INTERNATIONAL STANDARD

ISO 11992-4

First edition
2005-07-15

Road vehicles — Interchange of digital information on electrical connections between towing and towed vehicles —

Part 4: Diagnostics

Véhicules routiers — Échange d'informations numériques sur les connexions électriques entre véhicules tracteurs et véhicules tractés —

Partie 4: Diagnostics

Reference number
ISO 11992-4:2005(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web 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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

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.

ISO 11992-4 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 11992 consists of the following parts, under the general title *Road vehicles — Interchange of digital information on electrical connections between towing and towed vehicles*:

— *Part 1: Physical and data-link layers*

— *Part 2: Application layer for brakes and running gear*

— *Part 3: Application layer for equipment other than brakes and running gear*

— *Part 4: Diagnostics*

Introduction

ISO 11992 has been established in order to define the data interchange between road vehicles and their towed vehicles using a Controller Area Network (CAN) serial data link as specified in ISO 11898[4].

The description of this part of ISO 11992 is based on the Open Systems Interconnection (OSI) Basic Reference Model in accordance with ISO/IEC 7498[2] (and ISO/IEC 10731[3]), which structures communication systems into seven layers.

When mapped on this model, the communication system specified by ISO 11992 is broken down into:

Layer 7
Application layer for brakes and running gear.
Application layer for equipment other than brakes and running gear.
Application layer for diagnostics.

Layer 3
Network layer for diagnostics.

Layer 2
Data link layer for all communication types.

Layer 1
Physical layer for all communication types.

Table 1 — Applicability and relationship between International Standards

Applicability	**Normal communication**		**Diagnostic communication**
	Brakes and running gear	Equipment other than brakes and running gear	All applications
Layer 7: Application layer	**ISO 11992-2**	**ISO 11992-3**	**ISO 11992-4** ISO 14229-1
Layer 6: Presentation layer	No functions specified for this layer.		
Layer 5: Session layer	No functions specified for this layer.		
Layer 4: Transport layer	No functions specified for this layer.		
Layer 3: Network layer	No functions specified for this layer.		**ISO 11992-4** ISO 15765-2
Layer 2: Data link layer	**ISO 11992-1**		
Layer 1: Physical layer	**ISO 11992-1**		

Road vehicles — Interchange of digital information on electrical connections between towing and towed vehicles —

Part 4: Diagnostics

1 Scope

This part of ISO 11992 specifies the data communication for diagnostic purposes on a serial data link between a road vehicle and its towed vehicle(s).

This part of ISO 11992 is applicable to road vehicles of a maximum authorized total mass greater than 3 500 kg.

2 Normative references

The following referenced documents are indispensable for the application 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 11992-1, *Road vehicles — Interchange of digital information on electrical connections between towing and towed vehicles — Part 1: Physical and data-link layers*

ISO 11992-2, *Road vehicles — Interchange of digital information on electrical connections between towing and towed vehicles — Part 2: Application layer for brakes and running gear*

ISO 11992-3, *Road vehicles — Interchange of digital information on electrical connections between towing and towed vehicles — Part 3: Application layer for equipment other than brakes and running gear*

ISO 14229-1, *Road vehicles — Unified diagnostic services (UDS) — Part 1: Specification and requirements*

ISO 15765-2, *Road vehicles — Diagnostics on Controller Area Networks (CAN) — Part 2: Network layer services*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 11992-1, ISO 14229-1 and ISO 15765-2 apply.

4 Syntax applied

For the description of services and service parameters of this part of ISO 11992, the following syntax is used:

Name: Type	Parameter name and type specification
Name	Mandatory parameter value
<Name>	Parameter name representing a set of mandatory parameter values
[Name]	Optional parameter value
[<Name>]	Parameter name representing a set of optional parameter values
{Name 1;Name 2}	List of mandatory parameter values
{<Name 1>;<Name 2>}	List of parameter names representing sets of mandatory parameter values
[<Name 1>;<Name 2>]	List of parameter names representing sets of optional parameter values
{<Name 1>|<Name 2>}	Parameter names selection list representing sets of mandatory parameter values
[<Name 1>|<Name 2>]	Parameter names selection list representing sets of optional parameter values
Name.req	Service request primitive
Name.ind	Service indication primitive
Name.rsp	Service response primitive
Name.rsp-	Service negative response primitive
Name.rsp+	Service positive response primitive
Name.con	Service confirmation primitive
Name.con-	Service negative confirmation primitive
Name.con+	Service positive confirmation primitive

5 Diagnostic application specification

5.1 General

The diagnostic applications are divided into basic diagnostic applications and enhanced diagnostic applications.

Functions, services and protocols of the layers 1 to 4 shall be identical for basic diagnostics and enhanced diagnostics.

5.2 Basic diagnostics

The purpose of the basic diagnostics is to provide vehicle-independent identification and diagnostic information.

All basic diagnostic functions and services shall be provided under all operation conditions in the default diagnostic session without the need for specific access rights.

5.3 Enhanced diagnostics

The support and the conditions under which enhanced diagnostic functions and services are provided are manufacturer-specific. It is the responsibility of the manufacturer to secure a server against unauthorized access and to guarantee performance and safe operation in all operation modes allowing enhanced diagnostics.

5.4 Client and server state diagrams

5.4.1 General

The client and server state diagrams describe the diagnostic service processing of the client and server application entity.

5.4.2 Client service primitives handling

The client service primitives handling shall be as specified in Figure 1 and Figure 2.

Client states while processing a diagnostic service shall be as specified in Table 2, events resulting in a client state change shall be as specified in Table 3.

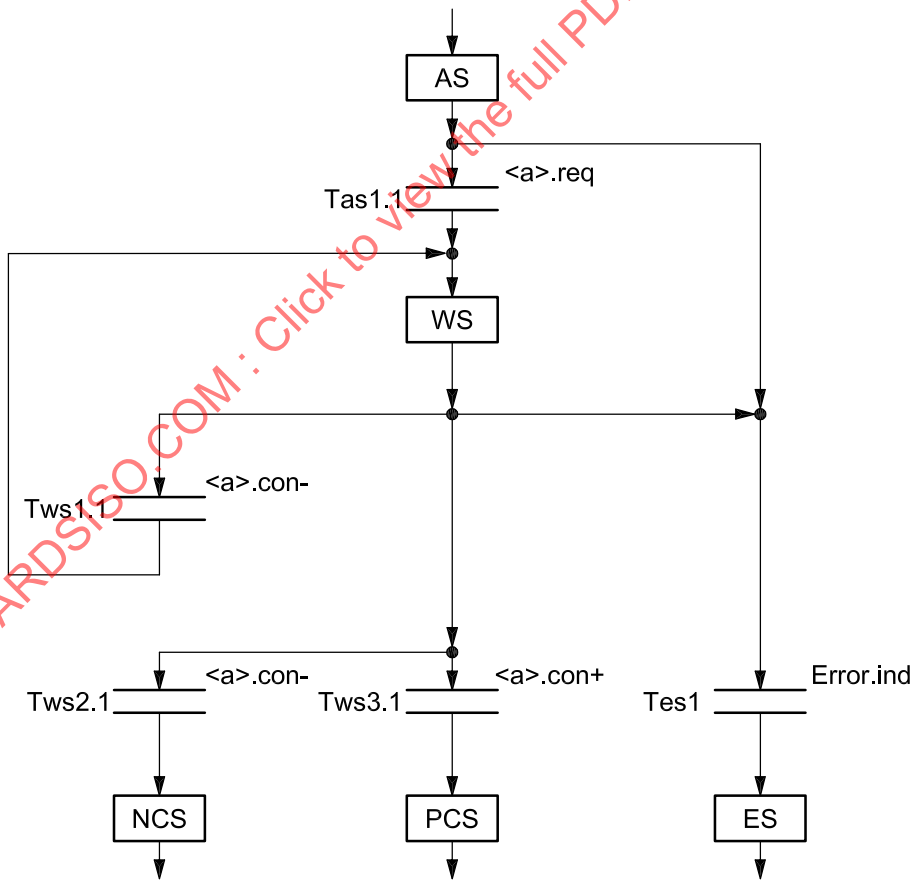

Figure 1 — Client service state diagram — Service request with physical server target address

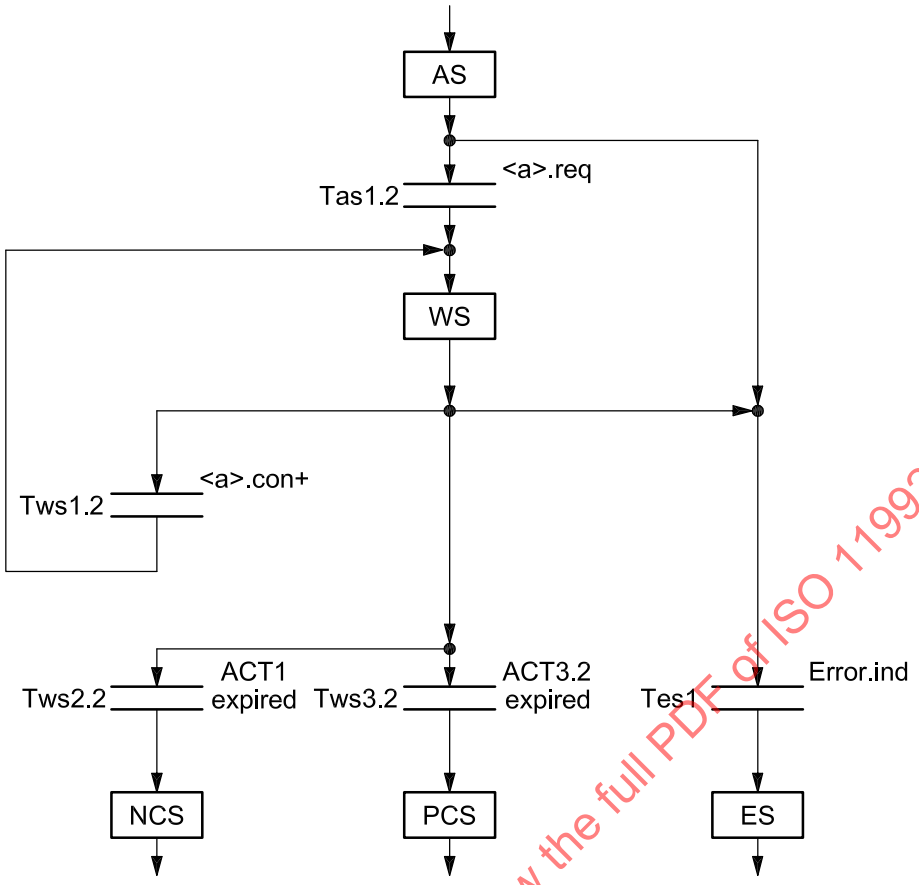

Figure 2 — Client service state diagram — Service request with functional server target address

Table 2 — Client state description

Client state	Description
AS	Any client state in which a service request can take place.
WS	Client state while waiting for a confirmation from the server.
NCS	a) Following a service request with a physical server target address: client state after the reception of a negative confirmation from the server. b) Following a service request with a functional server target address: client state if no positive service confirmation has been received.
PCS	Client state after the reception of a positive confirmation from a server.
ES	Client state for error handling, e.g. in case of a time out condition.

Table 3 — Client event description

Client event		Description
Tas1.1	Transmit <a.>.req	The client transmits an <a> service request with a physical server target address.
Tas1.2	Transmit <a.>.req	The client transmits an <a> service request with a functional server target address.
Tws1.1	Receive <a>.con-	Following a service request with a physical server target address: the client receives a negative <a> service confirmation with the response code 'Request correctly received - response pending'. The client shall then reset the time outs and enter the WS state again.
Tws1.2	Receive <a>.con+	Following a service request with a functional server target address: the client receives a positive <a> service confirmation. The client shall then process the positive service confirmation and enter the WS state again.
Tws2.1	Receive <a>.con-	Following a service request with a physical server target address: the <a> service request has been rejected, a corresponding negative <a> service confirmation with a response code has been received. The client shall then change to the NCS state.
Tws2.2	ACT1 expired	Following a service request with a functional server target address: the time ACT1 for the reception of the first service confirmation has expired and no positive service confirmation has been received. The client shall then change to the NCS state.
Tws3.1	Receive <a>.con+	Following a service request with a physical server target address: the <a> service has been executed, a positive <a> service confirmation, i.e. the result of the service, has been received. The client shall then change to the PCS state.
Tws3.2	ACT3.2 expired	Following a service request with a functional server target address the time ACT3.2 for the reception of consecutive service confirmations has expired and at least one positive service confirmation has been received. The client shall then change to the PCS state.
Tes1	Error.ind	An error condition, e.g. a time out condition, is signalled to the client. The client shall then change to the ES state for error handling.
NOTE Negative service responses with a response code 10_{16}, 11_{16} or 12_{16} shall not be sent by a server in case of a service request with a functional server target address.		
<a> Any diagnostic service.		

5.4.3 Server service primitives handling

The server diagnostic service primitives handling shall be as specified in Figure 3.

Server states while processing a diagnostic service shall be as specified in Table 4, events resulting in a server state change shall be as specified in Table 5.

Table 4 — Server state description

Server state	Description
AS	Any server state in which the reception of a service indication can take place.
PS	Server state while processing a service.
PCS	Server state after the diagnostic service has been executed.
ES	Server state error handling, e.g. after reaching a time out condition.

