INTERNATIONAL STANDARD

ISO 14229-3

Second edition
2022-03

Road vehicles — Unified diagnostic services (UDS) —

Part 3: Unified diagnostic services on CAN implementation (UDSonCAN)

Véhicules routiers — Services de diagnostic unifiés (SDU) —

Partie 3: SDU sur l'implémentation du gestionnaire de réseau de communication (SDUsurCAN)

Reference number
ISO 14229-3:2022(E)

COPYRIGHT PROTECTED DOCUMENT

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

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 14229-3:2012), 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 UDSonCAN document and related documents according to the OSI model.

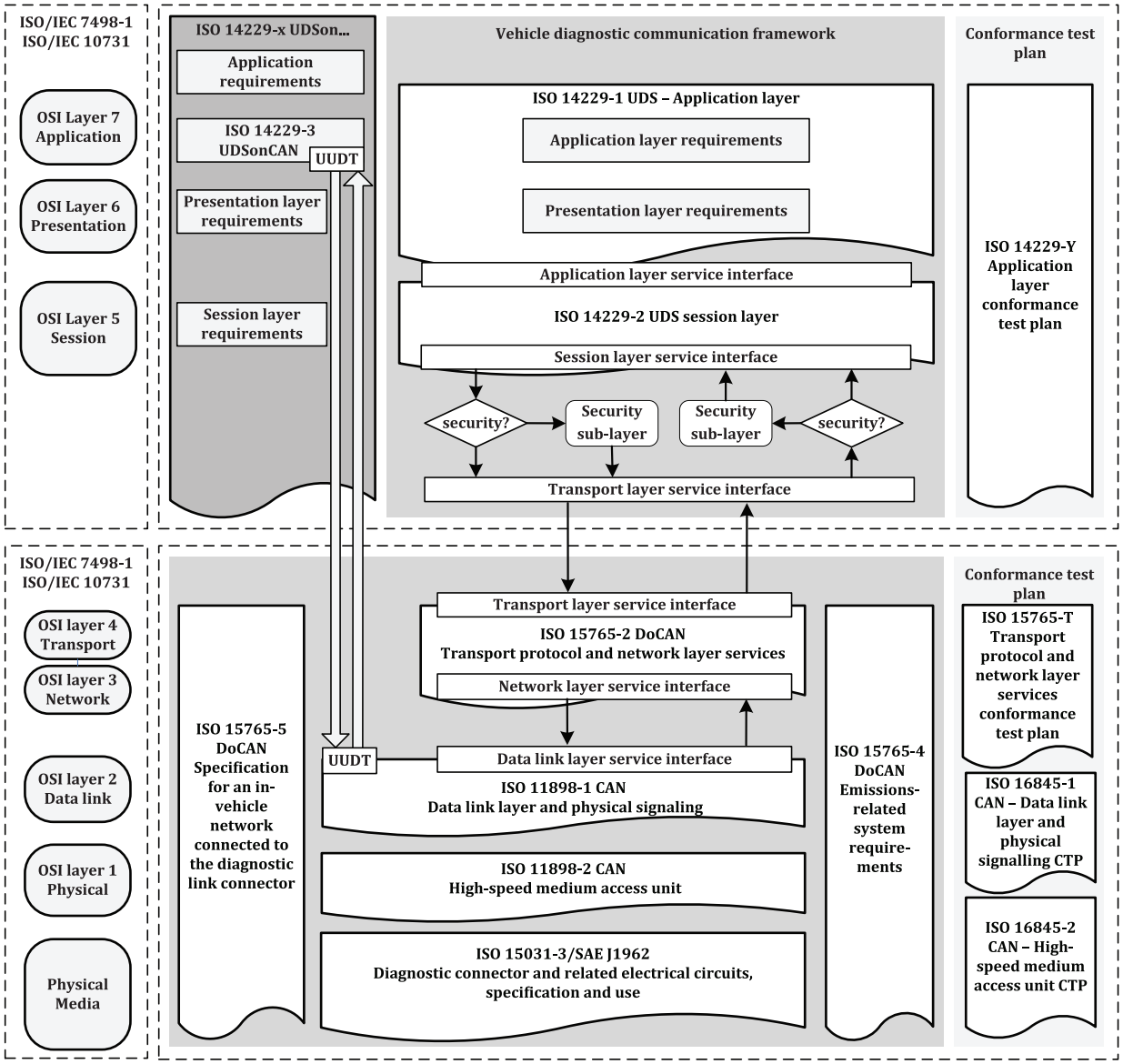


Figure 1 — ISO 14229-3 document reference according to OSI model

Road vehicles — Unified diagnostic services (UDS) —

Part 3: Unified diagnostic services on CAN implementation (UDSonCAN)

1 Scope

This document specifies an application profile for the implementation of unified diagnostic services (UDS) on controller area network (CAN) in road vehicles.

