INTERNATIONAL STANDARD

ISO 22901-2

First edition
2011-07-01

Road vehicles — Open diagnostic data exchange (ODX) —

Part 2: Emissions-related diagnostic data

Véhicules routiers — Échange de données de diagnostic ouvert (ODX) —

Partie 2: Données de diagnostic relatives aux émissions

Reference number
ISO 22901-2:2011(E)

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

Published in Switzerland

Contents

Page

Foreword

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

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

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

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO 22901-2 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 22901 consists of the following parts, under the general title *Road vehicles — Open diagnostic data exchange (ODX)*:

— *Part 1: Data model specification*

— *Part 2: Emissions-related diagnostic data*

Introduction

This International Standard has been established in order to define the data format for transferring standardized emissions-related diagnostic data of the vehicle's OBD system between system supplier, vehicle manufacturer and service dealerships and diagnostic tools of different vendors.

The standardized information is contained in the following standards:

— Diagnostic protocol information:

  — ISO 9141-2:1994, *Road vehicles — Diagnostic systems — Part 2: CARB requirements for interchange of digital information,*

  — ISO 9141-2:1994/Amd.1:1996, *Road vehicles — Diagnostic systems — Part 2: CARB requirements for interchange of digital information — Amendment 1*,

  — ISO 14230-4:2000, *Road vehicles — Diagnostic systems — Keyword Protocol 2000 — Part 4: Requirements for emissions-related systems*,

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

  — SAE J1850, *Class B Data Communications Network Interface*

  — ISO 15031-5, *Road vehicles — Communication between vehicle and external equipment for emissions-related diagnostics — Part 5: Emissions-related diagnostic services*;

— Emissions-related OBD data:

  — ISO 15031-4, *Road vehicles — Communication between vehicle and external equipment for emissions-related diagnostics — Part 4: External test equipment*,

  — ISO 15031-5, *Road vehicles — Communication between vehicle and external equipment for emissions-related diagnostics — Part 5: Emissions-related diagnostic services*,

  — ISO 15031-6, *Road vehicles — Communication between vehicle and external test equipment for emissions-related diagnostics — Part 6: Diagnostic trouble code definitions,*

  — SAE J1979-DA, *Digital Annex of E/E Diagnostic Test Modes*,

  — SAE J2012-DA, *Digital Annex of Diagnostic Trouble Code Definition*;

— OBD Conformance test cases:

  — SAE J1699-3, *OBD II Compliance Test Cases*.

The automotive industry mostly utilizes an informal description to document diagnostic data stream information of vehicle ECUs. Each user, who desires to use the ECU diagnostic data stream documentation to setup development tools or service diagnostic test equipment, has a requirement for a manual transformation of this documentation into a format readable by these tools. This effort will no longer be required if the diagnostic data stream information is provided in ODX format and if those tools support the ODX format.

Road vehicles — Open diagnostic data exchange (ODX) —

Part 2:
Emissions-related diagnostic data

1 Scope

This part of ISO 22901 is intended to ensure that diagnostic data stream information is available to diagnostic tool application manufacturers to simplify the support of the aftermarket automotive service industry. The ODX modelled diagnostic data are compatible with the software requirements of the Modular Vehicle Communication Interface (MVCI) (ISO 22900-2 and ISO 22900-3). The ODX modelled diagnostic data can enable an MVCI device to communicate with the vehicle [ECU(s)] and interpret the diagnostic data contained in the messages exchanged between the external test equipment and the ECU(s). For ODX-compliant external test equipment, no software programming is necessary to convert diagnostic data into technician-readable information for display by the external test equipment.

This part of ISO 22901 contains emissions-related OBD data examples described in ODX. The data examples derive from ISO 15031 (all parts).

EXAMPLES Diagnostic trouble codes, data parameters, identification data and communication parameters.

The emissions-related OBD ODX modelled diagnostic data describe

— the protocol specification from diagnostic communication of emissions-related ECUs;

— the communication parameters for the emissions-related OBD protocols and data link layers and for emissions-related ECU software;

— the related vehicle interface description (connectors and pin-out);

— the functional description of diagnostic capabilities of a network of ECUs.

This part of ISO 22901 is based on emissions-related diagnostic data derived and formatted according to the ISO 15765-4 DoCAN protocol. The definitions and XML representation is exemplary for all other protocols that are referenced in ISO 15031-5.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 15031 (all parts), *Road vehicles — Communication between vehicle and external equipment for emissions-related diagnostics*

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

ISO 22901-1, *Road vehicles — Open diagnostic data exchange (ODX) — Part 1: Data model specification*

3 Terms and definitions and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 22901-1 apply.

3.2 Abbreviated terms

MVCI Modular Vehicle Communication Interface

ODX-RT Open Diagnostic data eXchange — Run-Time format

4 Conventions

This part of ISO 22901 is based on the conventions discussed in the OSI Service Conventions (ISO/IEC 10731[11]) as they apply for diagnostic services.

