INTERNATIONAL STANDARD

ISO 26021-1

Second edition
2022-02

Road vehicles — End-of-life activation of in-vehicle pyrotechnic devices —

Part 1: Application and communication interface

Véhicules routiers — Activation de fin de vie des dispositifs pyrotechniques embarqués —

Partie 1: Interface des couches application et communication

Reference number
ISO 26021-1:2022(E)

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

Published in Switzerland

Contents

Page

Foreword

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

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

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

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

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

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

This second edition cancels and replaces the first edition (ISO 26021-1:2008, ISO 26021-2:2008, ISO 26021-2:2008/Cor 1:2009, ISO 26021-4:2009, ISO 26021-5:2009), which have been technically revised.

The main changes are as follows:

— restructuring of four parts into a single document including use cases and application requirements;

— introduction of requirement structure with numbering and name;

— support of ISO 13400 DoIP (diagnostic communication over Internet Protocol);

— support of ISO 13400-4 DoIP diagnostic connector.

A list of all parts in the ISO 26021 series can be found on the ISO website.

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

Introduction

End-of-life deployment activation of on-board pyrotechnic devices is a part of a wider regime designed to ensure that road vehicles are scrapped in a safe and environmentally acceptable condition after their use.

Newly designed products implement new security features like the authentication service. Such vehicle PCU(s) can not be supported by pyrotechnic device deployment tools (PDTs) without security implementation.

The ISO 26021 series is based on the Open Systems Interconnection (OSI) basic reference model specified in ISO/IEC 7498-1 and ISO/IEC 10731[1], which structures communication systems into seven layers. When mapped on this model, the application layer protocol and data link layer framework requirements specified/referenced in the ISO 26021 series are structured according to Figure 1.

Figure 1 illustrates a standard-based documentation concept, which consists of the following main clusters:

— vehicle diagnostic communication framework: covers all relevant basic vehicle diagnostic communication specifications of OSI layers 7, 6 and 5;

— vehicle diagnostic communication use case framework: covers the use cases and requirements of the subject matter of OSI layer 7;

— presentation layer framework: covers all data-relevant specifications of OSI layer 6;

— conformance test plan: covers the conformance test plan requirements of the use cases and communication requirements of OSI layers 7, 6 and 5;

— lower OSI layer framework: covers all vehicle diagnostic protocol standards of OSI layers 4, 3, 2 and 1, which are relevant and referenced by the use case specific standard.

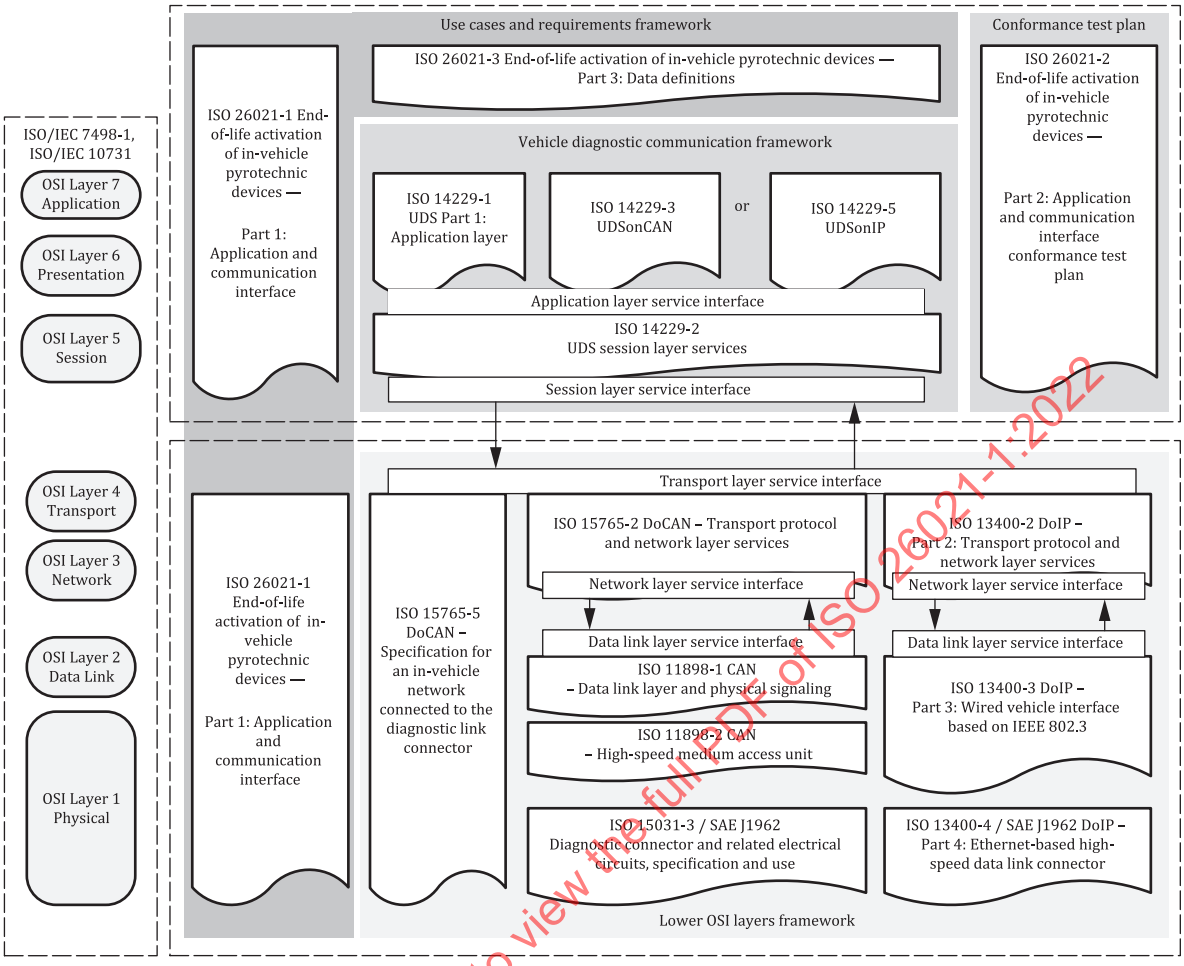


Figure 1 — ISO 26021 documents reference according to OSI model

Road vehicles — End-of-life activation of in-vehicle pyrotechnic devices —

Part 1: Application and communication interface

1 Scope

This document is applicable to road vehicles, where the electronic vehicle interface of the diagnostic link connector (DLC) is used to perform an end-of-life (EoL) activation of in-vehicle pyrotechnic devices. Apart from actual removal, this is the method to assure that no pyrotechnic substances are left in an EoL vehicle. On-board activation is an effective and safe method.

This document describes use cases and specifies technical requirements in order to support the end-of-life activation of in-vehicle pyrotechnic devices via the electronic communication interface. This document references the ISO 14229 series unified diagnostic services implemented on diagnostic communication over controller area network (DoCAN) and Internet Protocol (DoIP) along with the required provision of data definitions.

This document comprises:

— terminology definitions;

— definition of end-of-life activation of in-vehicle pyrotechnic devices relevant use cases;

— requirements for the establishment of communication between the pyrotechnic device deployment tool (PDT) and the vehicle's pyrotechnic control unit(s) (PCU(s));

— requirements for the optional usage of a credentials-based authentication and authorisation mechanism between the PDT and the vehicle;

— requirements for the protection against tampering of the defined end-of-life activation of in-vehicle pyrotechnic devices;

— PCU-relevant technical requirements.

PDT-relevant requirements are specified in a test equipment-specific standard with PDT-specific requirements.

2 Normative references

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

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

ISO/IEC 9834-1, *Information technology — Procedures for the operation of object identifier registration authorities: General procedures and top arcs of the international object identifier tree — Part 1:*

ISO 11898-1, *Road vehicles — Controller area network (CAN) — Part 1: Data link layer and physical signalling*

ISO 11898-2, *Road vehicles — Controller area network (CAN) — Part 2: High-speed medium access unit*

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

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

ISO 13400-4, *Road vehicles — Diagnostic communication over Internet Protocol (DoIP) — Part 4: Ethernet-based high-speed data link connector*

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

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

ISO 14229-3, *Road vehicles — Unified diagnostic services (UDS) — Part 3: Unified diagnostic services on CAN implementation (UDSonCAN)*

ISO 14229-5, *Road vehicles — Unified diagnostic services (UDS) — Part 5: Unified diagnostic services on Internet Protocol implementation (UDSonIP)*

ISO 14230-1, *Road vehicles — Diagnostic communication over K-Line (DoK-Line) — Part 1: Physical layer*

ISO 15031-3, *Road vehicles — Communication between vehicle and external equipment for emissions-related diagnostics — Part 3: Diagnostic connector and related electrical circuits: Specification and use*

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

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

ISO 26021-3,[1)] *Road vehicles — End-of-life activation of on-board pyrotechnic devices — Part 3: Data definitions*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 7498-1, ISO 14229-1 and the following apply.

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

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

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

3.1
key
data value sent from the external test equipment to the on-board controller in response to the *seed* (3.9) in order to gain access to the locked services

3.2
pyrotechnic control unit
PCU
electronic control unit in the vehicle network which controls the activation of pyrotechnic devices

3.3
pulse width modulation
PWM
signal linked by the ACL to the independent hardware path in the *pyrotechnic control unit* (3.2)

Note 1 to entry: The PWM signal is active during the deployment session.

1) Second edition under preparation. Stage at the time of publication: ISO/DIS 26021-3:2022.

3.4
pyrotechnic device deployment tool
PDT
tool designed to be plugged into the OBD interface in order to communicate via the internal computer network in an end-of-life vehicle with control units which are able to activate pyrotechnic devices

3.5
safing
mechanism whose primary purpose is to prevent an unintended functioning of the *pyrotechnic control unit* (3.2) processor prior to detection of a crash situation

3.6
safing unit
part of the *pyrotechnic control unit* (3.2) that allows the pyrotechnic component deployment microprocessor (µP) to deploy the pyrotechnic devices via the driver stage

EXAMPLE An electromechanically operated switch or a separate processor.

3.7
scrapping program module
module responsible for firing the selected pyrotechnic device loops one by one

3.8
scrapping program module loader
module responsible for converting the *scrapping program module* (3.7) to an executable format

3.9
seed
pseudo-random data value sent from the on-board controller to the external test equipment, which is processed by the security algorithm to produce the *key* (3.1)

4 Symbols and abbreviated terms

4.1 Symbols

Δ	delta
Δt_{P6_Client}	DoIP network design-dependent delays
Δt_{P6*_Client}	DoIP network design-dependent extended delays
Δt_{P2}	DoCAN network design-dependent delays
t	time
t_{S3_Client}	client session timer
t_{S3_Server}	server session timer
$t_{P2\_Server\_Max}$	server response timer maximum value
$t_{P2*\_Server\_Max}$	server extended response timer maximum value
$t_{P3\_Client\_Phys}$	time between end of server response and start of new client request

4.2 Abbreviated terms

ACL	additional communication line

AL	application layer
APP	application
BP	basic principle
CAN	controller area network
CANID	CAN identifier
DID	data identifier
DLC	diagnostic link connector
DLL	data link layer
DoCAN	diagnostic communication over CAN
DoIP	diagnostic communication over internet protocol
EoL	end-of-life
IDIS	international dismantling information system
IO	input, output
LSb	least significant bit
LSB	least significant byte
M	mandatory
MSb	most significant bit
MSB	most significant byte
MsgParam	message parameter
N/A	not applicable
NRC	negative response code
NL	network layer
O	optional
OBD	on-board diagnostic
OSI	open systems interconnection
PCU	pyrotechnic control unit
PDT	pyrotechnic device deployment tool
PDU	protocol data unit
PHY	physical layer
PL	presentation layer
PosRspMsgParam	positive response message parameter

PWM	pulse width modulation
RAM	random access memory
REQ	requirement
ReqMsgParam	request message parameter
RID	routine identifier
SA	source address
SL	session layer
SI	service interface
SIP	service interface parameter
SPL	scrapping program module loader
SPM	scrapping program module
SRS	supplemental restraint system
SF	SubFunction
TA	target address
TL	transport layer
µC	microcontroller
UDS	unified diagnostic services
VIN	vehicle identification number
VM	vehicle manufacturer

5 Conventions

This document is based on OSI service conventions as specified in ISO/IEC 10731[1].

6 Basic principles and use cases overview

6.1 Basic principles

Basic principles are established as a guideline to develop this document.

— BP1: use cases describe the interaction between the PDT and the vehicle's pyrotechnic device(s) utilising the vehicle's communication interface and/or additional communication line at the diagnostic link connector.

— BP2: use cases of the same subject are combined in one use case group.

— BP3: use cases described in this document are described from a vehicle's point of view.

— BP4: use cases are described independently of the vehicle system group, e.g. safety systems.

— BP5: all communication messages comply with the ISO 14229 series.

— BP6: all data definitions comply with the ISO 14229 series.

— BP7: fixed parameter values, for example, hardware deployment method, number of PCU(s), address information, VIN, dismantling information, and routine controls are assigned to standardized identifiers (DIDs, RIDs) and are mandatory for applicable use cases and interface requirements unless otherwise noted.

— BP8: requirements inherit the classification of the corresponding use cases.

— BP9: a "REQ X.Y" requirement specifies a single requirement (not multiple).

— BP10: only diagnostic services and data that are within the scope of the ISO 26021 series are guaranteed to work in the context of end-of-life deployment activation of on-board pyrotechnic devices.

6.2 Use case groups and associated use cases

Table 1 provides an overview of the main use case groups, associated use cases.

Table 1 — Use case groups and associated use cases

#	Use case group (UCG)	Use case reference
1	UCG 1 – Perform communication interface discovery	UC 1.1 – Discover DoCAN communication interface UC 1.2 – Discover DoIP communication interface
2	UCG-2 – Perform authentication	UC 2.1 – Perform PDT authentication UC 2.2 – Perform fixed-address PCU/PCU(s) authentication
3	UCG 3 – Perform system initialisation (Sys-Init)	UC 3.1 – Report PCU hardware deployment method UC 3.2 – Report number of PCU(s) UC 3.3 – Report address information of PCU(s) UC 3.4 – Report vehicle identification number UC 3.5 – Report dismantling documentation of PCU
4	UCG 4 – Perform PCU initialisation (PCU-Init)	UC 4.1 – Report PCU deployment loop identification table UC 4.2 – Initiate safetySystemDiagnosticSession UC 4.3 – Keep-alive safetySystemDiagnosticSession UC 4.4 – Unlock security of PCU UC 4.5 – Execute PCU(s) scrapping program module loader
5	UCG 5 – Perform PCU and ACL sequence (PCU- and ACL-Scrapping)	UC 5.1 – Report ACL deployment sequence (ACL-Init) UC 5.2 – Write dismantling documentation into PCU (Device-Deploy) UC 5.3 – Perform ACL deployment confirmation sequence (Device-Deploy) UC 5.4 – Perform PCU pyrotechnic device scrapping via loop identification (Device-Deploy)
6	UCG 6 – Terminate PCU pyrotechnic device deployment (PCU-End)	UC 6.1 – Terminate PCU pyrotechnic device scrapping via communication interface UC 6.2 – Terminate PCU pyrotechnic device scrapping via ACL

7 Use cases definition (UC)

7.1 UCG 1 – Perform communication interface discovery

7.1.1 UC 1.1 – Discover DoCAN communication interface

Table 2 defines the UC 1.1 – Discover DoCAN communication interface.

Table 2 — UC 1.1 – Discover DoCAN communication interface

Item	Description
Name	UC 1.1 – Discover DoCAN communication interface
Goal	The fixed-address PCU/gateway listens on the DoCAN communication interface connected to the vehicle's diagnostic link connector to discover the PDT selected protocol.
Actor	PDT and fixed-address PCU/gateway
Input	The fixed-address PCU/gateway listens on the DoCAN communication interface for a request message sent by the PDT.
Output	Transmission of a positive response message on the DoCAN communication interface by the fixed-address PCU/gateway
Function	The fixed-address PCU/gateway (depends on in-vehicle network architecture implementation) listens on the DoCAN communication interface(s) connected to the vehicle diagnostic link connector according to ISO 15031-3 for a request message sent by the PDT. The fixed-address PCU/gateway responds with a positive response message on the DoCAN communication interface.
Classification	Mandatory

7.1.2 UC 1.2 – Discover DoIP communication interface

Table 3 defines the UC 1.2 – Discover DoIP communication interface.

Table 3 — UC 1.2 – Discover DoIP communication interface

Item	Description
Name	UC 1.2 – Discover DoIP communication interface
Goal	The fixed-address PCU/gateway listens on the DoIP communication interface connected to the vehicle's diagnostic link connector to discover the PDT selected protocol.
Actor	PDT and fixed-address PCU/gateway
Input	The fixed-address PCU/gateway listens on the DoIP communication interface for a request message sent by the PDT.
Output	Transmission of positive response message on the DoIP communication interface by the fixed-address PCU/gateway
Function	The fixed-address PCU/gateway (depends on in-vehicle network architecture implementation) listens on the DoIP communication interface connected to the vehicle diagnostic link connector according to ISO 13400-4 for a request message sent by the PDT. The fixed-address PCU/gateway responds with a positive response message on the DoIP communication interface.
Classification	Mandatory

7.2 UCG-2 – Perform authentication

7.2.1 UC 2.1 – Perform PDT authentication

Table 4 defines the UC 2.1 – Perform PDT authentication.

Table 4 — UC 2.1 – Perform PDT authentication

Item	Description
Name	UC 2.1 – Perform PDT authentication
Goal	Authenticate the PDT against the fixed address PCU/gateway/PCU(s)
Actor	PDT and fixed-address PCU/gateway/PCU(s)
Input	Valid credentials sent by the PDT to the fixed address PCU/gateway/PCU(s)
Output	The fixed address PCU/gateway/PCU(s) send(s) the result of the authentication to the PDT. A positive authentication response message confirms the authenticity.
Function	The PDT uses credentials to authenticate against the fixed address PCU/gateway/PCU(s). The fixed address PCU/gateway/PCU(s) respond(s) with the result of the authentication and if successful, grant access rights to all diagnostic services and data necessary to fulfil the use cases specified in this document.
Classification	Optional

7.2.2 UC 2.2 – Perform fixed-address PCU/PCU(s) authentication

Table 5 defines the UC 2.2 – Perform fixed-address PCU/PCU(s) authentication.

Table 5 — UC 2.2 – Perform fixed-address PCU/PCU(s) authentication

Item	Description
Name	UC 2.2 – Perform fixed-address PCU/PCU(s) authentication
Goal	Authenticate the fixed address PCU(s)/gateway/PCU(s) against the PDT
Actor	PDT and fixed-address PCU/gateway/PCU(s)
Input	Valid credentials sent by the fixed address PCU(s)/gateway/PCU(s) to the PDT
Output	The PDT sends the result of the authentication to the fixed address PCU/gateway/PCU(s). A positive authentication response message confirms the authenticity.
Function	The fixed address PCU(s)/gateway/PCU(s) use(s) credentials to authenticate against the PDT. The PDT responds with the result of the authentication.
Classification	Optional

7.3 UCG 3 – Perform system initialisation (Sys-Init)

7.3.1 UC 3.1 – Report PCU hardware deployment method

Table 6 defines the UC 3.1 – Report PCU hardware deployment method.

Table 6 — UC 3.1 – Report PCU hardware deployment method

Item	Description
Name	UC 3.1 – Report PCU hardware deployment method
Goal	The fixed-address PCU provides the hardware deployment method information upon request by the PDT.
Actor	PDT and fixed-address PCU(s)
Input	Fixed-address PCU(s) receive(s) request message with DID = PCUHardwareDeploymentMethod.
Output	Fixed-address PCU(s) transmit(s) the positive response message with DID = PCUHardwareDeploymentMethod, its value, and the ACL type parameter to the PDT.
Function	The PDT requests the PCUHardwareDeploymentMethod. The fixed-address PCU(s) respond(s) with the PCUHardwareDeploymentMethod information.

Table 6 *(continued)*

Item	Description
Classification	Mandatory

7.3.2 UC 3.2 – Report number of PCU(s)

Table 7 defines the UC 3.2 – Report number of PCU(s).

Table 7 — UC 3.2 – Report number of PCU(s)

Item	Description
Name	UC 3.2 – Report number of PCU(s)
Goal	The fixed-address PCU provides the number of PCU(s) information upon request by the PDT.
Actor	PDT and fixed-address PCU(s)
Input	Fixed-address PCU(s) receive(s) request message with DID = NumberOfPCU.
Output	Fixed-address PCU(s) transmit(s) the positive response message with DID = NumberOfPCU and its value to the PDT.
Function	The PDT requests the NumberOfPCU. The fixed-address PCU(s) respond(s) with the NumberOfPCU information. The NumberOfPCU value specifies the outer loop of the scrapping sequence, which is stored for later use in the sequence executed by the PDT.
Classification	Mandatory

7.3.3 UC 3.3 – Report address information of PCU(s)

Table 8 defines the UC 3.3 – Report address information of PCU(s).

Table 8 — UC 3.3 – Report address information of PCU(s)

Item	Description
Name	UC 3.3 – Report address information of PCU(s)
Goal	The fixed-address PCU(s) provide(s) the address information of PCUs upon request by the PDT.
Actor	PDT and fixed-address PCU(s)
Input	Fixed-address PCU(s) receive(s) request message with DID = AddressInfoOfPCUs
Output	Fixed-address PCU(s) transmit(s) the positive response message with DID = AddressInfoOfPCUs and its information of each PCU installed in the vehicle to the PDT.
Function	The purpose of this feature is to support the backward compatibility of used addressing methods implemented in PCUs. The PDT requests the AddressInfoOfPCUs. The fixed-address PCU(s) respond(s) with the AddressInfoOfPCUs information. The AddressInfoOfPCUs information consists of the data record PCU address format #1, PCU request address #1, the PCU response address #1, up to PCU address format #N, PCU request address #N, the PCU response address #N.
Classification	Mandatory

7.3.4 UC 3.4 – Report vehicle identification number

Table 9 defines the UC 3.4 – Report vehicle identification number.

