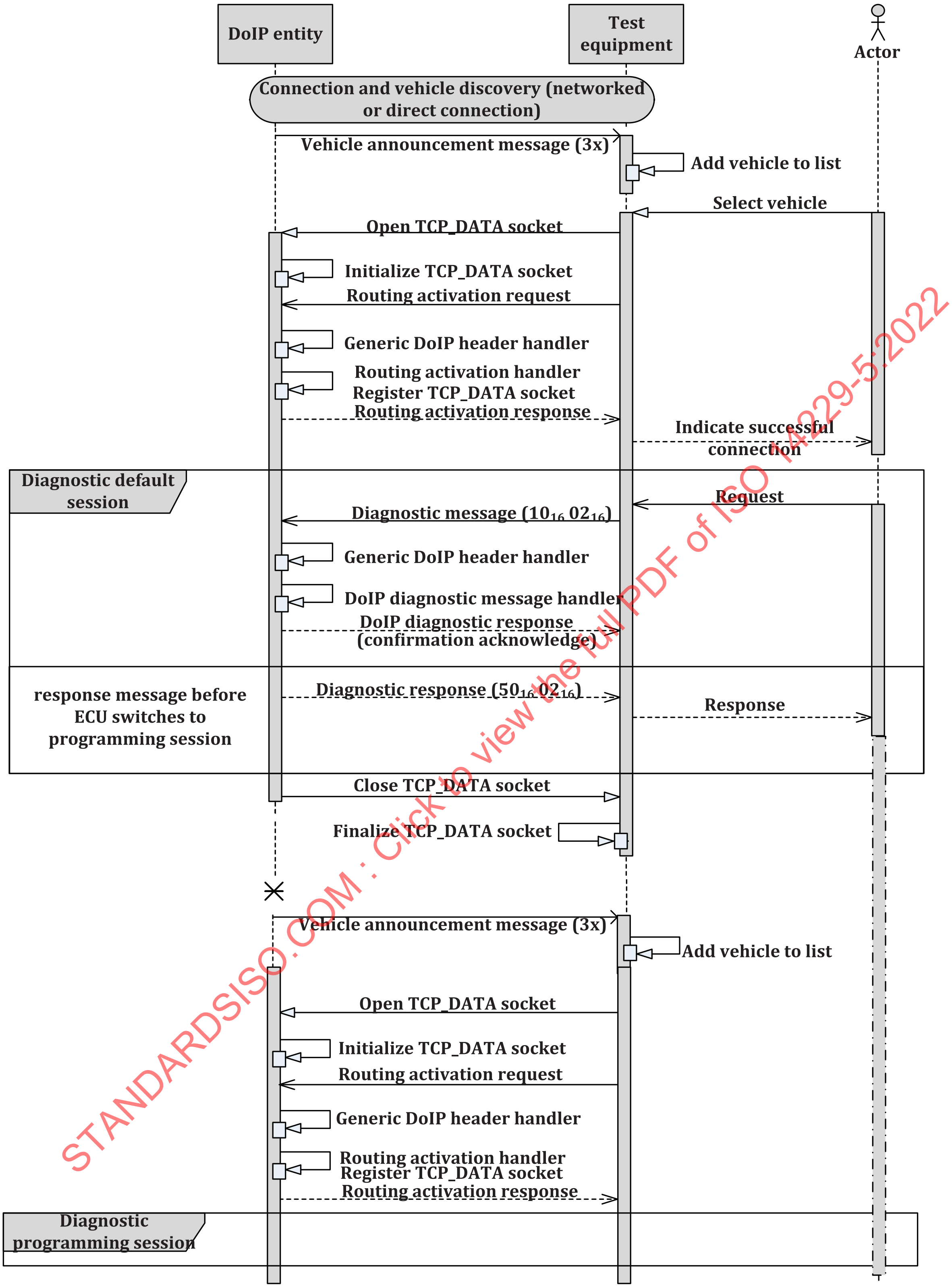

DoIP entity
Test equipment
Actor
Connection and vehicle discovery (networked or direct connection)
Vehicle announcement message (3x)
Add vehicle to list
Select vehicle
Open TCP_DATA socket
Initialize TCP_DATA socket
Routing activation request
Generic DoIP header handler
Routing activation handler
Register TCP_DATA socket
Routing activation response
Indicate successful connection
Diagnostic default session
Request
Diagnostic message (10_{16} 02_{16})
Generic DoIP header handler
DoIP diagnostic message handler
DoIP diagnostic response (confirmation acknowledge)
response message before ECU switches to programming session
Diagnostic response (50_{16} 02_{16})
Response
Close TCP_DATA socket
Finalize TCP_DATA socket
Vehicle announcement message (3x)
Add vehicle to list
Open TCP_DATA socket
Initialize TCP_DATA socket
Routing activation request
Generic DoIP header handler
Routing activation handler
Register TCP_DATA socket
Routing activation response
Diagnostic programming session

Figure 5 — Switch to programming session handling

8.6 ECUReset service UDSonIP implementation requirements

8.6.1 General

In addition to the generic implementation requirements stated in ISO 14229-1, the following requirements are applicable for an UDSonIP implementation.

8.6.2 TCP connection handling

This requirement specifies the UDSonIP ECUReset service TCP handling if a connection is disconnected.