5 ODX data in the ECU life cycle

Figure 1 shows the usage of ODX in the ECU life cycle. Engineering, manufacturing, and service specify that communication protocol and data should be implemented in the ECU. This information is documented in a structured format utilizing the XML standard and by an appropriate ODX authoring tool. There is potential to generate ECU software from the ODX file. Furthermore, the same ODX file is used to set up the diagnostic engineering tools to verify proper communication with the ECU and to perform functional verification and compliance testing. Once all quality goals are met, the ODX file may be released to a diagnostic database. Diagnostic information is now available to manufacturing, service, OEM franchised dealers and aftermarket service outlets via Intranet and Internet.

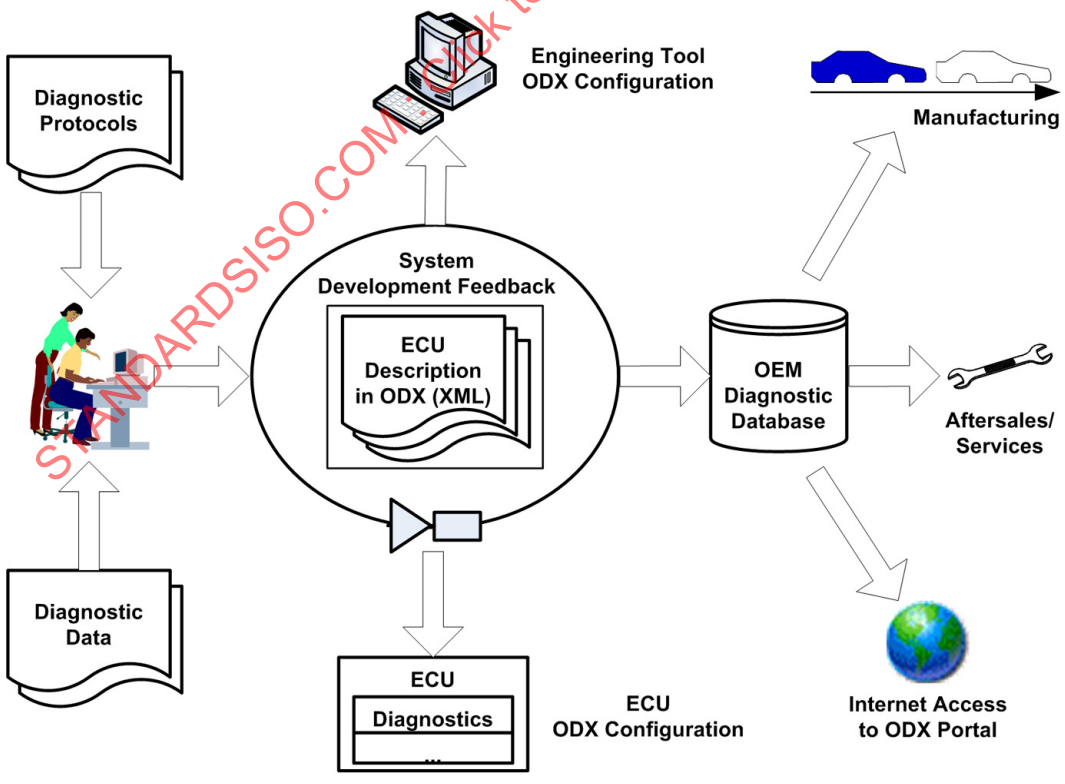


Figure 1 — Usage of ODX data in the ECU life cycle

The objective of this specification is to ensure that diagnostic data from any vehicle manufacturer is independent of the testing hardware and protocol software supplied by any test equipment manufacturer.

6 Emissions-related OBD ODX use cases

6.1 Use case 1 — OBD Scan Tool based on a Modular VCI architecture and ODX

This use case describes the usage of an OBD scan tool in accordance with ISO 15031-4 / SAE J1978 and implemented according to the Modular VCI specification (see ISO 22900, parts 1, 2 and 3) and ODX (see ISO 22901-1).

The benefits of an emissions-related OBD scan tool that is based on the Modular VCI and ODX standard are the following:

— no software programming to support the implementation of

  — new diagnostic trouble codes (see ISO 15031-6 / SAE J2012-DA),

  — new PIDs, Test IDs, Monitor IDs, Info Type IDs, and Scaling IDs (see ISO 15031-5 / SAE J1979-DA);

— OBD scan tool applications in accordance with ISO 15031-4 are developed only once and are not impacted by modifications / changes in the definition of emissions-related OBD data and formats;

— separation of application, communication logic and data items.

NOTE The Modular VCI software architecture supports the emissions-related OBD scan tool requirements as well as enhanced diagnostic protocols, data streams and applications.