Table 9 — UC 3.4 – Report vehicle identification number

Item	Description
Name	UC 3.4 – Report vehicle identification number
Goal	The fixed-address PCU(s) provide(s) the VIN.
Actor	PDT and fixed-address PCU(s)
Input	Fixed-address PCU(s) receive(s) request message with DID = VIN.
Output	Fixed-address PCU(s) transmit(s) the positive response message with DID = VIN and its information to the PDT.
Function	The PDT requests the VIN. The fixed-address PCU(s) respond(s) with the VIN. The VIN is used to uniquely identify the vehicle.
Classification	Mandatory

7.3.5 UC 3.5 – Report dismantling documentation of PCU

Table 10 defines the UC 3.5 – Report dismantling documentation of PCU.

Table 10 — UC 3.5 – Report dismantling documentation of PCU

Item	Description
Name	UC 3.5 – Report dismantling documentation of PCU
Goal	The fixed-address PCU(s) receive(s) a request to report the dismantling information from the vehicle's fixed-address PCU(s) protected, permanent memory.
Actor	PDT and fixed-address PCU(s)
Input	Fixed-address PCU(s) receive(s) request message with DID = DismantlerIdentification.
Output	The DismantlerIdentification number, the PCU deployment device identification value, the year, month, and the day of deployment is read from the fixed-address PCU's protected, permanent memory.
Function	The PDT requests the DismantlerIdentification data. The fixed-address PCU responds with the DismantlerIdentification information.
Classification	Mandatory

7.4 UCG 4 – Perform PCU initialisation (PCU-Init)

7.4.1 UC 4.1 – Report PCU deployment loop identification table

Table 11 defines the UC 4.1 – Report PCU deployment loop identification table.

Table 11 — UC 4.1 – Report PCU deployment loop identification table

Item	Description
Name	UC 4.1 – Report PCU deployment loop identification table
Goal	Determine the number and status of pyrotechnic devices supported in the vehicle.
Actor	PDT and PCU #1 to #N
Input	The addressed PCU #1 to #N listens on the supported communication interface for a ReadDataByIdentifer request message with DID = NumOfDeployLoopTableRecAndStatus sent by the PDT.
Output	Transmission of a ReadDataByIdentifer positive response message with the DeploymentLoopIdTable on the communication interface supported by the PCU #1 to #N

Table 11 *(continued)*

Item	Description
Description	The addressed PCU #1 to #N listens on the supported communication interface connected to the vehicle diagnostic link connector according to either ISO 15031-3, ISO 13400-4 for a ReadDataByIdentifer request message with DID = DeploymentLoopIdTable sent by the PDT. The addressed PCU #1 to #N responds with a positive response message on the supported communication interface.
Classification	Mandatory

7.4.2 UC 4.2 – Initiate safetySystemDiagnosticSession

Table 12 defines the UC 4.2 – Initiate safetySystemDiagnosticSession.

Table 12 — UC 4.2 – Initiate safetySystemDiagnosticSession

Item	Description
Name	UC 4.2 – Initiate safetySystemDiagnosticSession
Goal	Transit one of the PCUs into the safety system diagnostic session
Actor	PDT and PCU #1 to #N
Input	The addressed PCU #1 to #N listens on the supported communication interface for a DiagnosticSessionControl request message with the SubFunction parameter set to safetySystemDiagnosticSession sent by the PDT.
Output	The addressed PCU #1 to #N transmits a DiagnosticSessionControl positive response message on the communication interface.
Function	The addressed PCU #1 to #N (depends on in-vehicle network architecture implementation) listens on the supported communication interface connected to the vehicle diagnostic link connector according to either ISO 15031-3, ISO 13400-4 for a DiagnosticSessionControl request message with the SubFunction parameter set to safetySystemDiagnosticSession. The addressed PCU #1 to #N responds with a DiagnosticSessionControl positive response message on the supported communication interface.
Classification	Mandatory

7.4.3 UC 4.3 – Keep-alive safetySystemDiagnosticSession

Table 13 defines the UC 4.3 – Keep-alive safetySystemDiagnosticSession.

Table 13 — UC 4.3 – Keep-alive safetySystemDiagnosticSession

Item	Description
Name	UC 4.3 – Keep-alive safetySystemDiagnosticSession
Goal	Keep-alive active safetySystemDiagnosticSession
Actor	PDT and PCU
Input	The addressed PCU listens on the supported communication interface for any request message.
Output	Transmission of a positive response message on the supported communication interface by the addressed PCU
Description	The addressed PCU (depends on in-vehicle network architecture implementation) listens on the supported communication interface(s) connected to the vehicle diagnostic link connector according to either ISO 15031-3, ISO 13400-4 for any request message. The addressed PCU responds with a positive response message on the supported communication interface which keeps the active diagnostic session alive.
Classification	Mandatory

7.4.4 UC 4.4 – Unlock security of PCU

Table 14 defines the UC 4.4 – Unlock security of PCU.

Table 14 — UC 4.4 – Unlock security of PCU

Item	Description
Name	UC 4.4 – Unlock security of PCU
Goal	Enable security deployment with SecurityAccess service
Actor	PDT and PCU
Input	1. The addressed PCU listens on the supported communication interface for a SecurityAccess request message with a SecurityAccessType set to RequestDeploymentSeed. 2. The addressed PCU listens on the supported communication interface for a SecurityAccess request message with a SecurityAccessType set to SendDeploymentKey and the SecurityKey value required to successfully unlock the PCU.
Output	1. Transmission of a SecurityAccess positive response message with a SecuritySeed for RequestDeploymentSeed on the communication interface supported by the addressed PCU. 2. Transmission of a SecurityAccess positive response message with a SecurityAccessType set to sendKey.
Function	The SecurityAccess is a 2-step sequence between the PDT and the PCU. In the first step the addressed PCU (depends on in-vehicle network architecture implementation) listens on the supported communication interface connected to the vehicle diagnostic link connector according to either ISO 15031-3, ISO 13400-4 for a SecurityAccess request message with a SecurityAccessType set to RequestDeploymentSeed sent by the PDT. The addressed PCU responds with a SecurityAccess positive response message with a SecuritySeed for RequestDeploymentSeed. In the second step the addressed PCU listens on the supported communication interface for a SecurityAccess request message with a SecurityAccessType set to SendDeploymentKey and the SecurityKey value required to successfully unlock the PCU. The addressed PCU responds with a SecurityAccess positive response message with a SecurityAccessType set to SendKey.
Classification	Mandatory

7.4.5 UC 4.5 – Execute PCU(s) scrapping program module loader

Table 15 defines the UC 4.5 – Execute PCU(s) scrapping program module loader.

Table 15 — UC 4.5 – Execute PCU(s) scrapping program module loader

Item	Description
Name	UC 4.5 – Execute PCU(s) scrapping program module loader
Goal	PDT prepares the PCU for deployment of connected pyrotechnic devices.
Actor	PDT and PCU
Input	1. The addressed PCU listens on the supported communication interface for a RoutineControl request message with a routineControlType = startRoutine, a routineIdentifier = ExecuteSPL, and a routineControlOptionRecord = ExecuteSPL parameter. 2. The addressed PCU listens on the supported communication interface for a RoutineControl request message with a routineControlType = requestRoutineResult and a routineIdentifier = ExecuteSPL.

Table 15 *(continued)*

Item	Description
Output	1. Transmission of a RoutineControl positive response message on the communication interface supported by the addressed PCU after the routine stops and the scrapping module is in an executable form. 2. Transmission of a RoutineControl positive response message on the communication interface is supported by the addressed PCU with a routineStatusRecord.
Function	The scrapping program module (SPM) is stored in a non-executable format. The scrapping program loader (SPL) is responsible for converting the SPM into an executable format. The PDT requests RoutineControl with ExecuteSPL to the PCU. The SPL copies the SPM into a free RAM space. The RAM space is initialized after a reset. The SPL converts the SPM into an executable format, e.g. the SPL overwrites some values of the port input/output or function addresses with the correct values.
Classification	Mandatory

7.5 UCG 5 – Perform PCU and ACL sequence (PCU- and ACL-Scrapping)

7.5.1 UC 5.1 – Report ACL deployment sequence (ACL-Init)

Table 16 defines the UC 5.1 – Report ACL deployment sequence (ACL-Init).

Table 16 — UC 5.1 – Report ACL deployment sequence (ACL-Init)

Item	Description
Name	UC 5.1 – Report ACL deployment sequence (ACL-Init)
Goal	PCU(s) report(s) ACL deployment sequence via ACL with bidirectional communication/PWM signal.
Actor	PDT and PCU
Input	UC 4.1 – Report PCU deployment loop identification table; PCU receives an ACL request for deployment message/signal code via the ACL.
Output	PCU transmits the ACL command for deployment or signal code via the ACL.
Description	In this use case the fixed-address PCU reports via UC 4.1 – Report PCU deployment loop identification table that the vehicle architecture supports a PCU(s) with an ACL with bidirectional communication/PWM signal at the diagnostic link connector. The PCU awaits reception of an ACL request for deployment message/signal code via the ACL. Then the PCU transmits the ACL command for deployment or signal code via the ACL.
Classification	Optional

7.5.2 UC 5.2 – Write dismantling documentation into PCU (Device-Deploy)

Table 17 defines the UC 5.2 – Write dismantling documentation into PCU (Device-Deploy).

Table 17 — UC 5.2 – Write dismantling documentation into PCU (Device-Deploy)

Item	Description
Name	UC 5.2 – Write dismantling documentation into PCU (Device-Deploy)
Goal	The fixed-address PCU receives a request to write the dismantling information in the protected, permanent memory.
Actor	PDT and fixed-address PCU
Input	Fixed-address PCU receives a request message with DID = DismantlerIdentification.

Table 17 *(continued)*

Item	Description
Output	The DismantlerIdentification data of the PDT, the PCU deployment device identification value, year, month, and day of deployment is written into the fixed-address PCU's protected, permanent memory.
Function	The dismantler information is used to document the deployment process. Once the dismantler information record is written to the protected, permanent memory of the fixed-address PCU, such data become read-only.
Classification	Mandatory

7.5.3 UC 5.3 – Perform ACL deployment confirmation sequence (Device-Deploy)

Table 18 defines the UC 5.3 – Perform ACL deployment confirmation sequence (Device-Deploy).

Table 18 — UC 5.3 – Perform ACL deployment confirmation sequence (Device-Deploy)

Item	Description
Name	UC 5.3 – Perform ACL deployment confirmation sequence (Device-Deploy)
Goal	PCU reports the ACL deployment confirmation sequence via the ACL with bidirectional communication/PWM signal.
Actor	PDT and PCU
Input	UC 4.1 – Report PCU deployment loop identification table (e.g. hardware switch, ACL bidirectional communication with or without DoCAN/DoIP); PCU receives an ACL deployment confirmation message/signal code via the ACL.
Output	PCU transmits the ACL deployment confirmation response message/signal code via the ACL.
Function	In this use case the fixed-address PCU reports via UC 4.1 – Report PCU deployment loop identification table that the vehicle architecture supports a PCU(s) with an ACL with bidirectional communication/PWM signal at the diagnostic link connector. The PCU awaits reception of an ACL deployment confirmation message/signal code via the ACL. Then the PCU transmits the ACL deployment confirmation response message/signal code via the ACL.
Classification	Optional

7.5.4 UC 5.4 – Perform PCU pyrotechnic device scrapping via loop identification (Device-Deploy)

Table 19 defines the UC 5.4 – Perform PCU pyrotechnic device scrapping via loop identification (Device-Deploy).

Table 19 — UC 5.4 – Perform PCU pyrotechnic device scrapping via loop identification (Device-Deploy)

Item	Description
Name	UC 5.4 – Perform PCU pyrotechnic device scrapping via loop identification (Device-Deploy)
Goal	Scrapping of PCU's connected pyrotechnic devices
Actor	PDT and PCU
Input	The requested input is based on the output of UC 4.1 – Report PCU deployment loop identification table. The addressed PCU listens on the supported communication interface for a RoutineControl request message with a routineControlType = startRoutine, a routineIdentifier = DeployLoopRoutineID, and a routineControlOptionRecord with DeploymentLoopID information.

Table 19 *(continued)*

Item	Description
Output	The PCU determines the number of internal loops to execute the firing of one or multiple pyrotechnic devices. The PCU transmits a RoutineControl positive response message with a routineControlType = startRoutine, a routineIdentifier = DeployLoopRoutineID, and a routineStatusRecord including DeploymentLoopID information.
Description	To support the work of the dismantler, it is necessary to identify the number and area of installed PCU pyrotechnic devices. Every PCU can support a defined number, 1 to 255, of pyrotechnic devices. There is no standardized allocation between the PCU channel and the type of PCU generator. One or multiple pyrotechnic devices are connected to a PCU via mapped PCU channels. Each PCU has an internal loop identification table, which describes the link between the PCU channel and the system-specific allocation of the firing loops. The PCU receives a request message RoutineControl with startRoutine, RID = ExecuteSPL, and parameter loop ID to the PCU (for each pyrotechnic device connected to this PCU). — The SPM fires the pyrotechnic device. — The PDT receives the result. — The SPM updates the LoopStatus. — The PDT requests PCU-Reset, which clears the RAM and deletes the executable SPM.
Classification	Mandatory

7.6 UCG 6 – Terminate PCU pyrotechnic device deployment (PCU-End)

7.6.1 UC 6.1 – Terminate PCU pyrotechnic device scrapping via communication interface

Table 20 defines the UC 6.1 – Terminate PCU pyrotechnic device scrapping via communication interface.

Table 20 — UC 6.1 – Terminate PCU pyrotechnic device scrapping via communication interface

Item	Description
Name	UC 6.1 – Terminate PCU pyrotechnic device scrapping via communication interface
Goal	After deployment of all PCUs, the deployment termination becomes active via the communication interface.
Actor	PDT and PCU
Input	PCU does not receive any diagnostic messages for a timeout period specified in ISO 14229-2 or receives an ECUReset diagnostic request.
Output	PCU ends the safetySystemDiagnosticSession.
Function	To terminate the PCU pyrotechnic device scrapping via the communication interface, the PCU does not receive any diagnostic messages for a timeout period specified in ISO 14229-2 or receive an ECUReset diagnostic request. This causes the PCU to end the safetySystemDiagnosticSession.
Classification	Mandatory

7.6.2 UC 6.2 – Terminate PCU pyrotechnic device scrapping via ACL

Table 21 defines the UC 6.2 – Terminate PCU pyrotechnic device scrapping via ACL.

Table 21 — UC 6.2 – Terminate PCU pyrotechnic device scrapping via ACL

Item	Description
Name	UC 6.2 – Terminate PCU pyrotechnic device scrapping via ACL

Table 21 *(continued)*

Item	Description
Goal	After deployment of all PCUs the deployment termination becomes active via the ACL.
Actor	PDT and PCU
Input	PCU receives a deployment termination command via the ACL.
Output	PCU ends the safetySystemDiagnosticSession.
Description	To terminate the PCU pyrotechnic device scrapping via the ACL, the PCU receives a deployment termination command via the ACL. This causes the PCU to end the safetySystemDiagnosticSession.
Classification	Optional

8 Application (APP)

8.1 APP – Preconditions of end-of-life activation of pyrotechnic devices

Vehicle manufacturers require the fulfillment of preconditions to enable the end-of-life activation of pyrotechnic devices. Such preconditions vary across brand, vehicle type, model year, and model. Variations are caused by technical design decisions related to the in-vehicle network and PCU implementation. This impacts the actions to be performed by the dismantler person prior to the deployment of pyrotechnic devices.

REQ	8.1 APP – Preconditions of end-of-life activation of pyrotechnic devices – Access to vehicle interior
The vehicle shall provide access to the vehicle interior, grant a suitable identification method via ignition key or keyless entry unit, and shall be in standstill.	

NOTE 1 Ignition on status is visualized by the instrument panel cluster.

NOTE 2 Additional handling information is given in the International Dismantling Information System (IDIS) [2].

The vehicle systems are switched into operational state.

REQ	8.2 APP – Preconditions of end-of-life activation of pyrotechnic devices – Report PcuHardware-DeploymentMethodVersion
The fixed-address PCU in the vehicle shall report PcuHardwareDeploymentMethod information (PcuHardwareDeploymentMethodVersion, PcuIdentificationString) upon request by the PDT.	

The vehicle manufacturer documents preconditions required for each brand, vehicle type, model year, and model in the IDIS[2] to provide the information needed by the dismantler person to perform the deployment.

EXAMPLE Vehicle-manufacturer specific information on IDIS:

— disconnect/cutoff a diagnosable device as precondition;

— belt buckle;

— peripheral sensor;

— other diagnosable devices;

— network separation and security gateway (authentication, end to end security, etc.).

REQ	8.3 APP – Preconditions of end-of-life activation of pyrotechnic devices – Release loading of the scrapping program module
When the fulfilment of the vehicle manufacturer specific requirements (see IDIS[2]) are detected by the PCU, the PCU shall release the loading of the scrapping program module (SPM).	

REQ	8.4 APP – Preconditions of end-of-life activation of pyrotechnic devices – Reject loading of the scrapping program module
When the fulfilment of the vehicle manufacturer specific requirements (see IDIS[2]) are not detected by the PCU, the PCU shall not release the loading of the scrapping program module (SPM) and shall respond with a negative response message including NRC = conditionsNotCorrect (see ISO 14229-1) information upon a RoutineControl request message with the RID = ExecuteSPL.	

8.2 APP – Overview of end-of-life activation of pyrotechnic devices sequence

The end-of-life activation of pyrotechnic devices sequence consists of the following subsequences:

— APP – Discovery of communication interface (Com I/F-Discovery);

— APP – Perform authentication – Optional (Sys-Auth);

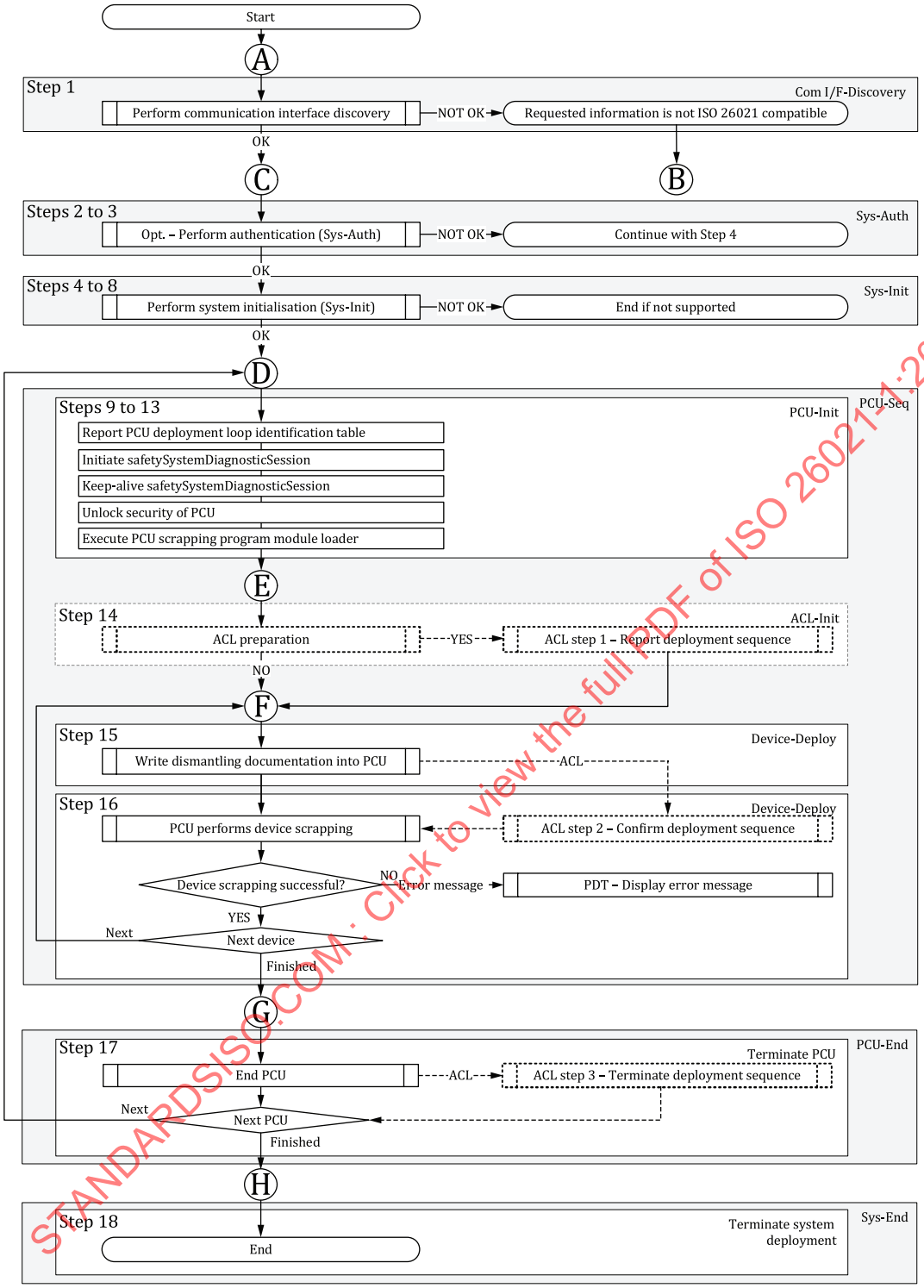
— APP – Perform system initialisation (Sys-Init);

— APP – Perform PCU initialisation (PCU-Seq);

— APP – Perform PCU and ACL scrapping (Device-Deploy); and

— APP – Terminate PCU and ACL pyrotechnic device deployment (PCU-End).

Figure 2 specifies the end-of-life activation of pyrotechnic device sequence.