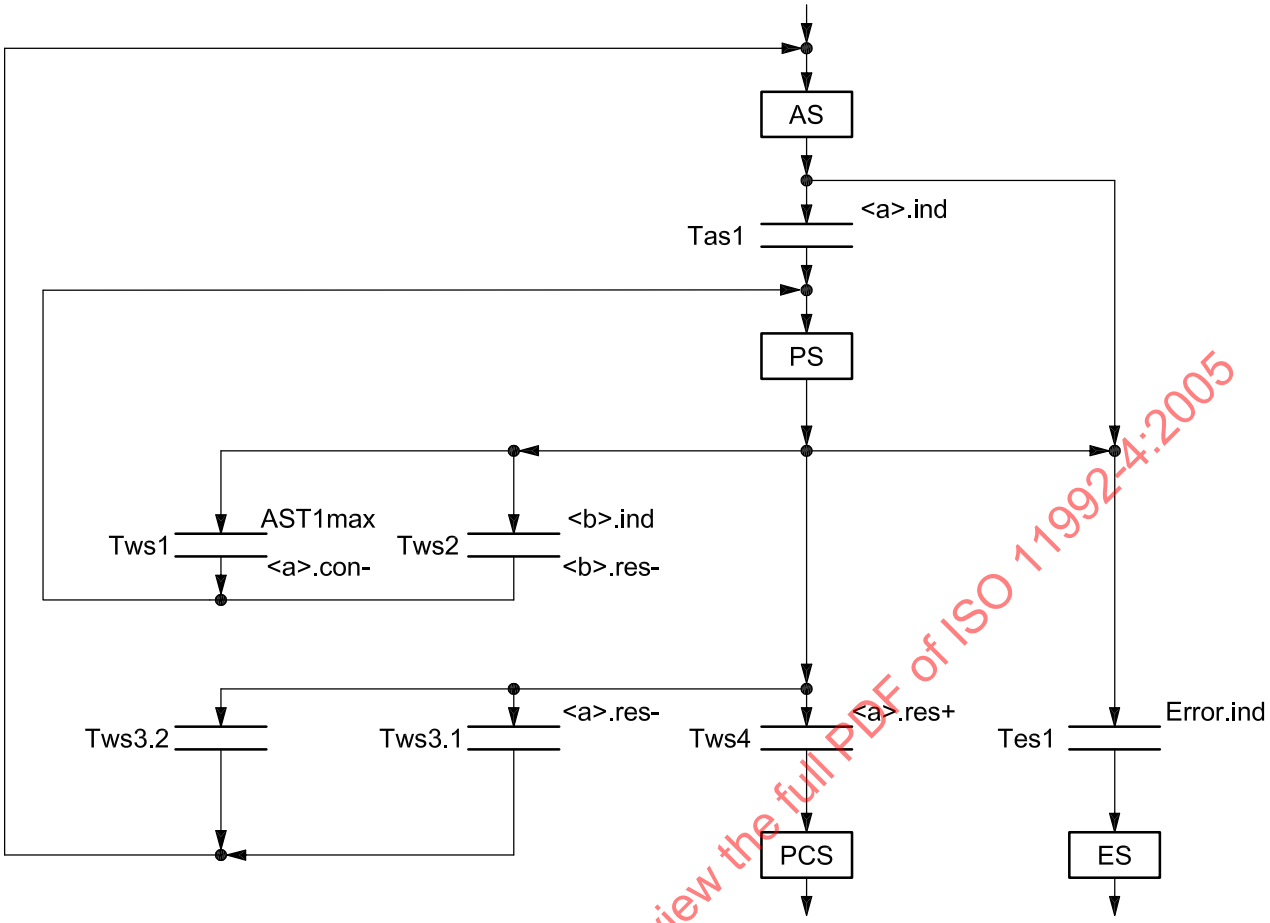

Figure 3 — Server state diagram

Table 5 — Server event description

Event		Description
Tas1	Receive <a.>.ind	The server receives any <a> service.indication.
Tws1	AST1max expired Transmit <a>.res-	The service execution time AST1max has expired. the server shall then send a negative service response with the response code 'Request correctly received - response pending' and change back to the PS state to proceed the service execution.
Tws2	Received <b>.ind Respond <b>.res±	The server receives a <b> service indication, while service <a> is in progress. the server shall reject the service <b> if service <b> ≠ service <a> and send a negative response with response code "Busy - Repeat Request". If service <b> = service <a>, the server shall send a negative service response with response code "Request Correctly Received - Response Pending". The server shall then enter again the PS state to proceed the service execution.
Tws3.1	Service execution not completed Respond <a>.res-	Following a service request with a physical server target address: the server rejects the service request. The server shall then send a negative <a> service response with a corresponding response code and change to the AS state.
Tws3.2	Service execution not completed No response	Following a service request with a functional server target address: the server rejects the service request. The server shall then send no response code and change to the AS state.
Tws4	Service execution completed Respond <a>.res+	The server sends a positive <a> service response. the service has been executed. The server shall then transmit a positive <a> service response, i.e. the service results, and shall change again to the PCS state.
Tes1	Error.ind	An error condition is indicated to the server, e.g. a time out condition. The server changes to the ES state for error handling.
NOTE Negative service responses with a response code 10_{16}, 11_{16} or 12_{16} shall not be sent by a server in case of a service request with a functional server target address.		
<a> Any first diagnostic service. <b> Any second diagnostic service.		

6 Application layer specification

6.1 General

The application layer function, service and protocol specifications comply with ISO 14229-1. In case of differences, the specifications of this part of ISO 11992 shall have precedence.

For the diagnostic communication between road vehicles and their towed vehicles, the restrictions described in this clause apply additionally.

6.2 Application layer functions

6.2.1 General

The application layer provides functions for the execution of the vehicle diagnostics. These functions are used by client and server applications requesting the respective application layer services.

6.2.2 Processing of diagnostic services requests and responses

Diagnostic service requests from the client application and diagnostic service responses from the server application shall be processed according to the service identifier. The diagnostic data shall be encoded as an application layer protocol data unit (A_PDU). The A_PDU shall be transmitted to the respective application layer peer entity by requesting services of the layers beneath the application layer.

6.2.3 Processing of diagnostic service indications and confirmations

Diagnostic data shall be received as an A_PDU from the layers beneath the application layer. If the received A_PDU is addressed to one of the local server or client application, the received A_PDU shall be decoded and processed according to the diagnostic service identifier and delivered to the server or client application as a service indication or confirmation.

6.2.4 Determination of network layer service parameters

Network layer service parameters are determined by the application layer service type, i.e. ClientIdentifier, ServerIdentifier, ServiceIdentifier and ServiceParameter. In addition the specified parameters Priority and ReservedBit shall be used.

NOTE As no specific functions have been specified for the presentation, session and transport layer, the PDUs of these layers are identical to the respective application layer PDUs.

6.2.5 Application layer protocol timing supervision

The peer application layer entities communicating shall supervise the specified timing and shall take the respective actions in case a specified time out expires.

6.2.6 Server and client addressing

6.2.6.1 Vehicle network architecture

Towed vehicle server and client applications shall be addressed and identified by means of remote network addressing. The physical sub-networks between towing and towed vehicles are part of the local motor vehicle network and share the same address range. The address type of the target address (TA) and of the remote address (RA) in the case of encoding a remote target address shall be identified by the target address type (TA_Type).

Figure 4 shows an example of the vehicle network architecture.

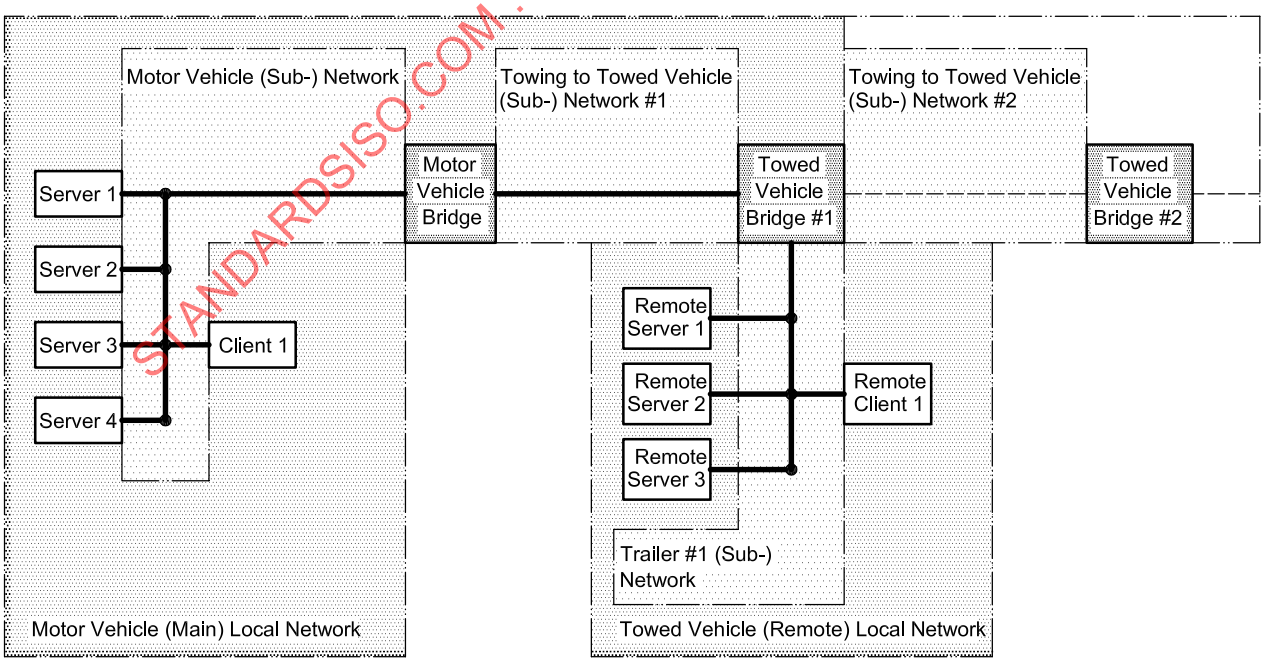

Figure 4 — Vehicle network architecture example

6.2.6.2 Towing to towed vehicle <Service>.Request and <Service>.Indication

For a diagnostic service request transmitted from a towing vehicle to a towed vehicle, the address parameters of the service primitives have the following meaning:

SA = <Towing vehicle client source address>

TA = <Towed vehicle target address>

RA = <Towed vehicle remote server target address>

TA_type = {
<Physical target addresses>|
<Functional target addresses>
}

See Annex A.

NOTE TA_type identifies the TA and the RA target address type.

6.2.6.3 Towed to towing vehicle <Service>.Response and <Service>.Confirmation

For a diagnostic service response transmitted from a towed vehicle to a towing vehicle, the address parameters of the service primitives have the following meaning:

SA = <Towed vehicle source address>

TA = <Towing vehicle client target address>

RA = <Towed vehicle remote server source address>

TA_type = <Physical target addresses>

NOTE TA_type identifies only the TA target address type.

6.2.6.4 Towed to towing vehicle <Service>.Request and <Service>.Indication

For a diagnostic service request transmitted from a towed vehicle to a towing vehicle, the address parameters of the service primitives have the following meaning:

SA = <Towed vehicle source address>

TA = <Towing vehicle server target address>

RA = <Towed vehicle remote client source address>

TA_type = {
<Physical target addresses>|
<Functional target addresses>
}

NOTE TA_type identifies only the TA target address type.

6.2.6.5 Towing to towed vehicle <Service>.Response and <Service>.Confirmation

For a diagnostic service response transmitted from a towing vehicle to a towed vehicle, the address parameters of the service primitives have the following meaning:

SA = <Towing vehicle server source address>

TA = <Towed vehicle target address>

RA = <Towed vehicle remote client target address>

TA_type = <Physical target addresses>

NOTE TA_type identifies the TA and RA target address type.

6.3 Application layer services

6.3.1 General

This subclause specifies the application layer services for diagnostics.

6.3.2 Application layer service parameters

6.3.2.1 General

This subclause specifies the general application layer service parameter and the respective parameter format. Service specific parameters are specified in 6.3.4.

6.3.2.2 Source address (SA)

The parameter source address (SA) contains the client or server source address. It represents the physical location of a client or server on the local network.

SA = {
<Towing vehicle client source address>|
<Towed vehicle source address>
}

6.3.2.3 Target address (TA)

The parameter target address (TA) contains the client or server target address. It represents the physical location of a client or server or a functional group of servers on the local network. The target address type is determined by the parameter TA_type.

TA = {
<Towed vehicle target address>|
<Towing vehicle client target address>|
<Towing vehicle server target address>
}

6.3.2.4 Target address type (TA_type)

The parameter target address type (TA_type) determines the address type of the target address TA and the remote address RA, in the case that the remote address corresponds to a target address.

TA_type = {
<Physical target addresses>|
<Functional target addresses>
}

6.3.2.5 Remote address (RA)

The remote address (RA) contains the remote addresses of servers or clients on a remote network. Depending on the respective application layer primitive, RA represents either a remote target address or a remote source address.

RA = {
<Towed vehicle remote server target address>|
<Towed vehicle remote server source address>|
<Towed vehicle remote client source address>|
<Towed vehicle remote client target address>
}

6.3.2.6 ServiceParameter

ServiceParameter is a record which contains the respective service parameters of the service primitive.

ServiceParameter = {
<<Service name> request service parameter>|
<<Service name> positive response service parameter>|
<<Service name> negative response service parameter>
}

<Service name> request service parameters are those following the <service name> request service identifier.

<Service name> positive response service parameters are those following the <service name> positive response service identifier.

<Service name> negative response service parameters are those following the negative response service identifier.

6.3.3 Application layer service data units (A_SDU)

The application layer service data units (A_SDU) of the diagnostic service primitives have the following formats:

<Service name>.Request = {
<SA>;
<TA>;
<TA_Type>;
<RA>;
<<Service name> request service identifier>;
<<Service name> request service parameter>
}

<Service name>.Indication = < <Service name>.Request>

<Service name>.Response = {
<SA>;
<TA>;
<TA_Type>;
<RA>;
<<Service name> response service identifier>;
<<Service name> response service parameter>
}

<Service name>.Confirmation = <<Service name>.Response>

Negative<Service name>.Response = {
<SA>;
<TA>;
<TA_Type>;
<RA>;
<Negative response service identifier>;
<<Service name> negative response service parameter>
}

Negative<Service name>.Confirmation = <Negative<Service name>.Response>

6.3.4 Basic diagnostic services

6.3.4.1 Services

The specified diagnostic services for basic diagnostics shall be as specified in Table 6.

Table 6 — Services for basic diagnostics

Service name	Request	Response
ReadDataByIdentifier	22_{16}	62_{16}
ReadDTCInformation	19_{16}	59_{16}
NegativeResponse [a]	—	$7F_{16}$
[a] The negative response service identifier is followed by the <Service name>RequestServiceIdentifier.		

The diagnostic services specification includes tables which list diagnostic services and respective service primitive parameters. For all services and service primitive parameters, the presence is specified by the convention values (Cvt) listed in Table 7.

Table 7 — Diagnostic service primitive parameter conventions

Cvt	Name	Description
M	Mandatory	The service or service primitive parameter shall be present.
C	Conditional	The service or service primitive parameter can be present, based on certain criteria (e.g. due to a certain sub-function).
S	Selection	The service or service primitive parameter is mandatory (unless otherwise specified) and is a selection from a given list of services or service primitive parameters.
U	User option	The service or service primitive parameter may or may not be present, depending on dynamic usage by the user.

NOTE A service identifier marked as mandatory does not imply that this service has to be supported.

6.3.4.2 ReadDTCInformation service

6.3.4.2.1 General

The service ReadDTCInformation is used to request diagnostic trouble code (DTC) and corresponding severity and functional unit information.

6.3.4.2.2 ReadDTCInformation service parameters

6.3.4.2.2.1 General

The service parameter of the ReadDTCInformation service specified in this subclause shall be supported for basic diagnostics.

6.3.4.2.2.2 Sub-function

For basic diagnostics the ReadDTCInformation sub-functions shown in Table 8 shall be supported to enable a client to request severity and functional unit information. The DTC format for basic diagnostics shall be as specified in Annex B.

Sub-function: 8 Bit = {
<ReportNumberOfDTCBySeverityMaskRecord>|
<ReportDTCBySeverityMaskRecord>|
<ReportSeverityInformationOfDTC>
}

Table 8 — ReadDTCInformation sub-functions for basic diagnostics

Hex	Definition	Cvt
07	ReportNumberOfDTCBySeverityMaskRecord Retrieve a count of the number of DTCs matching a client-defined severity mask record. This parameter specifies that the server shall transmit to the client the number of DTCs matching a client defined severity mask record.	M
08	ReportDTCBySeverityMaskRecord Retrieve a list of severity and functional unit information, which satisfies a client-defined severity mask record. This parameter specifies that the server shall transmit to the client a list with severity, functional unit, DTC, failure type and status information of stored DTCs matching a client-defined severity mask record.	M
09	ReportSeverityInformationOfDTC Retrieve severity and functional unit information for a client-defined DTCMaskRecord. This parameter specifies that the server shall transmit to the client the severity, functional unit, DTC, failure type and status information matching a client-defined DTCMaskRecord.	U

6.3.4.2.2.3 DTCSeverityMaskRecord

The DTCSeverityMaskRecord consists of the DTCSeverityMask and the DTCStatusMask.