Figure 2 illustrates external test equipment connected to the vehicle's diagnostic connector. The OBD scan tool's software architecture is compliant to the Modular VCI specifications. The diagnostic kernel is the key software component of the Modular VCI system. It implements the D-PDU API (see ISO 22900-2), the D-Server API (see ISO 22900-3) and the interface to the ODX derived runtime data.

The OBD scan tool application depends on standardized names or naming conventions as defined by this part of ISO 22901. These names are defined in the emissions-related ODX data and utilized by the OBD scan tool application to address logical links, services, and emission-related data. Using the standardized names and structures from this part of ISO 22901, the interface to implement the scan tool application against is clearly defined. This is indicated by the dashed line in Figure 2.

The D-PDU API is a software component of the tool supplier's Modular VCI protocol module. It connects the diagnostic kernel with any Modular VCI compatible vehicle communication interface.

The D-Server API of the diagnostic kernel provides a standardized interface to the OBD scan tool applications. These applications shall be in accordance with ISO 15031-4, which implements the standardized data and messages of ISO 15031-5 and ISO 15031-6.

The emissions-related ODX runtime data format is tool supplier specific. The runtime format is not contained in the ODX standard (see ISO 22901-1). Based on the use cases supported by the diagnostic tool, the content and structure of the ODX runtime data format and content may differ. However, for emissions-related OBD the OBD scan tool applications and ODX runtime data shall support the full scope of ISO 15031 (all parts) and the respective SAE J documents.

All emissions-related OBD data as specified in ISO 15031-5 and SAE J1979-DA, ISO 15031-6 and SAE J2012-DA shall be authored according to the requirements established in this part of ISO 22901.

This use case requires the unique and complete definition of all elements necessary for any OBD scan tool application compliant to ISO 22900.