<table>
<tr><th>REQ</th><th>7.10 UDSonIP – ECUReset TCP connection handling</th></tr>
<tr><td colspan="2">If a TCP connection is disconnected between the client and a server, a new TCP connection and routing activation, as specified in ISO 13400-2, shall be established with this server, where the ECUReset is applied.</td></tr>
</table>

8.6.3 TCP connection closing

This requirement specifies the UDSonIP ECUReset service TCP connection closing.

<table>
<tr><th>REQ</th><th>7.11 UDSonIP – ECUReset TCP connection closing</th></tr>
<tr><td colspan="2">The server shall initiate the closing of the TCP connection as specified in IETF RFC 793:1981, 3.5 after sending an ECUReset positive response message and prior to the execution of the ECUReset service.</td></tr>
</table>

8.7 ReadDataByPeriodicIdentifier service UDSonIP implementation requirements

8.7.1 General

This document specifies the specific UDSonIP ReadDataByPeriodicIdentifier service implementation requirements or restrictions.

8.7.2 Service interface

8.7.2.1 General

Service primitives are used to trigger the timing parameters of the application layer.

8.7.2.2 Service primitive – A_Data.req

The A_Data.req service primitive is issued by the sender node.

<table>
<tr><th>REQ</th><th>7.12 UDSonIP – ReadDataByPeriodicIdentifier – A_Data.req</th></tr>
<tr><td colspan="2">The service primitive shall request periodic transmission of <code>A_Data</code> with <code>A_Length</code> number of bytes from the sender to the receiver peer entities identified by the message type <code>A_Mtype</code> and address information in <code>A_TAtype</code>, <code>A_SA</code> and <code>A_TA</code>.</td></tr>
</table>

```
A_Data.req         (
                   A_Mtype,
                   A_SA,
                   A_TA,
                   A_TAtype,
                   A_Data[Data#1, Data#2, …, Data#n ],
                   A_Length,
                   )
```

8.7.2.3 Service primitive – A_Data.ind

The `A_Data.ind` service primitive is received by the receiver node.

REQ	**7.13 UDSonIP – ReadDataByPeriodicIdentifier – A_Data.ind**
The service primitive shall deliver `A_Data` with `A_Length` bytes received from a peer protocol entity identified by the message type `A_Mtype` and address information in `A_TAtype`, `A_SA` and `A_TA`.	

The parameters `A_Data` and `A_Length` are only valid when the service primitive is indicated. In case of a reception error no indication shall be generated.

```
A_Data.ind         (
                   A_Mtype,
                   A_SA,
                   A_TA,
                   A_TAtype,
                   A_Data[Data#1, Data#2, …, Data#n ],
                   A_Length,
                   )
```

8.7.2.4 Service primitive – A_Data.conf

The `A_Data.conf` service is received by the sender node.

REQ	**7.14 UDSonIP – ReadDataByPeriodicIdentifier – A_Data.conf**
The service primitive confirms the completion of a `A_Data.req` service identified by the message type `A_Mtype` and address information in `A_TAtype, A_SA` and `A_TA`. The parameter `A_Result` provides the status of the service request.	

```
A_Data.conf         (
                    A_Mtype,
                    A_SA,
                    A_TA,
                    A_TAtype,
                    A_Result,
                    A_Length,
                    )
```

8.7.3 Service primitive data types

The service primitive data types derive from ISO 14229-2.

REQ	**7.15 UDSonIP – ReadDataByPeriodicIdentifier – Service primitive parameters**
The data type definitions and the parameters `Mtype`, `TA`, `SA`, `Length`, `Data`, and `Result` shall be implemented as specified in ISO 14229-2.	

8.7.4 Periodic response message A_PDU format

One or more UDS periodic response A_PDUs of subnet server(s), connected to a DoIP gateway, can be mapped into a single UDSonIP periodic response message A_PDU.

REQ	**7.16 UDSonIP – ReadDataByPeriodicIdentifier – Periodic response message payload type**
A UDSonIP periodic response message A_PDU shall implement the payload type 8004_{16} and formatted as specified in ISO 13400-2.	

8.7.5 Periodic transmission response message handling

The usage of the periodic transmission response message considers a message size being supported for the entire in-vehicle network, which can consist of gateways and other connected data links than Internet Protocol.

REQ	**7.17 UDSonIP – ReadDataByPeriodicIdentifier – Periodic transmission response message handling**
The data record length of the message referenced by a periodicDataIdentifier (pDID) shall not exceed the length limitation of a non-segmented message of the UDSonIP protocol.	