DTCSeverityMaskRecord: 16 Bit = {
<DTCSeverityMask>;
<DTCStatusMask>
}

6.3.4.2.2.4 DTCSeverityMask

The DTCSeverityMask shall be used in the request message by a client to request DTC information with severity information matching the DTCSeverityMask. A DTC's severity value matches the DTCSeverityMask if any one of the DTC's actual severity bits is set to '1' and the corresponding severity bit in the DTCSeverityMask is also set to '1' (i.e., if the DTCSeverityMask is bit-wise logically ANDed with the DTC's actual severity and the result is non-zero, then a match has occurred).

DTCSeverityMask: 8 Bit = 00100000_2 .... 11100000_2

NOTE The value 00000000_2 shall not be used for the DTCSeverityMask

DTCSeverityMask values shall be as specified in Table 9.

The lower 5 bits of the DTCSeverityMask are reserved for future use and shall be set to 00000_2.

Table 9 — DTCSeverityMask specification

Value	Description	Cvt
xx1 00000_2	MaintenanceOnly This value indicates that the failure requests a check of the vehicle at the next maintenance.	M
x1x 00000_2	CheckAtNextHalt This value indicates that the failure requires a check of the vehicle at the next halt.	M
1xx 00000_2	CheckImmediately This value indicates that the failure requires an immediate check of the vehicle.	M
Bit position marked with 'x' might be set to '0' or '1'.		

6.3.4.2.2.5 DTCStatusMask

The DTCStatusMask shall be used in the request message by a client to request DTC information of DTCs with DTC status information matching the DTCStatusMask. A DTC's status matches the DTCStatusMask if any one of the DTC's actual status bits is set to "1" and the corresponding status bit in the DTCStatusMask is also set to "1" (i.e., if the DTCStatusMask is bit-wise logically ANDed with the DTC's actual status and the result is non-zero, then a match has occurred). If the client specifies a status mask that contains bits that the server does not support, then the server shall process the DTC information using only the bits that it does support. The DTCStatusMask bits shall be as specified in Table 10.

DTC status bits are specified in Annex B.

DTCStatusMask: 8 Bit = 00000001_2 .... 11111111_2

Table 10 — DTCStatusMask bits

Value	Description	Cvt
xxxx $xxx1_2$	TestFailed	M
xxxx $xx1x_2$	TestFailedThisMonitoringCycle	C1
xxxx $x1xx_2$	PendingDTC	U
xxxx $1xxx_2$	ConfirmedDTC	M
$xxx1xxxx_2$	TestNotCompletedSinceLastClear	C2
xx1x $xxxx_2$	TestFailedSinceLastClear	C2
x1xx $xxxx_2$	TestNotCompletedThisMonitoringCycle	M
1xxx $xxxx_2$	WarningIndicatorRequested	U
Bit position market with 'x' might be set to '0' or '1'.		
C1: Bit 1 (TestFailedThisMonitoringCycle) is mandatory if bit 2 (PendingDTC) is supported. Bit 1 (TestFailedThisMonitoringCycle) is user optional if bit 2 (PendingDTC) is not supported. C2: Bit 4 (TestNotCompletedSinceLastClear) and bit 5 (TestNotFailedSinceLastClear) shall always be supported together.		

6.3.4.2.2.6 ReportType

ReportType is an echo of the sub-function parameter in the request message from the client.

ReportType = <Sub-function>

6.3.4.2.2.7 DTCStatusAvailabilityMask

The DTCStatusAvailabilityMask represents the DTC status bits that are supported by the server. Bits that are not supported by the server shall be set to 0. The DTCStatusAvailabilityMask bits shall be as specified in Table 11.

DTC status bits are specified in Annex B.

DTCStatusAvailabilityMask: 8 Bit = 00000001_2 .... 11111111_2

Table 11 — DTCStatusAvailabilityMask bits

Value	Description	Cvt
xxxx xxx1_2	TestFailed	M
xxxx xx1x$_2$	TestFailedThisMonitoringCycle	C1
xxxx x1xx$_2$	PendingDTC	U
xxxx 1xxx$_2$	ConfirmedDTC	M
xxx1xxxx$_2$	TestNotCompletedSinceLastClear	C2
xx1x xxxx$_2$	TestFailedSinceLastClear	C2
x1xx xxxx$_2$	TestNotCompletedThisMonitoringCycle	M
1xxx xxxx$_2$	WarningIndicatorRequested	U
Bit position market with 'x' might be set to '0' or '1'.		
C1: Bit 1 (TestFailedThisMonitoringCycle) is mandatory if bit 2 (PendingDTC) is supported. Bit 1 (TestFailedThisMonitoringCycle) is user optional if bit 2 (PendingDTC) is not supported. C2: Bit 4 (TestNotCompletedSinceLastClear) and bit 5 (TestNotFailedSinceLastClear) shall always be supported together.		

6.3.4.2.2.8 DTCFormatIdentifier

The DTCFormatIdentifier parameter identifies the format of DTC's reported by the server.

DTCFormatIdentifier = <ISO11992DTCFormat>

ISO11992DTCFormat = 3_{10}

All other DTCFormatIdentifier values are reserved for future specification.

6.3.4.2.2.9 DTCCount

DTCCount contains the DTCCountHighByte and DTCCountLowByte parameters sent in response to a ReportNumberOfDTC request. DTCCount shall provide the number of DTC's that match the DTCStatusMask defined in the client's request.

DTCCount = {<DTCCountHighByte>;<DTCCountLowByte>} = 0_{10}...65535_{10}.

6.3.4.2.2.10 DTCAndSeverityRecord

The DTCAndSeverityRecord contains one or more groupings of DTCSeverity, DTCFunctionalUnit, DTCHighByte, DTCMiddleByte, DTCLowByte, and statusOfDTC.

```
DTCAndSeverityRecord = {
                       <DTCSeverity#1>;
                       <DTCFunctionalUnit#1>;
                       <DTCHighByte#1>;
                       <DTCMiddleByte#1>;
                       <DTCLowByte#1>;
                       <StatusOfDTC#1>
                       .....
                       <DTCSeverity#n>;
                       <DTCFunctionalUnit#n>;
                       <DTCHighByte#n>;
                       <DTCMiddleByte#n>;
                       <DTCLowByte#n>;
                       <StatusOfDTC#n>
                       }
```

6.3.4.2.2.11 DTCSeverity

The DTCSeverity identifies the importance of the failure for the vehicle operation and/or system function and enabless recommended actions to be displayed to the driver. The DTCSeverity shall be as specified in Table 12.

The lower 5 bits of the DTCSeverity are reserved for future use and shall be set to 00000_2.

```
DTCSeverity: 8 Bit = {
                     <MaintenanceOnly>|
                     <CheckAtNextHalt>|
                     <CheckImmediately>
                     }
```

Table 12 — DTCSeverity specification

Value	Description	Cvt
$000\ 00000_2$	NoSeverityAvailable There is no severity information available.	U
$001\ 00000_2$	MaintenanceOnly This value indicates that the failure requests a check of the vehicle at the next maintenance.	M
$010\ 00000_2$	CheckAtNextHalt This value indicates that the failure requires a check of the vehicle at the next halt.	M
$100\ 00000_2$	CheckImmediately This value indicates to the failure requires an immediate check of the vehicle.	M

6.3.4.2.2.12 DTCFunctionalUnit

The DTCFunctionalUnit identifies the corresponding basic vehicle and/or system function which reports the DTC.

DTCFunctionalUnit: 8 Bit = 0_{10} ... 255_{10}

DTCFunctionalUnit values are specified in Annex B.

6.3.4.2.2.13 DTCHighByte, DTCMiddleByte and DTCLowByte

DTCHighByte, DTCMiddleByte and DTCLowByte together represent a unique identification number for a specific diagnostic trouble code supported by a server. The DTCHighByte and DTCMiddleByte represent a circuit or system that is being diagnosed. The DTCLowByte represents the type of fault in the circuit or system (DTCFailureTypeByte, e.g. sensor open circuit, sensor shorted to earth, algorithm based failure, etc.).

DTC: 16 Bit = {<DTCHighByte>;<DTCMiddleByte>}

DTCFailureTypeByte: 8 Bit = <DTCLowByte>

6.3.4.2.2.14 StatusOfDTC

StatusOfDTC is the status of a particular DTC (e.g. pending DTC). The StatusOfDTC bits are specified in Annex B.

StatusOfDTC: 8 Bit = 00000000_2 ... 11111111_2

6.3.4.2.3 ReadDTCInformation — ReportNumberOfDTCBySeverityMaskRecord

The ReadDTCInformation service request parameters shown in Table 13 shall be used by the client to request the number of stored DTC's matching the DTCSeverityMaskRecord.

Table 13 — ReportNumberOfDTCBySeverityMaskRecord request parameter

A_Data byte	Parameter name	Cvt	Hex value
#1	ReadDTCInformation request service identifier	M	19
#2	Sub-function = ReportNumberOfDTCBySeverityMaskRecord	M	07
	DTCSeverityMaskRecord = {		
#3	DTCSeverityMask	M	00-FF
#4	DTCStatusMask }	M	00-FF

The ReadDTCInformation service response parameters shown in Table 14 shall be used by the server to send the number of stored DTC's matching the DTCSeverityMaskRecord to the client.

NOTE StatusOfDTC information supported by the server are indicated by the DTCStatusAvailabilityMask.

Table 14 — ReportNumberOfDTCBySeverityMaskRecord positive response parameter

A_Data byte	Parameter name	Cvt	Hex value
#1	ReadDTCInformation response service identifier	M	59
#2	ReportType = ReportNumberOfDTCBySeverityMaskRecord	M	07
#3	DTCStatusAvailabilityMask	M	00-FF
#4	DTCFormatIdentifier = ISO11992DTCFormat	M	03
	DTCCount = {		
#5	DTCCountHighByte	M	00-FF
#6	DTCCountLowByte }	M	00-FF

6.3.4.2.4 ReadDTCInformation — ReportDTCBySeverityMaskRecord

The ReadDTCInformation service request parameters shown in Table 15 shall be used by the client to request the stored DTC's matching the DTCSeverityMaskRecord.

Table 15 — ReportDTCBySeverityMaskRecord request parameter

A_Data byte	Parameter name	Cvt	Hex value
#1	ReadDTCInformation request service identifier	M	19
#2	Sub-function = ReportDTCBySeverityMaskRecord	M	08
	DTCSeverityMaskRecord = {		
#3	DTCSeverityMask	M	00-FF
#4	DTCStatusMask }	M	00-FF

The ReadDTCInformation service response parameters shown in Table 16 shall be used by the server to send the stored DTC's matching the DTCSeverityMaskRecord to the client.

NOTE StatusOfDTC information supported by the server are indicated by the DTCStatusAvailabilityMask.

Table 16 — ReportDTCBySeverityMaskRecord positive response parameter

A_Data byte	Parameter name	Cvt	Hex value
#1	ReadDTCInformation response service identifier	M	59
#2	ReportType = ReportDTCBySeverityMaskRecord	M	08
#3	DTCStatusAvailabilityMask	M	00-FF
	DTCAndSeverityRecord = {		
#4	DTCSeverity #1	C1	00-FF
#5	DTCFunctionalUnit #1	C1	00-FF
#6	DTCHighByte #1	C1	00-FF
#7	DTCMiddleByte #1	C1	00-FF
#8	DTCLowByte #1	C1	00-FF
#9	StatusOfDTC #1	C1	00-FF
:	:	:	:
#n-5	DTCSeverity #m	C2	00-FF
#n-4	DTCFunctionalUnit #m	C2	00-FF
#n-3	DTCHighByte #m	C2	00-FF
#n-2	DTCMiddleByte #m	C2	00-FF
#n-1	DTCLowByte #m	C2	00-FF
#n	StatusOfDTC #m}	C2	00-FF
C1: This parameter is only present if DTC information is available to be reported.			
C2: This parameter is only present if more than one piece of DTC information is available to be reported.			

6.3.4.2.5 ReadDTCInformation — ReportSeverityInformationOfDTC

The ReadDTCInformation service request parameters shown in Table 17 shall be used by the client to request the severity and functional unit information of stored DTC's matching the DTCMaskRecord.

Table 17 — ReportSeverityInformationOfDTC request parameter

A_Data byte	Parameter name	Cvt	Hex value
#1	ReadDTCInformation request service identifier	M	19
#2	Sub-function = ReportSeverityInformationOfDTC	M	09
	DTCMaskRecord = {		
#3	DTCHighByte	M	00-FF
#4	DTCMiddleByte	M	00-FF
#5	DTCLowByte }	M	00-FF

The ReadDTCInformation service response parameters shown in Table 18 shall be used by the server to send the stored DTC's matching the DTCMaskRecord to the client.

NOTE StatusOfDTC information supported by the server are indicated by the DTCStatusAvailabilityMask.

Table 18 — ReportSeverityInformationOfDTC positive response parameter

A_Data byte	Parameter name	Cvt	Hex value
#1	ReadDTCInformation response service identifier	M	59
#2	ReportType = ReportSeverityInformationOfDTC	M	09
#3	DTCStatusAvailabilityMask	M	00-FF
	DTCAndSeverityRecord = {		
#4	DTCSeverity #1	C1	00-FF
#5	DTCFunctionalUnit #1	C1	00-FF
#6	DTCHighByte #1	C1	00-FF
#7	DTCMiddleByte #1	C1	00-FF
#8	DTCLowByte #1	C1	00-FF
#9	StatusOfDTC #1	C1	00-FF
:	:	:	:
#n-5	DTCSeverity #m	C2	00-FF
#n-4	DTCFunctionalUnit #m	C2	00-FF
#n-3	DTCHighByte #m	C2	00-FF
#n-2	DTCMiddleByte #m	C2	00-FF
#n-1	DTCLowByte #m	C2	00-FF
#n	StatusOfDTC #m }	C2	00-FF
C1: This parameter is only present if DTC information is available to be reported.			
C2: This parameter is only present if more than one piece of DTC information is available to be reported.			

6.3.4.2.6 ReadDTCInformation negative response

The ReadDTCInformation service negative response parameters shown in Table 19 shall be used by the server to inform the client that the service request has not been performed.

NOTE Response codes are specified in Annex B.

Table 19 — ReadDTCInformation negative response parameter

A_Data byte	Parameter name	Cvt	Hex value
#1	NegativeResponse service identifier	M	7F
#2	ReadDTCInformation request service identifier	M	19
#3	ResponseCode	M	00-FF

6.3.4.3 ReadDataByIdentifier

6.3.4.3.1 General

The service ReadDataByIdentifier shall be used to request information from a server using a RecordDataIdentifier.

6.3.4.3.2 ReadDataByIdentifier service parameter

6.3.4.3.2.1 General

The service parameter of the ReadDataByIdentifier service specified in this subclause shall be supported for basic diagnostics.

For basic diagnostics RecordDataIdentifier and DataRecord specified in Annex B shall be supported.

6.3.4.3.2.2 RecordDataIdentifier

The RecordDataIdentifier identifies a server DataRecord requested by the client.

RecordDataIdentifier: 16 Bit = {<Byte 1 MSB>;<Byte 2 LSB>} = 0000_{16} .... $FFFF_{16}$

For basic diagnostics the RecordDataIdentifier and DataRecord specified in Annex B shall be supported.

6.3.4.3.2.3 DataRecord

The dataRecord parameter shall be used by the server to provide the requested data record to the client.

DataRecord: Record = {<Data #1>;<Data #2>; .....;<Data #m>}

For basic diagnostics RecordDataIdentifier and DataRecord shall be supported as specified in Annex B.