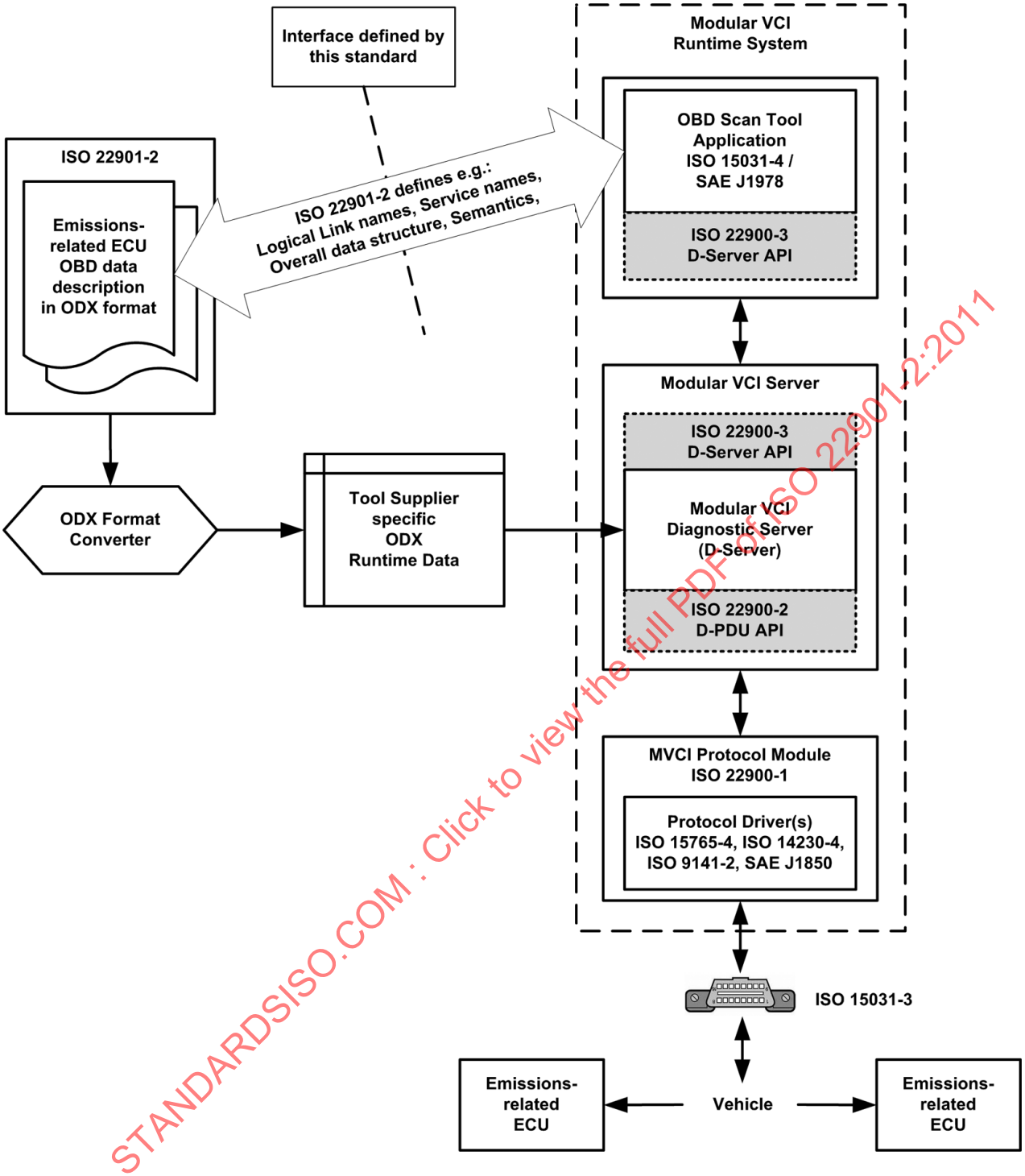


Figure 2 — OBD scan tool based on Modular VCI architecture and ODX

6.2 Use case 2 — Conversion of emissions-related OBD data to ODX format

This use case describes the conversion of emissions-related OBD data into the ODX format in order to provide various applications of external test equipment with emissions-related OBD data in an ODX-RT (runtime) format.

It is assumed that the external test equipment is based on ISO 22900.

The emissions-related OBD data files derive from the Registration Authority installation for ISO 15031.

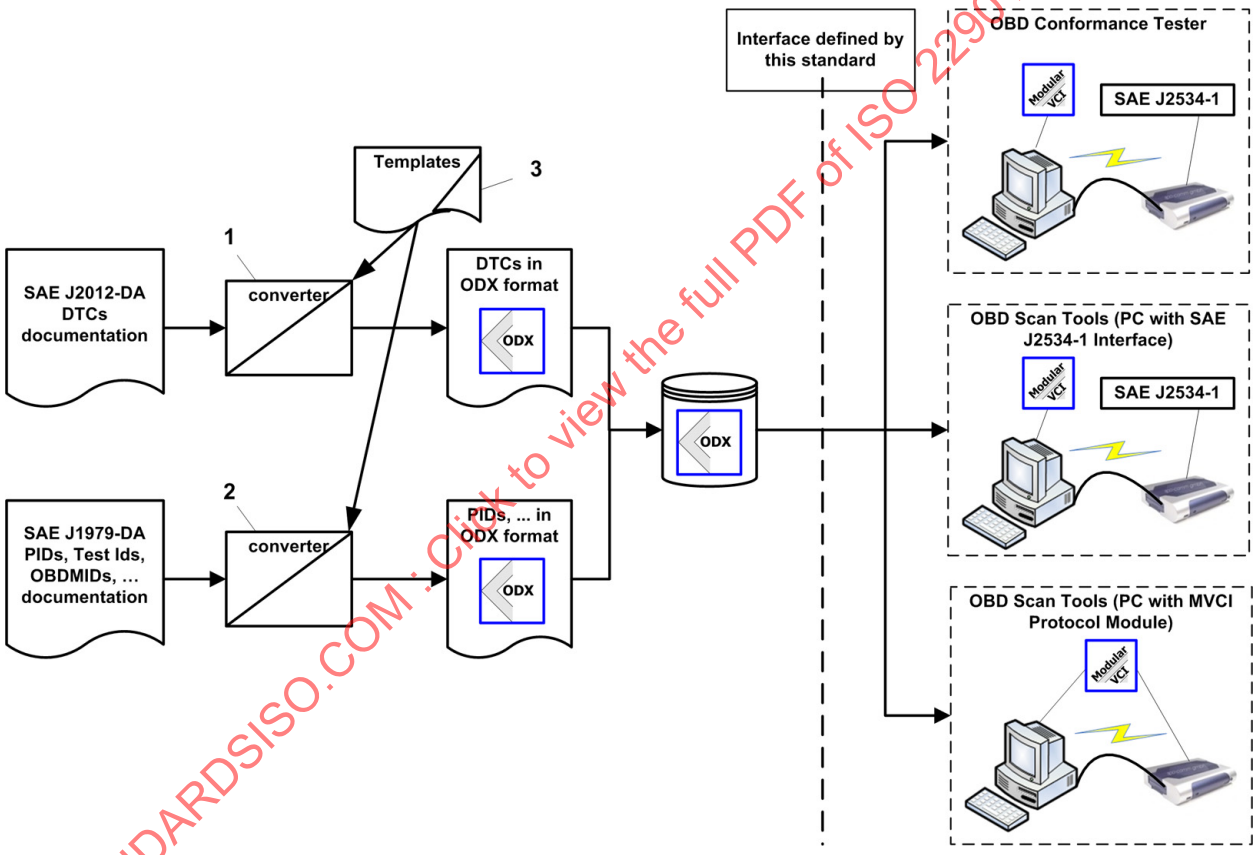
The applicable emissions-related OBD data files are

— SAE J1979-DA,

— SAE J2012-DA.