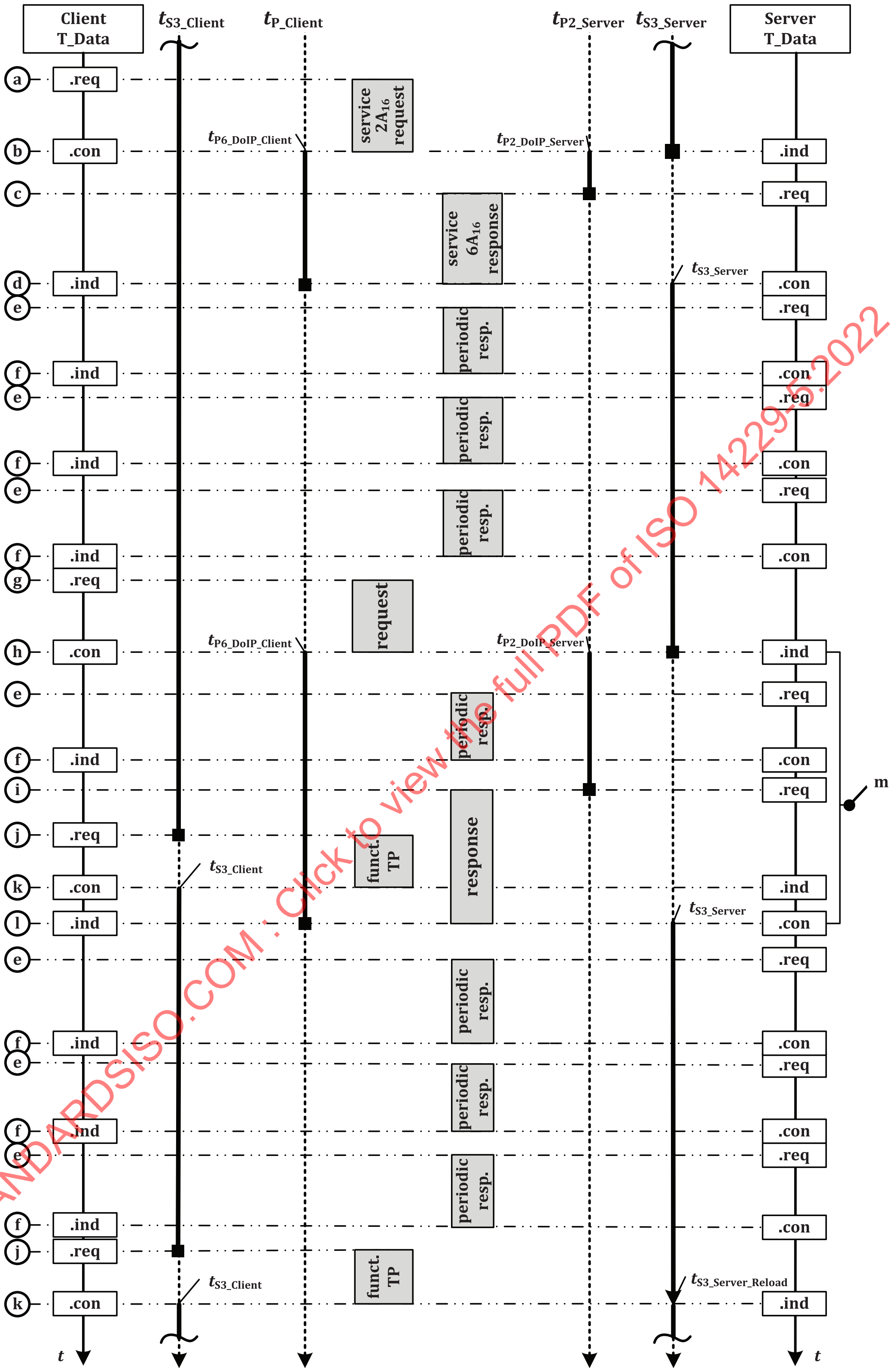
REQ	**7.18 UDSonIP – ReadDataByPeriodicIdentifier – Periodic transmission response message server restrictions**
Table 3 requirements shall be followed by the client and server for periodic transmission response messages.	

Table 3 — Periodic transmission — Requirements for periodic data response message mapping

Message type	Client request	Server response	Further server restrictions
Periodic data response message uses a different UDSonIP logical address (DoIP_AI) for periodic message transmissions	No restrictions	Only single UDSonIP response messages for periodic transmission	The request for periodic transmission is processed as a regular diagnostic request and the response is sent via the network layer (as a DoIP diagnostic message with service identifier $6A_{16}$).
			On receiving the `DoIP_Data.con` that indicates the completion of the transmission of the positive response, the application starts an independent scheduler, which handles the periodic transmission.
			The scheduler in the server processes the periodic transmission as a DoIP diagnostic message with an address (`DoIP_AI`) that is specific to periodic data responses. The address shall be chosen by the vehicle manufacturer to indicate a periodic response message.

8.7.6 Periodic transmission message flow

Figure 6 specifies the periodic transmission response messages handling and shows that the periodically transmitted response messages do not have any influence on the $t_{\mathtt{S3\_Server}}$ timer of the server. The ReadDataByPeriodicIdentifier service requires a non-defaultSession in order to be executed.



Key

a Client `T_Data.req`: the diagnostic application of the client starts the transmission of the ReadDataByPeriodicIdentifier ($2A_{16}$) request message by issuing a T_Data.req to its communication layer. The communication layer transmits the ReadDataByPeriodicIdentifier ($2A_{16}$) request message to the server.

b Client `T_Data.con`: the completion of the request message is indicated in the client via `T_Data.con`. Now the response timing as described in ISO 14229-2 applies.