6.3.4.3.3 ReadDataByIdentifier service request parameter

The ReadDataByIdentifier service request parameters in Table 20 shall be used by the client to request data records from the server identified by the RecordParameterIdentifier.

Table 20 — ReadDataByIdentifier request parameters

A_Data byte	Parameter name	Cvt	Hex value
#1	ReadDataByIdentifier request service identifier	M	22
	RecordDataIdentifier = {		
#2	Byte 1 (MSB)	M	00-FF
#3	Byte 2 (LSB)}	M	00-FF

6.3.4.3.4 ReadDataByIdentifier service response parameter

The ReadDataByIdentifier service response parameters shown in Table 21 shall be used by the server to send the stored data record identified by the RecordParameterIdentifier to the client.

Table 21 — ReadDataByIdentifier positive response parameters

A_Data byte	Parameter name	Cvt	Hex value
#1	ReadDataByIdentifier response service identifier	M	62
	RecordDataIdentifier = {		
#2	Byte 1 (MSB)	M	00-FF
#3	Byte 2 (LSB)}	M	00-FF
	DataRecord = {		00-FF
#4	Data#1	M	00-FF
:	:	:	:
#(m – 1)+4	Data#m}	C1	00-FF
C1: This parameter is only present if the data record consists of more than one data byte.			

6.3.4.3.5 ReadDataByIdentifier service negative response parameter

The ReadDataByIdentifier service negative response parameters shown in Table 22 shall be used by the server to inform the client that the service request has not been performed.

NOTE Response codes are specified in Annex B.

Table 22 — ReadDataByIdentifier negative response parameters

A_Data byte	Parameter name	Cvt	Hex value
#1	Negative service response identifier	M	7F
#2	ReadDataByIdentifier request service identifier	M	22
#3	ResponseCode	M	00-FF

6.3.5 Enhanced diagnostic services

Enhanced diagnostic services and service parameters shall comply with ISO 14229-1.

6.4 Application layer protocol

6.4.1 General

The application layer protocol specifies the communication between peer application layer entities on a data link between a towed and a towing vehicle for diagnostic purposes.

The application layer protocol data unit shall have the following general format:

```
<A_PDU> =   {
            <SA>;
            <TA>;
            <TA_type>;
            <RA>;
            <A_DATA>
            }
```

6.4.2 Application layer protocol parameter SA, TA, TA_Type, RA

The application layer protocol parameters SA, TA, TA_Type and RA shall correspond to the respective application layer service parameter.

6.4.3 Application layer protocol control information (A_PCI)

The application layer protocol control information (A_PCI) shall consist of the diagnostic service identifier.

```
A_PCI = {
        <ServiceIdentifier>
        }
```

The ServiceIdentifier identifies the service and respective service parameters of the service primitive.

```
ServiceIdentifier = {
                    <<Service Name> request service identifier>|
                    <<Service Name> response service identifier>|
                    {<NegativeResponse service identifier>;<<Service Name> request service identifier>}
                    }
```

6.4.4 Application layer protocol data (A_DATA)

The application layer protocol data shall correspond to the respective application layer service identifier and service parameter.

```
A_DATA =  {
          <ServiceIdentifier>;
          <ServiceParameter>
          }
```

6.4.5 Application layer protocol data units (A_PDU)

The application layer protocol data unit A_PDU consist of the data communicated between peer application layer entities.

```
A_PDU     =  {
             <Request>|
             <Indication>|
             <Response>|
             <Confirmation>
             }

Request   =  {
             <SA>;
             <TA>;
             <TA_Type>;
             <RA>;
             <<Service Name> request service identifier>;
             <<Service Name> request service parameter>
             }

Indication  =  <Request>
```

```
Response =  {
            <SA>;
            <TA>;
            <TA_Type>;
            <RA>;
            {<<Service Name> response service identifier>;
             <<Service Name> response service parameter>} |
            {<NegativeResponse service identifier>;
             <<Service Name> request service identifier>;
             <<Service Name> negative response service parameter>}
            }
```

Confirmation = <Response>

6.4.6 Application layer protocol timing

The specification of the general requirements for the communication of the application layer entities is independent of different diagnostic sessions. The monitoring of the application layer timing shall be performed by the client and server application layer entities.

The A_PDU timing diagram for confirmed services shall be as shown in Figure 5 for a service request with a physical target address and as shown in Figure 6 for a service request with a functional target address. The corresponding timing parameter values shall be as specified in Table 23.

Application layer time out values shall be as specified in Table 24, corresponding application layer time out conditions shall be as specified in Table 25.

In the case of a confirmed service a new service request shall not be sent before the service confirmation of the previous service request is received or the specific time out has been expired.

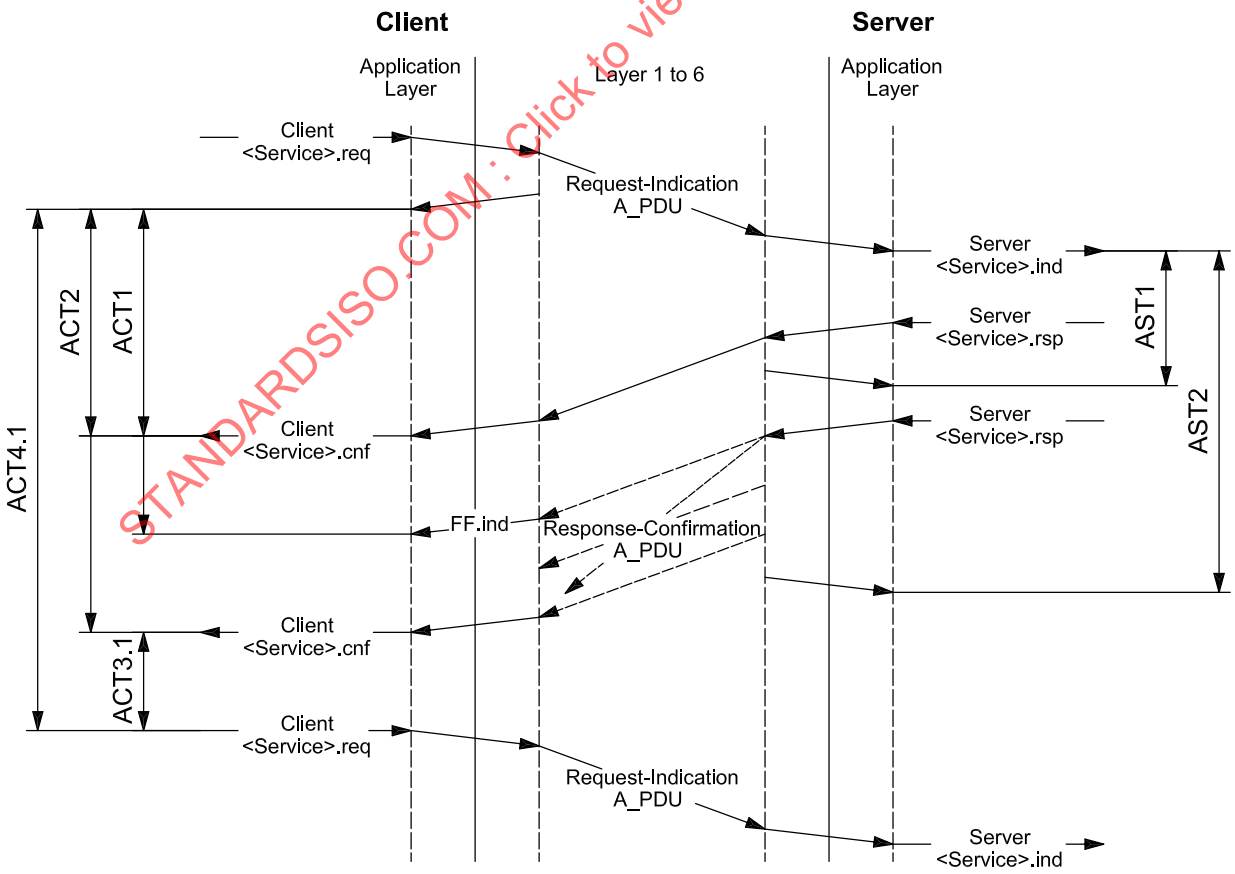

Figure 5 — A_PDU timing diagram confirmed services — physical target address

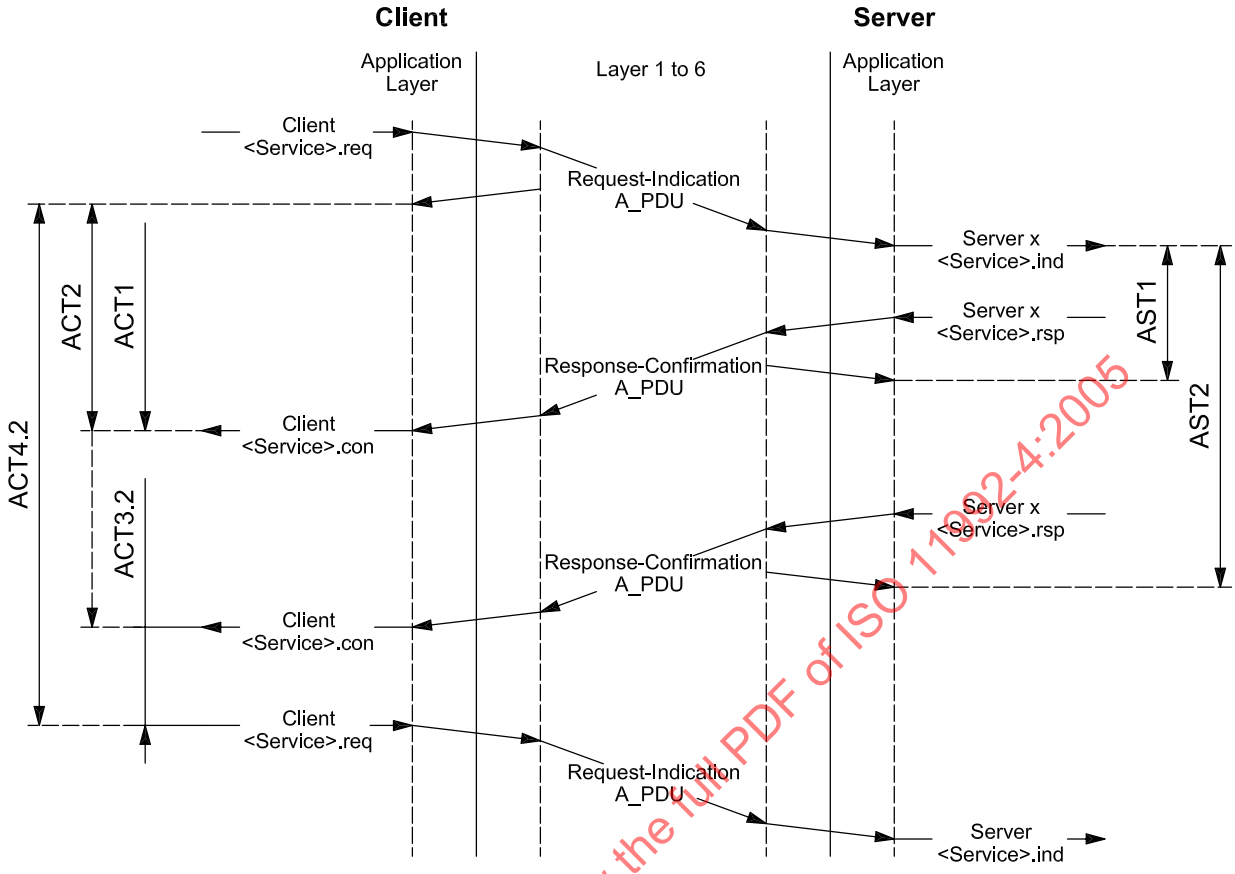

Figure 6 — A_PDU timing diagram confirmed services — functional target address

Table 23 — Application layer timing values

Name	Meaning		
ACT1	First client confirmation/indication after client request. Delay time between request and confirmation/indication.	min. max.	0 3 000 ms
ACT2	Final client confirmation. Total execution and transmission time.	min. max.	0 10 000 ms
ACT3.1	Client request after client confirmation. Service request delay time after service confirmation.	min. max.	0 ∞
ACT3.2	Client request after client confirmation. Service request delay time after service confirmation.	min. max.	1 000 ms ∞
ACT4.1	Client request repetition time. Determines the busload.	min. max.	100 ms ∞
ACT4.2	Client request repetition time. Determines the busload.	min. max.	ACT1max ∞
AST1	Server response time. Delay time between server indication and first response.	min. max.	0 1 000 ms
AST2	Server response time. Total server execution time.	min. max.	0 8 000 ms
NOTE A first confirmation is indicated after the reception of a negative service response with 'Request correctly received - response pending' (response code 78_{16}) or after the reception of a first frame of a multi frame message.			

Table 24 — Application layer time outs

Name	Time out	Purpose/Failure
ACTO1	ACT1max	Client positive or negative confirmation or FirstFrame indication after client request failed.
ACTO2	ACT2max	Final client confirmation failed after negative confirmation with response code 78_{16} or a FirstFrame indication had been received.
ASTO1	AST1max	Server response after server indication failed.
ASTO2	AST2max	Final server response failed after negative server response with response code 78_{16} had been sent.

Table 25 — Application layer time out conditions

Name	(Re-)Start conditions	Stop conditions
ACTO1	Service request primitive has been sent. > Service confirmation from network layer to client application layer.	FirstFrame indication or service positive/ negative confirmation primitive to client application.
ACTO2	Service request primitive has been sent. > Service confirmation from network layer to client application layer.	Final service positive or negative confirmation primitive (except with response code 78_{16}) to client application.
ASTO1	Service indication primitive to server application.	Service positive / negative response primitive has been sent. > Service confirmation from network layer to server application layer.
ASTO2	Service indication primitive to server application.	Final service positive or negative response primitive (except with response code 78_{16}) has been sent. > Service confirmation from network layer to server application layer.

6.4.7 Application layer protocol error handling

The protocol error handling on the application layer level is based on the application layer protocol timing specification. Table 26 specifies the client and the server reaction for the specified time outs.

Table 26 — Application layer error conditions and error handling

Error condition	Application layer entity activity	
	Client	Server
ACTO1 or ACTO2 expired	Report time out condition to the client application.	—
ASTO1 or ASTO2 expired	—	Report time out condition to the server application. A following service response from the application shall be executed.
Unknown service request/indication	—	Send a negative service response with an appropriate response code.
Unknown service response/confirmation	Report the error condition to the client application.	—
Parameter is missing in the service request/indication.	—	Send negative service response with an appropriate response code.
Parameter is missing in the service response/confirmation	Report the error condition to the client application.	—

7 Presentation layer specification

No specific presentation layer functions have been specified. Presentation layer primitives are one to one mapped on session layer primitives and vice versa without any further action.

8 Session layer specification

No specific session layer functions have been specified. Session layer primitives are one to one mapped on transport layer primitives and vice versa without any further action.

NOTE To start, stop and maintain a diagnostic session is the task of the client and server applications and is not a communication related function. Therefore, the necessary services are provided by the application layer entities to the respective application.

9 Transport layer specification

No specific transport layer functions have been specified. Transport layer primitives are one to one mapped on network layer primitives and vice versa without any further action.

NOTE The functionality for segmenting and reassembling of data records is provided by the network layer entities. Transport layer addresses are one to one mapped on network addresses.

10 Network layer specification

10.1 General

The network layer function, service and protocol specifications comply with ISO 15765-2. In case of differences the specifications of this part of ISO 11992 shall have precedence.