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

This document specifies

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

— specific restrictions in the implementation of UDS on the CAN 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 11898-1, *Road vehicles — Controller area network (CAN) — Part 1: Data link layer and physical signalling*

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*

ISO 15765-2, *Road vehicles — Diagnostic communication over Controller Area Network (DoCAN) — Part 2: Transport protocol and network layer services*

ISO 15765-5, *Road vehicles — Diagnostic communication over Controller Area Network (DoCAN) — Part 5: Specification for an in-vehicle network connected to the diagnostic link connector*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 14229-1, ISO 14229-2, ISO 15765-2, and ISO 15765-5 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_{P_Client}	client application layer timer
t_{P2_Server}	server application layer timer
$t_{P2\_CAN\_Client}$	client application layer timeout value for CAN
$t_{P2\_CAN\_Server}$	server application layer timeout value for CAN
t_{S3_Client}	client session layer timer
t_{S3_Server}	server session layer timer
$t_{S3\_Server\_Reload}$	server session layer timeout-reload value

4.2 Abbreviated terms

CAN	Controller Area Network
CBFF	classical base frame format
CEFF	classical extended frame format
DA	destination address
DLC	data length code
FBFF	FD base frame format
FEFF	FD extended frame format
IVN	in-vehicle network
PCI	protocol control information
SA	source address
SId	service identifier
SOM	start of message
STRT	serviceToRespondTo
TA	target address
UDS	unified diagnostic services
USDT	unacknowledged segmented data transfer
UUDT	unacknowledged unsegmented data transfer

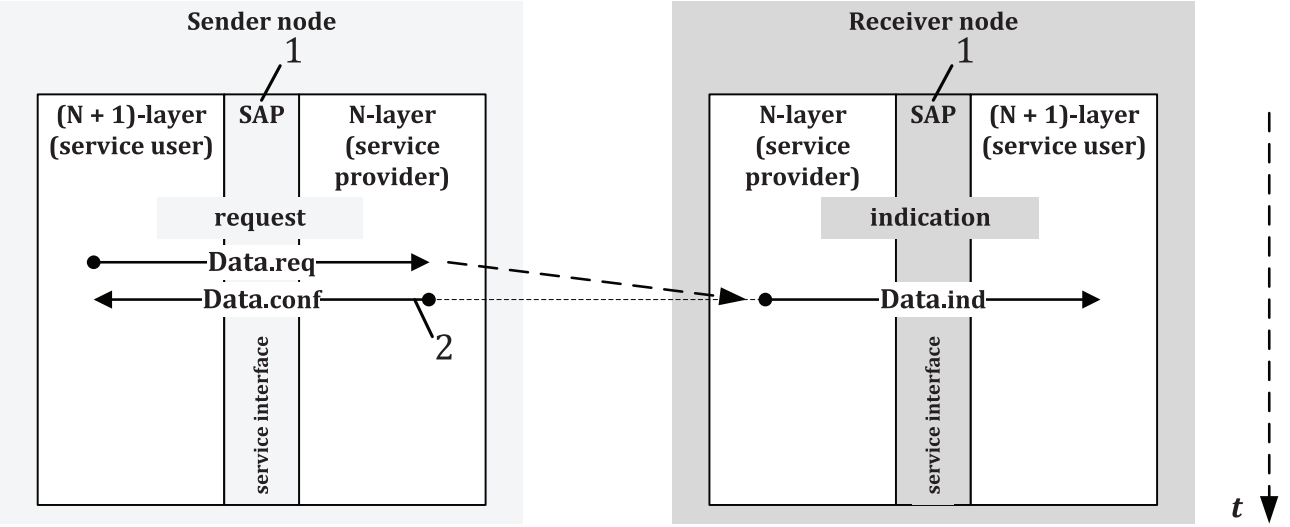
5 Conventions

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 primitive 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 about technical requirements and associated requirement numbers.

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	UDSonCAN specific requirements
7.4	No UDSonCAN-specific requirements
7.5	UUDT periodic transmission response message handling
7.6	UUDT periodic transmission response message server restrictions
7.7	UUDT OSI-layer support
7.8	Service primitive – A_UUData.req
7.9	Service primitive – A_UUData.ind
7.10	Service primitive – A_UUData.conf
7.11	General UUDT service primitive parameters
7.12	Specific UUDT service primitive parameters

Table 1 *(continued)*

OSI#.REQ#	Technical requirement title
7.13	UUDT message format
7.14	UUDT A_PDU size
7.15	Request and response message timing parameter values
7.16	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	USDT service primitive parameters
4.2	T_Data.req, T_Data.ind, and T_Data.conf service interface
3	**Network layer**
3.1	Service primitive parameter definition
3.2	N_Data.req, N_Data.ind, and N_Data.conf service interface
3.4	N_TAtype service primitive parameter
3.5	Same N_TAtype request and associated response message format
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

REQ	7.1 UDSonCAN – 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 UDSonCAN – 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 UDSonCAN services overview

The purpose of Table 2 is to reference ISO 14229-1 and ISO 14229-2 services as they are applicable for an implementation in this document. Table 2 contains the UDSonCAN applicable diagnostic services. Certain UDSonCAN 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 UDSonCAN specific requirements

Services that are marked "UDSonCAN-specific requirements" shall be implemented as specified in the referenced subclause number in accordance with Table 2 "Reference" column.