Server T_Data.ind: the completion of the request message is indicated in the server via the T_Data.ind. Now the response timing as described in ISO 14229-2 applies. Furthermore, the server stops its $t_{\mathtt{S3\_Server}}$ timer.

c Server T_Data.req: it is assumed that the client requires a response from the server. The server transmits the ReadDataByPeriodicIdentifier positive response message to indicate that the request has been processed and that the transmission of the periodic messages starts afterwards.

d Server T_Data.con: the completion of the transmission of the ReadDataByPeriodicIdentifier response message is indicated in the server via T_Data.con. Now the server restarts its $t_{\mathtt{S3\_Server}}$ timer, which keeps the activated non-default session active as long as it does not time out.

Client T_Data.ind: the reception of the response message is indicated in the client.

e Server T_Data.req: the server starts to transmit the periodic response messages. The transmission of the periodic response message has no influence on the $t_{\mathtt{S3\_Server}}$ timer.

f Server T_Data.con: the completion of the transmission of the periodic response message is indicated in the server.

Client T_Data.ind: the completion of the reception of the periodic response message is indicated in the client.

g Client T_Data.req: the diagnostic application of the client starts the transmission of the next request message by issuing a T_Data.req to its communication layer. The communication layer transmits the request message to the server.

h Client T_Data.con: the completion of the request message is indicated in the client via T_Data.con. Now the response timing, as described in ISO 14229-2, applies.

Server T_Data.ind: the completion of the request message is indicated in the server via the T_Data.ind. Now the response timing, as described in ISO 14229-2, applies.

i Server T_Data.req: for the figure given, it is assumed that the client requires a response from the server. The server transmits the positive (or negative) response message by issuing a T_Data.req to its communication layer.

j Client T_Data.req: when the $t_{\mathtt{S3\_Client}}$ timer times out in the client, the client then transmits a functionally addressed TesterPresent ($\mathtt{3E}_{16}$) request message to reset the $t_{\mathtt{S3\_Server}}$ timer in the server.

k Server T_Data.ind: the server is in the process of transmitting the response of the previous request. Therefore, the server does not act on the received TesterPresent ($\mathtt{3E}_{16}$) request message because its $t_{\mathtt{S3\_Server}}$ timer is not yet re-activated.

Client T_Data.con: the reception of the TesterPresent ($\mathtt{3E}_{16}$) request message is indicated in the server.

l Server T_Data.con: when the diagnostic service is completely processed, the server then restarts its $t_{\mathtt{S3\_Server}}$ timer. This means that any diagnostic service, including TesterPresent ($\mathtt{3E}_{16}$), resets the $t_{\mathtt{S3\_Server}}$ timer. A diagnostic service is meant to be in progress any time between the reception of the request message (T_Data.ind receive) and the completion of the transmission of the response message, where a response message is required, or the completion of any action that is caused by the request, where no response message is required (point in time reached that would cause the start of the response message). This includes negative response message(s) including negative response code $\mathtt{78}_{16}$.

m Any TesterPresent request message that is received during a disabled $t_{\mathtt{S3\_Server}}$ timer is ignored by the server.

Figure 6 — Periodic transmission response message handling

8.8 ResponseOnEvent service UDSonIP implementation requirements

8.8.1 General

This document specifies the specific UDSonIP ResponseOnEvent service implementation requirements or restrictions.

8.8.2 Activated storageState

An activated ResponseOnEvent logic causes the server to automatically execute a diagnostic service in case a specified event occurs in the server. The event occurrence may happen at any time while the server is awake. When the storageState (eventType subfunction bit 6) is set to '$\mathtt{1}_2$' the event logic resumes sending serviceToRespondTo-(STRT-) responses according to the ResponseOnEvent-setup after a reset or power on.