For the communication between road vehicles and their towed vehicles the restrictions described in this clause shall apply in addition.

10.2 Network layer functions

10.2.1 General

The network layer entities provide functions for the transport of diagnostic data via the respective communication link. These functions shall be used by application layer entities requesting the respective network layer service via the layers in between. The communication between two network layer entities shall be of the half-duplex type.

10.2.2 Message routing

The network layer entities shall provide message routing functions to support communication between servers and clients on the road vehicle (main) local network and the trailer (remote) local networks. The routing mechanism and possible address format conversion algorithms within each vehicle local network are left open to the user of this part of ISO 11992.

An example is given in Annex C.

10.2.3 Segmented message transmission

For the diagnostic message transmission, the network layer protocol is used on specified diagnostic communication channels (see 10.2.6).

NOTE For the diagnostic communication, the network layer protocol is used even if less than 8 bytes are transmitted (SingleFrame transmission).

10.2.4 Message addressing

Remote network addressing is used by means of the mixed address format. The network layer address information N_AI shall be encapsulated in the data link layer service parameters and the first byte of the data link layer data field. The relation between network layer address information, PDU1 data link layer parameters and the identifier specification according to ISO 11992-2 and ISO 11992-3 is given in 11.2.4.

10.2.5 Establishing, maintaining and termination of connections

For the diagnostic data transmission between towing and towed vehicles the following limitations apply:

a) multiframe (segmented) messages shall use point to point (1 to 1) connections with physical addressing;

b) single frame messages may use multi point (1 to n) connectionless transmission with functional addressing or point to point (1 to 1) connections with physical addressing.

In case of a multiframe message, connections between peer entities shall be established and maintained automatically with the transmission/reception of the first flow control frame from the receiver of the message. A connection shall be automatically terminated after the transmission/reception of the last consecutive frame or the expiration of a correspondent time out. Only one connection between two peer network layer entities shall be allowed at the same time.

10.2.6 Diagnostic communication channels (DCC)

The network layer parameter MTYPE (see 10.3.2) and the meaning of the network layer extended address N_AE (see 10.3.3.5) mapped in the first data byte of each CAN frame is specified by means of diagnostic communications channels DCC. The DCC naming convention is shown in Figure 7.

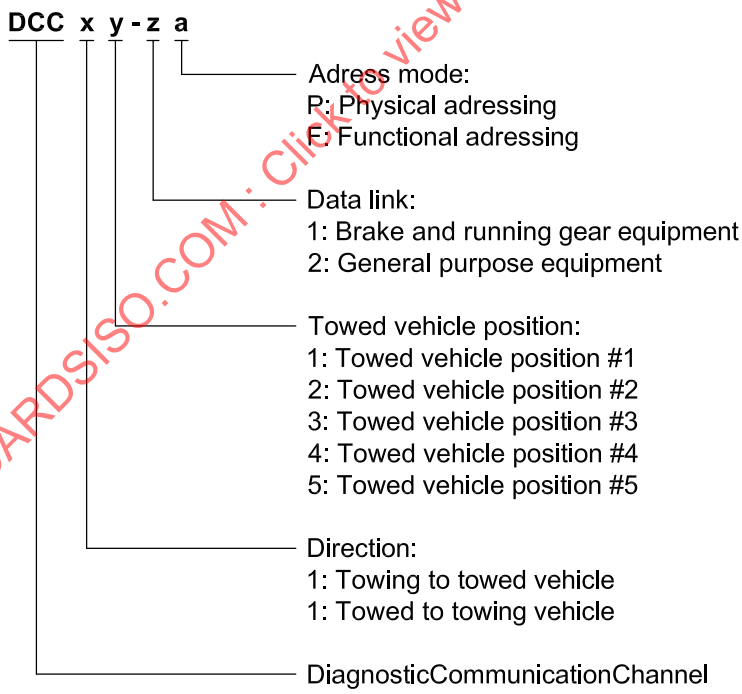

Figure 7 — DCC naming convention

Each communication channel is identified by the following identifier parameters:

a) DataPage DP;

b) ParameterFormat PF;

c) DestinationAddress DA or SourceAddress SA.

NOTE DA corresponds to the PS field according to the PDU1 data link layer format (see Clause 11).

The resulting diagnostic communication channels shall be as specified in Tables 27 to 30.

Table 27 — DCCs for braking and running gear equipment — Physical addressing

DiagnosticCommunicationChannel		PRIO	RES	DP	PF	DA	SA	Identifier (hex)	Data field AddressExtension
Commercial vehicle to towed vehicle 1	DCC11-1P	7	0	0	206	200	xx	1CCEC8xx	TargetAddressExtension
Commercial vehicle to towed vehicle 2	DCC12-1P	7	0	0	206	192	xx	1CCEC0xx	TargetAddressExtension
Commercial vehicle to towed vehicle 3	DCC13-1P	7	0	0	206	184	xx	1CCEB8xx	TargetAddressExtension
Commercial vehicle to towed vehicle 4	DCC14-1P	7	0	0	206	176	xx	1CCEB0xx	TargetAddressExtension
Commercial vehicle to towed vehicle 5	DCC15-1P	7	0	0	206	168	xx	1CCEA8xx	TargetAddressExtension
Towed vehicle 1 to commercial vehicle	DCC21-1P	7	0	0	206	xx	200	1CCExxC8	SourceAddressExtension
Towed vehicle 2 to commercial vehicle	DCC22-1P	7	0	0	206	xx	192	1CCExxC0	SourceAddressExtension
Towed vehicle 3 to commercial vehicle	DCC23-1P	7	0	0	206	xx	184	1CCExxB8	SourceAddressExtension
Towed vehicle 4 to commercial vehicle	DCC24-1P	7	0	0	206	xx	176	1CCExxB0	SourceAddressExtension
Towed vehicle 5 to commercial vehicle	DCC25-1P	7	0	0	206	xx	168	1CCExxA8	SourceAddressExtension

Table 28 — DCCs for braking and running gear equipment — Functional addressing

DiagnosticCommunicationChannel		PRIO	RES	DP	PF	DA	SA	Identifier (hex)	Data field AddressExtension
Commercial vehicle to towed vehicle 1	DCC11-1F	7	0	0	205	200	xx	1CCDC8xx	TargetAddressExtension
Commercial vehicle to towed vehicle 2	DCC12-1F	7	0	0	205	192	xx	1CCDC0xx	TargetAddressExtension
Commercial vehicle to towed vehicle 3	DCC13-1F	7	0	0	205	184	xx	1CCDB8xx	TargetAddressExtension
Commercial vehicle to towed vehicle 4	DCC14-1F	7	0	0	205	176	xx	1CCDB0xx	TargetAddressExtension
Commercial vehicle to towed vehicle 5	DCC15-1F	7	0	0	205	168	xx	1CCDA8xx	TargetAddressExtension
Towed vehicle 1 to commercial vehicle	DCC21-1F	7	0	0	205	xx	200	1CCDxxC8	SourceAddressExtension
Towed vehicle 2 to commercial vehicle	DCC22-1F	7	0	0	205	xx	192	1CCDxxC0	SourceAddressExtension
Towed vehicle 3 to commercial vehicle	DCC23-1F	7	0	0	205	xx	184	1CCDxxB8	SourceAddressExtension
Towed vehicle 4 to commercial vehicle	DCC24-1F	7	0	0	205	xx	176	1CCDxxB0	SourceAddressExtension
Towed vehicle 5 to commercial vehicle	DCC25-1F	7	0	0	205	xx	168	1CCDxxA8	SourceAddressExtension

Table 29 — DCCs for general purpose equipment — Physical addressing

DiagnosticCommunicationChannel		PRIO	RES	DP	PF	DA	SA	Identifier (hex)	Data field AddressExtension
Commercial vehicle to towed vehicle 1	DCC11-2P	7	0	0	206	201	xx	1CCEC9xx	TargetAddressExtension
Commercial vehicle to towed vehicle 2	DCC12-2P	7	0	0	206	193	xx	1CCEC1xx	TargetAddressExtension
Commercial vehicle to towed vehicle 3	DCC13-2P	7	0	0	206	185	xx	1CCEB9xx	TargetAddressExtension
Commercial vehicle to towed vehicle 4	DCC14-2P	7	0	0	206	177	xx	1CCEB1xx	TargetAddressExtension
Commercial vehicle to towed vehicle 5	DCC15-2P	7	0	0	206	169	xx	1CCEA9xx	TargetAddressExtension
Towed vehicle 1 to commercial vehicle	DCC21-2P	7	0	0	206	xx	201	1CCExxC9	SourceAddressExtension
Towed vehicle 2 to commercial vehicle	DCC22-2P	7	0	0	206	xx	193	1CCExxC1	SourceAddressExtension
Towed vehicle 3 to commercial vehicle	DCC23-2P	7	0	0	206	xx	185	1CCExxB9	SourceAddressExtension
Towed vehicle 4 to commercial vehicle	DCC24-2P	7	0	0	206	xx	177	1CCExxB1	SourceAddressExtension
Towed vehicle 5 to commercial vehicle	DCC25-2P	7	0	0	206	xx	169	1CCExxA9	SourceAddressExtension

Table 30 — DCCs for general purpose equipment — Functional addressing

DiagnosticCommunicationChannel		PRIO	RES	DP	PF	DA	SA	Identifier (hex)	Data field AddressExtension
Commercial vehicle to towed vehicle 1	DCC11-2F	7	0	0	205	201	xx	1CCDC9xx	TargetAddressExtension
Commercial vehicle to towed vehicle 2	DCC12-2F	7	0	0	205	193	xx	1CCDC1xx	TargetAddressExtension
Commercial vehicle to towed vehicle 3	DCC13-2F	7	0	0	205	185	xx	1CCDB9xx	TargetAddressExtension
Commercial vehicle to towed vehicle 4	DCC14-2F	7	0	0	205	177	xx	1CCDB1xx	TargetAddressExtension
Commercial vehicle to towed vehicle 5	DCC15-2F	7	0	0	205	169	xx	1CCDA9xx	TargetAddressExtension
Towed vehicle 1 to commercial vehicle	DCC21-2F	7	0	0	205	xx	201	1CCDxxC9	SourceAddressExtension
Towed vehicle 2 to commercial vehicle	DCC22-2F	7	0	0	205	xx	193	1CCDxxC1	SourceAddressExtension
Towed vehicle 3 to commercial vehicle	DCC23-2F	7	0	0	205	xx	185	1CCDxxB9	SourceAddressExtension
Towed vehicle 4 to commercial vehicle	DCC24-2F	7	0	0	205	xx	177	1CCDxxB1	SourceAddressExtension
Towed vehicle 5 to commercial vehicle	DCC25-2F	7	0	0	205	xx	169	1CCDxxA9	SourceAddressExtension

10.2.7 Protocol timing supervision

The peer network layer entities communicating over a connection shall supervise the protocol timing and shall take the respective actions in the case a specified time out occurs.

10.3 Network layer services

10.3.1 General

The network layer entities shall support the N_USDATA network layer service which provides functions to segment, transmit and assemble data records. The N_ChangeParameter service shall not be supported.

10.3.2 Network layer service parameter

10.3.2.1 General

This subclause specifies the network layer service parameters used for diagnostic communication.

10.3.2.2 Message type (MTYPE)

The parameter MTYPE identifies the message data content and address format. For the diagnostic communication between road vehicles and their towed vehicles the MTYPE parameter shall be coded by means of diagnostic communication channels (see 10.2.6).

MTYPE = <Remote Diagnostics>

10.3.2.3 Network layer address information (N_AI)

10.3.2.3.1 General

The network layer address information N_AI shall be a data record which consists of the following parameters.

```
N_AI: Record =  {
                <N_TAtype>;
                <N_TA>;
                <N_SA>;
                <N_AE>
                }
```

10.3.2.3.2 Network layer target address type (N_TAtype)

This parameter contains the message priority, the message data content, the target address type and the address type of the network layer address extension AN_AE in the case that it represents a target address extension.

```
N_TAtype: Record =  {
                    <Priority>;
                    <ReservedBit>;
                    <DataPage>;
                    <PDUFormat>
                    }
```

10.3.2.3.2.1 Priority (PRIO)

The priority shall be used to optimize the message latency time.

PRIO: 3 Bit = 7

10.3.2.3.2.2 Reserved bit (RES)

This bit is reserved for future expansion.

RES: 1 Bit = 0

10.3.2.3.2.3 Data page (DP)

The data page (DP) identifies together with the PDUFormat parameter the message data content.

DP: 1 Bit = 0

10.3.2.3.2.4 PDU format (PF)

The PDU format (PF) parameter determines together with the data page information (DP) the PDU format, the message data content, the target address type and the address type of the address extension in the case that N_AE represents a target address extension.

PF: 8 Bit = { 206_{10} | 205_{10} }

206_{10} = PDU1 (PS=DA); Diagnostic Message; Mixed addressing; Physical target addresses.

205_{10} = PDU1 (PS=DA); Diagnostic Message; Mixed addressing; Functional target addresses.

10.3.2.3.3 Network layer target address (N_TA)

The network layer target address N_TA represents the target address of the message receiver. The parameter shall correspond to the destination address DA of the ISO 11992 identifier definition.

N_TA: 8 Bit = {
<Towed vehicle target address>|
<Towing vehicle client target address>|
<Towing vehicle server target address>
}

10.3.2.3.4 Network layer source address (N_SA)

The network layer source address N_SA represents the source address of the message sender. This parameter shall correspond to the source address SA of the ISO 11992 identifier definition.

N_SA: 8 Bit = {
<Towed vehicle source address>|
<Towing vehicle client source address>|
<Towing vehicle server source address>
}

10.3.2.3.5 Network layer address extension (N_AE)

The network layer address extension N_AE represents a client or server local address on the towed vehicle local network.

Whether N_AE represents a target or source address is specified by means of diagnostic communication channels (see 10.2.6).

N_AE: 8 Bit = {<TargetAddressExtension> |
<SourceAddressExtension> }

10.3.2.4 Message data (A_DATA)

The MessageData represents the data record which shall be transmitted.

MessageData: Record = {<A_DATA>}

10.3.2.5 Message length (Length)

The parameter Length identifies the number of message data bytes. The maximum length allowed is 255_{10} bytes.

Length = { <N>; 0 <= N <= 255_{10}}

NOTE The USDT protocol allows a maximum of 4095_{10} MessageData bytes. For the diagnostic communication between towing and towed vehicles, the length is limited to 255_{10} bytes.

10.3.2.6 Network layer service request result (N_Result)

N_Result is an internal network layer parameter which identifies the result of a service request or indication.

```
N_Result =  {
            N_OK |
            N_TIMEOUT_A |
            N_TIMEOUT_Bs |
            N_TIMEOUT_Cr |
            N_WRONG_SN |
            N_INVALID_FS |
            N_UNEXP_PDU |
            N_WFT_OVRN |
            N_BUFFER_OVFLW |
            N_UNEXPECTED_DLC |
            N_ERROR
            }
```

where

N_OK:	The service execution has completed successfully.
N_TIMEOUT_A:	Timer N_Ar/N_As has passed its timeout value N_As_{max}/N_Ar_{max}.
N_TIMEOUT_Bs:	Timer N_Bs has passed its timeout value N_Bs_{max}.
N_TIMEOUT_Cr:	The timer N_Cr has passed its timeout value N_Cr_{max}.
N_WRONG_SN:	Reception of an unexpected sequence number (PCI.SN) value.
N_INVALID_FS:	Invalid or unknown FlowStatus value has been received in a flow control (FC) N_PDU.
N_UNEXP_PDU:	Reception of an unexpected protocol data unit.
N_WFT_OVRN:	Reception of a flow control WAIT frame that exceeds the maximum counter N_WFTmax.
N_BUFFER_OVFLW:	Reception of a flow control (FC) N_PDU with FlowStatus = OVFLW.
N_UNEXPECTED_DLC:	N_PDU has been received with fewer CAN data bytes than expected.
N_ERROR:	General error. An error has been detected by the network layer entity and no other parameter value can be used to better describe the error.