REQ	7.4 No UDSonCAN-specific requirements
Services specified in Table 2 that are marked "No UDSonCAN-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	No UDSonCAN-specific requirements	—
	ECUReset	No UDSonCAN-specific requirements	—
	SecurityAccess	No UDSonCAN-specific requirements	—
	CommunicationControl	No UDSonCAN-specific requirements	—
	TesterPresent	No UDSonCAN-specific requirements	—
	AccessTimingParameters	Not supported	—
	Authentication	No UDSonCAN-specific requirements	—
	SecuredDataTransmission	No UDSonCAN-specific requirements	—
	ControlDTCSetting	No UDSonCAN-specific requirements	—
	ResponseOnEvent	No UDSonCAN-specific requirements	—
	LinkControl	No UDSonCAN-specific requirements	—
Data transmission	ReadDataByIdentifier	No UDSonCAN-specific requirements	—
	ReadMemoryByAddress	No UDSonCAN-specific requirements	—
	ReadScalingDataByIdentifier	No UDSonCAN-specific requirements	—
	ReadDataByPeriodicIdentifier	UDSonCAN-specific requirements	see 8.5
	DynamicallyDefineDataIdentifier	No UDSonCAN-specific requirements	—
	WriteDataByIdentifier	No UDSonCAN-specific requirements	—
	WriteMemoryByAddress	No UDSonCAN-specific requirements	—
Stored data transmission	ReadDTCInformation	No UDSonCAN-specific requirements	—
	ClearDiagnosticInformation	No UDSonCAN-specific requirements	—
Input/output control	InputOutputControlByIdentifier	No UDSonCAN-specific requirements	—
Remote activation of routine	RoutineControl	No UDSonCAN-specific requirements	—
Upload/ download	RequestDownload	No UDSonCAN-specific requirements	—
	RequestUpload	No UDSonCAN-specific requirements	—
	TransferData	No UDSonCAN-specific requirements	—
	RequestTransferExit	No UDSonCAN-specific requirements	—
	RequestFileTransfer	No UDSonCAN-specific requirements	—

8.4 A_PDU definition

Figure 3 shows the A_PDU.

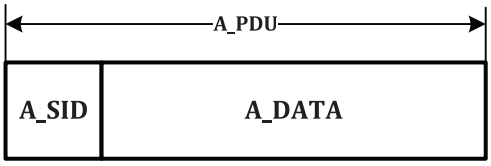


Figure 3 — A_PDU definition

8.5 ReadDataByPeriodicIdentifier service UDSonCAN implementation requirements

8.5.1 UUDT periodic transmission response message handling

The periodic transmission response message is based on a message size which is supported by the in-vehicle network (IVN). The IVN can consist of gateways and other connected data links than CAN.

REQ	7.5 UDSonCAN – ReadDataByPeriodicIdentifier – UUDT periodic transmission response message handling
The A_PDU length referenced by a periodicDataIdentifier (pDID) shall not exceed the length limitation of a non-segmented message.	

REQ	7.6 UDSonCAN – ReadDataByPeriodicIdentifier – UUDT periodic transmission response message server restrictions
Client receiving periodic response messages shall support the message mapping as specified in Table 3.	
Server transmitting periodic response messages shall support the message mapping as specified in Table 3.	

Table 3 — Periodic transmission response message mapping

Message type	Client request requirements	Server response requirements	Further server restrictions
A UUDT message shall use a different CAN identifier for periodic transmission.	No restrictions	1. Only single-frame responses for periodic transmission 2. Multi-frame responses to new (non-periodic-transmission) requests are possible	The request for periodic transmission is processed as a regular diagnostic request and the response is sent via the network layer (as a USDT message with service identifier $6A_{16}$).
			On receiving the `N_USData.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 single CAN frame response message in a bypass (i.e. writes the message directly to the CAN-controller/data link layer driver without using the network-layer).
			There is neither a protocol control information (PCI) nor a service identifier (SId) included in the response message. Only the periodic identifier and corresponding data are included.