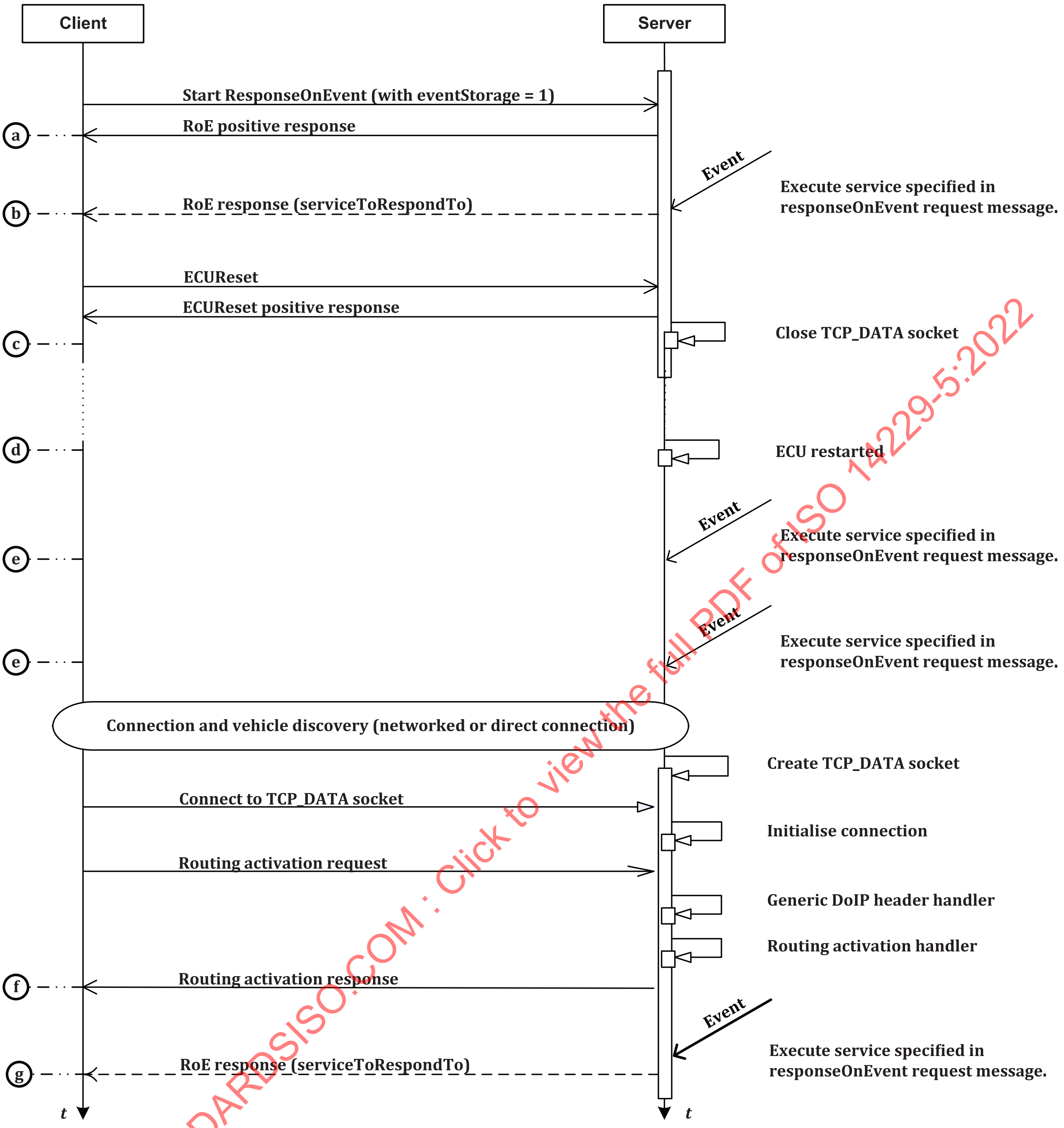
This behaviour on connection-less transport layer services is defined in ISO 15765-2 (DoCAN). This functionality is different on a connection-oriented transport layer. The transport layer specified in ISO 13400-2 is connection-oriented and limited to the client to establish the communication. As a result, unless the client has not actively established a connection, the server side cannot communicate on that transport layer.

In case the server is requested to send a STRT-response and no active connection on the transport layer service is established, the STRT-response cannot be sent. As a result, the message is not transmitted on the network, even though the event occurred in the server.

For the ResponseOnEvent service requested to send responses over UDSonIP the server shall treat the STRT-response provided by the application layer in a “fire-and-forget” manner. That means the ResponseOnEvent logic in the application layer defined by ISO 14229-1 runs independently from the connection state on the transport layer. Each occurring event triggers the processing of the STRT-response message (including all potential actions caused by this execution, e.g. ReadDataByIdentifier). The resulting response message is handed over to the DoIP transport layer. If and only if a DoIP connection from a client, that activated ResponseOnEvent logic is available, the DoIP transport layer sends the serviceToRespondTo-response message to the client. The ISO 14229-1 application layer discards a failed STRT-response (no retry) without further notice.

If the message is discarded it does not appear on the network and is not visible to any node connected to that network (e.g. network tracing and analysis tools).

Figure 7 graphically depicts the behaviour of an activated ResponseOnEvent logic in the server while events are occurring and the transport layer is in different connection states.


Client
Server
Start ResponseOnEvent (with eventStorage = 1)
a
RoE positive response
Event
Execute service specified in responseOnEvent request message.
b
RoE response (serviceToRespondTo)
ECUReset
ECUReset positive response
c
Close TCP_DATA socket
d
ECU restarted
Event
e
Execute service specified in responseOnEvent request message.
Event
e
Execute service specified in responseOnEvent request message.
Connection and vehicle discovery (networked or direct connection)
Create TCP_DATA socket
Connect to TCP_DATA socket
Initialise connection
Routing activation request
Generic DoIP header handler
Routing activation handler
f
Routing activation response
Event
Execute service specified in responseOnEvent request message.
g
RoE response (serviceToRespondTo)
t
t

Key

a The client activated the ResponseOnEvent logic in the server with the eventStorage bit set to 1. This results in keeping the activated ResponseOnEvent logic alive over power down or ECU reset until explicitly stopped by the client.

b The server detects an event matching the clients ResponseOnEvent setup conditions. The STRT service assigned to this event is processed in the server. The transport layer has an active connection and sends the response message to the client.

The server closes the transport layer connection after the processing of the EcuReset.

NOTE The TCP_IP socket also closes (lost) when the server is gracefully shut down or the power is shut off.

c The ECU is restarted and no client connection to the transport layer is yet established. The ResponseOnEvent logic in the server remains activated due to the eventStore bit set to 1 (see key a).

d The server detects an event matching the clients ResponseOnEvent setup conditions. The STRT service assigned to this event is processed in the server. The transport layer has no active connection to the client and discards the response message.