Key

A vehicle's fixed-address PCU listens to the DoCAN or DoIP network for a diagnostic request message

B vehicle's fixed-address PCU determines that the diagnostic request message on the DoCAN or DoIP network is not compatible with this document

C vehicle's fixed-address PCU determines that the diagnostic request message on the DoCAN or DoIP network is compatible with this document

D vehicle's fixed-address PCU determines that the authentication (optional) and the system initialisation is successful

E vehicle's PCU #1 to #N determines that the PCU initialisation is successful

F vehicle's PCU #1 to #N determines that ACL is not supported

G PCU sequence reaches the state PCU-End, next PCU

H PCU sequence reaches the state “finished”

Figure 2 — End-of-life activation of pyrotechnic device sequence

8.3 APP – Software provisions

8.3.1 APP – Scrapping program module (SPM)

The SPM is usually stored in a ROM/FLASH memory of the PCU. To reduce the RAM space, only the security-related part of the program, which enables the output stages, can be executed from RAM.

REQ	8.5 APP – Software provisions – APP – Scrapping program module (SPM)
The SPM shall be stored in the PCU in a non-executable format.	

8.3.2 APP – Scrapping program module loader (SPL)

The SPL is responsible for converting the SPM format to an executable format.

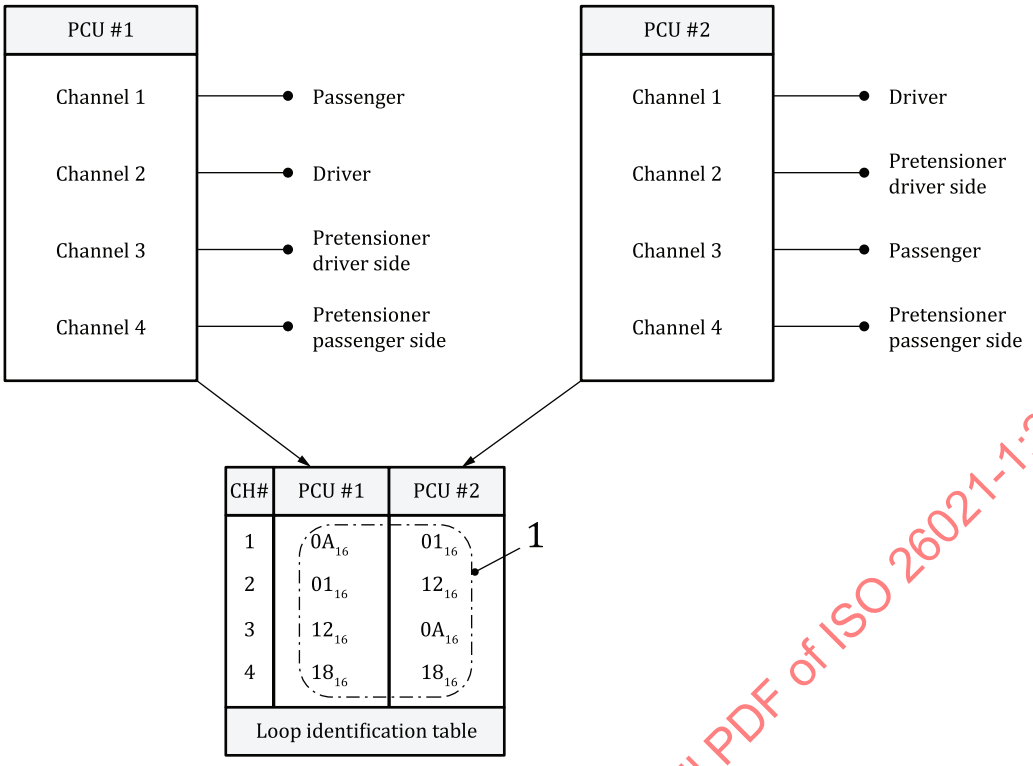
REQ	8.6 APP – Software provisions – APP – Scrapping program module loader (SPL)
The SPL shall convert the SPM into an executable format.	

8.3.3 APP – PCU loop identification table

To support the work of the dismantler, it is necessary to identify the number and area of installed PCU components. Every PCU can support a defined number of pyrotechnic devices (1 to 255). There is no standardized allocation between the PCU channel and the type of PCU generator.

REQ	8.7 APP – Software provisions – APP – PCU loop identification table – Channel identification
The APP – PCU loop identification table shall include channel identification for the purpose of simple identification of the mapped PCU channels.	

REQ	8.8 APP – Software provisions – APP – PCU loop identification table – Loop identifier
Each PCU shall have an internal APP – PCU loop identification table, which specifies the link between the PCU and the system-specific allocation of the firing loops identified by the loop identifier (see Figure 3).	



Key

1 loop identifier

Figure 3 — PCU loop identification table

The PCU loop identification table contains one or multiple PCU-specific channels. Each channel contains an identifier of a connected pyrotechnic device. A list of connected pyrotechnic device identifiers and description is specified in ISO 26021-3. The description provides an explanation to enable every recycler to find the relevant pyrotechnic device in the vehicle.

EXAMPLE 1

PCU #1 supports channel 1 with the loop identifier $0A_{16}$. The loop identifier $0A_{16}$ defines the “Airbag passenger side frontal 1st stage” (see ISO 26031-3).

EXAMPLE 2

PCU #2 supports channel 1 with the loop identifier 01_{16}”. The loop identifier 01_{16} defines the “Airbag driver side frontal 1st stage” (see ISO 26031-3).

The PCU #2 firing sequence is defined by the order of each loop identifier. PCU #2 uses AutoMode to fire the connected pyrotechnic devices in the following sequence:

a) 01_{16}: airbag driver side frontal 1st stage;

b) 12_{16}: 1st pretensioner — driver side;

c) $0A_{16}$: airbag passenger side frontal 1st stage;

d) 18_{16}: 1st pretensioner — passenger side.

8.4 APP – Mapping of use cases to requirements

Table 22 provides an overview about the technical requirements, associated requirement number and use case coverage. The "SEQ" column represents the order of requirements as they should occur in the

communication between the vehicle and the PDT external test equipment. The "REQ" column shows the requirement number for reference purposes by other documents. The "Technical requirement title" column contains the main title of the related requirement. The "Use case # and name" column lists all use cases related to the requirement in the same row.

Table 22 — Mapping of use cases to requirements

Use case # and name	REQ	Technical requirement title
UC 1.1 – Discover DoCAN communication interface	8.10 to 8.20	APP – Setup DoCAN communication interface
UC 1.2 – Discover DoIP communication interface	8.21 to 8.28	APP – Setup DoIP communication interface
UC 2.1 – Perform PDT authentication	8.30 to 8.33	APP – PDT authentication against fixed-address PCU – Optional (Sys-Auth)
UC 2.2 – Perform fixed-address PCU/PCU(s) authentication	8.34	APP – Fixed-address PCU authentication against PDT – Optional (Sys-Auth)
UC 3.1 – Report PCU hardware deployment method	8.35	APP – Report PcuHardwareDeploymentMethod
UC 3.2 – Report number of PCU(s)	8.36	APP – Report number of PCUs
UC 3.3 – Report address information of PCU(s)	8.37 to 8.39	APP – Report DoCAN address information of PCUs
	8.40 to 8.42	APP – Report DoIP address information of PCUs
UC 3.4 – Report vehicle identification number	8.43	APP – Report vehicle identification number
UC 3.5 – Report dismantling documentation of PCU	8.44	APP – Report dismantling documentation of PCU
UC 4.1 – Report PCU deployment loop identification table	8.45	APP – Report PCU deployment loop identification table
UC 4.2 – Initiate safetySystemDiagnosticSession	8.46	APP – Initiate safetySystemDiagnosticSession
UC 4.3 – Keep-alive safetySystemDiagnosticSession	8.47	APP – Keep-alive safetySystemDiagnosticSession
UC 4.4 – Unlock security of PCU	8.48	APP – Unlock security of PCU
UC 4.5 – Execute PCU(s) scrapping program module loader	8.49	APP – Execute PCU scrapping program module loader
UC 5.1 – Report ACL deployment sequence (ACL-Init)	8.50	APP – Report ACL deployment sequence
UC 5.2 – Write dismantling documentation into PCU (Device-Deploy)	8.51	APP – ACL-Prep – Write dismantler information into PCU
UC 5.3 – Perform ACL deployment confirmation sequence (Device-Deploy)	8.53	APP – Device-Deploy – Confirm ACL deployment sequence
UC 5.4 – Perform PCU pyrotechnic device scrapping via loop identification (Device-Deploy)	8.54 to 8.56	APP – Device-Deploy – Perform device scrapping APP – Device-Deploy – Evaluation of device scrapping APP – Device-Deploy – Next pyrotechnic device
UC 6.1 – Terminate PCU pyrotechnic device scrapping via communication interface	8.57	APP – Terminate PCU pyrotechnic device scrapping
UC 6.2 – Terminate PCU pyrotechnic device scrapping via ACL	8.58	APP – Terminate PCU pyrotechnic device scrapping via ACL

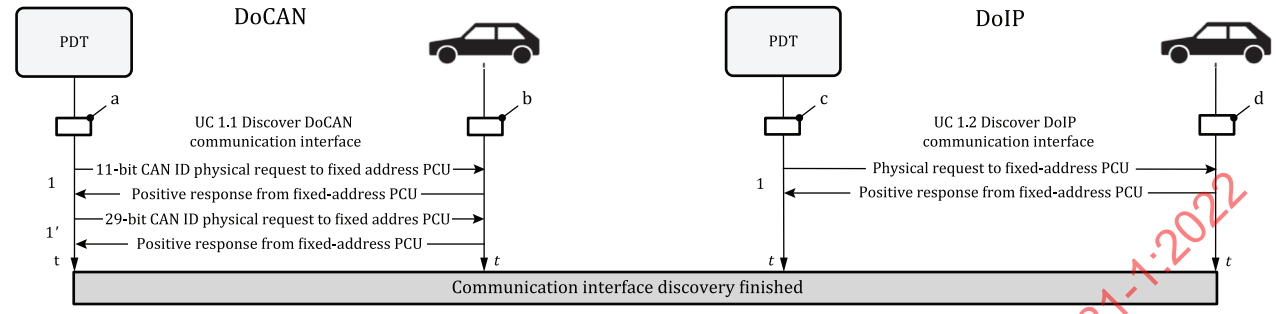
8.5 APP – Application timing definition

REQ	8.9 APP – Application timing definition – $t_{P3\ Client\ Phys}$ timing specification
The $t_{P3\ Client\ Phys}$ time between end of server responses and start of new client request shall be set to 100 ms.	

8.6 APP – Discovery of communication interface (Com I/F-Discovery)

8.6.1 APP – Overview of discovery of communication interface (Com-Discovery)

Figure 4 shows the discovery of communication interface – (Com I/F-Discovery).



Key

t time

a Set-up the PDT DoCAN data link interface.

b Set-up the vehicle's DoCAN communication interface.

c Set-up PDT DoIP data link interface.

d Set-up vehicle's DoIP communication interface.

Figure 4 — Discovery of communication interface (Com I/F-Discovery)

8.6.2 APP – Setup DoCAN communication interface

The requirements specified in this subclause are applicable if a DoCAN communication interface framework is used for the deployment of pyrotechnic devices.

Applicable use case: UC 1.1 – Discover DoCAN communication interface

REQ	8.10 APP – Setup DoCAN communication interface – DoCAN diagnostic link connector
The vehicle shall meet the requirements stated in ISO 15031-3.	

REQ	8.11 APP – Setup DoCAN communication interface – DoCAN connection on diagnostic link connector
The vehicle shall support the ISO 11898-1, ISO 11898-2 CAN, ISO 15765-5 data link and ISO 15765-2 DoCAN transport and network layer. The vehicle shall have installed the diagnostic link connector and pin assignment to support DoCAN. — ISO 15031-3; — Pin 6: CAN_H line; and — Pin 14: CAN_L line.	

REQ	8.12 APP – Setup DoCAN communication interface – DoCAN network bit rate
The vehicle shall only support one arbitration bit rate:	

REQ	8.12 APP – Setup DoCAN communication interface – DoCAN network bit rate
— 250 kbit/s; — 500 kbit/s. Requirements to support either bit rate are specified in ISO 15765-5.	

REQ	8.13 APP – Setup DoCAN communication interface – DoCAN configuration of 11-bit CANID protocol
The vehicle shall support the 11-bit CAN identifiers as specified in Table 23 if REQ 8.14 is not supported.	

Table 23 — Specification of 11-bit CAN identifier format – Normal addressing

Message type	Bit 10 to 0	Description
Physical request	$7F1_{16}$	Physical request CAN identifier from the PDT to the fixed-address PCU
Physical response	$7F9_{16}$	Physical response CAN identifier from the fixed-address PCU to the PDT

REQ	8.14 APP – Setup DoCAN communication interface – DoCAN configuration of 29-bit CANID protocol
The vehicle shall support the 29-bit CAN identifiers as specified in Table 24 if REQ 8.13 is not supported.	

Table 24 — Specification of 29-bit CAN identifier format – Normal fixed addressing

Message type	Bit 28 to 24	Bit 23 to 16	Target address (TA)	Source address (SA)	Description
Physical request	18_{16}	DA_{16}	53_{16}	$F1_{16}$	TA = Fixed-address PCU SA = PDT
Physical response	18_{16}	DA_{16}	$F1_{16}$	53_{16}	TA = PDT SA = Fixed-address PCU

REQ	8.15 APP – Setup DoCAN communication interface – DoCAN addressing formats during initialisation configuration
The CAN communication interface shall support either the normal addressing format with 11-bit CAN identifier or the normal fixed addressing format with 29-bit CAN identifier during the communication initialisation according to ISO 15765-5.	

REQ	8.16 APP – Setup DoCAN communication interface – Support of "Classical CAN"
The CAN communication interface shall support "Classical CAN" according to ISO 15765-5.	

REQ	8.17 APP – Setup DoCAN communication interface – DoCAN 29-bit CANID maximum number of ECUs
The vehicle shall have equal or less than 240 PCU-relevant servers/ECUs installed.	

REQ	8.18 APP – Setup DoCAN communication interface – DoCAN physical addressing
The PCU(s) shall support physically addressed request messages (`TAtype = physical`).	

REQ	8.19 APP – Setup vehicle's DoCAN data link framework – Size of A_PDU in the request message
The PCU(s) shall be able to process physically addressed services (request message) with only one DID.	

REQ	8.20 APP – Setup DoCAN communication interface – DoCAN application message timing
The DoCAN application message timing specification shall be in accordance with ISO 14229-2 and REQ 8.9 $t_{P3\_Client\_Phys}$ timing specification.	

8.6.3 APP – Setup DoIP communication interface

The requirements specified in this subclause are applicable if a DoIP communication interface framework is used for the deployment of pyrotechnic devices.

Applicable use case: UC 1.2 – Discover DoIP communication interface

REQ	8.21 APP – Setup DoIP communication interface – DoIP diagnostic link connector
The vehicle shall meet the requirements stated in ISO 13400-4.	

REQ	8.22 APP – Setup DoIP communication interface – DoIP configuration of protocol
The vehicle shall use the DoIP configuration and shall meet the requirements stated in ISO 13400-2 and ISO 13400-3.	

REQ	8.23 APP – Setup DoIP communication interface – Fixed-address PCU DoIP physical logical address
The vehicle shall support the physical logical address assignment according to ISO 13400-2. According to this document, the DoIP physical logical source address and physical target address for physical request and physical response messages shall be implemented as specified in Table 25.	

Table 25 — Fixed-address PCU – DoIP physical logical address assignment

Addressing message type	Target address (TA)	Source address (SA)	Description
Physical request	$E002_{16}$	$0E02_{16}$	TA = fixed-address PCU/gateway SA = PDT
Physical response	$0E02_{16}$	$E002_{16}$	TA = PDT SA = fixed-address PCU/gateway

REQ	8.24 APP – Setup DoIP communication interface – Any DoIP PCU physical logical address
The vehicle's PCU(s) shall support the physical logical address assignments according to ISO 13400-2. According to this document, the DoIP physical logical source address and physical target address for physical request and response messages shall be implemented as specified in Table 26.	

Table 26 — Any PCU – DoIP physical logical address assignment

Addressing message type	Target address (TA)	Source address (SA)	Description
Physical request	$YYXX_{16}$[a]	$0E02_{16}$	TA = fixed-address PCU/gateway and any other PCU SA = PDT

[a] Any PCU address $YYXX_{16}$ ranges from 0001_{16} to $0DFF_{16}$ or 1000_{16} to $7FFF_{16}$.

Table 26 *(continued)*

Addressing message type	Target address (TA)	Source address (SA)	Description
Physical response	$0E02_{16}$	$YYXX_{16}$[a]	TA = PDT SA = fixed-address PCU/gateway and any other PCU

[a] Any PCU address $YYXX_{16}$ ranges from 0001_{16} to $0DFF_{16}$ or 1000_{16} to $7FFF_{16}$.

REQ	8.25 APP – Setup DoIP communication interface – DoIP maximum number of ECUs
The vehicle shall have equal or less than 240 PCU-relevant servers/ECUs installed.	

REQ	8.26 APP – Setup DoIP communication interface – DoIP physical addressing
The fixed-address PCU shall be able to process physically addressed services (request message).	

REQ	8.27 APP – Setup DoIP communication interface – Size of A_PDU in the request message
The fixed-address PCU shall be able to process physically addressed services (request message) with only one DID. The ECU can respond to longer requests, i.e. certificates.	

REQ	8.28 APP – Setup DoIP communication interface – DoIP application message timing specification in defaultSession
The DoIP application message timing specification in the defaultSession shall be in accordance with the following specifications. For the parameters which are standardised in ISO 26021-3, the vehicle shall be able to support an external test equipment with the following timings: t_{P6_Client} = 5 000 ms; t_{P6*_Client} = 10 000 ms.	

For vehicle manufacturer-specific diagnostic services, the communication parameters defined by the vehicle manufacturer are applicable.

These timings can only be achieved if the total length of diagnostic request and response is smaller than 4 kbyte. Thus, ISO 26021-3 does not define standardised services greater than this boundary.

The application message timing specification specified in this subclause ensures that an ISO 26021-1-conformant vehicle can respond within its response performance required.

TCP is a stream-oriented protocol. It is possible that more than one DoIP response message is sent via a single TCP segment. Also, TCP uses acknowledging and automatic retries, which depend on the overall network performance and reliability. Thus, it is not possible to map individual responses into individual IP packets. Therefore, the message sequence charts only provide a logical view of multiple messages on the Ethernet/IP/TCP side, which can differ from the actual IP packet transmission.

8.6.4 APP – Determination of DoCAN or DoIP communication interface in the vehicle

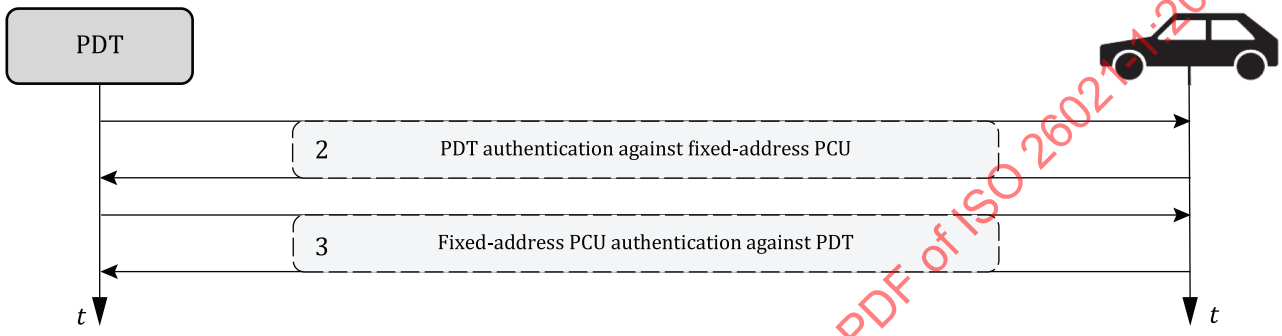
The requirement specified in this subclause is applicable to the mixture of pyrotechnic devices (PCUs) across both communication interfaces, DoCAN or DoIP.

REQ	8.29 APP – Determination of DoCAN or DoIP communication interface in the vehicle
The vehicle shall only listen to the communication interface selected by the PDT for all further requests as specified in this document. Two PDTs, one connected to DoCAN communication interface and another connected to the DoIP communication interface shall not be supported by the vehicle.	

8.7 APP – Perform authentication – Optional (Sys-Auth)

8.7.1 APP – Overview of the authentication – Optional (Sys-Auth)

Figure 5 shows an overview of the authentication – Optional (Sys-Auth).



Key

t time

Figure 5 — Overview of the authentication – Optional (Sys-Auth)

8.7.2 APP – PDT authentication against fixed-address PCU – Optional (Sys-Auth)

The implementation requirements of the PDT authentication specify the message sequence and relevant requirements.

Applicable use case: UC 2.1 – Perform PDT authentication

REQ	8.30 APP – PDT authentication against fixed-address PCU – General requirement #1
The authentication of the PDT against the ECU(s) shall be in accordance with ISO 14229-1 “Authentication” or “SecurityAccess” services.	

REQ	8.31 APP – PDT authentication against fixed-address PCU – General requirement #2
If the PDT sends valid PDT credentials, the vehicle shall grant access rights to the PDT for at least the pyrotechnic device relevant content, which is also used for end-of-life deployment of pyrotechnic devices. The vehicle shall grant access to at least the activation of routine(s) as defined in ISO 26021-3. NOTE All relevant credentials required by the PDT are delivered offline.	

It is recommended to use the same algorithm as specified in ISO 14229-1 as for other diagnostic use cases.

REQ	8.32 APP – PDT authentication against fixed-address PCU – Allowed security concepts for using service “Authentication”
Authentication shall be implemented using PKI Certificate Exchange (APCE), see ISO 14229-1.	

REQ	8.33 APP – PDT authentication against fixed-address PCU – Publication of used algorithms
If an algorithm is used for authentication, its reference shall be published in the Object Identifier (OID) repository http://oid-info.com/ according to ISO/IEC 9834-1 under the node being used for vehicle system and PDT authentication.	

8.7.3 APP – Fixed-address PCU authentication against PDT – Optional (Sys-Auth)

Applicable use case: UC 2.2 – Perform fixed-address PCU/PCU(s) authentication

REQ	8.34 APP – Fixed-address PCU authentication
The authentication of the fixed-address PCU(s) against the PDT shall be in accordance with ISO 14229-1.	