Figure 3 illustrates the process to be followed in order to convert SAE J2012-DA, and SAE J1979-DA data file information (i.e. Excel or equivalent format) into a standardized ODX format[1],[2] which enrich the emissions-related OBD data with template[3] information. The dotted line depicts the interface that this part of ISO 22901 defines. ODX data providers can deliver ODX data in the format defined here, while tester and scan tool developers can create their tools in accordance with this part of ISO 22901. Thus, both parties can work independently and their products will work together.

How far the converter processes can be automated depends solely on the concrete format of the digital annex. This part of ISO 22901 defines the target format of these processes.



Key

1 SAE J2012-DA DTC converter into ODX format

2 SAE J1979-DA PIDs, OBDMIDs, ... converter into ODX format

3 ODX converter templates to determine standardized ODX parts i.e. PROTOCOLS, COMPARAMS, ... and ODX usage as defined by this part of ISO 22901

Figure 3 — Emissions-related OBD data converter to ODX-RT format

The benefits of implementing this use case are

— the setup/update of an ISO 22900 Modular VCI based OBD test equipment utilizing a SAE J2534-1 compliant vehicle communication interface or an MVCI compliant Protocol Module with emissions-related OBD data (ODX-RT format) that derive from a conversion of the applicable SAE Digital Annexes;

— the setup/update of an ISO 22900 Modular VCI based OBD conformance tester with emissions-related OBD data (ODX-RT format) that implements the test cases as specified in SAE J1699-3.

7 Emissions-related OBD ODX application examples

7.1 OBD conformance tester according to SAE J1699-3

This application example describes the implementation of an OBD conformance tester in accordance with SAE J1699-3 and based on the Modular VCI software architecture. The base architecture as shown in use case 1 applies. The major difference between the emissions-related OBD scan tool and the OBD conformance tester is implemented in the test applications. While the emissions-related OBD scan tool is in accordance with ISO 15031-4, the OBD conformance tester is in accordance with SAE J1699-3. This specification describes very specific test cases in order to achieve vehicle emissions-related system compliance. These test cases have been introduced and referenced by legislation in order to reduce emissions-related diagnostic software implementation deviations in the ECUs from ISO 15031 (all parts) and the respective SAE J documents.

The benefits of an OBD conformance tester based on the Modular VCI and ODX standard are

— no software programming to support the implementation of

  — new diagnostic trouble codes (see ISO 15031-6 / SAE J2012-DA),

  — new PIDs, Test IDs, Monitor IDs, Info Type IDs and Scaling IDs (see ISO 15031-5 / SAE J1979-DA);

— conformance test applications implement the test logic but not the data items (derive from emissions-related ODX runtime);

— clear separation of application and communication logic as well as from all data items.

Figure 4 is based on the architecture as shown in use case 1.