After sending the routing activation response of the DoIP (see ISO 13400-2), the client has established a transport layer connection with the server and the server is ready to send. From this point all events of the activated ResponseOnEvent logic can be sent to the client.

e The client activated the ResponseOnEvent logic in the server with the eventStorage bit set to 1. This results in keeping the activated ResponseOnEvent logic alive over power down or ECU reset until explicitly stopped by the client.

f The server detects an event matching the clients ResponseOnEvent setup conditions. The STRT service assigned to this event is processed in the server. The transport layer has an active connection and sends the response message to the client.

g The server detects an event matching the clients ResponseOnEvent setup conditions. The STRT service assigned to this event is processed in the server. The transport layer has an active connection and sends the response message to the client.

Figure 7 — RoE STRT transmissions depending on different transport layer connection states

8.9 Timing parameter definition

8.9.1 Request and response message timing parameter values

The request and response message timing parameters belong to the application layer.

REQ	**7.19 UDSonIP – Timing parameter definition**
The request and response message timing parameter values shall be implemented as specified in ISO 14229-2.	

8.9.2 Unsolicited response messages

Unsolicited messages are those transmitted by the server(s) based on either a periodic scheduler (see service ReadDataByPeriodicIdentifier in 8.7) or a configured trigger, such as a change of a DTC status or a dataIdentifier value change.

REQ	**7.20 UDSonIP – Unsolicited response messages**
Any unsolicited transmitted response message shall not reset the t_{S3_Server} timer in the server.	

This avoids a diagnostic session keep-alive latch-up effect in the server for cases where a periodic message transmission is active or a timer-triggered event is configured in the server where the time interval between the events is smaller than t_{S3_Server}. The t_{S3_Server} timer is only reset if the transmitted response message is the direct result of processing a request message and transmitting the final response message (such as the initial positive response that indicates that a request to schedule one or more periodicDataIdentifiers is performed).

9 Presentation layer

The presentation layer specification is not in the scope of this document.

10 Session layer

10.1 Service primitive parameter definition

This document is part of the ISO 14229 series and therefore, the service primitive parameter implementation follows the ISO 14229-2 specification.

REQ	**5.1 UDSonIP – Service primitive parameter definition**
The service primitive parameters shall be implemented as specified in ISO 14229-2.	

10.2 S_Data.req, S_Data.ind, and S_Data.conf service interface

This document is part of the ISO 14229 series and therefore, the service interface implementation follows the ISO 14229-2 specification.

REQ	5.2 UDSonIP – `S_Data.req`, `S_Data.ind`, and `S_Data.conf` service interface
The `S_Data.req`, `S_Data.ind`, and `S_Data.conf` service interface shall be implemented as specified in ISO 14229-2.	

11 Transport layer

11.1 Service primitive parameter definition

This document refers to ISO 13400-2 DoIP and therefore the service primitive parameters follow this specification.

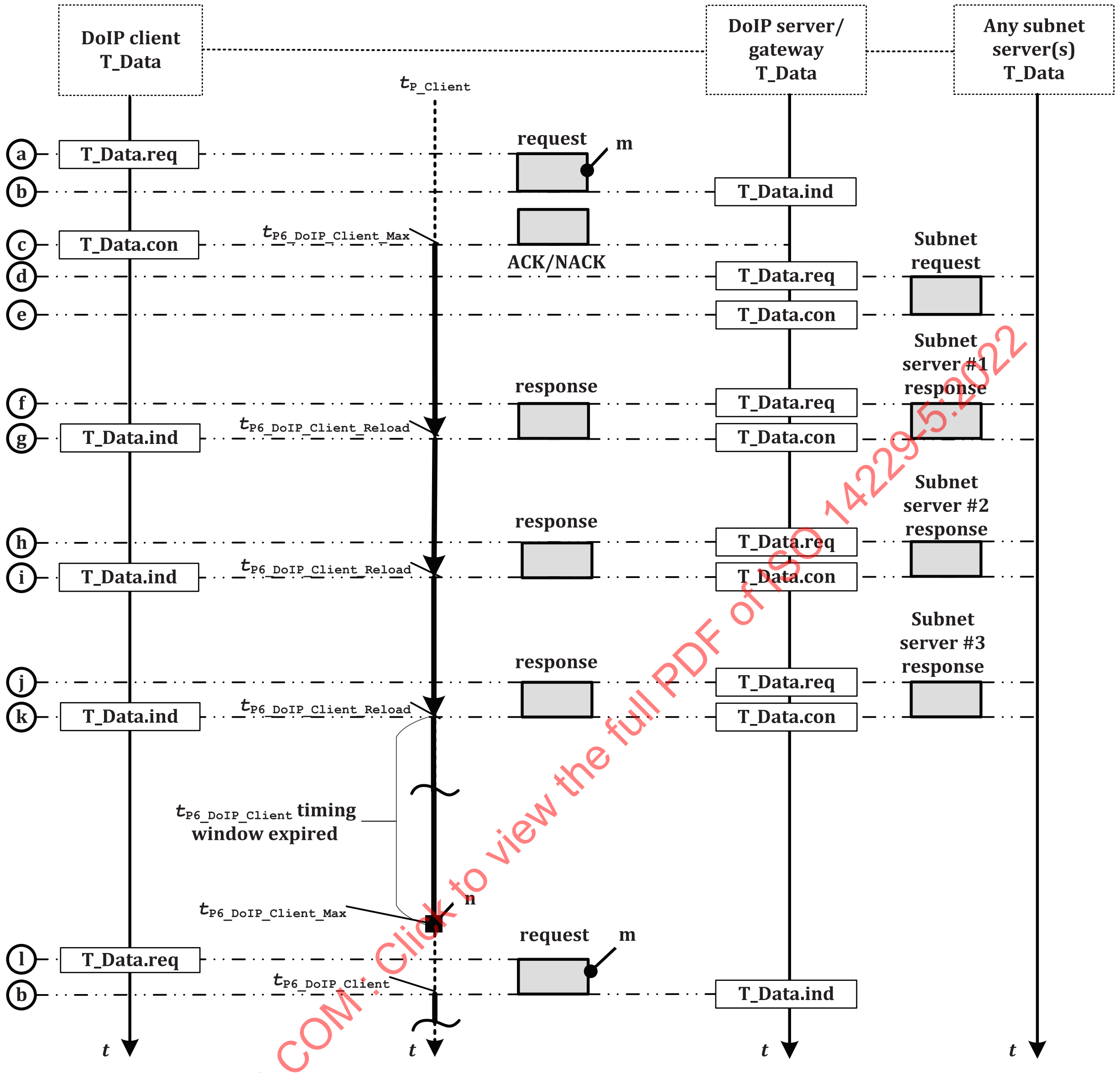
REQ	4.1 UDSonIP – Service primitive parameter definition
The service primitive parameters shall be implemented as specified in ISO 13400-2.	