8.5.2 Service interface – UUDT

8.5.2.1 General

The unconfirmed unsegmented data transfer (UUDT) is a response message type. UUDT messages are communicated between the application layer and the data link layer. UUDT periodic response

messages neither support the service identifier information (application layer) nor the protocol control information (transport layer).

REQ	7.7 UDSonCAN – ReadDataByPeriodicIdentifier – UUDT OSI-layer support
UUDT messages shall bypass the session layer, transport layer, and the network layer and implemented in the data link layer.	

8.5.2.2 Service primitive – A_UUData.req

The `A_UUData.req` service primitive is issued by the sender node.

REQ	7.8 UDSonCAN – UUDT service primitive – A_UUData.req
The service primitives shall request periodic transmission of `A_Data` with `A_Length` number of bytes from the sender to the receiver peer entities identified by the message type `A_Mtype` and address information in `A_UUTAtype`, `A_SA` and `A_TA`.	

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

8.5.2.3 Service primitive – A_UUData.ind

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

REQ	7.9 UDSonCAN – UUDT service primitive – A_UUData.ind
The service primitives 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_UUTAtype`, `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 is generated.

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

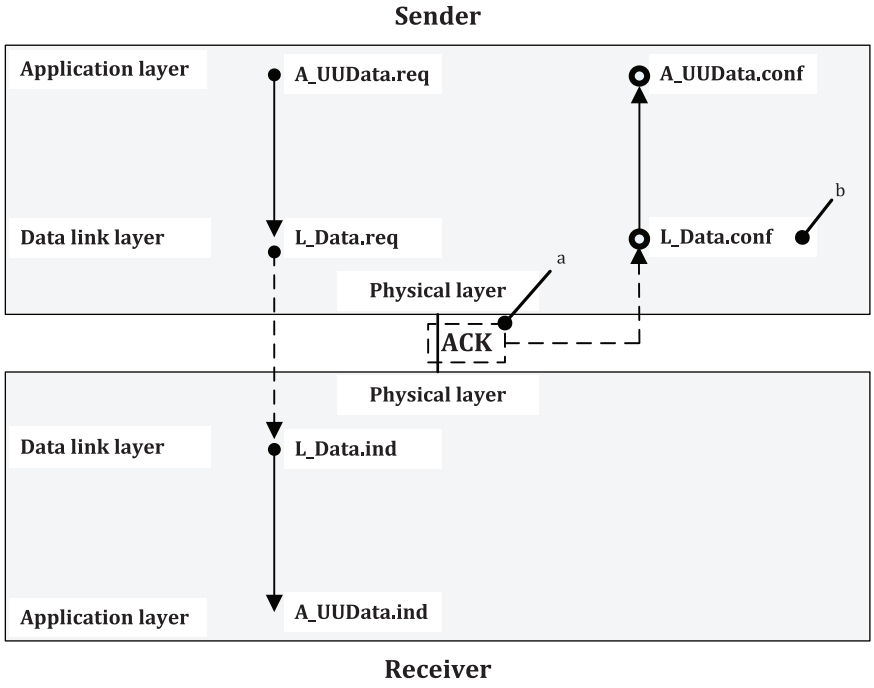
8.5.2.4 Service primitive – A_UUData.conf

The A_UUData.conf service is received by the sender node.

REQ	7.10 UDSonCAN – UUDT service primitive – A_UUData.conf
The service primitives shall confirm the completion of a A_UUData.req service identified by the message type A_Mtype and address information in A_UUTAtype, A_SA, and A_TA. The parameter A_Result provides the status of the service request.	

```
A_UUData.conf    (
                 A_Mtype,
                 A_SA,
                 A_TA,
                 A_UUTAtype,
                 A_Result
                 A_Length,
                 )
```

Figure 4 shows the UUDT service primitive interface. The sender node implementation of L_Data.conf and A_UUData.conf is not in the scope of this document.



a CAN ACK (acknowledge) bit is set by a receiver node and indicated to the sender.

b The data link layer of the sender node initiates a L_Data.conf service primitive to indicate to the application layer.

Figure 4 — UUDT service primitive interface

8.5.3 UUDT service primitive parameters

8.5.3.1 Service primitive data types

UUDT messages are exchanged directly between the application layer and the data link layer.

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

REQ	7.11 UDSonCAN – ReadDataByPeriodicIdentifier – General UUDT 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.	

8.5.3.2 UUDT specific parameters

The parameter `A_UUTAtype` is UUDT-specific and is a configuration attribute to the `A_TA` parameter. It is used to encode the UUDT communication model used by the communicating peer entities.