8.8 APP – Perform system initialisation (Sys-Init)

8.8.1 APP – Overview of the system initialisation (Sys-Init)

In step 4 the vehicle's fixed-address PCU reports the PCU hardware deployment method. The PCU hardware deployment method contains:

— PCU hardware deployment method version: the version identifies the edition of the ISO 26021 series document, e.g. ISO 26021-2 edition 1, ISO 26021-1 edition 2 (this document) to determine preconditions (see 9.1), diagnostic protocol services (see Clause 11), and the sequence used for the PCU deployment (see 9.2),

— PCU identification string, and

— additional data for future use.

In step 5 the vehicle's fixed-address PCU reports the number of PCU(s) information. The NumberOfPCU value specifies the outer loop of the scrapping sequence, which is stored for later use in the sequence executed by the PDT.

In step 6 the vehicle's fixed-address PCU reports the DoCAN/DoIP address information of each PCU (#1 to #N). The address information is used by the PDT to individually address each PCU on the identified network configuration.

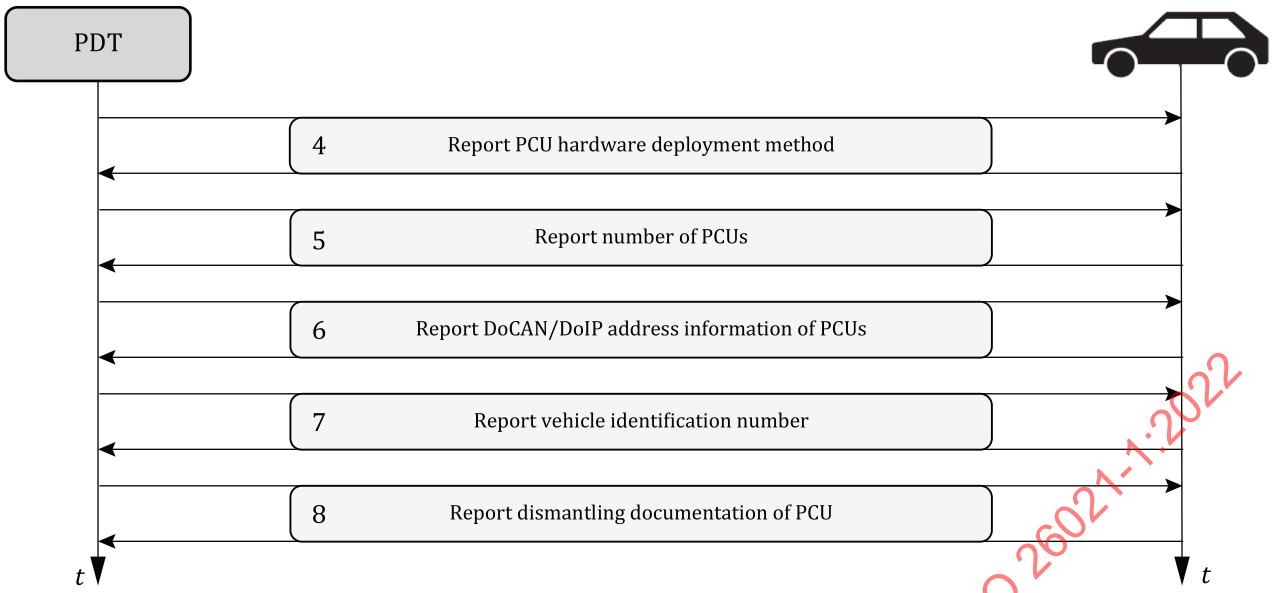
In step 7 the vehicle's fixed-address PCU reports the vehicle identification number. The support of the VIN is optional.

In step 8 the vehicle's fixed-address PCU reports the dismantling information. The dismantling information contains:

— the PCU deployment device identification value,

— the year, the month, and the day of deployment.

Figure 6 shows an overview of the system initialisation sequence (Sys-Init).



Key

t time

Figure 6 — Overview of the system initialisation (Sys-Init)

8.8.2 APP – Report PcuHardwareDeploymentMethod (Sys-Init)

The fixed-address PCU provides the hardware deployment method information upon request by the PDT. The PcuHardwareDeploymentMethod version identifies the version of the diagnostic protocol services and the sequence used for the PCU deployment. The PCU identification string is assigned to the fixed-address PCU by the vehicle manufacturer.

Applicable use case: UC 3.1 – Report PCU hardware deployment method

REQ	8.35 APP – Report PcuHardwareDeploymentMethod (Sys-Init) – PCU hardware deployment method
Upon request the fixed-address PCU shall report the PCU hardware deployment method to inform the PDT about the deployment method to be used.	

8.8.3 APP – Report number of PCUs (Sys-Init)

The PDT requests the number of PCUs from the fixed-address PCU. This number, which specifies the outer loop of the scrapping sequence, is stored for later use in the sequence of the PDT.

A PCU on the network/subnet is part of the reported count if it connects to a pyrotechnic device(s). A PCU without a connected pyrotechnic device(s) is not counted.

Applicable use case: UC 3.2 – Report number of PCU(s)

REQ	8.36 APP – Sys-Init – Report number of PCUs
Upon request the fixed-address PCU shall report the number of PCUs on the network/subnet which connects a pyrotechnic device(s).	

8.8.4 APP – Report DoCAN address information of PCUs (Sys-Init)

Upon request the fixed-address PCU reports by PCU address format identifier 01_{16} to 06_{16} the address format used by each PCU. Both the request and the response address information of each PCU are included in a 32-bit field.

Applicable use case: UC 3.3 – Report address information of PCU(s)

REQ	8.37 APP – Sys-Init – DoCAN address mapping into 32-bit field
The fixed-address PCU shall report upon request the address format information for PCU #1 to #N, which use the address format identified by 01_{16} to 06_{16} as specified in Tables 27 and 28.	

REQ	8.38 APP – Sys-Init – DoCAN format identifiers
The vehicle manufacturer shall only implement DoCAN format identifiers in the fixed-address PCU as specified in Tables 27 and 28.	

The diagnostic service and data record, which includes the addressFormatIdPcu, requestMsgAddrPcu, and responseMsgAddrPcu, is specified in 10.6. The following PCU address format identifiers are described.

— PCU address FMT ID 01_{16}:

In Table 27 the PCU address format identifier 01_{16} specifies a DoCAN 11-bit address parameter (service primitive interface parameter N_AI). This N_AI[] format is used for the report of the request and response CAN identifier of each PCU (see 10.6). The network architecture 'A' is referenced but other network architectures are possible and left to the discretion of the vehicle manufacturer.

— PCU address FMT ID 02_{16}:

In Table 27 the PCU address format identifier 02_{16} specifies a DoCAN 11-bit address parameter (service primitive interface parameter N_AI) and a target address N_AI[TA] of a PCU connected to the DoCAN subnet behind the gateway/fixed-address PCU. The N_AI[TA] format is used for the report of the request and response CAN identifier of each PCU (see 10.6). The network architecture 'B' is referenced but other network architectures are possible and left to the discretion of the vehicle manufacturer.

— PCU address FMT ID 03_{16}:

In Table 27 the PCU address format identifier 03_{16} specifies a DoCAN 11-bit address parameter (service primitive interface parameter N_AI) and a target address N_AI[TA] of a PCU connected to the DoCAN subnet behind the gateway/fixed-address PCU. The N_AI[TA] format is used for the report of the request and response CAN identifier of each PCU (see 10.6). The network architecture 'B' is referenced but other network architectures are possible and left to the discretion of the vehicle manufacturer.

— PCU address FMT ID 04_{16}:

In Table 27 the PCU address format identifier 04_{16} specifies a DoCAN 29-bit address parameter (service primitive interface parameter N_AI). The N_AI[TA] format is used for the report of the request and response CAN identifier of each PCU (see 10.6). The network architecture 'A' is referenced but other network architectures are possible and left to the discretion of the vehicle manufacturer.

— PCU address FMT ID 05_{16}:

In Table 27 the PCU address format identifier 05_{16} specifies a DoCAN 29-bit address parameter (service primitive interface parameter N_AI) and a N_AI[TA] of a PCU connected to the DoCAN subnet behind the gateway/fixed-address PCU. The N_AI[TA] format is used for the report of the request and response CAN identifier of each PCU (see 10.6). The network architecture 'B' is referenced but other network architectures are possible and left to the discretion of the vehicle manufacturer.

Table 27 — Mapping of DoCAN N_PDU parameters into the 32-bit address

PCU address FMT ID	Network architecture (see Annex B) and addressing	32-bit field request/response address[a] Bit 31 (MSb)					Bit 0 (LSb)
		31 to 24	23 to19	18 to 16	15 to 11	10 to 8	7 to 0
01_{16}	'A' (see Figure B.1) DoCAN 11 bit normal addressing	00_{16}	$0000\ 0_2$	N_AI[TA/SA] (fixed-address DoCAN PCU)			FF_{16}
02_{16}	'B' (see Figure B.2) DoCAN 11 bit extended addressing	00_{16}	$0000\ 0_2$	N_AI[SA] (fixed-address DoCAN PCU)			TA (any other DoCAN PCU)
03_{16}	'B' (see Figure B.2) DoCAN 11 bit mixed addressing	00_{16}	$0000\ 0_2$	N_AI[TA/SA] (fixed address DoCAN PCU)			AE (any other DoCAN PCU)
04_{16}	'A' (see Figure B.1) DoCAN 29 bit normal fixed addressing	00_{16}	N_AI[TA] (fixed-address DoCAN PCU and any other PCU)		N_AI[SA] (DoCAN client 1 PDT)		FF_{16}
05_{16}	'B' (see Figure B.2) DoCAN 29 bit mixed addressing	00_{16}	N_AI[TA] (fixed-address DoCAN PCU)		N_AI[SA] (DoCAN client 1 PDT)		AE (any other DoCAN PCU)

[a] Mapping of N_AI, N_TA, N_SA, and N_AE address information.

The diagnostic service and data record, which includes the addressFormatIdPcu, requestMsgAddrPcu, and responseMsgAddrPcu, is specified in 10.6. The PCU address FMT ID 06_{16} is specified in Table 28.

Table 28 — Mapping of DoCAN unique addressing N_PDU parameters into the 32-bit address

PCU address FMT ID	Network architecture (see Annex B) and addressing	32-bit field request/response address[a] Bit 31 (MSb)				Bit 0 (LSb)
		31 to 24	23	22	21 to11	10 to 0
06_{16}	DoCAN unique addressing	00_{16}	1_2	1_2	unique N_AI[TA/SA] (DoCAN client 1 PDT)	unique N_AI[SA/TA]

[a] It is the mapping of N_AI[TA] and N_AI[SA] address information.

REQ	8.39 APP – Sys-Init – DoCAN addressing format after initialisation configuration
After the initialisation configuration, the vehicle's fixed-address PCU shall support any vehicle manufacturer-specific combination of addressing formats as specified in REQ 8.37 and REQ 8.38.	

8.8.5 APP – Report DoIP address information of PCUs (Sys-Init)

Upon request the fixed-address PCU reports by PCU address format identifier 10_{16} the address format used by each PCU. Both the request and the response address information of each PCU are included in a 32-bit field.

Applicable use case: UC 3.3 – Report address information of PCU(s)

REQ	8.40 APP – Sys-Init – DoIP address mapping into 32-bit field
The fixed-address PCU shall report upon request the address format information for PCU #1 to #N, which use the address format identified by 10_{16} as specified in Table 29.	

REQ	8.41 APP – Sys-Init – DoIP format identifier
The vehicle manufacturer shall implement the DoIP format identifier in the fixed-address PCU as specified in Table 29.	

The diagnostic service and data record, which includes the addressFormatIdPcu, requestMsgAddrPcu, and responseMsgAddrPcu, is specified in 10.6. The PCU address FMT ID 10_{16} specifies the DoIP address parameter (service primitive interface parameter N_AI). The network architecture 'D' to 'F' is referenced but other network architectures are possible and left to the discretion of the vehicle manufacturer.

Table 29 — Mapping of DoIP addressing N_PDU parameters into the 32-bit address

PCU address FMT ID	Network architecture (see Annex B) and addressing	32-bit field request/response address[a] Bit 31 (MSb) 31 to 16	Bit 0 (LSb) 15 to 0
10_{16}	'D' to 'F'(see Figures 5 to 7) DoIP addressing	N_AI[TA] (physical logical request address DoIP PCU and any other PCU) (DoIP response address client 1 PDT)	N_AI[SA] (DoIP request address client 1 PDT) (physical logical response address DoIP PCU and any other PCU)

[a] It is the mapping of N_AI[TA] and N_AI[SA] address information.

REQ	8.42 APP – Sys-Init – DoIP addressing format after initialisation configuration
After the initialisation configuration, the vehicle's fixed-address PCU shall support the DoIP addressing format as specified in REQ 8.40 and REQ 8.41.	

8.8.6 APP – Report vehicle identification number (Sys-Init)

Applicable use case: UC 3.4 – Report vehicle identification number

REQ	8.43 APP – Sys-Init – Report vehicle identification number
The fixed-address PCU shall report the vehicle identification number (VIN) upon request by the PDT, if supported by the vehicle.	

8.8.7 APP – Report dismantling documentation of PCU (Sys-Init)

Applicable use case: UC 3.5 – Report dismantling documentation of PCU

REQ	8.44 APP – Sys-Init – Report dismantler documentation of PCU
The fixed-address PCU shall report the dismantler documentation upon request by the PDT. The dismantler documentation shall contain the DismantlerIdentification data, the PCU deployment device identification value, the year, month, and the day of deployment. The dismantler documentation shall be stored in the fixed-address PCU's protected, permanent memory.	

8.9 APP – Perform PCU initialisation (PCU-Seq)

8.9.1 APP – Overview of the PCU initialisation (PCU-Seq)

In step 9 the vehicle's PCU #1 to PCU #N reports the PCU deployment loop identification table. The PCU deployment loop identification table contains:

— whether an additional communication line (ACL) is used and what type of ACL configuration is implemented in the vehicle,

— the ACL method version,

— the number of loop table records,

— the loop identification #1 to loop identification #N, and

— the loop status #1 to loop status #N.

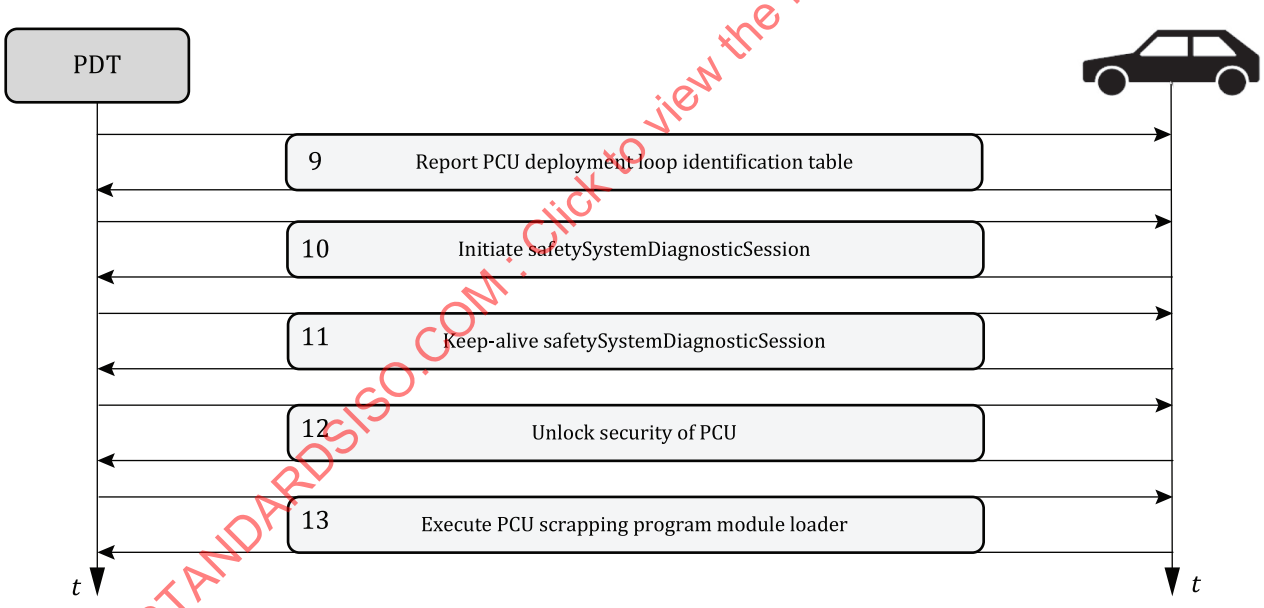
In step 10 the vehicle's PCU changes the diagnostic session to the safetySystemDiagnosticSession upon request by the PDT.

In step 11 the vehicle's PCU safetySystemDiagnosticSession is kept-alive by PDT request messages.

In step 12 the vehicle's PCU security is changed to the unlock state upon request by the PDT.

In step 13 the vehicle's PCU executes the scrapping program module loader (SPL). The SPL copies the SPM into free RAM space. The RAM space is initialised after a reset. The SPL converts the scrapping program module (SPM), which is stored in a non-executable format, into an executable form. The SPM inside the PCU performs the steps to control the output stages, overrides the safing unit (alternative use of ACL), and carries out the communication to the PDT. After a further communication sequence of the PDT with the PCU, the SPM communicates to the independent electronic safing unit (if no ACL is available), activates the output stages and records this event for each deployment loop. If an ACL is present in the vehicle, the unlock signal on this line is present during the deployment event and evaluated by the independent safing unit to release the output. After the deployment sequence is completed of this PCU, the PDT requests a reset of the PCU and the PCU exits the scrapping mode.

Figure 7 shows an overview of the PCU initialisation (PCU-Seq).



Key

t time

Figure 7 — Overview of the PCU initialisation (PCU-Seq)

8.9.2 APP – Report PCU deployment loop identification table (PCU-Seq)

Applicable use case: UC 4.1 – Report PCU deployment loop identification table

REQ	8.45 APP – PCU-Seq – Report PCU deployment loop identification table
The PCU in the vehicle shall report the deployment loop identification table upon request by the PDT. The deployment loop identification table shall contain:	

REQ	8.45 APP – PCU-Seq – Report PCU deployment loop identification table
—	the type of ACL required by the diagnostic protocol services and the sequence used for deployment of the pyrotechnic device, — the ACL method version, — the number of loop table records, — the loop identification #1, — the loop status #1, — the loop identification #N, and — the loop status #N.

8.9.3 APP – Initiate safetySystemDiagnosticSession (PCU-Seq)

Applicable use case: UC 4.2 – Initiate safetySystemDiagnosticSession

REQ	8.46 APP – PCU-Seq – Initiate safetySystemDiagnosticSession
The addressed PCU in the vehicle shall support the safetySystemDiagnosticSession upon request by the PDT.	

8.9.4 APP – Keep-alive safetySystemDiagnosticSession (PCU-Seq)

Applicable use case: UC 4.3 – Keep-alive safetySystemDiagnosticSession

REQ	8.47 APP – PCU-Seq – Keep-alive safetySystemDiagnosticSession
The addressed PCU in the vehicle shall keep the safetySystemDiagnosticSession active as long as it receives the keep-alive TesterPresent request messages in intervals of the t_{S3_Client} timer reload value specified in ISO 14229-2 by the PDT.	

8.9.5 APP – Unlock security of PCU (PCU-Seq)

The securityAccess key value from the PDT is used to enable the SPL to load the SPM into the RAM and convert the SPM to an executable format.

Applicable use case: UC 4.4 – Unlock security of PCU

REQ	8.48 APP – PCU-Seq – Unlock security of PCU
The addressed PCU in the vehicle shall support the security unlock sequence upon request by the PDT. See Table 31 “SecurityAccess (enable scrapping)” for “requestSeed” and “sendKey” values.	

8.9.6 APP – Execute PCU scrapping program module loader (PCU-Seq)

Applicable use case: UC 4.5 – Execute PCU(s) scrapping program module loader

REQ	8.49 APP – PCU-Seq – Execute PCU scrapping program module loader
The addressed PCU in the vehicle shall listen on the supported communication interface for a reception of a RoutineControl request message with a routineControlType = startRoutine, a routineIdentifier = ExecuteSPL, and a routineControlOptionRecord = ExecuteSPL parameter. Upon successful reception the PCU shall start the SPL to load the SPM into the executable RAM and shall convert the SPM to an executable format.	

8.10 APP – Perform PCU and ACL scrapping (Device-Deploy)

8.10.1 APP – Overview of the PCU- and ACL-Scrapping (Device-Deploy)

In step 14, the vehicle's fixed-address PCU reports to the PDT the PCU hardware deployment method that the vehicle architecture supports. There are two hardware deploment methods: a PCU(s) with an ACL supporting bidirectional communication or a PWM signal at the diagnostic link connector.