10.3.3 Network layer service data units (N_SDU)

10.3.3.1 General

The network layer service data units (N_SDU) have the following general format:

```
<Service name>.<Service primitive> =  {
                                      <MTYPE>
                                      <N_AI>
                                      <Service primitive parameter>
                                      }
```

10.3.3.2 N_USData.Request service primitive

The service primitive requests transmission of <MessageData> with <Length> bytes from the sender to the receiver peer entity identified by <N_AI>.

```
N_USData.Request =  {
                    <MTYPE>;
                    <N_AI>;
                    <MessageData>;
                    <Length>;
                    }
```

10.3.3.3 N_USData.Confirmation service primitive

The service primitive confirms the completion of a N_USData service request identified by <N_AI> parameter. The parameter <N_Result> provides the status of the service request.

```
N_USData.Confirmation = {
                        <MTYPE>;
                        <N_AI>;
                        <N_Result>;
                        }
```

10.3.3.4 N_USData_FF.Indication service primitive

The service primitive indicates the reception of a first frame (FF) of a segmented message with <Length> bytes from a peer entity identified by <N_AI> to the adjacent upper layer.

```
N_USData.Indication =   {
                        <MTYPE>;
                        <N_AI>;
                        <Length>;
                        }
```

10.3.3.5 N_USData.Indication service primitive

The service primitive indicates <N_Result> events and delivers <MessageData> with <Length> bytes received from a peer entity identified by <N_AI> to the adjacent upper layer.

```
N_USData.Indication =   {
                        <MTYPE>;
                        <N_AI>;
                        <MessageData>;
                        <Length>;
                        <N_Result>
                        }
```

10.4 Network layer protocol

10.4.1 General

The network layer protocol describes the communication between peer network layer entities on a data link between a towed and a towing vehicle.

10.4.2 Network layer protocol parameter

10.4.2.1 Network layer protocol address information (N_AI)

The network layer protocol address information N_AI shall correspond to the respective network layer address information.

10.4.2.2 Network layer protocol control information (N_PCI)

10.4.2.2.1 General

The network layer protocol control information N_PCI shall be used in the network layer PDU. It encodes the network layer PDU type and network layer protocol control information.

N_PCI = {
<SingleFrame PCI>|
<FirstFrame PCI>|
<ConsecutiveFrame PCI>|
<FlowControl PCI>
}

The encoding of the network layer protocol information N_PCI shall be as shown in Table 31.

Table 31 — Encoding of network layer protocol control information

N_PCI		Protocol control information (N_PCI)									
		Byte #1								Byte #2	Byte #3
PCI name	Mnemonic	7	6	5	4	3	2	1	0		
SingleFrame PCI	SF_PCI	0	0	0	0	SF.DL				–	–
FirstFrame PCI	FF_PCI	0	0	0	1	FF.DL					–
						high nibble				low byte	–
ConsecutiveFrame PCI	CF_PCI	0	0	1	0	SN				–	–
FlowControl PCI	FC_PCI	0	0	1	1	FS				BS	STmin
ReservedByDocument	RBD	40_{16} to FF_{16}								reserved	reserved

10.4.2.2.2 SingleFrame.DataLength (SF.DL)

The parameter SingleFrame.DataLength (SF.DL) determines the number of valid message data bytes of a single frame message.

SF.DL: 4 bit = { $1 \leqslant$ <N> $\leqslant 6_{10}$ }

NOTE The network layer protocol using the mixed address format allows a maximum of six (6) MessageData bytes in a single frame message.

10.4.2.2.3 FirstFrame.DataLength (FF.DL)

The parameter FirstFrame.DataLength (FF.DL) determines the number of message data bytes of a segmented multi frame message.

FF.DL: 8 bit = { $7_{10} \leqslant$ <N> $\leqslant 255_{10}$ }

NOTE The USDT protocol allows a maximum of 4095_{10} MessageData bytes. For the diagnostic communication between towing and towed vehicles the length is limited to 255_{10} bytes.

10.4.2.2.4 SequenceNumber (SN)

The parameter SequenceNumber (SN) shall be used to check the sequence of consecutive frames in a segmented transmission. The SequenceNumber shall be set to one for the first consecutive frame of a segmented transmission and shall be incremented by one for each following consecutive frame. After the SequenceNumber reaches the value of fifteen, it shall be set to zero for the next consecutive frame.

SN: 4 bit = 0 ... 15_{10}

10.4.2.2.5 BlockSize (BS)

The parameter BlockSize (BS) shall be used by the receiving network layer peer entity in the flow control frame to request the transmission of a maximum number of consecutive frames by the sending network layer peer entity without an intermediate flow control frame.

BS: 8 bit = 1_{10} ... 15_{10}

The value BS = 0, i.e. no intermediate flow control frames, shall not be used.

10.4.2.2.6 SeparationTime (STmin)

The parameter SeparationTime (STmin) shall be used by the receiving network layer peer entity in the flow control frame to request a minimum time gap between the transmission of consecutive frames from the sending network layer peer entity.

STmin: 8 bit = 10_{10} ms to 127_{10} ms

10.4.2.2.7 FlowStatus (FS)

The FlowStatus (FS) shall be used by the receiving network layer peer entity in the FlowControl frame to indicate to the sender whether it is ready to receive <BS> consecutive frames sent with <ST> separation time.

FS: 4 bit = {
<ContinueToSend>|
<Wait>|
<Overflow>
}

ContinueToSend = CTS = 0000_2

Wait = WT = 0001_2

Overflow = OVFLW = 0010_2

BS and STmin transmitted with the first flow control frame shall be valid during the transmission of the whole message.

10.4.2.2.8 Network layer protocol data (N_DATA)

The network layer parameter N_DATA contains the segments of the message data record.

N_DATA = { < record of <N> MessageData bytes >; $1 \leqslant N \leqslant 6_{10}$ }

NOTE For diagnostic communication between towing and towed vehicles this data record corresponds to the segmented application layer parameter ServiceIdentifier and ServiceParameter (A_DATA).

10.4.2.3 Maximum number of FC.Wait frame transmissions (N_WFTmax)

The local entity parameter maximum number of FC.Wait frame transmissions N_WFTmax defines the allowed maximum number of consecutive FlowControl frames with FlowStatus set to Wait.

N_WFTmax = 10_{10}

10.4.3 Network layer protocol data units (N_PDU)

10.4.3.1 General

Four different network layer protocol data units (N_PDU) are specified which correspond to the MixedAddressFormat:

N_PDU = {
<SingleFrame>|
<FirstFrame>|
<ConsecutiveFrame>|
<FlowControl>
}

10.4.3.2 SingleFrame N_PDU

Unsegmented single frame message with up to six message data bytes.

SingleFrame = {
<N_AI>;
<SingleFrame PCI>;
<N_DATA>
}

10.4.3.3 FirstFrame N_PDU

First frame of a segmented message transmission with a message length from 7_{10} up to 255_{10} message data bytes.

FirstFrame = {
<N_AI>;
<FirstFrame PCI>;
<N_DATA>
}

10.4.3.4 ConsecutiveFrame N_PDU

Consecutive frame of a segmented transmission.

ConsecutiveFrame = {
<N_AI>;
<ConsecutiveFrame PCI>;
<N_DATA>
}

10.4.3.5 FlowControl N_PDU

Flow control frames are sent by the message receiver to control the transmission of consecutive frames sent by the message transmitter.

FlowControlFrame = {
<N_AI>;
<FlowControlFrame PCI>
}

10.4.4 Network layer protocol timing

The network layer timing diagram is shown in Figures 8 and 9, the corresponding timing parameters values shall be as specified in Table 32. Network layer time out values shall be as specified in Table 33, corresponding network layer time out conditions shall be as specified in Table 34.

NOTE Modified timeout values for timing parameters N_AS, N_AR, N_BS and N_CR compared with ISO 15765-2.

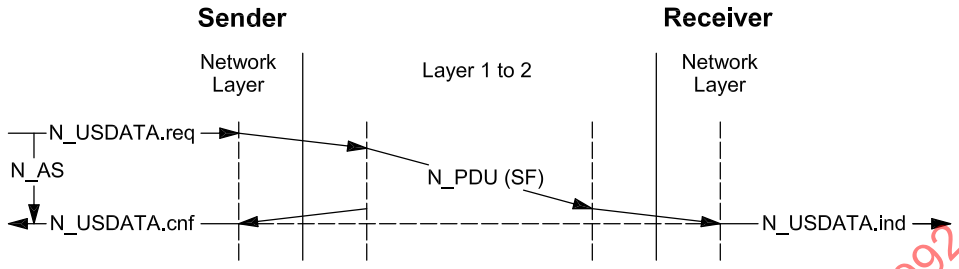

Figure 8 — Network layer timing diagram — Single frame message transmission

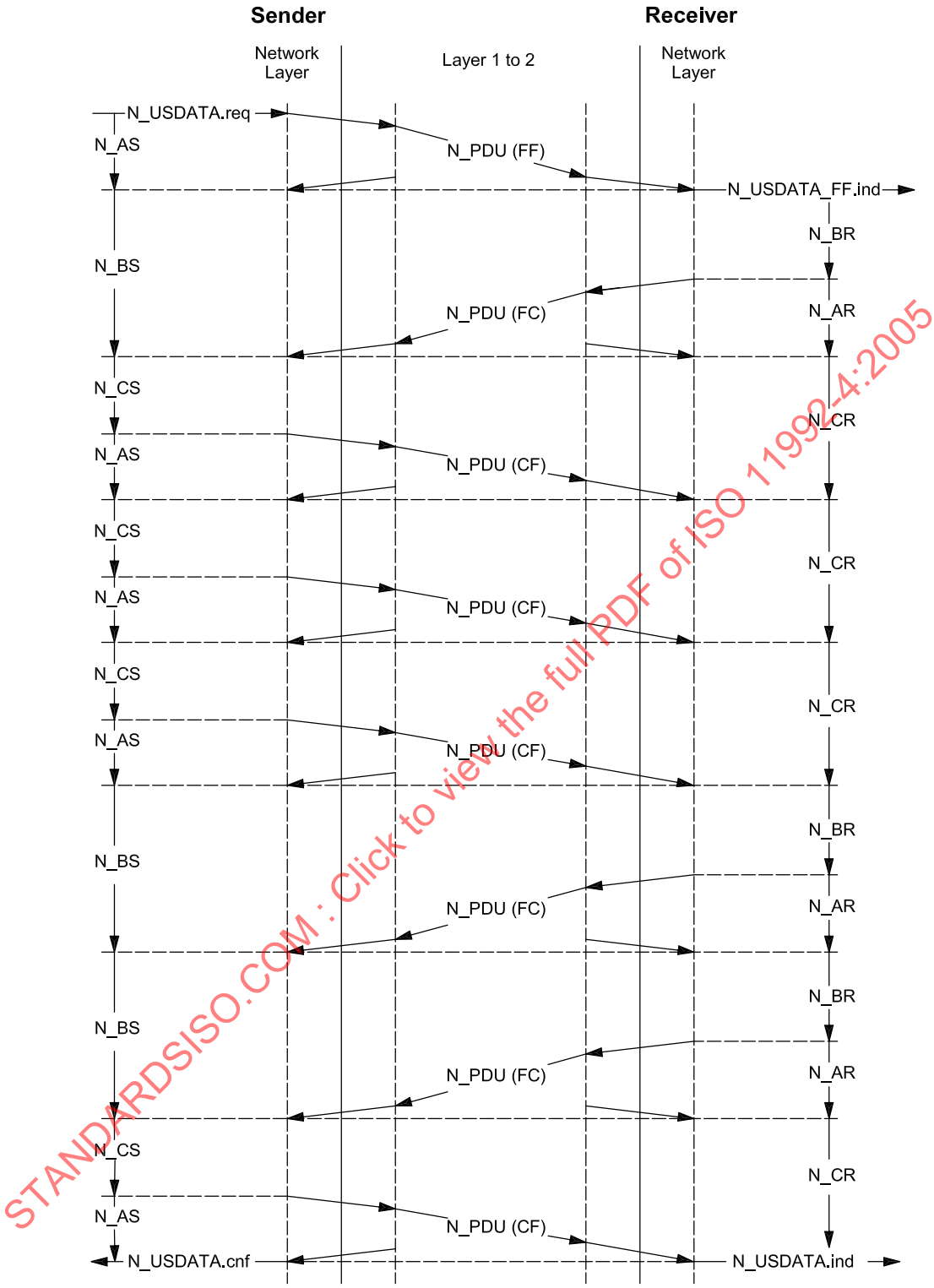

Figure 9 — Network layer timing diagram — Segmented message transmission

Table 32 — Network layer timing values

Name	Description		
N_As	Transmission delay time of a N_PDU frame on the sender side	min. max.	0 100 ms
N_Ar	Transmission delay time of a N_PDU frame on the receiver side	min. max.	0 100 ms
N_Bs	Time between transmission of a FirstFrame or ConsecutiveFrame and reception of a FlowControlFrame or time between two consecutive FlowControlFrames.	min. max.	0 150 ms
N_Br	Time between reception of a FirstFrame or ConsecutiveFrame and transmission request of a FlowControlFrame.	min. max.	0 35 ms[a]
N_Cs	Time between reception of a FlowControlFrame or transmission of a ConsecutiveFrame and transmission request of next ConsecutiveFrame	min. max.	0 35 ms[b]
N_Cr	Time between transmission of a FlowControlFrame or reception of a ConsecutiveFrame and reception of a ConsecutiveFrame	min. max.	0 150 ms

[a] Sender internal performance requirement $(N\_Br_{max} + N\_Ar_{max}) < (0.9 \times N\_Bs_{max})$

[b] Receiver internal performance requirement $(N\_Cs_{max} + N\_As_{max}) < (0.9 \times N\_Cr_{max})$

Table 33 — Network layer time outs

Name	Time out	Description
NSTO1	N_As max	N_PDU transmission failed on sender side
NRTO1	N_Ar max	N_PDU transmission failed on receiver side
NSTO2	N_Bs max	FlowControlFrame not received on sender side
NRTO2	N_Cr max	ConsecutiveFrame not received on receiver side

Table 34 — Network layer time out conditions

Name	(Re-)Start conditions	Stop conditions
NSTO1	N_PDU transmission request from sender network layer entity > L_DATA request service primitive from network layer to data link layer entity	N_PDU transmission confirmation to sender network layer entity > L_DATA confirmation service primitive from data link layer to network layer entity
NRTO1	N_PDU transmission request from receiver network layer entity > L_DATA request service primitive from network layer to data link layer entity	N_PDU transmission confirmation to receiver network layer entity > L_DATA confirmation service primitive from data link layer to network layer entity
NSTO2	Transmission confirmation of FirstFrame or last ConsecutiveFrame of BlockSize frame number to sender network layer entity > L_DATA (FF or CF) confirmation service primitive from data link layer to network layer entity	FlowControl indication to sender network layer entity > L_DATA indication service primitive from data link layer to network layer entity
NRTO2	FlowControlFrame transmission confirmation or reception indication of a ConsecutiveFrame within BlockSize frame number to receiver network layer entity > L_DATA (FC) confirmation primitive or L_DATA (CF) indication service primitive from data link layer to network layer entity	Last ConsecutiveFrame reception confirmation of a message transmission to receiver network layer entity > L_DATA (CF) confirmation service primitive from data link layer to network layer

10.4.5 Network layer protocol error handling

The error handling on the network layer level is based on the network layer protocol timing specification. Table 35 specifies the sender and receiver reaction for the error conditions.