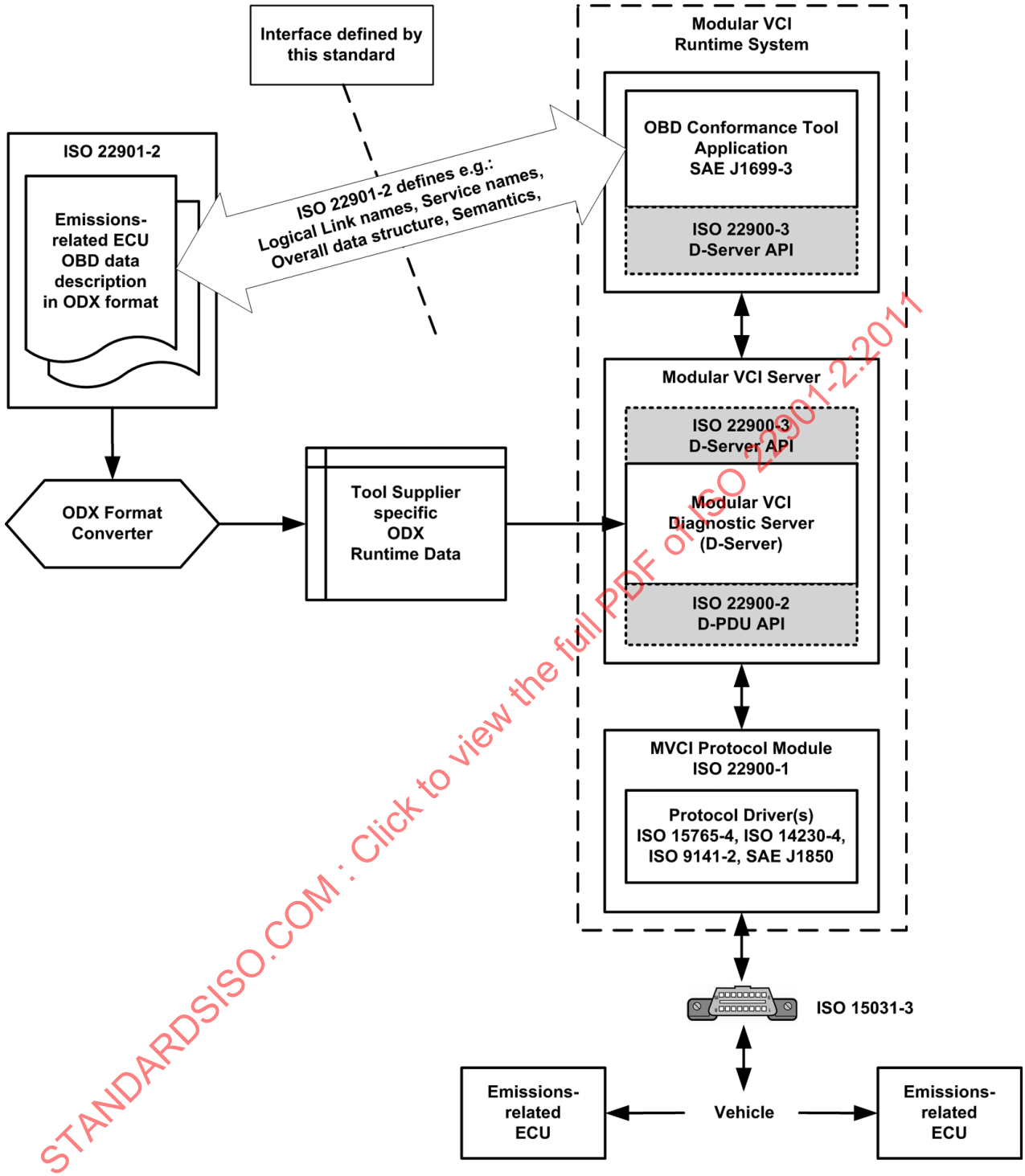


Figure 4 — ODX emissions OBD Modular VCI based OBD conformance tester

7.2 Usage of ODX as a configuration for standardized ECU software

This application example describes how to drive the implementation of the emissions-related OBD diagnostic software module of the ECU by the OBD ODX data. This may be done either by using the OBD ODX data as configuration for a generic diagnostic software module or by utilizing a software generation process, which is controlled by the OBD ODX configuration data.

Once the OBD behaviour of an ECU is defined in ODX format, this file can be used to configure a standardized software part in the ECU.

The benefits of implementing this use case are that

— it is necessary to test standardized ECU software modules only once;

— standardized ECU software modules can be reused in different projects;

— ECU behaviour fits exactly to the behaviour described in the ODX file (because the software as well as the documentation are derived from the same data source).

Figure 5 depicts an example of an ECU diagnostic software module and configuration data derived from ODX.