REQ	7.12 UDSonCAN – ReadDataByPeriodicIdentifier – Specific UUDT service primitive parameters
The `A_UUTAtype` parameter shall be of data type `Enum` (see ISO 14229-2) and is used to identify the UUDT target address type. Range: [`A_UUTAtype` #1: physical addressing, UUDT classical base frame format (CBFF), 11-bit CAN identifier, `A_UUTAtype` #2: physical addressing, UUDT FD base frame format (FBFF), 11-bit CAN identifier, `A_UUTAtype` #3: physical addressing, UUDT classical extended frame format (CEFF), 29-bit CAN identifier, `A_UUTAtype` #4: physical addressing, UUDT FD extended frame format (FEFF), 29-bit CAN identifier]	

8.5.4 UUDT message format

8.5.4.1 General

A UUDT `A_PDU` consists of three fields.

REQ	7.13 UDSonCAN – ReadDataByPeriodicIdentifier – UUDT message format
A UUDT message shall be implemented as specified in Table 4.	

Table 4 — UUDT message (A_PDU) format

UUDT message identifer	Data field
periodicDataIdentifier	dataRecord

8.5.4.2 UUDT address information

The `A_AI(A_TA, A_SA)` is provided by the service primitive interface and is used to identify the communicating peer entities for the periodic message (`A_PDU`).

8.5.4.3 UUDT length information

The length information of the A_PDU is provided by the service primitive interface and is used to identify the length of the UUDT message.

REQ	7.14 UDSonCAN – ReadDataByPeriodicIdentifier – UUDT A_PDU size
The maximum size of the `A_PDU` data field of the UUDT message shall be in accordance with the parameter `TX_DL` as specified in ISO 15765-2.	

8.5.4.4 UUDT message identifier

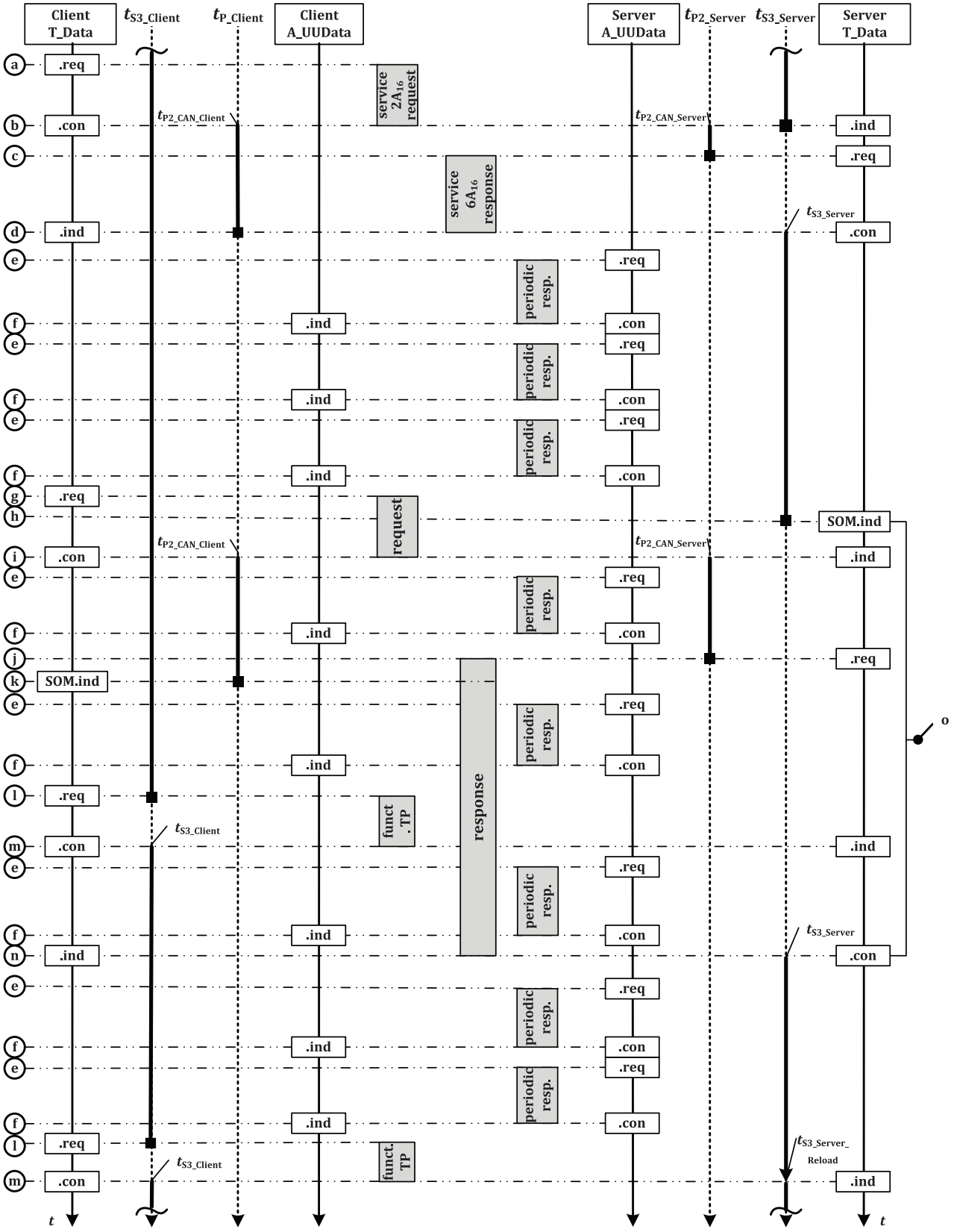
The UUDT message identifier contains the periodicDataIdentifier (pDID).