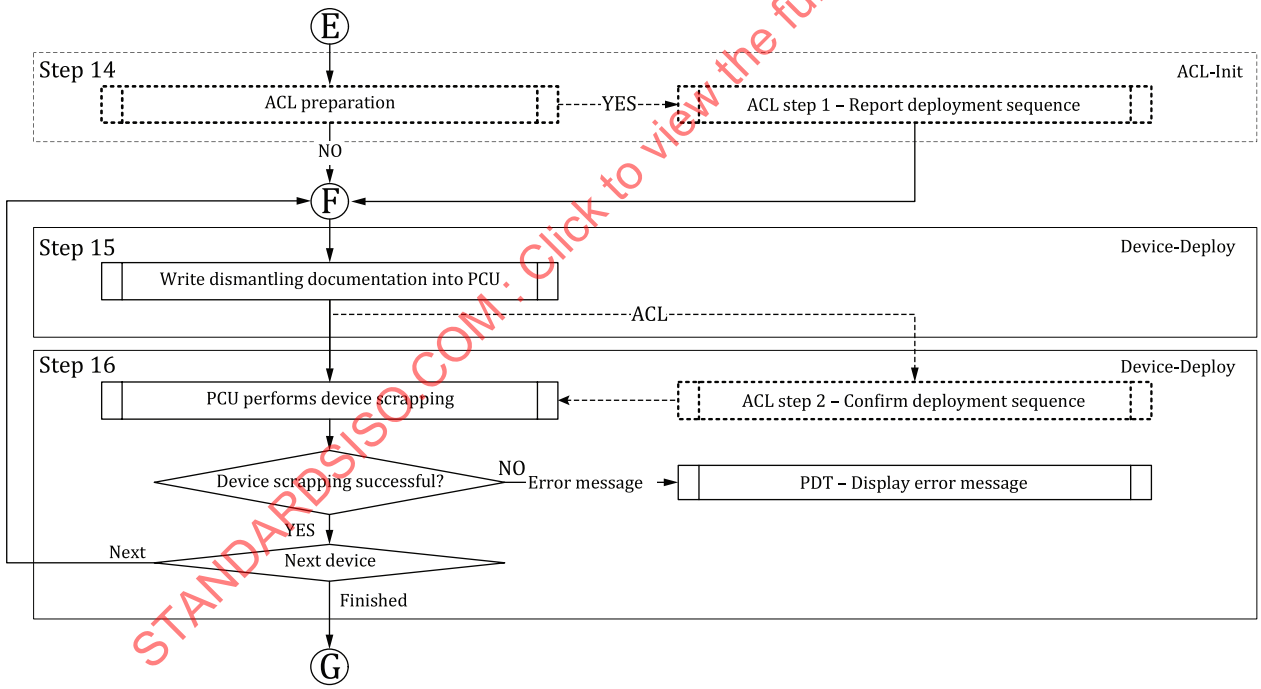
In step 15, the vehicle's fixed-address PCU(s) waits on the reception of a request to write the dismantling information in the protected, permanent memory of the PCU. The dismantling information consists of:

— the PCU deployment device identification value,

— the year, the month, and the day of deployment.

In step 16, the vehicle's PCU determines the deployment communication method of pyrotechnic devices based on the PCU deployment loop identification table content. The deployment communication method is either based on the ACL deployment confirmation response message, the signal code via the ACL, or the supported communication interface (DoCAN, DoIP). The PCU determines the number of internal loops to execute the firing of one or multiple pyrotechnic devices. The PCU transmits a RoutineControl positive response message with a routineControlType = startRoutine, a routineIdentifier = DeployLoopRoutineID, and a routineStatusRecord including deploymentLoopID information. After the device scrapping finished successful the next PCU performs the same device deployment.

Figure 8 shows an overview of the PCU- and ACL-Scrapping (Device-Deploy).



Key

E vehicle's PCU determines that the PCU initialisation is successful

F vehicle's PCU determines that ACL is not supported

G PCU sequence has reached the state PCU-End

Figure 8 — Overview of the PCU- and ACL-Scrapping (PCU- and ACL-Scrapping)

8.10.2 APP – Report ACL deployment sequence (ACL-Prep)

Applicable use case: UC 5.1 – Report ACL deployment sequence (ACL-Init)

REQ	8.50 APP – ACL-Prep – Report ACL deployment sequence
The vehicle's fixed-address PCU shall report/transmit the ACL command for deployment upon a PDT request with bidirectional communication or PWM signal code via the ACL.	

8.10.3 APP – Write dismantling documentation into PCU (Device-Deploy)

The purpose of writing the dismantler documentation into the PCU's memory is to automatically detect the in-vehicle configuration of the deployment method and version of the pyrotechnic devices. Writing of the dismantler information is performed during activation of the first firing loop or during the preparation of the deployment session. In case there is already a DismantlerIdentification data written in the PCU's memory then the newly received updated information overwrites the stored information.

Applicable use case: UC 5.2 – Write dismantling documentation into PCU (Device-Deploy)

REQ	8.51 APP – ACL-Prep – Write dismantler information into PCU
The vehicle's fixed-address PCU shall write the DismantlerIdentification data of the PDT, the PCU deployment device identification value, year, month, and day of deployment values into the fixed-address PCU's protected, permanent memory.	

REQ	8.52 APP – ACL-Prep – PCU restart of the deployment-program
The vehicle's fixed-address PCU shall support a restart of the deployment program upon request of the PDT.	

After the PDT detects the PCU, the PDT carries out 8.10.4 to 8.10.7 to perform the deployment process. In the Sys-Init deployment process, the vehicle is requested to provide data on the specific configuration. The PDT stores the data for later use in the sequence. The PCUs are handled in exactly the sequence in which they are stored in the fixed-address PCU.

8.10.4 APP – Confirm ACL deployment sequence (Device-Deploy)

Applicable use case: UC 5.3 – Perform ACL deployment confirmation sequence (Device-Deploy)

REQ	8.53 APP – Device-Deploy – Confirm ACL deployment sequence
The vehicle's PCU shall transmit the ACL deployment confirmation via the bidirectional communication or via PWM signal code on the ACL.	

8.10.5 APP – Perform device scrapping (Device-Deploy)

Applicable use case: UC 5.4 – Perform PCU pyrotechnic device scrapping via loop identification (Device-Deploy)

REQ	8.54 APP – Device-Deploy – Perform device scrapping
The vehicle's PCU shall determine the number of internal loops to execute the firing of one or multiple pyrotechnic devices.	

The information is contained in the loop identification table.

8.10.6 APP – Evaluation of device scrapping (Device-Deploy)

Applicable use case: UC 5.4 – Perform PCU pyrotechnic device scrapping via loop identification (Device-Deploy)

REQ	8.55 APP – Device-Deploy – Evaluation of device scrapping
The vehicle's PCU shall evaluate the pyrotechnic device scrapping of the independent safing unit during the deployment event. If the deployment event is not successful, the vehicle's PCU shall send an error message to the PDT.	

8.10.7 APP – Next pyrotechnic device (Device-Deploy)

Applicable use case: UC 5.4 – Perform PCU pyrotechnic device scrapping via loop identification (Device-Deploy)

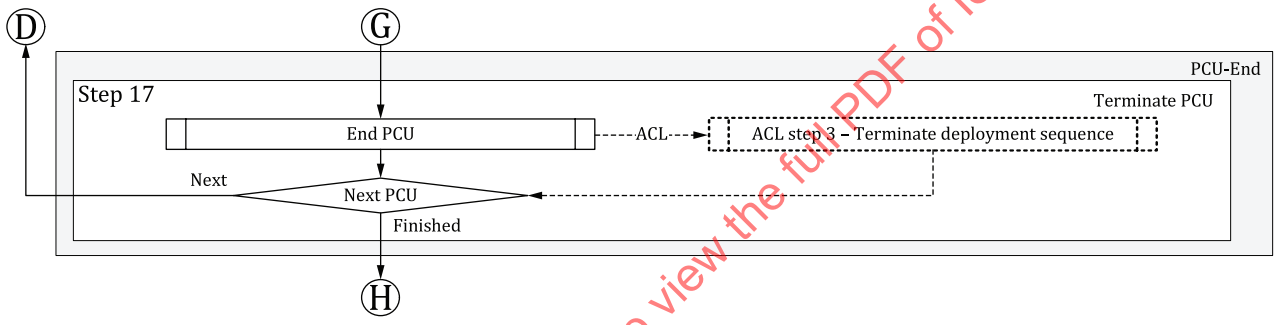
REQ	8.56 APP – Device-Deploy – Next pyrotechnic device
The vehicle's PCU shall check whether the pyrotechnic device scrapping for the given PCU is completed. If not completed the given PCU shall continue with the next pyrotechnic device by writing the dismantler information into the PCU. If completed the given PCU shall end the pyrotechnic device scrapping for the given PCU.	

8.11 APP – Terminate PCU and ACL pyrotechnic device deployment (PCU-End)

8.11.1 APP – Overview of the PCU- and ACL-Termination (PCU-End)

In step 17 the PCU terminates the device scrapping. If the optional ACL is implemented the PCU also terminates the ACL bidirectional communication/PWM signals. After termination of this PCU the next PCU is processed.

Figure 9 shows an overview of the PCU- and ACL-Termination (PCU-End).



Key

D vehicle's fixed-address PCU determines that the authentication (optional) and the system initialisation is successful

G PCU sequence has reached the state PCU-End

H PCU sequence has reached the state PCU-End

Figure 9 — Overview of the PCU- and ACL-Termination (PCU-End)

8.11.2 APP – Terminate PCU pyrotechnic device scrapping (PCU-End)

The PCU terminates the safetySystemDiagnosticSession if either a reset command messsage is received or a communication interface timeout period is exceeded.

Applicable use cases:

— UC 6.1 – Terminate PCU pyrotechnic device scrapping via communication interface,

— UC 6.2 – Terminate PCU pyrotechnic device scrapping via ACL.

REQ	8.57 APP – PCU-End – Terminate PCU pyrotechnic device scrapping
The vehicle's PCU shall end the safetySystemDiagnosticSession after the reception of an ECUReset diagnostic request message or if a communication interface timeout period, as specified in ISO 14229-2, is detected by the application.	

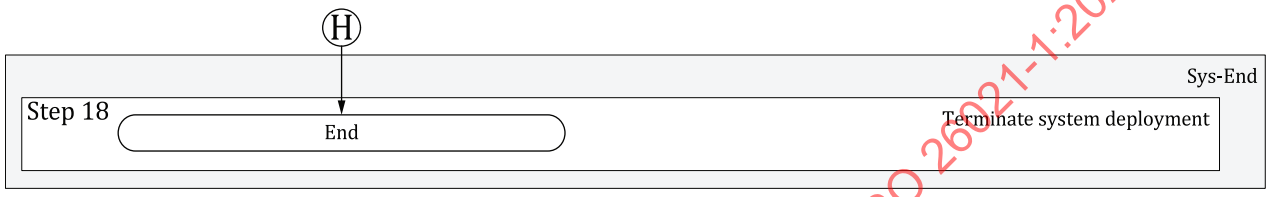
8.11.3 APP – Terminate PCU pyrotechnic device scrapping via ACL (PCU-End)

To terminate the PCU pyrotechnic device scrapping via the ACL the PCU ends the safetySystemDiagnosticSession.

REQ	8.58 APP – PCU-End – Terminate PCU pyrotechnic device scrapping via ACL
The PCU shall end the safetySystemDiagnosticSession after the reception of a deployment termination command via the ACL.	

8.12 APP – Terminate system deployment (Sys-End)

Figure 10 shows the termination of the system deployment (Sys-End).



Key

H PCU sequence has reached the state PCU-End

Figure 10 — Terminate system deployment (Sys-End)

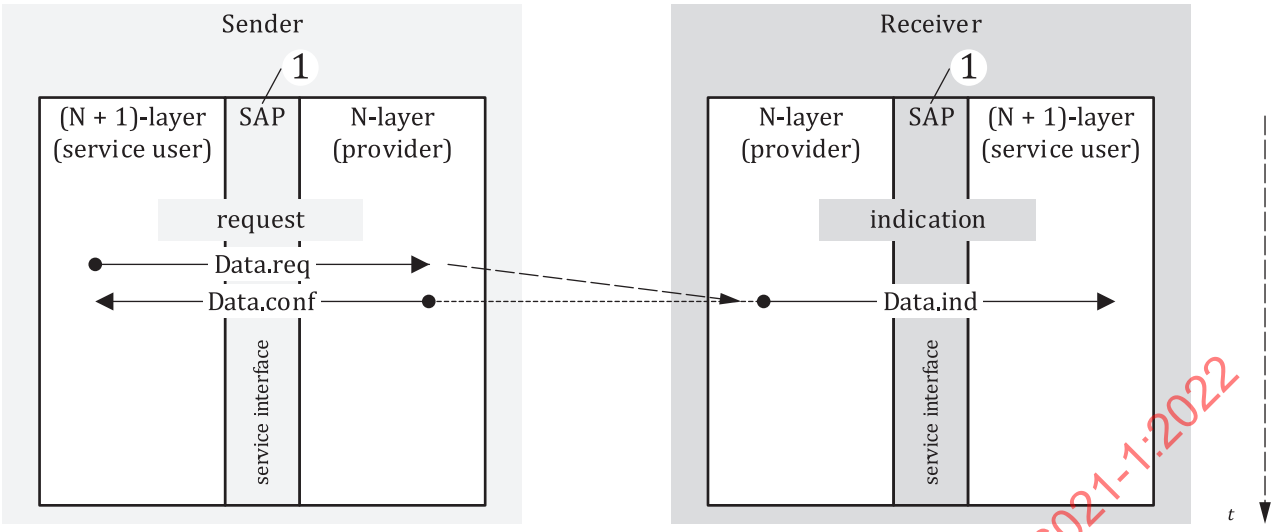
REQ	8.59 APP – Sys-End – Termination of deployment sequence
The vehicle shall terminate the sequence and end the process with documentation of the vehicle status.	

REQ	8.60 APP – Sys-End – Deployment of remaining pyrotechnic devices
The remaining pyrotechnic devices shall be deployed in a conventional way (by physical removal) or by using another method.	

9 Service interface (SI) definition between application and OSI layers

9.1 SI — A_Data.req, A_Data.ind, and A_Data.conf service interface (SI)

The service interface defines the service and parameter mapping to the application and the lower OSI layers. Figure 11 shows the A_Data.req, A_Data.ind, and A_Data.conf service interface.



Key

t time

1 service access point

Figure 11 — A_Data.req, A_Data.ind, and A_Data.conf service interface

9.2 SI — A_Data.req, A_Data.ind, and A_Data.conf service interface (SI) parameter mapping

This requirement specifies the application service interface (SI) parameter mapping between application and OSI layers.

REQ	0.1 SI – A_Data.req, A_Data.ind, and A_Data.conf service interface parameter mapping between application and OSI layers
Table 30 specifies the Data.req and Data.ind service interface parameter mapping between the application and OSI layers.	

Table 30 — A_Data.req, A_Data.ind, and A_Data.conf service interface parameter mapping between application and OSI layers

APP	AL	PL	SL	TL	NL	DLL	Description
Mtype	A_Mtype	A_Mtype	S_Mtype	T_Ptype	N_Ptype	L_Ftype	message, packet, frame type [DiagNormAddr, DiagNormFixAddr, DiagExtAddr, RDiagMixAddr]
TAtype	A_TAtype	A_TAtype	S_TAtype	T_TAtype	N_TAtype	TAtype	addressing type [functional, physical]
AE	A_AE	A_AE	S_AE	T_AE	N_AE	—[a]	address extension[b]
TA	A_TA	A_TA	S_TA	T_TA	N_TA	—[a]	target address
SA	A_SA	A_SA	S_SA	T_SA	N_SA	—[a]	source address
Length	A_Length	A_Length	S_Length	T_Length	N_Length	L_Length	length of protocol data unit
PDU	A_PDU	A_PDU	S_PDU	T_PDU	N_PDU	L_PDU	protocol data unit

[a] Not supported = "—".

[b] Includes address extension if N_Ptype = remote diagnostics (RDiagMixAddr).

Table 30 *(continued)*

APP	AL	PL	SL	TL	NL	DLL	Description
Result	A_Result	A_Result	S_Result	T_Result	N_Result	L_Result	result of OSI layer transaction

[a] Not supported = "—".

[b] Includes address extension if N_Ptype = remote diagnostics (RDiagMixAddr).

9.3 Service interface parameters (SIP)

9.3.1 SIP – General

The following subclauses specify the service interface parameters (SIP) and data types, which are used between application and OSI layer services.

9.3.2 SIP – Data type definitions

REQ	0.2 SIP – Data type definitions
The data types shall be in accordance to: — `Enum`: 8-bit enumeration, — `Unsigned Byte`: 8-bit unsigned numeric value, — `Unsigned Word`: 16-bit unsigned numeric value, — `Unsigned Long`: 32-bit unsigned numeric value, — `Byte Array`: sequence of 8-bit aligned data, — `Bit String`: 8-bit binary coded.	

9.3.3 SIP – Mtype, message type

REQ	0.3 SIP – Mtype, message type
The `Mtype` parameter shall be of data type `Enum` and shall be used to identify the message type and range of address information included in a service call. Range: [`Diag`]	

9.3.4 SIP – TAtype, target address type

REQ	0.4 SIP – TAtype, target address type
The `TAtype` parameter shall be of data type `Enum` and shall be used to identify the target address type to be used with the request address. Range: [`physical`]	

9.3.5 SIP – AE, address extension

REQ	0.5 SIP – AE, address extension
The `AE` parameter shall be of data type `Unsigned Word` and shall be used to extend the available address range for large networks and to encode both, sending and receiving network layer entities of sub-networks other than the local network where the communication takes place. `AE` is only part of the addressing information if Mtype is set to remote diagnostics (RDiagMixAddr). Range: [0000_{16} to $FFFF_{16}$]	

9.3.6 SIP – TA, target address

REQ	0.6 SIP – TA, target address
The TA parameter shall be of data type Unsigned Word and shall contain the target address of the node. Range: [0000_{16} to $FFFF_{16}$]	

9.3.7 SIP – SA, source address

REQ	0.7 SIP – SA, source address
The SA parameter shall be of data type Unsigned Word and shall contain the soure address of the node. Range: [0000_{16} to $FFFF_{16}$]	

9.3.8 SIP – Length, length of PDU

REQ	0.8 SIP – Length, length of PDU
The Length parameter shall be of data type Unsigned Long and shall contain the length of the PDU to be transmitted/received. Range: [$0000\ 0000_{16}$ to $FFFF\ FFFF_{16}$]	

9.3.9 SIP – PDU, protocol data unit

REQ	0.9 SIP – PDU, protocol data unit
The PDU parameter shall be of data type Byte Array and shall contain the message data (PDU) content of the request or response message to be transmitted/received. Range: [00_{16} to FF_{16}]	

9.3.10 SIP – Result, result

REQ	0.10 SIP – Result, result
The Result parameter shall be of data type Bit String and shall contain the status relating to the outcome of a service execution (request field and response field sequence). If two or more errors are discovered at the same time, then the application layer entity shall set the appropriate error bit in the result parameter. Range: [OK, Err_AL_Length, Err_TL_PCI_Type, Err_TL_PCI_SF_DL_Value, Err_NL_AddrFmt, Err_DLL_Byte] The result OK shall be issued to the service user when the service execution is successfully completed. The OK shall be issued to a service user on both the sender and receiver side. The ERR_... shall be issued to the service user when an error is detected by a lower layer (provider). The ERR_... shall be issued to the service user on both the sender and the receiver side.	

10 Application layer (AL)

10.1 AL – Applicable ISO 14229-1 UDS functionality

The requirements specified in this subclause are applicable to the ISO 14229-1 UDS functionality.

REQ	7.1 AL – Applicable ISO 14229-1 UDS functionality – Pyrotechnical device applicable diagnostic services
If implemented, the vehicle interface(s) to the PCU-related systems shall be in accordance to the services, sub-Functions and data ranges as they are applicable for an implementation according to ISO 14229-3 UDSonCAN or ISO 14229-5 UDSonIP as specified in Table 31.	

REQ	7.2 AL – Applicable ISO 14229-1 UDS functionality – No use of Authentication subFunction transmitCertificate
The ISO 14229-1 Authentication service, subFunction transmitCertificate shall not be used for the purpose of this document.	

Table 31 — ISO 14229-1 Unified diagnostic services applicable to pyrotechnic device

Diagnostic service name	SubFunction	default-Session	safetySystem-DiagnosticSession
Authentication[a]	All subFunctions except subFunction transmitCertificate	O	O
DiagnosticSessionControl (scrapping in progress)	defaultSession: 01_{16} safetySystemDiagnosticSession: 04_{16}	M	M
TesterPresent (scrapping in progress)	zeroSubFunction (00_{16})	M	M
SecurityAccess (enable scrapping)	requestSeed: $5F_{16}$ sendKey: 60_{16}	N/A	M
ReadDataByIdentifier (report VIN, report number of PCUs, report ACL version, ...)	N/A	M	M
WriteDataByIdentifier (write dismantler information)	N/A	N/A	M
RoutineControl (carry out scrapping)	startRoutine: 01_{16} stopRoutine: 02_{16} requestRoutineResults: 03_{16}	N/A	M[b]
EcuReset (end of scrapping)	hardReset: 01_{16}	N/A	M

[a] The Authentication service is specified in ISO 14229-1.

[b] Secured routines require a SecurityAccess service in a non-default diagnostic session. A routine that requires to be stopped actively by the PDT also requires a non-default session.

10.2 AL – PCU timing parameters

REQ	7.3 AL – PCU timing parameters
The vehicle's PCU-related systems shall be in accordance to the application layer timing parameters as specified in ISO 14229-2.	

10.3 AL – Authentication

10.3.1 AL – Requirements specification – PDT authentication

The implementation of the PDT authentication is optional.

The implementation requirements of the PDT authentication specify the message sequence and relevant requirements.

Applicable use case: UC 2.1 – Perform PDT authentication

REQ	7.4 AL – PDT authentication – General requirement #1
The authentication of the PDT against the PCU(s) shall be in accordance to ISO 14229-1:2020, "Authentication service" or a later edition. NOTE Legacy PDTs not supporting PDT authentication are incompatible with PCU(s) supporting authentication.	

REQ	7.5 AL – PDT authentication – General requirement #2
If the PDT sends valid pyrotechnic device credentials, the vehicle shall grant access rights to the PDT for at least[a] the safety-relevant content, which is used for end-of-life deployment activation of pyrotechnic devices including the reading of: — hardware deployment method, — number of PCUs, — PCU address information, — VIN, and — dismantler documentation, and the writing of the dismantling documentation to the PCU. The PCU(s) shall grant access to the activation of routine(s) as defined in ISO 26021-3.	
[a] More information beside safety-relevant content can be provided by the vehicle manufacturer for simplicity of PCU implementations.	

All relevant credentials required by the PDT are delivered offline. It is recommended to use the same algorithm as specified in ISO 14229-1 for diagnostic use cases.

REQ	7.6 AL – PDT authentication – Allowed security concepts for using service "Authentication"
Authentication shall be implemented using PKI Certificate Exchange (APCE) as specified in ISO 14229-1:2020.	

REQ	7.7 AL – PDT authentication – Publication of used algorithms
If an algorithm is used for authentication, its reference shall be published in the Object Identifier (OID) repository http://oid-info.com/ according to ISO/IEC 9834-1 under the node being used for PCU(s) and PDT authentication.	