Table 35 — Network layer error conditions and error handling

Error condition	Network layer entity activity	
	Sender	Receiver
NSTO1 expired	Abort message transmission and issue N_USData.confirmation service primitive with N_Result = N_TIMEOUT_A	–
NRTO1 expired	–	Abort message reception and issue N_USData.indication service primitive with N_Result = N_TIMEOUT_A
NSTO2 expired	Abort message transmission and issue N_USData.confirmation service primitive with N_Result = N_TIMEOUT_Bs	–
NRTO2 expired	–	Abort message reception and issue N_USData.indication service primitive with N_Result = N_TIMEOUT_Cr
FlowControlFrame lost on sender side	Abort message transmission and issue N_USData.confirmation service primitive with N_Result = N_TIMEOUT_Bs	–
FirstFrame or ConsecutiveFrame lost on receiver side	–	Abort message reception and issue N_USData.indication service primitive with N_Result = N_TIMEOUT_Cr
Maximum number of wait frames N_WFTmax exceeded	Abort message transmission and issue N_USData.confirmation service primitive with N_Result = WFT_OVRN	–
No frame reception possible after maximum number of wait frames N_WFTmax has been sent	–	Abort message reception and issue N_USData.indication service primitive with N_Result = WFT_OVRN
Reception of an unexpected sequence number value	–	Abort message reception and issue N_USData.indication service primitive with N_Result = N_WRONG_SN
Invalid or unknwon FlowStatus value	Abort message transmission and issue N_USData.confirmation service primitive with N_Result = N_INVALID_FS	–
Reception of an unexpected N_PDU	Abort message transmission and issue N_USData.confirmation service primitive with N_Result = N_UNEXP_PDU	Abort message reception and issue N_USData.indication service primitive with N_Result = N_UNEXP_PDU
Buffer overflow	–	Abort message reception, issue N_USData.indication primitive with N_Result = N_BUFFER_OVFL and send FlowControlFrame with FlowStatus = <Overflow>
Reception of FlowControlFrame with FlowStatus = <Overflow>	Abort message transmission and issue N_USData.confirmation service primitive with N_Result = N_BUFFER_OVFL	–
Reception of N_PDU with less than 8 data bytes	Abort message transmission and issue N_USData.confirmation service primitive with N_Result = N_UNEXPECTED_DLC	Abort message reception and issue N_USData.indication service primitive with N_Result = N_UNEXPECTED_DLC
Detection of an undefined error	Abort message transmission and issue N_USData.confirmation service primitive with N_Result = N_ERROR	Abort message reception and issue N_USData.indication service primitive with N_Result = N_ERROR

11 Data link layer specification

11.1 General

In accordance with ISO 11992-1, ISO 11992-2 and ISO 11992-3 the data link layer shall comply with ISO 11898-1 with the following limitations:

a) only 29-bit CAN identifier shall be used;

b) CAN remote request frames shall not be used.

See Annex D.

The PDU1 type message format shall be used for diagnostic communication between a road vehicle and its towed vehicles.

11.2 Data link layer service parameter

11.2.1 General

The specified data link layer service parameter shall comply with ISO 11898-1.

11.2.2 Data length code (DLC)

The number of DATA bytes of a CAN message shall be fixed at 8 bytes.

DLC: 4 Bit = 8_{10}

11.2.3 Message data (DATA)

This parameter contains the network layer extended address information, protocol control information and data record.

DATA: Record = {
<N_AE>
<N_PCI>
<N_DATA>
}

DATA shall always consist of 8 bytes. Unused bytes may have any value.

The encoding of data link layer DATA bytes and the relation to the network layer parameter shall be as specified Table 36.

Table 36 — Encoding of N_PDU information in the DATA bytes

Description		DATA bytes							
	(mnemonic)	1	2	3	4	5	6	7	8
SingleFrame	SF	<N_AE>	<SF PCI>	<N_DATA>					
FirstFrame	FF	<N_AE>	<FF PCI>		< N_DATA >				
ConsecutiveFrame	CF	<N_AE>	<CF PCI>	< N_DATA >					
FlowControl frame	FC	<N_AE>	<FC PCI>			< N_DATA >			

11.2.4 CAN message identifier (IDENTIFIER)

The CAN message identifier is determined by the network layer entities based on the parameters N_TAtype, N_TA and N_SA of the network layer address information N_AI.

IDENTIFIER: 29 bit = {
<N_TAtype>
<N_TA>
<N_SA>
}

The encoding of the data link layer IDENTIFIER parameter and the relation to the network layer parameter is specified in Table 37.

Table 37 — Network layer address encoding in the IDENTIFIER parameter

N parameter	**N_AI — (Network layer address information)**							
Bit	–				7 0	7 0	7 0	
	N_TAtype				**N_TA**	**N_SA**	**N_AE**	
ISO 11992 identifier	**PRIO**		**RES**	**DP**	**PF**	**PS (DA)**	**SA**	–
Bit	28 26		25	24	23 16	15 8	7 0	7 0
Byte								Byte 1
DL parameter	**IDENTIFIER**							**DATA**

12 Physical layer specification

The physical layer shall be as specified in ISO 11992-1.

Annex A
(normative)

Addresses

The physical addresses of the data link between towing and towed vehicle and of the local trailer network shall be in accordance with Tables A.1 and A.2.

The functional addresses of the data link between towing and towed vehicle and of the local trailer network shall be in accordance with Tables A.3 and A.4.

Table A.1 — Towing–towed vehicle data link — Physical addresses

Address		Name
0_{10} – 31_{10}	0_{16} – $1F_{16}$	Reserved
32_{10}	20_{16}	Tractor bridge for braking and running gear equipment
33_{10} – 167_{10}	21_{16} – $A7_{16}$	Reserved
168_{10}	$A8_{16}$	Trailer #5 Braking and running gear equipment
169_{10}	$A9_{16}$	Trailer #5 Other than braking and running gear equipment
170_{10} – 175_{10}	AA_{16} – AF_{16}	Reserved
176_{10}	$B0_{16}$	Trailer #4 Braking and running gear equipment
177_{10}	$B1_{16}$	Trailer #4 Other than braking and running gear equipment
178_{10} – 183_{10}	$B2_{16}$ – $B7_{16}$	Reserved
184_{10}	$B8_{16}$	Trailer #3 Braking and running gear equipment
185_{10}	$B9_{16}$	Trailer #3 Other than braking and running gear equipment
186_{10} – 191_{10}	BA_{16} – BF_{16}	Reserved
192_{10}	$C0_{16}$	Trailer #2 Braking and running gear equipment
193_{10}	$C1_{16}$	Trailer #2 Other than braking and running gear equipment
194_{10} – 199_{10}	$C2_{16}$ – $C7_{16}$	Reserved
200_{10}	$C8_{16}$	Trailer #1 Braking and running gear equipment
201_{10}	$C9_{16}$	Trailer #1 Other than braking and running gear equipment
202_{10} – 234_{10}	CA_{16} – EA_{16}	Reserved
235_{10}	EB_{16}	Tractor bridge for other than braking and running gear equipment
236_{10} – 255_{10}	EC_{16} – FF_{16}	Reserved

Table A.2 — Trailer local network — Physical addresses

Address		Name
0_{10}	0_{16}	Trailer gateway application
1_{10}	1_{16}	Brake & running gear, tractor-trailer interface
2_{10}	2_{16}	General purpose, tractor-trailer interface
3_{10} – 254_{10}	3_{16} – FE_{16}	System or vehicle manufacturer specific
255_{10}	FF_{16}	Reserved

Table A.3 — Towing–towed data link — Functional addresses

Address		Name
0_{10} – 254_{10}	0_{16} – FE_{16}	Reserved
255_{10}	FF_{16}	GLOBAL (All/Any Node)

Table A.4 — Trailer local network — Functional addresses

Address		Name
0_{10} – 254_{10}	0_{16} – FE_{16}	System or vehicle manufacturer specific
255_{10}	FF_{16}	GLOBAL (All/Any Node)

Annex B
(normative)

Basic diagnostic service parameters

B.1 Diagnostic trouble codes (DTC) format

The format for diagnostic trouble code records shall be in accordance with Figure B.1.

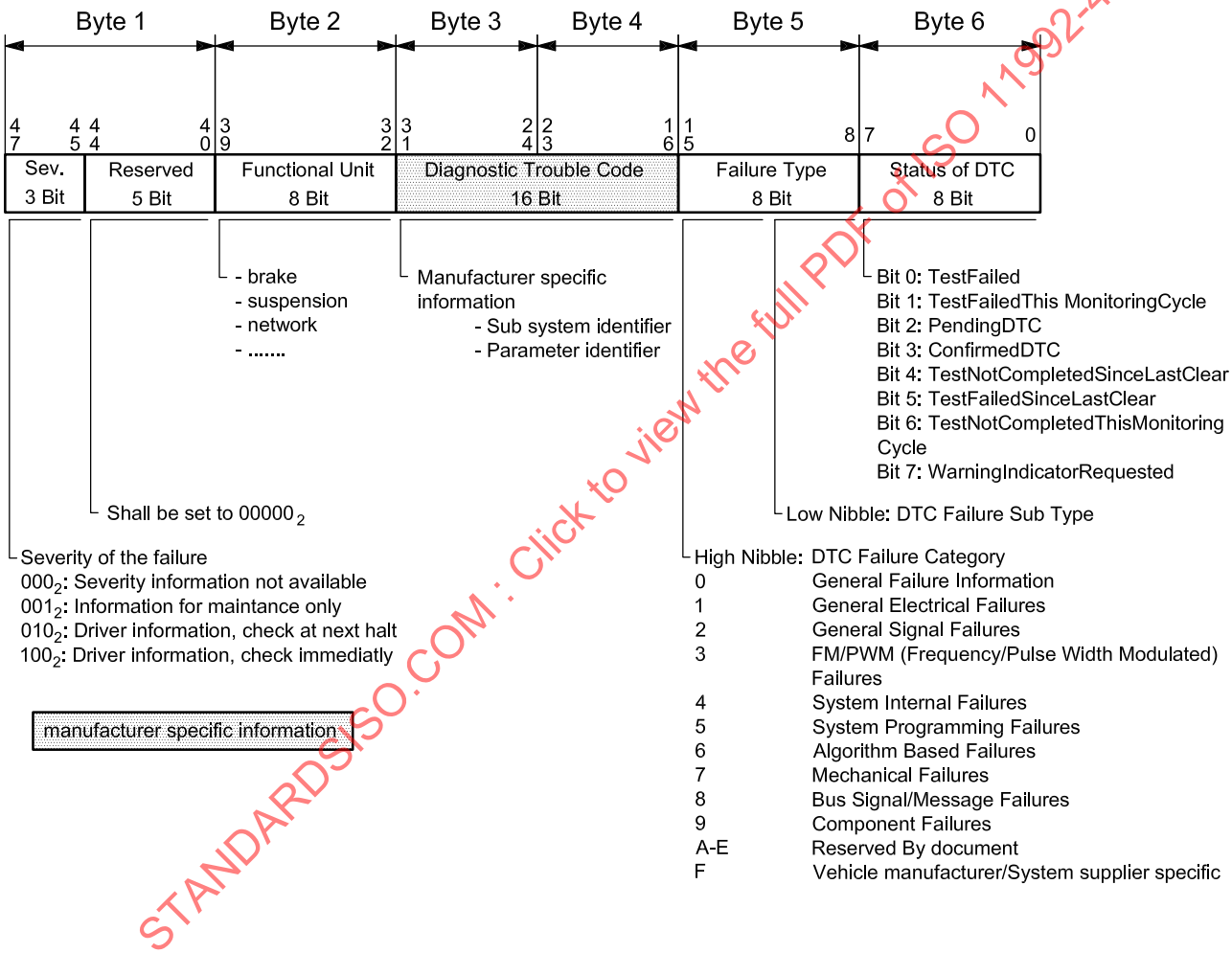

Figure B.1 — Diagnostic trouble code record for basic diagnostics

B.2 DTC severity information

This clause defines the mapping of the DTCSeverityMask/DTCSeverity parameters used with the ReadDTCInformation service. Every server shall adhere to the convention for storing bit-packed DTC severity information as specified in the Figure B.2.

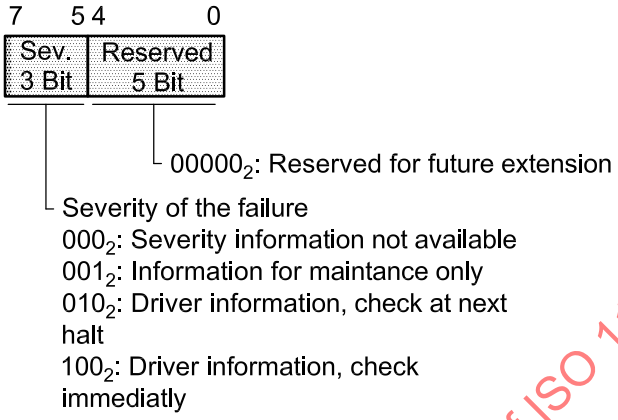

Figure B.2 — Severity information

B.3 DTC functional unit information

The DTC functional unit identifier shall be in accordance with Table B.1.

Table B.1 — Functional unit identifier

Description	Functional unit
Telematics (GPS, GSM)	0_{10}
General braking	1_{10}
ABS	2_{10}
EBS	3_{10}
Stability support	4_{10}
Retarder	5_{10}
Tyre	6_{10}
Suspension	7_{10}
Axle	8_{10}
Lift axle	9_{10}
Steering axle	10_{10}
General body application	11_{10}
Lights	12_{10}
Power take-off	13_{10}
Back-up assistance (rear obstacle detection, camera, ...)	14_{10}
Security	15_{10}
Loading ramp application (lift, ramp control, ...)	16_{10}
Temperature control (cooler, heater, ...)	17_{10}
Temperature Recorder	18_{10}
Auxiliary Power Unit	19_{10}
Local trailer communication (not ISO 11992)	20_{10}
On-board diagnostic/data recorder	21_{10}
Tractor power supply	22_{10}
Trailer battery power supply	23_{10}
Hitch (Trailer coupling)	24_{10}
Tractor-trailer communication (ISO 11992)	25_{10}
Reserved	$26_{10} – 254_{10}$
Manufacturer specific	255_{10}

B.4 StatusOfDTC bit specification

The bit specifications of StatusOfDTC shall be in accordance with Table B.2. The bit states '0' and '1' represent logical values.