8.5.4.5 UUDT data field information

The data field contains the dataRecord(s) which corresponds to the periodicDataIdentifier (pDID).

8.5.5 Periodic transmission message flow

Figure 5 specifies the periodic response messages handling and shows that the periodically transmitted response messages do not have any influence on the t_{S3_Server} timer of the server. For this figure it is assumed that a non-defaultSession has been activated prior to the configuration of the periodic scheduler (the ReadDataByPeriodicIdentifier service requires a non-defaultSession in order to be executed).



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 transport/network layer. The application layer transmits the ReadDataByPeriodicIdentifier ($2A_{16}$) request message to the server. The request message can either be a segmented or unsegmented message (depends on the value of the periodicDataIdentifier contained in the request message). For the example given, it is assumed that the request message is less or equal eight byte.

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 specified 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 specified in ISO 14229-2 applies. Furthermore, the server stops its t_{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.

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_{S3_Server} timer, which keeps the activated non-default session active as long as it does not time out. The transmission of the periodic messages is started.

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

e Server A_UUData.req: the server starts to transmit the periodic response messages. Each periodic message, which neither includes any PCI information nor service identification, uses a different source address than USDT response messages. Therefore, the server issues a A_UUData.req each time a periodic message is transmitted independent of any other service currently processed by the server. This means, that the transmission of the periodic response messages continues, even when the server is in the process of handling another diagnostic service request. The transmission of the periodic response messages has no influence on the t_{S3_Server} timer.

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

Client A_UUData.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. The request message can either be an unsegmented or segmented message. For the example given, it is assumed that the request message is a multi-frame message.

h Server T_Data_SOM.ind: the start of a request message is indicated in the server via T_Data_SOM.ind while a periodic scheduler is active. The server does not stop the periodic scheduler for the duration of processing the received request message. This means that the server transmits further periodic messages for the duration of processing the diagnostic service. The client is aware of receiving these periodic response messages. Furthermore, any time the server is in the process of handling any diagnostic service it stops its t_{S3_Server} timer.

i 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 multi-frame request message is indicated in the server via the T_Data.ind. Now the response timing as described in ISO 14229-2 applies.

j Server T_Data.req: it is assumed that the client requires a response from the server. The server transmits the positive (or negative) response message via issuing a T_Data.req to its transport/network layer. In this example it is assumed that the response is a multi-frame message. While the multi-frame response message is transmitted by the transport/network layer, the periodic scheduler continues to transmit the periodic response messages.

k Client T_Data_SOM.ind: the start of the response message is indicated in the client.

l Client T_Data.req: when the tS3_Client timer times out in the client, then the client transmits a functionally addressed TesterPresent ($3E_{16}$) request message to restart the t_{S3_Server} timer in the server.

m Client T_Data.con: the reception of the TesterPresent ($3E_{16}$) request message is indicated in the client.

Server T_Data.ind: the server is in the process of transmitting the multi-frame response message of the previous request. Therefore, the server does not act on the received TesterPresent ($3E_{16}$) request message, because its t_{S3_Server} timer is not yet re-activated.

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

Server T_Data.con: when the diagnostic service is completely processed, then the server restarts its t_{S3_Server} timer. This means that any diagnostic service, including TesterPresent ($3E_{16}$), restarts the t_{S3_Server} timer. A diagnostic service is meant to be in progress any time between the start of the reception of the request message (T_Data_SOM.ind or 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 messages including response code 78_{16}.

- Any TesterPresent that is received during a disabled t_{S3_Server} timer is ignored by the server.

Figure 5 — Periodic transmission response message handling

8.6 Timing parameter definition

8.6.1 Request and response message timing parameter values

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

REQ	7.15 UDSonCAN – Request and response message timing parameter values
The request and response message timing parameter values shall be implemented as specified in ISO 14229-2.	

8.6.2 Unsolicited response messages

Unsolicited messages are those transmitted by the server(s) based on either a periodic scheduler (see service ReadDataByPeriodicIdentifier in 8.5) or a configured trigger, such as a change of a DTC status or a dataIdentifier value change.

REQ	7.16 UDSonCAN – 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 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

REQ	5.1 UDSonCAN – 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 UDSonCAN – 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 USDT service primitive parameters

REQ	4.1 UDSonCAN – USDT service primitive parameters
The service primitive parameters shall be implemented as specified in ISO 15765-5.	