10.3.2 AL – Requirements specification – Fixed-address PCU/PCU(s) authentication

The implementation of the fixed-address PCU authentication is optional. All other PCUs can implement the authentication. The implementation requirements of the fixed-address PCU/PCU(s) authentication specify the message sequence and relevant requirements.

Applicable use case: UC 2.2 – Perform fixed-address PCU/PCU(s) authentication

REQ	7.8 AL – PCU authentication – Vehicle PCU authentication
The authentication of the fixed-address PCU/PCU(s) against the PDT shall be in accordance to ISO 14229-1:2020 "Authentication service". NOTE Legacy PDTs not supporting fixed-address PCU/PCU(s) authentication are incompatible with PCU(s) supporting authentication.	

10.4 AL – ReadDataByIdentifier – Read PCU hardware deployment method

10.4.1 AL – Requirements specification – Read PCU hardware deployment method

The ReadDataByIdentifier service is used to read the hardware deployment method and version of the fixed-address PCU/server.

Applicable use case: UC 3.1 – Report PCU hardware deployment method

REQ	7.9 AL – ReadDataByIdentifier – General requirement
The vehicle's PCU shall report the hardware deployment method information upon a PDT physical request message using the ReadDataByIdentifier service with the DID = PcuDeploymentMethod.	

REQ	7.10 AL – ReadDataByIdentifier – Request and response message processing
After the successful reception and processing of the ReadDataByIdentifier request message the vehicle's PCU shall send a ReadDataByIdentifier positive response message as specified in Table 32.	

REQ	7.11 AL – ReadDataByIdentifier – MsgParam – PcuHardwareDeploymentMethod
The DID PcuHardwareDeploymentMethod parameter shall be used to request the PCU hardware deployment method information.	

REQ	7.12 AL – ReadDataByIdentifier – PosRspMsgParam – PcuHardwareDeploymentMethod version
The dataRecord parameter PcuHardwareDeploymentMethod version shall be used to report the PcuHardware-DeploymentMethod version information.	

REQ	7.13 AL – ReadDataByIdentifier – PosRspMsgParam – PcuIdentificationString
The dataRecord parameter pcuIdentification shall be used to report the PcuIdentificationString assigned by the vehicle manufacturer.	

10.4.2 AL – Message sequence requirements – Read PcuHardwareDeploymentMethod

The implementation requirements of a ReadDataByIdentifier specify the message sequence and relevant requirements (see Table 32).

Table 32 — ReadDataByIdentifier service – Read PcuHardwareDeploymentMethod

Content	A_PDU definition	REQ	Cvt
1 byte	**ReadDataByIdentifier request message SID**	7.10	M
2 byte	DID = PcuHardwareDeploymentMethod	7.11	M
1 byte	**ReadDataByIdentifier positive response message SID**	7.10	M
2 byte	DID = PcuHardwareDeploymentMethod	7.11	M
—	dataRecord = [		M
1 byte	PcuHardwareDeploymentMethodVersion	7.12	M
9 byte	PcuIdentificationString]	7.13	M

10.4.3 AL – Message sequence example – Read PcuHardwareDeploymentMethod

The objective of this example is to show the message communication between the PDT and the vehicle's PCU to report the hardware deployment method (see Tables 33 and 34).

Table 33 — Message sequence example – Read PcuHardwareDeploymentMethod with Version = ISO 26021-2 Edition 1

Content	A_PDU definition
22_{16}	**ReadDataByIdentifier request message SID**
$FA01_{16}$	DID = PcuHardwareDeploymentMethod
62_{16}	**ReadDataByIdentifier positive response message SID**
$FA01_{16}$	DID = PcuHardwareDeploymentMethod
—	dataRecord = [

Table 33 *(continued)*

Content	A_PDU definition
01_{16}	PcuHardwareDeploymentMethodVersion = ISO 26021-2 Edition 1
00_{16} 00_{16} 00_{16} 00_{16} 00_{16} 00_{16} 00_{16} 00_{16} 00_{16} 00_{16}	PcuIdentificationString = default]

Table 34 — Message sequence example – Read PcuHardwareDeploymentMethod with Version = ISO 26021-1 Edition 2

Content	A_PDU definition
22_{16}	**ReadDataByIdentifier request message SID**
$FA01_{16}$	DID = PcuHardwareDeploymentMethod
62_{16}	**ReadDataByIdentifier positive response message SID**
$FA01_{16}$	DID = PcuHardwareDeploymentMethod
—	dataRecord = [
02_{16}	PcuHardwareDeploymentMethodVersion = ISO 26021-1 Edition 2
00_{16} 00_{16} 00_{16} 00_{16} 00_{16} 00_{16} 00_{16} 00_{16} 00_{16} 00_{16}	PcuIdentificationString = default]

10.5 AL – ReadDataByIdentifier – Read NumberOfPcu in vehicle

10.5.1 AL – Requirements specification – Read NumberOfPcu in vehicle

The ReadDataByIdentifier service is used to read the NumberOfPcu installed in the vehicle of the fixed-address PCU/server.

Applicable use case: UC 3.2 – Report number of PCU(s)

REQ	7.14 AL – ReadDataByIdentifier – General requirement
The vehicle's PCU shall report the number of PCUs in vehicle information upon a PDT physical request message using the ReadDataByIdentifier service with the DID = NumberOfPcu.	

REQ	7.15 AL – ReadDataByIdentifier – Request and response message processing
After the successful reception and processing of the ReadDataByIdentifier request message the vehicle's PCU shall send a ReadDataByIdentifier positive response message as specified in Table 35.	

REQ	7.16 AL – ReadDataByIdentifier – MsgParam – NumberOfPcu
The DID NumberOfPcu parameter shall be used to report the number of PCU information.	

REQ	7.17 AL – ReadDataByIdentifier – PosRspMsgParam – NumberOfPcu
The dataRecord parameter numberOfPCUs shall be used to report the number of PCUs in the vehicle.	

10.5.2 AL – Message sequence requirements – Read number of PCUs in vehicle

The implementation requirements of a ReadDataByIdentifier specify the message sequence and relevant requirements (see Table 35).

Table 35 — ReadDataByIdentifier service – Read number of PCUs in vehicle

Content	A_PDU definition	REQ	Cvt
1 byte	**ReadDataByIdentifier request message SID**	7.15	M
2 byte	DID = NumberOfPcu	7.16	M
1 byte	**ReadDataByIdentifier positive response message SID**	7.15	M
2 byte	DID = NumberOfPcu	7.16	M
—	dataRecord = [		M
1 byte	NumberOfPcu]	7.17	M

10.5.3 AL – Message sequence example – Read NumberOfPcu in vehicle

The objective of this example is to show the message communication between the PDT and the vehicle's PCU to report the NumberOfPcu installed in the vehicle (see Table 36). In this example the NumberOfPcu = 2.

Table 36 — Message sequence example – Read NumberOfPcu in vehicle

Content	A_PDU definition
22_{16}	**ReadDataByIdentifier request message SID**
$FA00_{16}$	DID = NumberOfPcu
62_{16}	**ReadDataByIdentifier positive response message SID**
$FA00_{16}$	DID = NumberOfPcu
—	dataRecord = [
02_{16}	NumberOfPcu]

10.6 AL – ReadDataByIdentifier – Read PcuAddressInfo

10.6.1 AL – Requirements specification – Read PcuAddressInfo

The ReadDataByIdentifier service is used to read the PcuAddressInfo installed in the vehicle of the fixed-address PCU/server.

Applicable use case: UC 3.3 – Report address information of PCU(s)

REQ	7.18 AL – ReadDataByIdentifier – General requirement #1
The vehicle's fixed-address PCU shall report the address information of PCUs installed in the vehicle upon a PDT physical request message using the ReadDataByIdentifier service with the DID = PcuAddressInfo.	

REQ	7.19 AL – ReadDataByIdentifier – General requirement #2
The order of the PCU address information reported by the vehicle's fixed-address PCU shall correspond to the order the PCU shall be communicated with.	

REQ	7.20 AL – ReadDataByIdentifier – Request and response message processing
After the successful reception and processing of the ReadDataByIdentifier physically addressed (normal addressing format or normal fixed addressing format) request message the vehicle's fixed-address PCU/server shall send a ReadDataByIdentifier positive response message as specified in Table 37.	

REQ	7.21 AL – ReadDataByIdentifier – MsgParam – PcuAddressInfo
The DID PcuAddressInfo parameter shall be used to request the PcuAddressInfo.	

REQ	7.22 AL – ReadDataByIdentifier – PosRspMsgParam – PcuAddressFormatId
The dataRecord parameter PcuAddressFormatId shall be used to report the address format of a PCU.	

REQ	7.23 AL – ReadDataByIdentifier – PosRspMsgParam – PcuRequestMsgAddr
The dataRecord parameter PcuRequestMsgAddr shall be used to report the request message address of a PCU.	

REQ	7.24 AL – ReadDataByIdentifier – PosRspMsgParam – PcuResponseMsgAddr
The dataRecord parameter PcuResponseMsgAddr shall be used to report the response message address of a PCU.	

10.6.2 AL – Message sequence requirements – Read PcuAddressInfo of PCU

The implementation requirements of a ReadDataByIdentifier specify the message sequence and relevant requirements (see Table 37).

Table 37 — ReadDataByIdentifier service – Read PcuAddressInfo of PCU

Content	A_PDU definition	REQ	Cvt
1 byte	**ReadDataByIdentifier request message SID**	7.20	M
2 byte	DID = PcuAddressInfo	7.21	M
1 byte	**ReadDataByIdentifier positive response message SID**	7.20	M
2 byte	DID = PcuAddressInfo	7.21	M
—	dataRecord = [		M
m × 1 byte	PcuAddressFormatId (#1 to #N)	7.22	M
m × 4 byte	PcuRequestMsgAddr (#1 to #N)	7.23	M
m × 4 byte	PcuResponseMsgAddr (#1 to #N)]	7.24	M

10.6.3 AL – Message sequence example – Read PcuAddressInfo of DoCAN PCU

The objective of this example is to show the message communication between the PDT and the vehicle's PCU to report the address information of PCUs installed in the vehicle. The example shown in Table 38 reads address information of one PCU. The example shown in Table 39 reads address information of two DoCAN PCUs.

Table 38 — Message sequence example – Read PcuAddressInfo of one DoCAN PCU

Content	A_PDU definition
22_{16}	**ReadDataByIdentifier request message SID**
$FA02_{16}$	DID = PcuAddressInfo
62_{16}	**ReadDataByIdentifier positive response message SID**
$FA02_{16}$	DID = PcuAddressInfo
—	dataRecord = [
01_{16}	PcuAddressFormatId #1; DoCAN normal addressing
00_{16} 07_{16} $F2_{16}$ FF_{16}	PcuRequestMsgAddr #1
00_{16} 07_{16} FA_{16} FF_{16}	PcuResponseMsgAddr #1]

Table 39 — Message sequence example – Read PcuAddressInfo of two DoCAN PCUs

Content	A_PDU definition
22_{16}	**ReadDataByIdentifier request message SID**
$FA02_{16}$	DID = PcuAddressInfo
62_{16}	**ReadDataByIdentifier positive response message SID**
$FA02_{16}$	DID = PcuAddressInfo
—	dataRecord = [
01_{16}	PcuAddressFormatId #2; DoCAN normal addressing
$0007\ F2FF_{16}$	PcuRequestMsgAddr #1
$0007\ FAFF_{16}$	PcuResponseMsgAddr #1
02_{16}	PcuAddressFormatId #1; DoCAN extended addressing
$0006\ F177_{16}$	PcuRequestMsgAddr #2
$0006\ 77F1_{16}$	PcuResponseMsgAddr #2]

10.6.4 AL – Message sequence example – Read PcuAddressInfo of DoIP PCU

The objective of this example is to show the message communication between the PDT and the vehicle's PCU to report the address information of PCUs installed in the vehicle. The example shown in Table 40 reads address information of one PCU. The example shown in Table 41 reads address information of two DoIP PCUs.

Table 40 — Message sequence example – Read PcuAddressInfo of one DoIP PCU

Content	A_PDU definition
22_{16}	**ReadDataByIdentifier request message SID**
$FA02_{16}$	DID = PcuAddressInfo
62_{16}	**ReadDataByIdentifier positive response message SID**
$FA02_{16}$	DID = PcuAddressInfo
—	dataRecord = [
10_{16}	PcuAddressFormatId #1; DoIP addressing
$E002\ 0E02_{16}$	PcuRequestMsgAddr #1
$0E02\ E002_{16}$	PcuResponseMsgAddr #1]

Table 41 — Message sequence example – Read PcuAddressInfo of two DoIP PCUs

Content	A_PDU definition
22_{16}	**ReadDataByIdentifier request message SID**
$FA02_{16}$	DID = PcuAddressInfo
62_{16}	**ReadDataByIdentifier positive response message SID**
$FA02_{16}$	DID = PcuAddressInfo
—	dataRecord = [
10_{16}	PcuAddressFormatId #1; DoIP addressing
$E002\ 0E02_{16}$	PcuRequestMsgAddr #1
$0E02\ E002_{16}$	PcuResponseMsgAddr #1
10_{16}	PcuAddressFormatId #2; DoIP addressing
$E003\ 0E02_{16}$	PcuRequestMsgAddr #2
$0E02\ E003_{16}$	PcuResponseMsgAddr #2]

10.7 AL – ReadDataByIdentifier – Report VIN from PCU

10.7.1 AL – Requirements specification – Report VIN from PCU

The ReadDataByIdentifier service is used to report the VIN stored in the fixed-address PCU/server.

Applicable use case: UC 3.4 – Report vehicle identification number

REQ	7.25 AL – ReadDataByIdentifier – General requirement
At least one ECU shall report the vehicle identification number upon an PDT physical request message using the the ReadDataByIdentifier service with the DID = VIN.	

REQ	7.26 AL – ReadDataByIdentifier – Request and response message processing
After the successful reception and processing of the ReadDataByIdentifier physical request message the ECU shall send a ReadDataByIdentifier positive response message as specified in Table 42.	

REQ	7.27 AL – ReadDataByIdentifier – MsgParam – VIN
The DID VIN parameter shall be used to request the vehicle identification number. The DID value for VIN is specified in ISO 14229-1.	

REQ	7.28 AL – ReadDataByIdentifier – PosRspMsgParam – dataRecord
The dataRecord parameter vinData shall be used to report the vehicle identification number.	

REQ	7.29 AL – ReadDataByIdentifier – PosRspMsgParam – VinData
The vinData shall contain the VIN.	

10.7.2 AL – Message sequence requirements – Report VIN from PCU

Table 42 specifies the report VIN from PCU message sequence.

Table 42 — ReadDataByIdentifier service – Report VIN from PCU

Length	A_PDU definition	REQ	Cvt
1 byte	**ReadDataByIdentifier request message SID**	7.26	M
2 byte	DID = VIN	7.27	M
1 byte	**ReadDataByIdentifier positive response message SID**	7.26	M
2 byte	DID = VIN	7.27	M
—	dataRecord = [	7.28	M
17 byte	VinData]	7.29	M

10.7.3 AL – Message sequence example – Report VIN from PCU

Table 43 gives an example of the report VIN from PCU message sequence.

Table 43 — Message sequence example – Report VIN from PCU

A_PDU	A_PDU definition
22_{16}	**ReadDataByIdentifier request message SID**
$F190_{16}$	DID = VIN
62_{16}	**ReadDataByIdentifier positive response message SID**
$F190_{16}$	DID = VIN
—	dataRecord = [
57_{16} 30_{16} $4C_{16}$ 30_{16} 30_{16} 30_{16} 30_{16} 34_{16} 33_{16} $4D_{16}$ 42_{16} 35_{16} 34_{16} 31_{16} 33_{16} 32_{16} 36_{16}	W0L0; VinData 0004; VinData 3MB5; VinData 4132; VinData 6]; VinData

NOTE If the server fails to respond with either a positive or negative response, the client continues with the deployment sequence.

10.8 AL – ReadDataByIdentifier – Report dismantler information

10.8.1 AL – Requirements specification – Report dismantler information

The ReadDataByIdentifier service is used to report the dismantler information stored in the fixed-address PCU/server.

Applicable use case: UC 3.5 – Report dismantling documentation of PCU

REQ	7.30 AL – ReadDataByIdentifier – General requirement
At least one PCU shall report the dismantler information upon a PDT physical request message using the ReadDataByIdentifier service with the DID = DismantlerIdentification.	

REQ	7.31 AL – ReadDataByIdentifier – Request and response message processing
After the successful reception and processing of the ReadDataByIdentifier physical request message the PCU shall send a ReadDataByIdentifier positive response message as specified in Table 44.	

REQ	7.32 AL – ReadDataByIdentifier – MsgParam – DismantlerIdentification
The DID DismantlerIdentification parameter DismantlerIdentification shall be used to request the dismantler identification number.	

REQ	7.33 AL – ReadDataByIdentifier – PosRspMsgParam – dismantlerNumber
The dataRecord parameter dismantlerNumber shall contain the PDT dismantler number.	

REQ	7.34 AL – ReadDataByIdentifier – PosRspMsgParam – pdtDeviceIdentification
The dataRecord parameter pdtDeviceIdentification shall contain the identifier of the PDT.	

REQ	7.35 AL – ReadDataByIdentifier – PosRspMsgParam – deploymentDate
The dataRecord parameter deploymentDate shall contain the deployment year, month, and day of the PCU deployment.	

10.8.2 AL – Message sequence requirements – Report dismantler information

Table 44 specifies the report dismantler information message sequence.

Table 44 — ReadDataByIdentifier service – Report dismantler information

Length	A_PDU definition	REQ	Cvt
1 byte	**ReadDataByIdentifier request message SID**	7.31	M
2 byte	DID = DismantlerIdentification	7.32	M
1 byte	**ReadDataByIdentifier positive response message SID**	7.31	M
2 byte	DID = DismantlerIdentification	7.32	M
—	dataRecord = [		M
8 byte	DismantlerNumber	7.33	M
4 byte	PdtDeviceIdentification	7.34	M
4 byte	DeploymentDate]	7.35	M

10.8.3 AL – Message sequence example – Report dismantler information

Table 45 specifies an example of report dismantler information. The parameters provide information about the DismantlerNumber, PdtDeviceIdentification, and DeploymentDate. The deployment date is either equal to the original production value or equal to the actual deployment date.

Table 45 — Message sequence example – Report dismantler information

Content	A_PDU definition	
22_{16}	**ReadDataByIdentifier request message SID**	
$FA07_{16}$	DID = DismantlerIdentification	
62_{16}	**ReadDataByIdentifier positive response message SID**	
$FA07_{16}$	DID = DismantlerNumber	
—	dataRecord = [	
00_{16} 00_{16} 00_{16} 00_{16}	DismantlerNumber #1 to #4	
00_{16} 00_{16} 00_{16} 00_{16}	DismantlerNumber #5 to #8	
00_{16} 00_{16} 00_{16} 00_{16}	PdtDeviceIdentification #1 to #4	
07_{16} $D6_{16}$	DeploymentDate: Year (MSB, LSB);	2006
05_{16}	DeploymentDate: Month;	May
01_{16}	DeploymentDate: Day];	01

10.9 AL – ReadDataByIdentifier – Read deployment loop identification table

10.9.1 AL – Requirements specification – Read deployment loop identification table

The purpose of this service is to obtain the number of deployment loops installed in a PCU. One PCU can be responsible for different deployment loop numbers.

The ReadDataByIdentifier service is used to read the hardware deployment method and version from the fixed-address PCU/server.

Applicable use case: UC 4.1 – Report PCU deployment loop identification table

REQ	7.36 AL – ReadDataByIdentifier – General requirement
The vehicle's PCU shall report the deployment loop identification table information upon a PDT request message using the ReadDataByIdentifier service with the DID = DeploymentLoopIdTable.	

REQ	7.37 AL – ReadDataByIdentifier – Request and response message processing
After the successful reception and processing of the ReadDataByIdentifier request message, the vehicle's PCU shall send a ReadDataByIdentifier positive response message as specified in Table 46.	

REQ	7.38 AL – ReadDataByIdentifier – MsgParam – DeploymentLoopIdTable
The DID DeploymentLoopIdTable parameter shall be used to report the number of connected pyrotechnic devices to the PCU, the communication type implemented, the communication version, and the status information.	

The AclType identifies the ACL hardware interface type.

REQ	7.39 AL – ReadDataByIdentifier – PosRspMsgParam – AclType
The dataRecord parameter AclType shall be used to report the additional communication line type definition information.	

The AclMethodVersion identifies the version of the ACL diagnostic protocol services and sequence used for PCU deployment.

REQ	7.40 AL – ReadDataByIdentifier – PosRspMsgParam – AclMethodVersion
The dataRecord parameter AclMethodVersion shall be used to report the additional communication line method information. It shall be incremented every time an ACL protocol service or a data identifier changes in the ISO 26021 series and is no longer backward-compatible.	

REQ	7.41 AL – ReadDataByIdentifier – PosRspMsgParam – NumOfLoopTableRecords
The dataRecord parameter NumOfLoopTableRecords shall be used to report the number of connected pyrotechnic devices to the PCU.	

The DeploymentLoopStatus of a DeploymentLoopID reports, e.g. operating, deactivation, not installed, deployed, failure status.

REQ	7.42 AL – ReadDataByIdentifier – PosRspMsgParam – DeploymentLoopId, DeploymentLoopStatus
The dataRecord parameters DeploymentLoopId and DeploymentLoopStatus shall be used to report the deployment loop identification and deployment loop status information of a pyrotechnic device connected to a PCU.	

10.9.2 AL – Message sequence requirements – Read deployment loop identification table

Table 46 specifies the message sequence requirements – Read deployment loop identification table.