Table B.2 — DTC status bit description

<table>
<tr><th>Bit</th><th colspan="3">Description</th><th>Cvt</th></tr>
<tr><td rowspan="4">0</td><td colspan="3">TestFailed
This bit shall indicate the result of the most recently performed test. A ‘1’ shall indicate that the last test failed. Reset to '0' if the result of the most recently performed test returns a “pass” result, or after a call has been made to ClearDiagnosticInformation. Additional reset conditions may be defined by the vehicle manufacturer/implementation.</td><td rowspan="2">U</td></tr>
<tr><td colspan="3">Bit state after successful ClearDiagnosticInformation service: '0'</td></tr>
<tr><td>State:</td><td>Definition:</td><td>Recommended test equipment display text:</td><td rowspan="3"></td></tr>
<tr><td>'0'</td><td>Most recent return from DTC test indicated no failure detected.</td><td>DTC test is not failed at time of request</td></tr>
<tr><td>'1'</td><td>Most recent return from DTC test indicated a failed result.</td><td>DTC test failed at time of request</td></tr>
<tr><td rowspan="5">1</td><td colspan="3">TestFailedThisMonitoringCycle
This bit shall indicate whether or not a diagnostic test has reported a TestFailed result at any time during the current monitoring cycle (or that a TestFailed result has been reported during the current monitoring cycle and after the last time a call was made to ClearDiagnosticInformation). Reset to '0' when a new monitoring cycle is initiated or after a call to ClearDiagnosticInformation.
Note that once this bit is set to '1', it shall remain a '1' until a new monitoring cycle is started.</td><td rowspan="2">C1</td></tr>
<tr><td colspan="3">Bit state after successful ClearDiagnosticInformation service: '0'</td></tr>
<tr><td>State:</td><td>Definition:</td><td>Recommended test equipment display text:</td><td rowspan="3"></td></tr>
<tr><td>'0'</td><td>TestFailed result has not been reported during the current monitoring cycle or after a call was made to ClearDiagnosticInformation during the current monitoring cycle.</td><td>DTC test is not failed at time of request</td></tr>
<tr><td>'1'</td><td>TestFailed result was reported at least once during the current monitoring cycle.</td><td>DTC test failed at time of request</td></tr>
<tr><td rowspan="5">2</td><td colspan="3">PendingDTC
This bit shall indicate whether or not a diagnostic test has reported a TestFailed result at any time during the current or last completed monitoring cycle. The status shall be updated only if the test runs and completes. The criteria to set the PendingDTC bit and the TestFailedThisMonitoringCycle bit are the same. The difference is that the TestFailedThisMonitoringCycle is cleared at the end of the current monitoring cycle and the PendingDTC bit is not cleared until a monitoring cycle has completed where the test has passed at least once and never failed.
If the test did not complete during the current monitoring cycle, the status bit shall not be changed; e.g. if a monitor stops running after a confirmed DTC is set, the PendingDTC must remain set ='1'.</td><td rowspan="2">U</td></tr>
<tr><td colspan="3">Bit state after successful ClearDiagnosticInformation service: '0'</td></tr>
<tr><td>State:</td><td>Definition:</td><td>Recommended test equipment display text:</td><td rowspan="3"></td></tr>
<tr><td>'0'</td><td>This bit shall be set to '0' after completing a monitoring cycle during which the test completed and a malfunction was not detected or upon a call to the ClearDiagnosticInformation service.</td><td>DTC was not failed on the current or previous monitoring cycle</td></tr>
<tr><td>'1'</td><td>This bit shall be set to '1' and latched if a malfunction is detected during the current monitoring cycle.</td><td>DTC failed on the current or previous monitoring cycle</td></tr>
<tr><td colspan="5">C1: Bit 1 (TestFailedThisMonitoringCycle) is mandatory if bit 2 (PendingDTC) is supported.
Bit 1 (TestFailedThisMonitoringCycle) is user optional if bit 2 (PendingDTC) is not supported.</td></tr>
</table>

Table B.2 (*continued*)

<table>
<tr><th>Bit</th><th colspan="3">Description</th><th>Cvt</th></tr>
<tr><td rowspan="6">3</td><td colspan="3">ConfirmedDTC
This bit shall indicate whether a malfunction was detected enough times to warrant that the DTC be stored in long-term memory (PendingDTC has been set = '1' one or more times, depending on the DTC confirmation criteria).
A ConfirmedDTC does not always indicate that the malfunction is necessarily present at the time of the request. (PendingDTC or TestFailedThisMonitoringCycle can be used to determine if a malfunction is present at the time of the request.).
Reset to '0' after a call to ClearDiagnosticInformation or after ageing criteria has been satisfied.
Note: DTC confirmation and ageing criteria are defined by the vehicle manufacturer or mandated by On Board Diagnostic regulations.</td><td rowspan="2">M</td></tr>
<tr><td colspan="3">Bit state after successful ClearDiagnosticInformation service: '0'</td></tr>
<tr><td>State:</td><td>Definition:</td><td>Recommended test equipment display text:</td><td rowspan="3"></td></tr>
<tr><td>'0'</td><td>DTC has never been confirmed since the last call to ClearDiagnosticInformation or after the ageing criteria have been satisfied for the DTC.</td><td>DTC is not confirmed at the time of the request</td></tr>
<tr><td>'1'</td><td>DTC confirmed at least once since the last call to ClearDiagnosticInformation and ageing criteria have not yet been satisfied.</td><td>DTC is confirmed at the time of the request</td></tr>
<tr><td rowspan="5">4</td><td colspan="3">TestNotCompletedSinceLastClear
This bit shall indicate whether a DTC test has ever run and completed since the last time a call was made to ClearDiagnosticInformation. '1' shall indicate that the DTC test has not run to completion. If the test runs and passes or if the test runs and fails (TestFailedThisMonitoringCycle = '1') then the bit shall be set to a '0' (and latched).</td><td rowspan="2">C2</td></tr>
<tr><td colspan="3">Bit state after successful ClearDiagnosticInformation service: '1'</td></tr>
<tr><td>State:</td><td>Definition:</td><td>Recommended test equipment display text:</td><td rowspan="3"></td></tr>
<tr><td>'0'</td><td>DTC test has returned either a passed or failed test result at least one time since the last time diagnostic information was cleared.</td><td>DTC test completed since the last code clear.</td></tr>
<tr><td>'1'</td><td>DTC test has not run to completion since the last time diagnostic information was cleared.</td><td>DTC test not completed since the last code clear.</td></tr>
<tr><td rowspan="6">5</td><td colspan="3">TestFailedSinceLastClear
This bit shall indicate whether a DTC test has ever returned a TestFailedThisMonitoringCycle = '1' result since the last time a call was made to ClearDiagnosticInformation. (latched TestFailedThisMonitoringCycle = '1').
A '0' shall indicate that the test has not run or that the DTC test ran and passed (but never failed). If the test runs and fails then the bit shall remain latched at a '1'. Reset to '0' after a call to ClearDiagnosticInformation.</td><td rowspan="2">C2</td></tr>
<tr><td colspan="3">Bit state after successful ClearDiagnosticInformation service: '0'</td></tr>
<tr><td>State:</td><td>Definition:</td><td>Recommended test equipment display text:</td><td rowspan="3"></td></tr>
<tr><td>'0'</td><td>DTC test has not indicated a TestFailedThisMonitoringCycle = '1' result since the last time diagnostic information was cleared.</td><td>DTC test never failed since last code clear</td></tr>
<tr><td>'1'</td><td>DTC test returned a TestFailedThisMonitoringCycle = '1' result at least once since the last time diagnostic information was cleared.</td><td>DTC test failed at least once since last code clear.</td></tr>
<tr><td colspan="5">C2: Bit 4 (TestNotCompletedSinceLastClear) and bit 5 (TestNotFailedSinceLastClear) shall always be supported together.</td></tr>
</table>

Table B.2 (*continued*)

<table>
<tr><th>Bit</th><th colspan="3">Description</th><th>Cvt</th></tr>
<tr><td rowspan="3">6</td><td colspan="3">TestNotCompletedThisMonitoringCycle
This bit shall indicate whether a DTC test has ever run and completed during the current monitoring cycle (or completed during the current monitoring cycle after the last time a call was made to ClearDiagnosticInformation).
A '1' shall indicate that the DTC test has not run to completion during the current monitoring cycle. If the test runs and passes or fails then the bit shall be set (and latched) to '0' until a new monitoring cycle is started. Reset to '1' after a call to ClearDiagnosticInformation.</td><td rowspan="2">M</td></tr>
<tr><td colspan="3">Bit state after successful ClearDiagnosticInformation service: '1'</td></tr>
<tr><td rowspan="3"></td><td>State:</td><td>Definition:</td><td>Recommended test equipment display text:</td><td rowspan="3"></td></tr>
<tr><td>'0'</td><td>DTC test has returned either a passed or TestFailedThisMonitoringCycle = '1' result during the current drive cycle (or since the last time diagnostic information was cleared during the current monitoring cycle).</td><td>DTC test completed this monitoring cycle.</td></tr>
<tr><td>'1'</td><td>DTC test has not run to completion this monitoring cycle (or since the last time diagnostic information was cleared this monitoring cycle).</td><td>DTC test not completed this monitoring cycle.</td></tr>
<tr><td rowspan="3">7</td><td colspan="3">WarningIndicatorRequested
This bit shall report the status of any warning indicators associated with a particular DTC. Warning outputs may consist of indicator lamp(s), displayed text information, etc. If no warning indicators exist for a given system or particular DTC, this status shall default to a '0' state.
Conditions for activating the warning indicator shall be defined by the vehicle manufacturer/implementation, but if the warning indicator is on for a given DTC, then ConfirmedDTC shall be also be set to '1'.
Reset to a ‘0’ after a call to ClearDiagnosticInformation. Additional reset conditions defined by vehicle manufacturer/implementation.</td><td rowspan="2">U</td></tr>
<tr><td colspan="3">Bit state after successful ClearDiagnosticInformation service: '0'</td></tr>
<tr><td rowspan="3"></td><td>State:</td><td>Definition:</td><td>Recommended test equipment display text:</td><td rowspan="3"></td></tr>
<tr><td>'0'</td><td>Warning indicator requested to be OFF.</td><td>DTC does not request warning indication.</td></tr>
<tr><td>'1'</td><td>Warning indicator requested to be ON.</td><td>DTC does request warning indication.</td></tr>
</table>

B.5 Record data identifiers

The RecordDataIdentifier identifies a data record with a unique meaning for all systems within a vehicle and shall be in accordance with Table B.3.

For basic diagnostics the meaning shall be unique for all vehicle complying to this part of ISO 11992.

RecordDataIdentifier: 16 Bit = 0000_{16} ... $FFFF_{16}$

Table B.3 — RecordDataIdentifier and DataRecord specification for basic diagnostics

Identifier value	Description	Cvt
$F000_{16}$ – $F00F_{16}$	**NetworkConfigurationDataForTractorTrailerApplication** This range of values shall be used to request the remote addresses of all trailer systems independent of their functionality.	M
$F000_{16}$	**NetworkConfigurationData - TrailerRemoteAddress** This value shall be used to request the remote addresses of all trailer systems independent of their functionality.	U
$F001_{16}$	**NetworkConfigurationData - BrakingAndRunningGearTrailerRemoteAddress** This value shall be used to request the remote addresses of braking and running gear trailer systems according to ISO 11992-2. This information/PID shall be supported by all servers belonging to braking and running gear applications.	M
$F002_{16}$	**NetworkConfigurationData - GeneralPurposeTrailerRemoteAddress** This value shall be used to request the remote addresses of general purpose trailer systems according to ISO 11992-3. This information/PID shall be supported by all servers belonging to general purpose applications.	M
$F187_{16}$	**VehicleManufacturerSparePartNumber** This value shall be used to reference the vehicle manufacturer spare part number. Record data content and format shall be ASCII and defined by the vehicle manufacturer.	U
$F188_{16}$	**VehicleManufacturerECUSoftwareNumber** This value shall be used to reference the vehicle manufacturer ECU software number. Record data content and format shall be ASCII and defined by the vehicle manufacturer.	U
$F189_{16}$	**VehicleManufacturerECUSoftwareVersionNumber** This value shall be used to reference the vehicle manufacturer ECU software version number. Record data content and format shall be ASCII and defined by the vehicle manufacturer.	U
$F18A_{16}$	**SystemSupplierIdentifier** This value shall be used to reference the system supplier name and address information. Record data content and format shall be ASCII and defined by the system supplier.	U
$F18B_{16}$	**ECUManufacturingData** This value shall be used to reference the ECU manufacturing date. Record data content and format shall be ASCII and shall be ordered as Year, Month, Day.	U
$F18C_{16}$	**ECUSerialNumber** This value shall be used to reference the ECU serial number. Record data content and format shall be ASCII and defined by the system supplier.	U
$F18D_{16}$	**SupportedFunctionalUnits** This value shall be used to request the functional units implemented in a server. This information shall be provided by all servers.	M
$F190_{16}$	**VIN** This value shall be used to reference the VIN number. Record data content and format shall be ASCII and specified by the vehicle manufacturer. This information shall be at least provided by the interfaces in the towing and towed vehicles.	M

Table B.3 (*continued*)

Identifier value	Description	Cvt
$F191_{16}$	**VehicleManufacturerECUHardwareNumber** This value shall be used by reading services to reference the vehicle manufacturer specific ECU hardware number. Record data content and format shall be ECU specific and defined by vehicle manufacturer.	U
$F192_{16}$	**SystemSupplierECUHardwareNumber** This value shall be used to reference the system supplier specific ECU hardware number. Record data content and format shall be ECU specific and defined by the system supplier.	U
$F193_{16}$	**SystemSupplierECUHardwareVersionNumber** This value shall be used to reference the system supplier specific ECU hardware version number. Record data content and format shall be ECU specific and defined by the system supplier.	U
$F194_{16}$	**SystemSupplierECUSoftwareNumber** This value shall be used to reference the system supplier specific ECU software number. Record data content and format shall be ECU specific and defined by the system supplier.	U
$F195_{16}$	**SystemSupplierECUSoftwareVersionNumber** This value shall be used to reference the system supplier specific ECU software version number. Record data content and format shall be ECU specific and defined by the system supplier.	U
$F197_{16}$	**SystemNameOrEngineType** This value shall be used to reference the system name or engine type. Record data content and format shall be ASCII and defined by the vehicle manufacturer. This information should be provided by all servers. NOTE: The maximum length is limited by the maximum length of a segmented message to 255 bytes.	M
$F19E_{16}$	**ODXFileIdentifier** This value shall be used to reference the ODX (Open Diagnostic Data Exchange) file of the server to be used to interpret and scale the server data.	U
$FD00_{16}$ – $FEFF_{16}$	**SystemSupplierSpecific** This range of values shall be used to reference system supplier specific record data identifiers and input/output identifiers within the server.	U

B.6 Response codes

The response codes specify the reason for a reject of a diagnostic service request and shall be in accordance with Table B.4.

ResponseCode: 8 Bit = <Response code>

NOTE All response codes not mentioned in Table B.4 are reserved for future definition.

Table B.4 — Response codes for basic diagnostics

Response code	Code	Meaning
GeneralReject	10_{16}	The service request is rejected without any specific reason. This response code shall only be used if none of the other negative response codes specified in this document describes the reason of the service reject.
ServiceNotSupported	11_{16}	The server does not support the requested service.
SubFunctionNotSupported	12_{16}	The service execution is not possible with the given service request parameters.
BusyRepeatRequest	21_{16}	The Server has understood the service request, but it cannot execute the service at this time and will not start the service at any time later, e.g. due to the fact that a diagnostic service is already in progress. The client shall repeat the request later.
RequestOutOfRange	31_{16}	The requested action would have exceeded a pre-defined parameter range.
RequestCorrectlyReceived - ResponsePending	78_{16}	The request has been correctly received and is executed by the server. The final positive or negative response is delayed because of the execution time needed for this service. A further service request will be rejected.