11.2 T_Data.req, T_Data.ind, and T_Data.conf service interface

This document supports the service interface definition as specified in ISO 13400-2.

REQ	4.2 UDSonIP – `T_Data.req`, `T_Data.ind`, and `T_Data.conf` service interface
The `T_Data.req`, `T_Data.ind`, and `T_Data.conf` service interface implementation shall be implemented as specified in ISO 13400-2.	

Figure 8 specifies a request message (logical functional TA) with service primitives and timing. The client communicates request messages to the DoIP server/gateway which the DoIP server/gateway forwards onto the connected subnet(s). The addressed server(s) on the subnet transmit response messages to the DoIP server/gateway which the DoIP server/gateway transmits to the client.



Key

a Client `T_Data.req`: transport layer issues a request message with a logical TA = functional address to the DoIP network layer.

b DoIP server/gateway `T_Data.ind`: transport layer issues to DoIP server/gateway session layer the indication of the reception of a request message.

c Client `T_Data.con`: client network layer issues to transport layer the confirmation of the completion of the request message with a logical TA = functional address based on the reception of a DoIP ACK or NACK message. Client starts the $t_{\text{P\_Client}}$ timer with the value of $t_{\text{P6\_DoIP\_Client\_Max}}$.

d DoIP server/gateway subnet `T_Data.req`: DoIP server/gateway transport layer forwards the functionally addressed request message to the subnet transport layer by issuing a `T_Data.req`.

e DoIP server/gateway subnet `T_Data.con`: DoIP server/gateway network layer issues to the transport layer the confirmation of the completion of the request message.

f DoIP server/gateway receives a response message from the subnet server #1.

DoIP server/gateway `DoIP_Data.req`: DoIP server/gateway transport layer forwards the response message by issuing a `T_Data.req` to the DoIP network layer.

g DoIP server/gateway `DoIP T_Data.con`: DoIP server/gateway network layer issues to DoIP server/gateway transport layer the confirmation of the completion of the response message.

Client `T_Data.ind`: client network layer issues to transport layer the indication of the reception of the response message. Client reloads the $t_{\text{P\_Client}}$ timer with the value of $t_{\text{P6\_DoIP\_Client\_Max}}$.

h DoIP server/gateway receives a response message from the subnet server #2.

DoIP server/gateway `DoIP_Data.req`: DoIP server/gateway transport layer forwards the response message by issuing a `T_Data.req` to the DoIP network layer.

i Server/gateway DoIP `T_Data.con`: DoIP server/gateway network layer issues to DoIP server/gateway transport layer the confirmation of the completion of the response message.

Client `T_Data.ind`: client network layer issues to transport layer the indication of the reception of the response message. Client reloads the $t_{\mathtt{P\_Client}}$ timer with the value of $t_{\mathtt{P6\_DoIP\_Client\_Max}}$.

j DoIP server/gateway receives a response message from the subnet server #3.

DoIP server/gateway `DoIP_Data.req`: DoIP server/gateway transport layer forwards the response message by issuing a `T_Data.req` to the DoIP network layer.

k Server/gateway DoIP `T_Data.con`: DoIP server/gateway network layer issues to DoIP server/gateway transport layer the confirmation of the completion of the response message.

Client `T_Data.ind`: client network layer issues to transport layer the indication of the reception of the response message. Client reloads the $t_{\mathtt{P\_Client}}$ timer with the value of $t_{\mathtt{P6\_DoIP\_Client\_Max}}$.