Table 46 — Message sequence requirements – Read deployment loop identification table

Length	A_PDU definition	REQ	Cvt
1 byte	**ReadDataByIdentifier request message SID**	7.37	M
2 byte	DID = DeploymentLoopIdTable	7.38	M
1 byte	**ReadDataByIdentifier positive response message SID**	7.37	M
2 byte	DID = DeploymentLoopIdTable	7.38	M

Table 46 *(continued)*

Length	A_PDU definition	REQ	Cvt
—	dataRecord = [		M
1 byte	AclTypeDefinition	7.39	M
1 byte	AclMethodVersion	7.40	M
1 byte	NumOfLoopTableRecords	7.41	M
k × 2 byte	(#1 to #k) × (DeploymentLoopId, DeploymentLoopStatus)]	7.42	M

10.9.3 AL – Message sequence example – Read deployment loop identification table

Table 47 specifies the message sequence example – Read deployment loop identification table with ACL = CommMode12V and one connected pyrotechnic device.

Table 47 — Message sequence example – Read deployment loop identification table

Content	A_PDU definition	
22_{16}	**ReadDataByIdentifier request message SID**	
$FA06_{16}$	DID = DeploymentLoopIdTable	
62_{16}	**ReadDataByIdentifier positive response message SID**	
$FA06_{16}$	DID = DeploymentLoopIdTable	
02_{16}	AclTypeDefinition = CommMode12V	
01_{16}	AclMethodVersion = 1	
01_{16}	NumOfLoopTableRecords = 1	
—	dataRecord = [	
05_{16}	DeploymentLoopId #1;	airbag left side frontal 2nd stage; see ISO 26021-3
20_{16}	DeploymentLoopStatus #1];	deployed, see ISO 26021-3

Table 48 specifies the message sequence example – Read deployment loop identification table with ACL = CAN only and three connected pyrotechnic devices.

Table 48 — Message sequence example – Read deployment loop identification table

Content	A_PDU definition	
22_{16}	**ReadDataByIdentifier request message SID**	
$FA06_{16}$	DID = DeploymentLoopIdTable	
62_{16}	**ReadDataByIdentifier positive response message SID**	
$FA06_{16}$	DID = DeploymentLoopIdTable	
01_{16}	AclTypeDefinition = No_ACL_Line	
01_{16}	AclMethodVersion = 1	
03_{16}	NumOfLoopTableRecords = 3	
—	dataRecord = [	
$A8_{16}$	DeploymentLoopId #1;	sidebag inside – driver side – seat mounted; see ISO 26021-3
20_{16}	DeploymentLoopStatus #1;	deployed, see ISO 26021-3
$A9_{16}$	DeploymentLoopId #2;	sidebag inside – passenger side – seat mounted; see ISO 26021-3
00_{16}	DeploymentLoopStatus #2;	not deployed; see ISO 26021-3
AC_{16}	DeploymentLoopId #3;	farside (centre) airbag – 1st row; see ISO 26021-3
20_{16}	DeploymentLoopStatus #3];	deployed, see ISO 26021-3

NOTE The order of the loop identifier in the table determines the order in which the PDT deploys the pyrotechnic devices.

10.10 AL – DiagnosticSessionControl – safetySystemDiagnosticSession

10.10.1 AL – Requirements specification – safetySystemDiagnosticSession

The safetySystemDiagnosticSession is initiated in the PCU/server with the DiagnosticSessionControl service. This session is a timed session which requires a diagnostic request service to be sent by the PDT for the session to stay active.

Applicable use case: UC 4.2 – Initiate safetySystemDiagnosticSession

REQ	7.43 AL – safetySystemDiagnosticSession – General requirement
The vehicle's PCU shall transit into the SafetySystemDiagnosticSession upon a PDT request message using the DiagnosticSessionControl service with the DiagnosticSessionType = SafetySystemDiagnosticSession.	

REQ	7.44 AL – safetySystemDiagnosticSession – Request and response message processing
After the successful reception and processing of the DiagnosticSessionControl request message the vehicle's PCU shall send a DiagnosticSessionControl positive response message as specified in Table 49.	

REQ	7.45 AL – safetySystemDiagnosticSession – MsgParam – DiagnosticSessionType
The DiagnosticSessionType parameter shall be used to request and confirm the SafetySystemDiagnosticSession.	

REQ	7.46 AL – safetySystemDiagnosticSession – PosRspMsgParam – data #1 to data #4
The sessionParameterRecord parameters data #1 to data #4 shall be used to report the session specific parameter values.	

10.10.2 AL – Message sequence requirements – safetySystemDiagnosticSession

Table 49 specifies the DiagnosticSessionControl – safetySystemDiagnosticSession service.

Table 49 – DiagnosticSessionControl service – safetySystemDiagnosticSession

Content	A_PDU definition	REQ	Cvt
1 byte	**DiagnosticSessionControl request message SID**	7.44	M
1 byte	DiagnosticSessionType = SafetySystemDiagnosticSession	7.45	M
1 byte	**DiagnosticSessionControl positive response message SID**	7.44	M
1 byte	DiagnosticSessionType = SafetySystemDiagnosticSession	7.45	M
—	sessionParameterRecord = [		
1 byte	data#1	7.46	M
—	:	7.46	M
1 byte	data#4]	7.46	M

10.10.3 AL – Message sequence example – safetySystemDiagnosticSession

The objective of this example is to show the message communication between the PDT and the vehicle's PCU to transit the PCU into the SafetySystemDiagnosticSession.

Table 50 provides an example of the DiagnosticSessionControl service message sequence.

Table 50 — Example of the DiagnosticSessionControl service message sequence

Content	A_PDU definition
10_{16}	**DiagnosticSessionControl request message SID**
04_{16}	SF = DiagnosticSessionType: SafetySystemDiagnosticSession, SuppressPosRspMsgIndicationBit = FALSE
50_{16}	**DiagnosticSessionControl positive response message SID**
04_{16}	SF = DiagnosticSessionType: SuppressPosRspMsgIndicationBit = FALSE SafetySystemDiagnosticSession,
—	sessionParameterRecord = [
00_{16}	data #1; $t_{P2\_Server\_Max}$ (MSB): 50 ms
32_{16}	data #2; $t_{P2\_Server\_Max}$ (LSB): 50 ms
01_{16}	data #3; $t_{P2*\_Server\_Max}$ (MSB): 5 000 ms
$F4_{16}$	data #4; $t_{P2*\_Server\_Max}$ (LSB): 5 000 ms]

10.11 AL – TesterPresent

10.11.1 AL – Requirements specification – TesterPresent

This service is used to indicate to all PCUs that a PDT is still connected to the vehicle and that certain diagnostic services and/or communications, that have been previously activated, are to remain active.

This service is used to keep the selected PCUs in the safetySystemDiagnosticSession. This is either done by periodic TesterPresent request messages in case of absence of other diagnostic services or by other diagnostic services to prevent the PCU/server from automatically returning to the defaultSession.

Applicable use case: UC 4.3 – Keep-alive safetySystemDiagnosticSession

REQ	7.47 AL – TesterPresent – General requirement
The vehicle's PCU shall keep the current diagnostic session active upon a PDT request message using the TesterPresent service.	

REQ	7.48 AL – TesterPresent – Request and response message processing
After the successful reception and processing of the TesterPresent request message the vehicle's PCU shall send a TesterPresent positive response message as specified in Table 51.	

REQ	7.49 AL – TesterPresent – MsgParam – SuppressPosRspMsgIndicationBit
The suppressPosRspMsgIndicationBit parameter shall be used to control the transmission of a PCU's response message.	

10.11.2 AL – Message sequence requirements – TesterPresent

Table 51 specifies the TesterPresent – zeroSubFunction service.

Table 51 — TesterPresent service – suppressPosRspMsgIndicationBit

Content	A_PDU definition	REQ	Cvt
1 byte	**TesterPresent request message SID**	7.48	M
1 byte	SF = [suppressPosRspMsgIndicationBit, zeroSubFunction]	7.49	M
NOTE The TesterPresent positive response message is only sent if the suppressPosRspMsgIndicationBit = FALSE.			

Table 51 *(continued)*

Content	A_PDU definition	REQ	Cvt
1 byte	**TesterPresent positive response message SID**	7.48	M
1 byte	SF = [suppressPosRspMsgIndicationBit, zeroSubFunction]	7.49	M
NOTE The TesterPresent positive response message is only sent if the suppressPosRspMsgIndicationBit = FALSE.			

10.11.3 AL – Message sequence example – TesterPresent

The objective of this example is to show the message communication between the PDT and the vehicle's PCU to keep alive the current diagnostic session in the PCU.

Table 52 provides an example of the TesterPresent service message sequence with positive response.

Table 52 — Example of the TesterPresent service message sequence with positive response

Content	A_PDU definition
$3E_{16}$	**TesterPresent request message SID**
00_{16}	SubFunction = [suppressPosRspMsgIndicationBit = FALSE, zeroSubFunction]
$7E_{16}$	**TesterPresent positive response message SID**
00_{16}	SubFunction = [suppressPosRspMsgIndicationBit = FALSE, zeroSubFunction]

10.12 AL – SecurityAccess

10.12.1 AL – Requirements specification – SecurityAccess

The purpose of this service is to provide a means to access diagnostic services which have restricted access. This method is part of a set of provisions designed to fulfil security requirements.

The use of diagnostic services for scrapping pyrotechnic devices is a situation where security access is required. To avoid the improper use of the scrapping mode without correct PDT implementation, the security concept uses a seed and key relationship while a safetySystemDiagnosticSession is active. It is required that the PDT first initiates the safetySystemDiagnosticSession. This session is a timed session which requires a diagnostic request service to be sent by the PDT for the session to stay active. After successful initiation of the safetySystemDiagnosticSession, the PDT sends a securityAccess request service with the SecurityAccessType set to RequestDeploymentSeed. If it supports the SecurityAccessType value, the PCU shall respond with a securityAccess positive response message. The positive response message shall also include the securitySeed value. The PDT then sends a SecurityAccess request message with the SecurityAccessType set to the SendDeploymentKey value and the securityKey value required to successfully unlock the PCU. If the securityKey value received by the PCU is incorrect, this shall be regarded as an erroneous attempt and shall be answered with a negative response message including the appropriate negative response code. The negative response codes shall be applied as specified in ISO 14229-1. There is no time period which needs to be inserted between access attempts.

The implementation requirements of SecurityAccess specify the message sequence and relevant requirements.

Applicable use case: UC 4.4 – Unlock security of PCU

REQ	7.50 AL – SecurityAccess – General requirement
The vehicle's PCU shall report the SecurityAccess information upon a PDT request message using the SecurityAccess service with the SecurityAccessType set to RequestDeploymentSeed.	

REQ	7.51 AL – SecurityAccess – Request and response message processing
After the successful reception and processing of the SecurityAccess request message the vehicle's PCU shall send a SecurityAccess positive response message as specified in Table 53.	

REQ	7.52 AL – SecurityAccess – MsgParam – SecurityAccessType RequestDeploymentSeed
The subFunction parameter SecurityAccessType RequestDeploymentSeed value $5F_{16}$ shall be used in the SecurityAccess request message to request the deployment seed value from the PCU #1 to PCU #N.	

REQ	7.53 AL – SecurityAccess – MsgParam – DeploymentSeed value
The DeploymentSeed value shall consist of a 2-byte value. The MSB value 01_{16} shall specify the PCU deployment method version. The LSB value shall be any value between 00_{16} and FF_{16} (e.g. random value or part of serial number or fixed value) for all PCUs.	

REQ	7.54 AL – SecurityAccess – MsgParam – SecurityAccessType SendDeploymentKey
The subFunction parameter SecurityAccessType SendDeploymentKey value 60_{16} shall be used in the SecurityAccess request message to send the deployment key value to the PCU #1 to PCU #N.	

REQ	7.55 AL – SecurityAccess – MsgParam – DeploymentKey value
The DeploymentKey value shall consist of a 2-byte value and shall be the 1's complement of the DeploymentSeed value.	

REQ	7.56 AL – RoutineControl – PosRspMsgParam – RequestDeploymentSeed (#1 to #s)
The data parameter value DeploymentSeed (#1 to #s) shall be reported to the PDT in order to enable the PDT to identify the DeploymentKey value.	

REQ	7.57 AL – RoutineControl – ReqMsgParam – DeploymentKey (#1 to #k)
The value of the data parameter DeploymentKey (#1 to #k) shall be used to unlock the PCU #1 to PCU #N if the value matches the expected value.	

10.12.2 AL – Message sequence requirements – SecurityAccess

Table 53 specifies the SecurityAccess service with SecurityAccessType = RequestDeploymentSeed.

Table 53 — SecurityAccess service – SecurityAccessType = RequestDeploymentSeed

Content	A_PDU definition	REQ	Cvt
1 byte	**SecurityAccess request message SID**	7.51	M
1 byte	SF = [SuppressPosRspMsgIndicationBit, SecurityAccessType = RequestDeploymentSeed]	7.52	M
1 byte	**SecurityAccess positive response message SID**	7.51	M
1 byte	SF = [SuppressPosRspMsgIndicationBit, SecurityAccessType = RequestDeploymentSeed]	7.52	M

Table 53 *(continued)*

Content	A_PDU definition	REQ	Cvt
—	dataRecord = [		M
s × 1 byte	DeploymentSeed (#1 to #s)]	7.56	M

Table 54 specifies the SecurityAccess service with SecurityAccessType = SendKey.

Table 54 — SecurityAccess service – SecurityAccessType = SendKey

Content	A_PDU definition	REQ	Cvt
1 byte	**SecurityAccess request message SID**	7.51	M
1 byte	SF = [SuppressPosRspMsgIndicationBit, SecurityAccessType = SendDeploymentKey]	7.52	M
—	dataRecord = [		M
k × 1 byte	DeploymentKey (#1 to #k)]	7.57	M
1 byte	**SecurityAccess positive response message SID**	7.51	M
1 byte	SF = [SuppressPosRspMsgIndicationBit, SecurityAccessType = SendDeploymentKey]	7.52	M

10.12.3 AL – Message sequence example – SecurityAccessType = RequestSeed

The objective of this example is to request the DeploymentSeed value of the PCU.

Table 55 provides an example of the SecurityAccess service message sequence with SecurityAccessType = RequestDeploymentSeed.

Table 55 — Example of the SecurityAccess service message sequence – SecurityAccessType = RequestSeed

Content	A_PDU definition
27_{16}	**SecurityAccess request message SID**
$5F_{16}$	SF = SecurityAccessType = [SuppressPosRspMsgIndicationBit = FALSE, RequestDeploymentSeed]
67_{16}	**SecurityAccess positive response message SID**
$5F_{16}$	SF = SecurityAccessType = [SuppressPosRspMsgIndicationBit = FALSE, RequestDeploymentSeed]
—	dataRecord = [
01_{16}, (00_{16} to FF_{16})	DeploymentSeed (MSB, LSB)]

10.12.4 AL – Message sequence example – SecurityAccessType = SendDeploymentKey

The objective of this example is to send the DeploymentKey value from the PDT to the PCU to unlock the PCU.

Table 56 provides an example of the SecurityAccess service message sequence with SecurityAccessType = SendDeploymentKey.

Table 56 — Example of the SecurityAccess service message sequence – SecurityAccessType = SendKey

Content	A_PDU definition
27_{16}	**SecurityAccess request message SID**
60_{16}	SF = SecurityAccessType = [SuppressPosRspMsgIndicationBit = FALSE, SendDeploymentKey]
—	dataRecord = [

Table 56 *(continued)*

Content	A_PDU definition
FE_{16}, (00_{16} to FF_{16})	DeploymentKey (MSB, LSB)]
67_{16}	**SecurityAccess positive response message SID**
60_{16}	SF = SecurityAccessType = [SuppressPosRspMsgIndicationBit = FALSE, SendDeploymentKey]

10.13 AL – WriteDataByIdentifier – Write dismantler information

10.13.1 AL – Requirements specification – Write dismantler identification information

The WriteDataByIdentifier service allows the PDT/client to write information into the PCU #1 to PCU #N at an internal location specified by the data identifier provided.

The request message contains the dismantler identification information dataRecord. The dataRecord is identified by a dataIdentifier (see ISO 26021-3 DismantlerIdentification).

10.13.2 AL – Message sequence requirements – Write dismantler identification information

Applicable use case: UC 5.2 – Write dismantling documentation into PCU (Device-Deploy)

REQ	7.58 AL – WriteDataByIdentifier – General requirement
The vehicle's PCU #1 to PCU #N shall write the dismantler identification information upon a PDT physical addressed request message using the WriteDataByIdentifier service with the DID = DismantlerIdentification.	

REQ	7.59 AL – WriteDataByIdentifier – Request and response message processing
After the successful reception and processing of the WriteDataByIdentifier physical request message the PCU shall send a WriteDataByIdentifier positive response message as specified in Table 57.	

REQ	7.60 AL – WriteDataByIdentifier – MsgParam – DismantlerIdentification
The DID DismantlerIdentification parameter shall be used to write the PDT dismantler identification number.	

REQ	7.61 AL – WriteDataByIdentifier – ReqMsgParam – DismantlerIdentification
The dataRecord parameter DismantlerIdentification shall contain the PDT dismantler number.	

REQ	7.62 AL – WriteDataByIdentifier – ReqMsgParam – PdtDeviceIdentification
The dataRecord parameter PdtDeviceIdentification shall contain the identifier of the PDT.	

REQ	7.63 AL – WriteDataByIdentifier – ReqMsgParam – DeploymentDate
The dataRecord parameter DeploymentDate shall contain the deployment year, month, and day of the PCU deployment.	

Table 57 specifies the write dismantler identification information message sequence.

Table 57 — Message sequence requirements – Write dismantler identification information

Length	A_PDU definition	REQ	Cvt
1 byte	**WriteDataByIdentifier request message SID**	7.59	M
2 byte	DID = DismantlerIdentification	7.60	M
—	dataRecord = [		M
8 byte	DismantlerNumber	7.61	M
4 byte	PdtDeviceIdentification	7.62	M
4 byte	DeploymentDate]	7.63	M
1 byte	**WriteDataByIdentifier positive response message SID**	7.59	M
2 byte	DID = DismantlerIdentification	7.60	M

10.13.3AL – Message sequence example – Write dismantler identification information

This example includes parameter values of the dismantlerNumber, pdtDeviceIdentification, and deploymentDate. The deployment date is either equal to the original production value or equal to the actual deployment date. Table 58 specifies the message sequence example – Write dismantler information.

Table 58 — Message sequence example – Write dismantler identification information

Content	A_PDU definition	
$2E_{16}$	**WriteDataByIdentifier request message SID**	
$FA07_{16}$	DID = DismantlerIdentification	
—	dataRecord = [	
00_{16} 00_{16} 00_{16} 00_{16}	DismantlerNumber #1 to #4	
00_{16} 00_{16} 00_{16} 00_{16}	DismantlerNumber #5 to #8	
00_{16} 00_{16} 00_{16} 00_{16}	PdtDeviceIdentification #1 to #4	
07_{16} $D6_{16}$	DeploymentDate: Year (MSB, LSB);	2006
05_{16}	DeploymentDate: Month;	May
01_{16}	DeploymentDate: Day;	1
$6E_{16}$	**WriteDataByIdentifier positive response message SID**	
$FA07_{16}$	DID = DismantlerIdentification	

10.14 AL – RoutineControl

10.14.1AL – Requirements specification – RoutineControl

The RoutineControl service is used in the PCU to start the SPL or the SPM.

The PCU allows execution of the ExecuteSPL and DeployLoopRoutineId while unlocked by the SecurityAccess services. The implementation requirements of ExecuteSPL and DeployLoopRoutineId specify the message sequence and relevant requirements.

The PDT transmits a RoutineControl service request message to the PCU. The PCU accepts the request and starts the SPL. The PCU does not send a positive response until the routine is stopped and the scrapping program module (SPM) is in an executable form.

In testing use cases, the PDT requests routine results. The PDT transmits a RoutineControl service with the SubFunction parameter set to requestRoutineResult request message to the PCU. The PCU accepts the request and responds with the routineControlOption SPL number. The result is either 00_{16} (SPL without conversion) or 01_{16} (SPL with conversion).

Applicable use case: UC 4.5 – Execute PCU(s) scrapping program module loader

REQ	7.64 AL – RoutineControl – General requirement #1
The PCU shall activate the routine upon a PDT physical request message using the RID.	

NOTE The description of necessary preconditions for the activation is not part of this document. If applicable, they are documented by the vehicle manufacturer.

REQ	7.65 AL – RoutineControl - General requirement #2
Routines, which are available in the vehicle and are assigned to a standardized identifier (RID) shall be mandatory for applicable use cases and interface requirements.	

REQ	7.66 AL – RoutineControl - Vehicle status following routine activation
Routines, which are assigned to a standardized identifier (RID) shall return the control to the PCU after activation.	

REQ	7.67 AL – RoutineControl - Request and response message processing
After the successful reception and processing of the physical RoutineControl request message the PCU shall send a RoutineControl positive response message as specified in Table 59.	

REQ	7.68 AL – RoutineControl – MsgParam – routineControlType
This subFunction parameter routineControlType (startRoutine, stopRoutine, requestRoutineResults) shall be used to select the control of the routine as specified by the RID.	

REQ	7.69 AL – RoutineControl – MsgParam – RID
The parameter RID shall determine the routine as specified in ISO 26021-3, which shall be started/stopped.	

REQ	7.70 AL – RoutineControl – ReqMsgParam – routineControlOptionRecord[]
The parameter routineControlOptionRecord[] shall be implemented according to the specification in ISO 14229-1.	

REQ	7.71 AL – RoutineControl – ReqMsgParam – routineControlOption
The parameter routineControlOption shall be implemented according to the specification in ISO 26021-3. Supported routineControlOption values: — routineControlOption #1: 00_{16} — ExecuteSPL in RAM without conversion (testing use cases only); — routineControlOption #1: 01_{16} — ExecuteSPL in RAM with conversion (deployment process).	

REQ	7.72 AL – RoutineControl – PosRspMsgParam – routineInfo
The parameter routineInfo shall be implemented according to the specification in ISO 14229-1.	

REQ	7.73 AL – RoutineControl – PosRspMsgParam – routineStatusRecord[]
The parameter routineStatusRecord[] shall be implemented according to the specification in ISO 14229-1.	

REQ	7.74 AL – RoutineControl – PosRspMsgParam – routineStatus
The parameter routineStatus shall be implemented according to the specification in ISO 26021-3.	