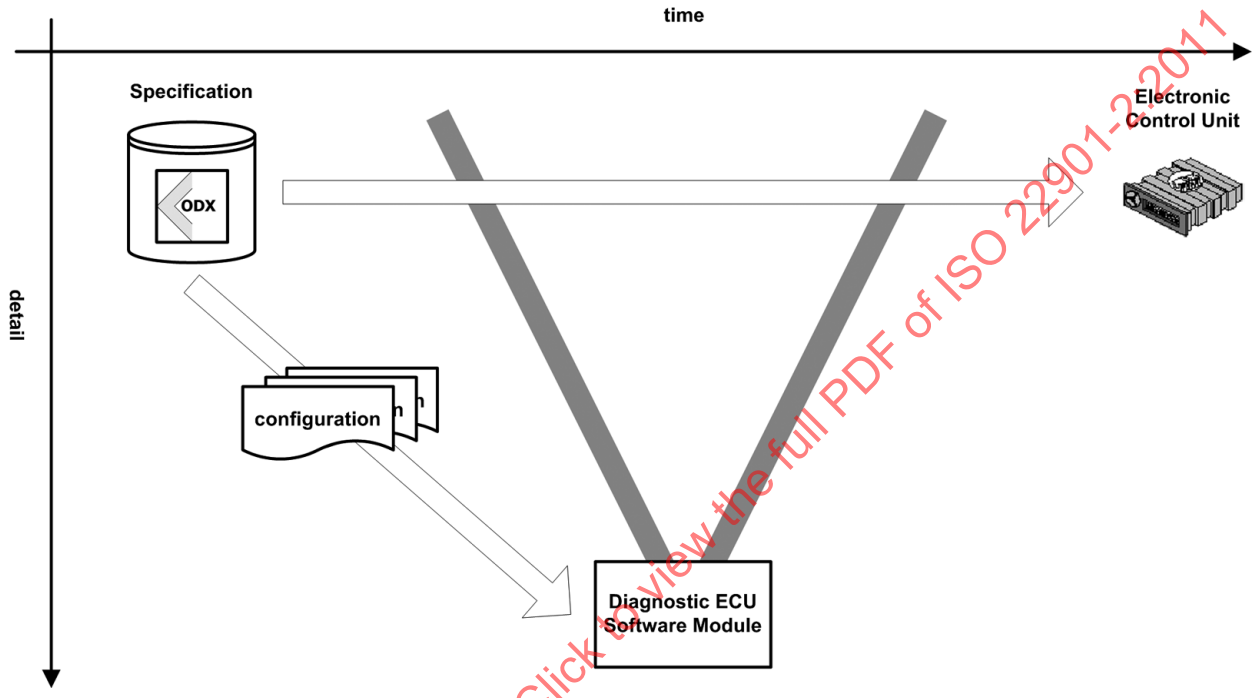


Figure 5 — Example of an ECU diagnostic software module and configuration data derived from ODX

7.3 Usage of ODX checker rules for ECU development

This application example describes the usage of ODX checker rules, which represent a subset of the SAE J1699-3 test cases.

For ODX, adaptable checkers exist. These allow to check for ODX compliance and may be extended with individual checker rules. With these, OBD compliance may be checked before the ECU is implemented, only if the emissions-related OBD ODX data follow the requirements of this part of ISO 22901.

EXAMPLE When specifying the behaviour of an individual ECU in ODX, the support of Infotype 0x0A (ECU-name) for model year 2010 and later can be checked before the ECU code is implemented.

The benefits of implementing this application example are:

— early check for errors (before ECU is implemented in the vehicle);

— checker rules may be provided by a third party and made available to interested users.

Figure 6 depicts an emissions-related OBD compliance test during ECU specification phase.

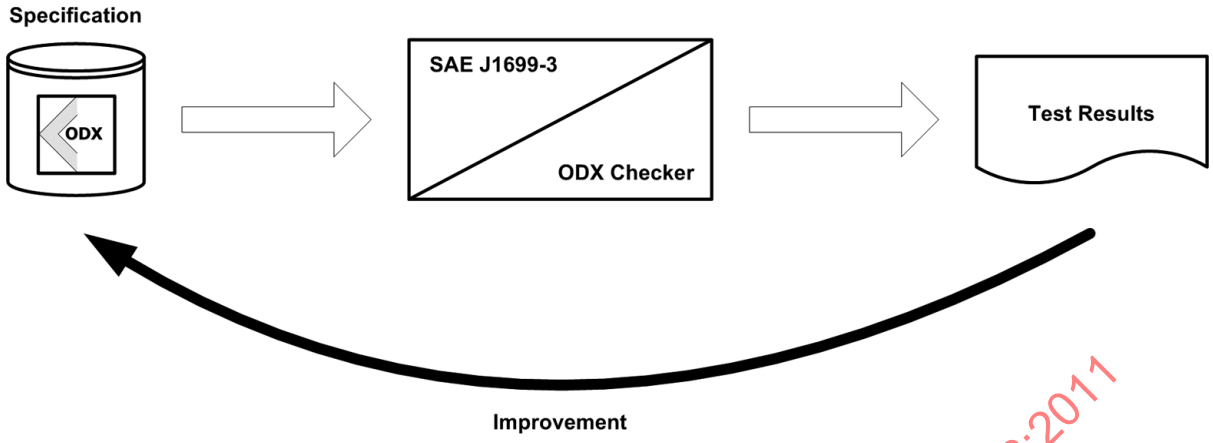


Figure 6 — Emissions-related OBD compliance test during ECU specification phase

8 Specification release version information

8.1 Specification release version location

The release version of the ODX standard can be obtained from every ODX file instance. It is contained in the MODEL-VERSION attribute.

```
<ODX MODEL-VERSION="2.2.0">
```

8.2 Specification release version

The specification release version of this document is: 2.2.0

9 OBD authoring in ODX

9.1 ODX layering

9.1.1 Relationship between ODX layers

Figure 7 illustrates the partitioning of the emissions-related OBD protocols and their associated ComParamSpec from the ECU-Shared-Data and Functional Groups 1 and 2. The Vehicle-Info specifies the Logical Links to the Protocols and Functional Groups. The light and dotted parts are user extensions that can be integrated, if protocols other than ISO 15765-4 are to be supported.

This part of ISO 22901 covers only the ISO 15765-4 DoCAN case. If other physical layers are modelled in ODX as well, the naming as defined in this part of ISO 22901 are to be used.