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 15765-5.

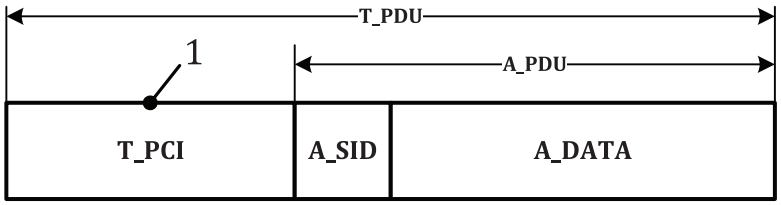
REQ	4.2 UDSonCAN – T_Data.req, T_Data.ind, and T_Data.conf service interface
The `T_Data.req`, `T_Data.ind`, and `T_Data.conf` service interface implementation shall be implemented as specified in ISO 15765-5.	

11.3 Transport protocol

REQ	4.3 UDSonCAN – Transport protocol
The transport protocol shall be implemented as specified in ISO 15765-2.	

11.4 T_PDU definition

Figure 6 shows the T_PDU.



Key

1 transport layer protocol control information

Figure 6 — T_PDU definition

11.5 DoCAN transport and network layer interface adaptation

11.5.1 Mapping of data link independent service primitives onto CAN data link-dependent service primitives

REQ	4.4 UDSonCAN – Mapping of data link independent service primitives onto CAN data link-dependent service primitives
The parameter mapping interface shall be implemented as specified in ISO 15765-2, DoCAN transport protocol and network layer services, and the session layer services as specified in ISO 14229-2 for the transmission and reception of diagnostic messages. The parameter mapping interface shall be implemented as specified in Table 5.	

Table 5 — Mapping of T_PDU service primitives onto N_PDU service primitives

Session to transport layer service primitives (data link independent according to ISO 14229-2)	DoCAN network layer service primitives (data link dependent according to ISO 15765-2)
T_Data.request	N_USData.request
T_DataSOM.indication	N_USDataFF.indication
T_Data.indication	N_USData.indication
T_Data.confirm	N_USData.confirm

11.5.2 Mapping of T_PDU onto N_PDU

The service primitive parameters of the application layer PDU defined to request the transmission of a diagnostic service request message (client) and response message (server) are mapped onto the parameters of the transport/network layer PDU.

REQ	4.5 UDSonCAN – Mapping of T_PDU onto N_PDU
The service primitive parameter mapping of T_PDU onto N_PDU shall be implemented as specified in Table 6.	

Table 6 — Mapping of T_PDU parameter onto N_PDU parameter

T_PDU parameter (data link independent according to ISO 14229-2)	N_PDU parameter (CAN data link dependent according to ISO 15765-2)
T_Ptype	N_Ptype
T_SA	N_SA
T_TA	N_TA
T_TAtype	N_TAtype
T_AE[a]	N_AE
T_Data[ ]	N_Data[ ]
T_Length	N_Length
T_Result	N_Result

[a] If Ptype = diagnostics, then the address information shall consist of the parameters SA, TA, and TAtype.

If Ptype = remote diagnostics, then the address information shall consist of the parameters SA, TA, TAtype, and AE.

12 Network layer

12.1 Service primitive parameter definition

This document specifies the implementation of the network layer of the ISO 15765 DoCAN series and therefore the service primitive parameter implementation follows the ISO 15765-5 specification.

REQ	3.1 UDSonCAN – USDT service primitive parameters
The service primitive parameters shall be implemented as specified in ISO 15765-5.	

12.2 N_Data.req, N_Data.ind, and N_Data.conf service interface

This document follows the service interface definition as specified in ISO 15765-5.

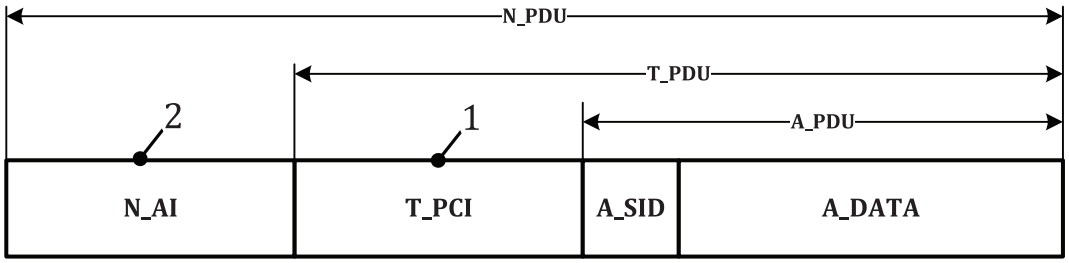
REQ	3.2 UDSonCAN – N_Data.req, N_Data.ind, and N_Data.conf service interface
The N_Data.req, N_Data.ind, and N_Data.conf service interface implementation shall be implemented as specified in ISO 15765-5.	