10.14.2 AL – Message sequence requirements – RoutineControl

Table 59 specifies the RoutineControl message sequence.

Table 59 — RoutineControl message sequence

Content	A_PDU definition	REQ	Cvt
1 byte	**RoutineControl request message SID (PCU #1 to #N)**		M
1 byte	SF = [suppressPosRspMsgIndicationBit, routineControlType]		M
2 byte	RID = RoutineIdentifier		M
—	routineControlOptionRecord = [		M
k byte	routineControlOption (byte #1 to #k)]		C
1 byte	**RoutineControl positive response message SID (PCU #1 to #N)**		M
1 byte	SF = [SuppressPosRspMsgIndicationBit, routineControlType]		M
2 byte	RID = RoutineIdentifier		M
—	routineStatusRecord = [		C
1 byte	routineStatus #1 = routineInfo: VM-specific		C
m × 1 byte	routineStatus #m]		C

10.14.3 AL – Message sequence example – ExecuteSPL with SF = startRoutine

Upon reception of a RoutineControl request message with the RID = ExecuteSPL:

— the PCU copies the SPM into free RAM space. This RAM space is initialised after reset;

— the PCU converts the SPM into an executable format, e.g. the SPL overwrites some values of port-IO or function addresses.

If the belt buckle contact is intact, the PCU responds with a negative response message including NRC = conditionsNotCorrect information upon a RoutineControl request message with the RID = ExecuteSPL.

Table 60 specifies the message sequence example – ExecuteSPL with SF = startRoutine and routineControlOption = SPLConversionId:

— SPLConversionId used during testing process: 00_{16}: ExecuteSPL into RAM without converting;

— SPLConversionId used in deployment process: 01_{16}: ExecuteSPL into RAM with conversion.

Table 60 — Message sequence example – ExecuteSPL with SF = startRoutine

Content	A_PDU definition
31_{16}	**RoutineControl request message SID**
01_{16}	SF = [suppressPosRspMsgIndicationBit, startRoutine]
$E200_{16}$	RID = ExecuteSPL
—	routineControlOptionRecord = [
01_{16}	routineControlOption = SPLConversionId]; load SPM to RAM with conversion
71_{16}	**RoutineControl positive response message SID**
01_{16}	SF = [suppressPosRspMsgIndicationBit, startRoutine]
$E200_{16}$	RID = ExecuteSPL
—	routineStatusRecord = [

Table 60 *(continued)*

Content	A_PDU definition
00_{16}	routineStatus #1 = routineInfo: VM-specific
01_{16}	routineStatus #2 = SPLConversionId]; load SPM to RAM with conversion
$7F_{16}$	**RoutineControl negative response message SID**
31_{16}	RoutineControl request message SID
22_{16}	responseCode; conditionsNotCorrect

10.14.4AL – Message sequence example – ExecuteSPL with SF = requestRoutineResult

The RoutineControl service is used by the PDT to address the PCU in order to start the SPL or the SPM.

The PCU allows execution of the ExecuteSPL and DeployLoopRoutineId routines while unlocked by the SecurityAccess service. The implementation requirements of ExecuteSPL and DeployLoopRoutineId routines specify the message sequence and relevant requirements.

Upon reception of a RoutineControl request message with the RID = DeployLoopRoutineId and the parameter DeploymentLoopId (for each pyrotechnic device connected to this PCU):

— the SPM fires the pyrotechnic device;

— the PCU sends a response with the result;

— the SPM updates the DeploymentLoopId status.

When the PCU receives an EcuReset request message the RAM is cleared and the executable SPM is deleted.

Table 61 specifies the message sequence example – ExecuteSPL with SF = request routine results.

Table 61 — Message sequence example – ExecuteSPL with SF = requestRoutineResult

Content	A_PDU definition
31_{16}	**RoutineControl request message SID**
03_{16}	SF = [SuppressPosRspMsgIndicationBit, requestRoutineResult]
$E200_{16}$	RID = ExecuteSPL
71_{16}	**RoutineControl positive response message SID**
03_{16}	SF = [SuppressPosRspMsgIndicationBit, requestRoutineResult]
$E200_{16}$	RID = ExecuteSPL
—	routineStatusRecord = [
00_{16}	routineStatus #1 = routineInfo: VM-specific
00_{16}	routineStatus #2 = # of ExecuteSPL]

10.14.5AL – Message sequence example – DeployLoopRoutineID with SF = startRoutine

The PCU allows execution of the DeployLoopRoutineID routines while unlocked by the SecurityAccess service. The implementation requirements of DeployLoopRoutineId specify the message sequence and relevant requirements.

The DeployLoopRoutineId is selected by the routineControlOption. The DeploymentLoopId parameter is specified in ISO 26021-3.

Table 62 specifies the message sequence example – DeployLoopRoutineID with SF = startRoutine.

Table 62 — Message sequence example – DeployLoopRoutineID with SF = startRoutine

Content	A_PDU definition
31_{16}	**RoutineControl request message SID**
01_{16}	SF = [SuppressPosRspMsgIndicationBit, startRoutine]
$E201_{16}$	RID = DeployLoopRoutineId
—	routineControlOptionRecord = [
53_{16}	routineControlOption = DeploymentLoopId]; kneebag — driver side (see ISO 26021–3:—, Table B.1[2)] Second edition under preparation. Stage at the time of publication: ISO/DIS 26021-3:2021.)
71_{16}	**RoutineControl positive response message SID**
01_{16}	SF = [SuppressPosRspMsgIndicationBit, startRoutine]
$E201_{16}$	RID = DeployLoopRoutineId
—	routineStatusRecord = [
00_{16}	routineStatus #1 = routineInfo: VM-specific
53_{16}	routineStatus #2 = DeploymentLoopId
20_{16}	routineStatus #3 = DeploymentLoopId status]; “Deployed” (see ISO 26021–3:—, Table B.2)

10.14.6AL – Message sequence example – DeployLoopRoutineID with SF = requestRoutineResult

The PDT transmits a RoutineControl service request message to the PCU. The PCU accepts the request message and starts the routine. For reliability reasons, the scrapping program module (SPM) is executed out of, e.g. RAM memory (see 8.9.6). The PCU does not send a positive response until the routine is stopped and the loop status is updated.

In order to keep the communication with the client active, the PCU may need to transmit negative response messages including response code 78_{16} (requestCorrectlyReceived-ResponsePending). When the routine is completely executed, the server transmits the RoutineControl positive response message that includes the results of the check performed on the programming dependencies.

Table 63 specifies the message sequence example – DeployLoopRoutineId with SF = requestRoutineResult.

Table 63 — Message sequence example – DeployLoopRoutineID with SF = requestRoutineResult

Content	A_PDU definition
31_{16}	**RoutineControl request message SID**
03_{16}	SF = [SuppressPosRspMsgIndicationBit, requestRoutineResult]
$E201_{16}$	RID = DeployLoopRoutineId
—	routineControlOptionRecord = [
01_{16}	routineControlOption = DeploymentLoopId]; airbag driver side frontal 1st stage
71_{16}	**RoutineControl positive response message SID**
03_{16}	SF = [suppressPosRspMsgIndicationBit, requestRoutineResult]
$E201_{16}$	RID = DeployLoopRoutineID
—	routineStatusRecord = [
00_{16}	routineStatus #1 = routineInfo: VM-specific
01_{16}	routineStatus #2 = DeploymentLoopId; airbag driver side frontal 1st stage

2) Second edition under preparation. Stage at the time of publication: ISO/DIS 26021-3:2021.

Table 63 *(continued)*

Content	A_PDU definition
10_{16}	routineStatus #3 = DeploymentLoopStatus]; Failure: an electrical fault in the firing loop, e.g. interrupted, short to ground, has deactivated this loop (CAUTION — external deployment is required).

10.15 AL – ACL request deployment sequence (optional)

10.15.1 AL – Requirements specification – ACL request deployment sequence

The ACL request deployment sequence service is used to enable the PCU to report its internal deployment ready status.

Applicable use case: UC 5.1 – Report ACL deployment sequence (ACL-Init)

REQ	7.75 AL – ACL request deployment sequence (optional) – General requirement
The vehicle's PCU shall report the ACL request deployment sequence response upon a PDT physical request message using the ACL request deployment sequence service.	

REQ	7.76 AL – ACL request deployment sequence (optional) – ACL request deployment sequence – Request and positive response message processing
After the successful reception and processing of the ACL request deployment sequence request message the vehicle's PCU shall send an ACL request deployment sequence positive response message as specified in Table 64.	

REQ	7.77 AL – ACL request deployment sequence (optional) – ACL request deployment sequence – Request and negative response message processing
After the successful reception and processing of the ACL request deployment sequence request message the vehicle's PCU shall send an ACL request deployment sequence negative response message (as specified in Table 64) in case the PCU cannot proceed to the activation process due to internal conditions.	

10.15.2 AL – Message sequence requirements – ACL request deployment sequence

The implementation requirements of the ACL request deployment sequence specify the message sequence and relevant requirements (see Table 64).

Table 64 — ACL request deployment sequence service

Content	A_PDU definition	REQ	Cvt
21_{16}	ACL request deployment sequence request message SID	7.76	M
61_{16}	ACL request deployment sequence positive response message SID	7.76	M
$7F_{16}$	ACL request deployment sequence negative response message SID	7.77	M

10.16 AL – ACL confirm deployment sequence (optional)

10.16.1 AL – Requirements specification – ACL confirm deployment sequence

The ACL confirm deployment sequence service is used to confirm to the PDT that the PCU is ready to perform the deployment sequence.

Applicable use case: UC 5.3 – Perform ACL deployment confirmation sequence (Device-Deploy)

REQ	7.78 AL – ACL confirm deployment sequence (optional) – General requirement
The vehicle's PCU shall report the ACL confirm deployment sequence response upon a PDT physical request message using the ACL confirm deployment sequence service.	

REQ	7.79 AL – ACL confirm deployment sequence (optional) – Request and positive response message processing
After the successful reception and processing of the ACL confirm deployment sequence request message the vehicle's PCU shall send an ACL confirm deployment sequence positive response message as specified in Table 65.	

REQ	7.80 AL – ACL confirm deployment sequence (optional) – Request and negative response message processing
After the successful reception and processing of the ACL request deployment sequence request message the vehicle's PCU shall send an ACL request deployment sequence negative response message (as specified in Table 65) in case the PCU cannot proceed to the activation process due to internal conditions.	

10.16.2 AL – Message sequence requirements – ACL confirm deployment sequence (optional)

The implementation requirements of the ACL confirm deployment sequence specify the message sequence and relevant requirements (see Table 65).

Table 65 — ACL confirm deployment sequence service (optional)

Content	A_PDU definition	REQ	Cvt
27_{16}	ACL confirm deployment sequence request message SID	7.79	M
67_{16}	ACL confirm deployment sequence positive response message SID	7.79	M
$7F_{16}$	ACL confirm deployment sequence negative response message SID	7.80	M

10.17 AL – ACL terminate deployment sequence (optional)

10.17.1 AL – Requirements specification – ACL terminate deployment sequence (optional)

The ACL terminate deployment sequence service is used to terminate the PCU deployment sequence.

Applicable use case: UC 6.2 – Terminate PCU pyrotechnic device scrapping via ACL

REQ	7.81 AL – ACL terminate deployment sequence (optional) – General requirement
The vehicle's PCU shall report the ACL terminate deployment sequence response upon a PDT physical request message using the ACL terminate deployment sequence service.	

REQ	7.82 AL – ACL terminate deployment sequence (optional) – Request and positive response message processing
After the successful reception and processing of the ACL terminate deployment sequence request message the vehicle's PCU shall send an ACL terminate deployment sequence positive response message as specified in Table 66.	

10.17.2 AL – Message sequence requirements – ACL terminate deployment sequence

The implementation requirements of the ACL terminate deployment sequence specify the message sequence and relevant requirements (see Table 66).

Table 66 — ACL terminate deployment sequence service

Content	A_PDU definition	REQ	Cvt
30_{16}	ACL terminate deployment sequence request message SID	7.82	M
70_{16}	ACL terminate deployment sequence positive response message SID	7.82	M

10.18 AL – EcuReset

10.18.1 AL – Requirements specification – EcuReset

The EcuReset service is used by the PDT/client to request a PCU/server reset. This message flow shows the preferred way of ending the diagnostic session "SafetySystemDiagnosticSession" in a PCU.

Applicable use case: UC 6.1 – Terminate PCU pyrotechnic device scrapping via communication interface

REQ	7.83 AL – EcuReset– General requirement
The vehicle's PCU software shall perform a reset upon a PDT request message using the EcuReset service.	

REQ	7.84 AL – EcuReset– Request and response message processing
After the successful reception and processing of the EcuReset request message the vehicle's PCU shall send an EcuReset positive response message as specified in Table 67.	

REQ	7.85 AL – EcuReset– MsgParam – SuppressPosRspMsgIndicationBit
The ResetType parameter shall be used to set the type of reset and to control the transmission of a PCU's response message via the SuppressPosRspMsgIndicationBit.	

10.18.2 AL – Message sequence requirements – EcuReset

Table 67 specifies the EcuReset– ResetType service.

Table 67 — EcuReset service – ResetType

Content	A_PDU definition	REQ	Cvt
1 byte	**EcuReset request message SID**		M
1 byte	SF = ResetType = [SuppressPosRspMsgIndicationBit, HardReset]		M
1 byte	**EcuReset positive response message SID**		M
1 byte	SF = ResetType = [SuppressPosRspMsgIndicationBit, HardReset]		M
NOTE The EcuReset positive response message is only sent if the SuppressPosRspMsgIndicationBit = FALSE.			

10.18.3 AL – Message sequence example – hardReset

The objective of this example is to show the message communication between the PDT and the vehicle's PCU to terminate the current diagnostic session in the PCU.

Table 68 provides an example of the EcuReset service message sequence with positive response.

Table 68 — Example of the EcuReset service message sequence with positive response

Content	A_PDU definition
11_{16}	**EcuReset request message SID**
01_{16}	SF = ResetType = [SuppressPosRspMsgIndicationBit = FALSE, HardReset]

Table 68 *(continued)*

Content	A_PDU definition
51_{16}	**EcuReset positive response message SID**
01_{16}	SF = ResetType = [SuppressPosRspMsgIndicationBit = FALSE, HardReset]

Table 69 provides an example of the EcuReset service message sequence without positive response. The PDT/client requests not to receive a positive response message by setting the SuppressPosRspMsgIndicationBit (bit 7 of the SubFunction parameter) to "TRUE" ("1").

Table 69 — Example of the EcuReset service message sequence without positive response

Content	A_PDU definition
11_{16}	**EcuReset request message SID**
81_{16}	SF = ResetType = [SuppressPosRspMsgIndicationBit = TRUE, HardReset]

11 Presentation layer (PL)

11.1 PL – Data type UNUM8

REQ	6.1 PL – Data type – UNUM8
The UNUM8 data type shall be defined as an 8-bit unsigned numeric value, which defines the range of 00_{16} to FF_{16}.	

11.2 PL – Data type UNUM16

REQ	6.1 PL – Data type – UNUM16
The UNUM16 data type shall be defined as a 16-bit unsigned numeric value, which defines the range of 0000_{16} to $FFFF_{16}$.	

11.3 PL – Data type UNUM32

REQ	6.1 PL – Data type – UNUM32
The UNUM32 data type shall be defined as a 32-bit unsigned numeric value, which defines the range of $0000\ 0000_{16}$ to $FFFF\ FFFF_{16}$.	

11.4 PL – Data type UCHAR8

REQ	6.1 PL – Data type – UCHAR8
The UCHAR8 data type shall be defined as an 8-bit unsigned character and is defined in the range of 00_{16} to FF_{16}.	

12 Session layer (SL)

12.1 SL – Timing parameters

REQ	5.1 SL – Timing parameters – ISO 14229-2
The vehicle's PCU-related systems shall be in accordance to the session layer timing parameters as specified in ISO 14229-2.	

REQ	5.2 SL – Timing parameters – Δt_{P2} timing
The Δt_{P2} timing parameter shall be in the range between 0 ms and 500 ms.	

REQ	5.3 SL – Timing parameters – Monitor t_{S3_Server} timeout
The PDT shall monitor the diagnostic communication as soon as the safetySystemDiagnosticSession is confirmed with a positive response message by the PCU. With no diagnostic communication for more than the timeout value of t_{S3_Server} (5 s), the PCU in the vehicle shall end the safetySystemDiagnosticSession.	

12.2 SL – Error detection

REQ	5.4 SL – Error detection
SL timing parameter timeouts shall be detected and reported via the service primitive parameter `Result`. The SL supports the following Result parameter values: Range: [`Err_SL_SESSION_TIMEOUT`]	

In the case of a communication error, it is assumed that the PDT/client and the PCU/server implement the application and session layer timing as defined in ISO 14229-2. The error handling is implemented in the PDT/client.

13 Transport layer (TL)

13.1 TL – DoCAN

This requirement specifies the normative references to DoCAN TL standards applicable to this document.

REQ	3.1 TL – DoCAN – Normative reference
The vehicle PCU(s) shall support the requirements specified in ISO 15765-5 and ISO 15765-2.	

13.2 TL – DoIP

This requirement specifies the normative references to DoIP TL standards applicable to this document.

REQ	4.2 TL – DoIP – Normative reference
The vehicle PCU(s) shall support the requirements specified in ISO 13400-2.	

14 Network layer (NL)

14.1 NL – DoCAN

This requirement specifies the normative references to DoCAN NL standards applicable to this document.

REQ	3.1 NL – DoCAN – Normative reference
The vehicle PCU(s) shall support the requirements specified in ISO 15765-5 and ISO 15765-2.	

REQ	3.2 NL – DoCAN – ISO 15765-2 FC.WAIT
The NL in the PCU shall not transmit the FC.WAIT packets in the segmented response message.	

REQ	3.3 NL – DoCAN – ISO 15765-2 FC.BS non-zero block size
The NL in the PCU shall not transmit the FC packet with the non-zero block size (BS) parameter in the segmented response message.	

REQ	3.4 NL – DoCAN ISO 15765-2 FC.STmin non-zero separation time
The NL shall not transmit the FC packet with the non-zero separation time (t_{ST_min}) parameter in the segmented response message.	

14.2 NL – DoIP

This requirement specifies the normative references to DoIP NL standards applicable to this document.

REQ	3.5 NL – DoIP – Normative reference
The vehicle PCU(s) shall support the requirements specified in ISO 13400-2.	

15 Data link layer (DLL)

15.1 DLL – CAN L_Data frame padding bytes

CAN frame padding bytes are used to achieve a consistent frame length to support a minimum time between back to back CAN frames at the receiver side.

REQ	2.1 DLL – CAN L_Data frame padding bytes – CAN data length code (DLC)
The vehicle PCU(s) shall add padding bytes into the frame to match the value included in the CAN DLC.	

15.2 DLL – ACL with bidirectional communication

15.2.1 DLL – `tP4_Sender` timing specification

REQ	2.2 DLL – ACL with bidirectional communication – DLL – $t_{P4\ Sender}$ timing specification
The $t_{P4\ Sender}$ inter-byte timing parameter of the sender shall be set to: 1 ms < $t_{P4\ Sender}$ < 1 000 ms.	

15.2.2 DLL – Bit rate and byte format specification

REQ	2.3 DLL – ACL with bidirectional communication – Bit rate
The PCU shall set the ACL bit rate to 10 400 kbit/s.	

REQ	2.4 DLL – ACL with bidirectional communication – ACL byte format and bit order
A byte shall consist of a start bit, bit 0 to bit 7, and a stop bit. Bit 0 shall be the LSb and bit 7 shall be the MSb. The LSb shall follow the start bit. The MSb shall be followed by the stop bit (see Figure 12).	

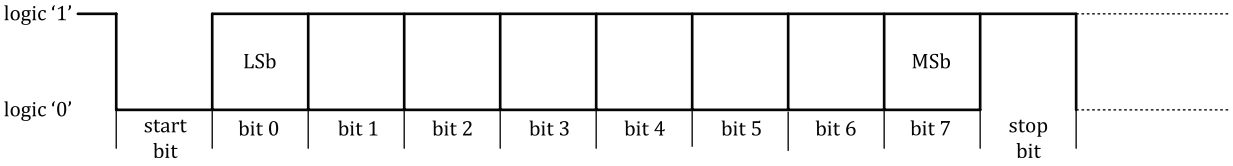


Figure 12 — DLL – ACL byte format and bit order

16 Physical layer (PHY)

16.1 PHY – Connection between PDT and vehicle PCU(s)

The connection between the PDT and the vehicle's PCU(s) is realized via the diagnostic link connector (DLC). Two communication links exist, either DoCAN or DoIP. Optionally, the vehicle manufacturer can implement an ACL with bidirectional communication.

REQ	1.1 PHY – Connection between PDT and vehicle PCU(s) – DLC with DoCAN or DoIP hardware provision
The vehicle shall meet the requirements stated in ISO 15031-3 if ISO 15765 DoCAN is supported or it shall meet the requirements stated in ISO 13400-4 if ISO 13400 DoIP is supported.	

REQ	1.2 PHY – Connection between PDT and vehicle PCU(s) – DLC with optional ACL
If the vehicle supports an ACL with bidirectional communication or PWM signal, the ACL wire shall be conneted to pin 15 of the ISO 15031-3 DLC.	

This document specifies the vehicle interface of the PCU(s) and the PDT. The PCU(s) is (are) connected with the vehicle's diagnostic link connector (DLC, see Figure 13 key 2). It is the vehicle manufacturer's choice, whether ISO 15765 DoCAN or ISO 13400 DoIP is supported by the vehicle. The PDT communicates with the PCU(s) and enables deployment with bidirectional communication.

Depending upon the vehicle-specific architecture, the communication link of the PCU(s) can be connected via a fixed-address PCU (gateway, see Figure 13 key 3) to the DLC, thus a communication link in the PCU (see Figure 13 key a and 4) for the mandatory link is not required.

Figure 13 key b represents the ACL with bidirectional communication, which is connected to the PCUs (see Figure 13 key 4 and key 5).

Figure 13 shows the DLC with DoCAN, DoIP, and ACL.