The $t_{\mathtt{P\_Client}}$ timer expires when it reaches $t_{\mathtt{P6\_DoIP\_Client\_Max}}$ (key b) indicating that no more responses are forthcoming.

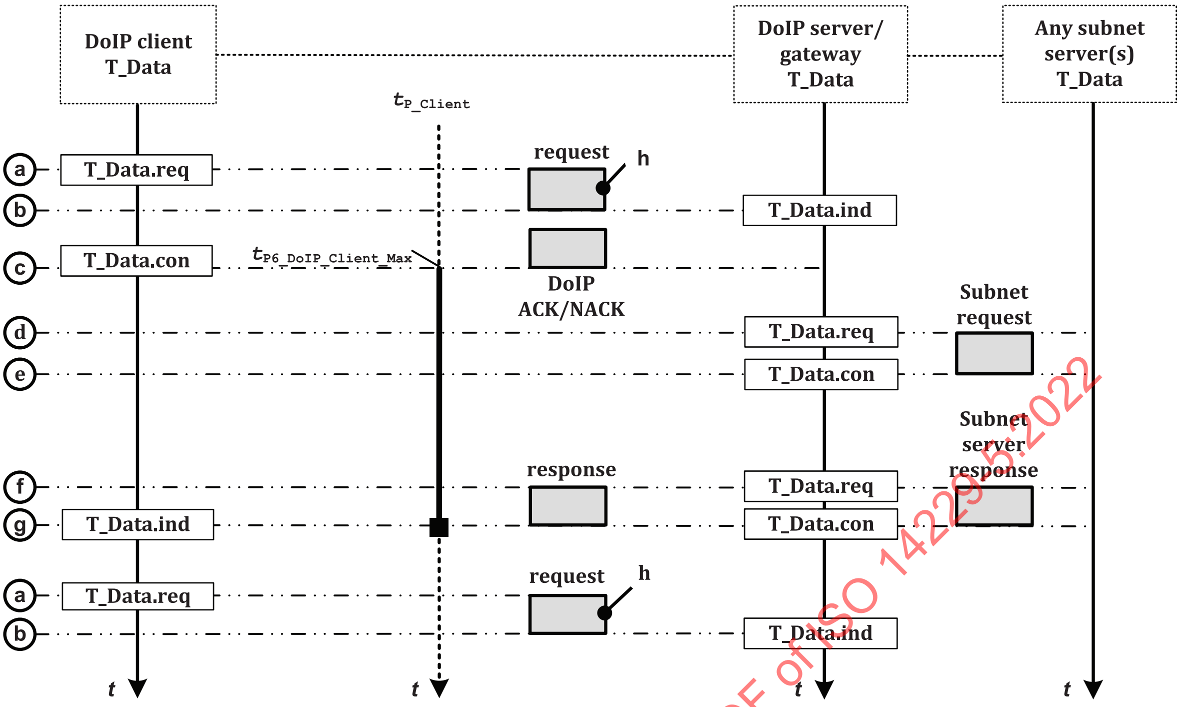
l Client `T_Data.req`: transport layer issues a request message with a logical TA = physical address to the DoIP network layer.

m DoIP request message with a logical TA = functional address.

n Client: $t_{\mathtt{P6\_DoIP\_Client\_Max}}$ timeout.

Figure 8 — Request message (logical functional TA) with service primitives and timing

Figure 9 specifies a request message (logical physical TA) with service primitives and timing. The client communicates all request messages to the DoIP server/gateway which the DoIP server/gateway forwards onto the connected subnet(s). The addressed server on the subnet transmits the response message to the DoIP server/gateway which the DoIP server/gateway transmits to the client.



Key

a Client `T_Data.req`: transport layer issues a request message with a logical TA = physical address to the DoIP network layer.

b DoIP server/gateway `T_Data.ind`: transport layer issues to DoIP server/gateway session layer the indication of the reception of a request message.

c Client `T_Data.con`: client network layer issues to transport layer the confirmation of the completion of the request message with a logical TA = functional address based on the reception of a DoIP ACK or NACK message. Client starts the $t_{\mathtt{P\_Client}}$ timer with the value of $t_{\mathtt{P6\_DoIP\_Client\_Max}}$.

d DoIP server/gateway subnet `T_Data.req`: DoIP server/gateway transport layer forwards the functionally addressed request message to the subnet transport layer by issuing a `T_Data.req`.

e DoIP server/gateway subnet `T_Data.con`: DoIP server/gateway network layer issues to the transport layer the confirmation of the completion of the request message.

f DoIP server/gateway receives a response message from the subnet server.

DoIP server/gateway `DoIP_Data.req`: DoIP server/gateway transport layer forwards the response message by issuing a `T_Data.req` to the DoIP network layer.

g DoIP server/gateway `DoIP T_Data.con`: DoIP server/gateway network layer issues to DoIP server/gateway transport layer the confirmation of the completion of the response message.

Client `T_Data.ind`: client network layer issues to transport layer the indication of the reception of the response message. Client stops the $t_{\mathtt{P\_Client}}$ timer.

h DoIP request message with a logical TA = physical address.

Figure 9 — Request message (logical physical TA) with service primitives and timing

11.3 T_PDU definition

Figure 10 shows the T_PDU which consists of:

- TCP/UDP header;
- protocol version;
- payload type;
- payload length;