12.3 Network layer services

REQ	3.3 UDSonCAN – Network layer services
The network layer services shall be implemented as specified in ISO 15765-2.	

12.4 N_PDU definition

Figure 7 shows the N_PDU.



Key

1 transport layer protocol control information

2 network layer address information

Figure 7 — N_PDU definition

12.5 N_TAtype service primitive parameter

The parameter `N_TAtype` is a configuration attribute to the `N_TA` parameter. It is used to encode the USDT communication by the peer entities.

REQ	3.4 UDSonCAN – N_TAtype service primitive parameter
The `N_TAtype` parameter shall be of data type `Enum` and shall be used to identify the USDT target address type.	

12.6 Same N_TAtype request and associated response message format

ISO 14229-1-defined services use the same request and response message format independent of the address type (AddrType) selected.

REQ	3.5 UDSonCAN – Same N_TAtype request and associated response message format
If a server implements a particular `N_TAtype` for a received request message, then it shall use the same `N_TAtype` as specified in Table 7 in the corresponding response message.	

Table 7 — N_Data service primitive parameter mapping

NL	.req	.ind	.conf	Description
N_AI[N_TAtype]	X	X	—	[AddrType: DiagNormAddr, DiagNormFixAddr, DiagExtAddr, RDiagMixAddr; N_TAtype #1: physical addressing, CBFF, 11-bit CAN identifier, N_TAtype #2: functional addressing, CBFF, 11-bit CAN identifier, N_TAtype #3: physical addressing, FBFF, 11-bit CAN identifier, N_TAtype #4: functional addressing, FBFF, 11-bit CAN identifier, N_TAtype #5: physical addressing, CEFF, 29-bit CAN identifier, N_TAtype #6: functional addressing, CEFF, 29-bit CAN identifier, N_TAtype #7: physical addressing, FEFF, 29-bit CAN identifier, N_TAtype #8: functional addressing, FEFF, 29-bit CAN identifier;]
N_AI[TA]	X	X	—	target address to be added to PDU if N_TAtype = DiagExtAddr.
N_AI[AE]	X	X	—	address extension to be added to PDU if N_TAtype = RDiagMixAddr.
Key X = supported — = not supported				

13 Data link layer

13.1 Service primitive parameter definition

This document follows the service primitive parameter definition as specified in ISO 11898-1.

REQ	2.1 UDSonCAN – Service primitive parameter definition
The service primitive parameters shall be implemented in accordance with ISO 11898-1.	

13.2 L_Data.req, L_Data.ind, and L_Data.conf service interface

This document follows the service interface of the data link layer in accordance with ISO 11898-1.

REQ	2.2 UDSonCAN – L_Data.req, L_Data.ind, and L_Data.conf service interface
The L_Data.req, L_Data.ind, and L_Data.conf service interface shall be implemented as specified in ISO 11898-1.	

13.3 Usage of ISO 15765-4-defined 11-bit CAN identifiers for enhanced diagnostics

The ISO 15765-4-defined 11-bit CAN identifiers may be used for enhanced diagnostics (e.g. the functional request CAN identifier may be used for the functionally addressed TesterPresent $3E_{16}$ request message to keep a non-defaultSession active).

If the 11-bit CAN identifiers are used for enhanced diagnostics as specified in ISO 15765-4, then the following permissions apply:

a) the network layer timing parameters according ISO 15765-4 may be used for enhanced diagnostics;

b) the CAN data length code may be set to eight and unused bytes in the CAN frame are padded.

13.4 Usage of ISO 15765-4-defined 29-bit CAN identifiers for enhanced diagnostics

The ISO 15765-4-defined 29-bit CAN identifiers comply with the normal fixed addressing format specified in ISO 15765-2 and can be used for enhanced diagnostics.

If the 29-bit CAN identifiers as specified in ISO 15765-4 are used for enhanced diagnostics, then the following permissions apply:

a) the network layer timing parameters as specified in ISO 15765-4 may be used for enhanced diagnostics;

b) the CAN data length code may be set to eight and unused bytes in the CAN frame are padded.

14 Physical layer

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

Bibliography

[1] ISO/IEC 7498-1, *Information technology — Open Systems Interconnection — Basic Reference Model: The Basic Model*

[2] ISO/IEC 10731, *Information technology — Open Systems Interconnection — Basic Reference Model — Conventions for the definition of OSI services*

[3] ISO 15765-4, *Road vehicles — Diagnostic communication over Controller Area Network (DoCAN) — Part 4: Requirements for emissions-related systems*

ICS 43.180

Price based on 19 pages