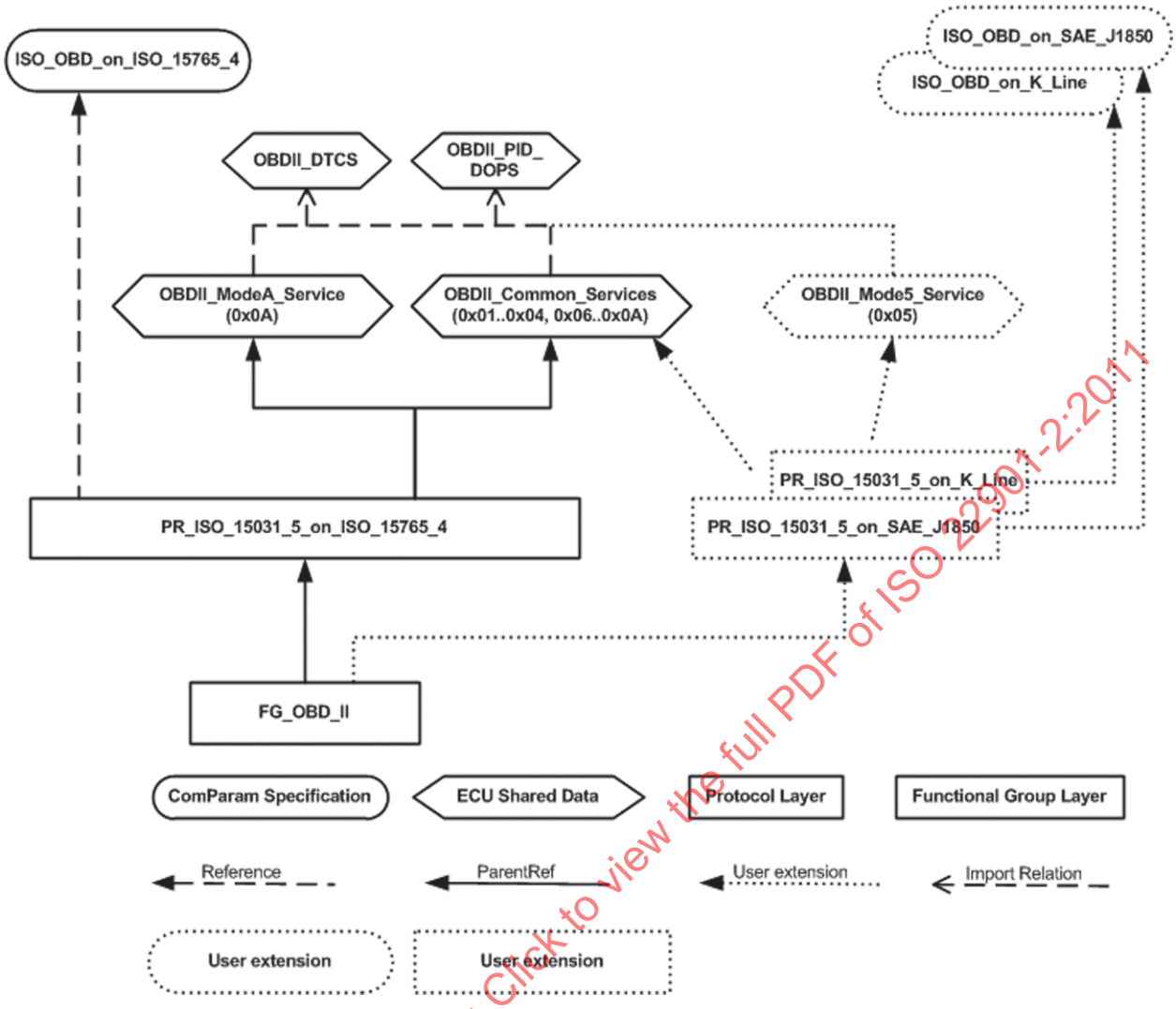


Figure 7 — ComParam-Specs for emissions-related OBD protocols and data

Each DIAG-LAYER should reside in a DIAG-LAYER-CONTAINER of its own.

9.1.2 Authoring of Functional Groups

Functional Groups specify data for a group of emissions-related ECUs, i.e. Engine Control Module and Transmission Control Module which contain all required information to enable the Modular VCI compliant emissions-related OBD test equipment to perform functional communication.

A Functional Group named “FG_OBD_II” specifies the data relevant to the OBD message protocol for all of the available and supported physical link layers (ISO_OBD_on_ISO_15765_4 and also ISO_OBD_on_SAE_J1850, ISO_OBD_on_K_Line, if present). The ComParamSpec as defined by ISO 22900-2 specifies the protocol specific message framing, message timing and message addressing information. For SAE J1993-73, this part of ISO 22901 defines only the name for the protocol layer.

9.1.3 Authoring of emissions-related protocols

The PROTOCOL class in ODX is used to capture communication data like message layout, parameters in diagnostic requests and responses, conversion information to convert from coded values to physical values and vice versa.

For emission-related data, three ODX protocol layers named “PR_ISO_15031_5_on_ISO_15765_4”, “PR_ISO_15031_5_on_SAE_J1850” and “PR_ISO_15031_5_on_K_Line” are defined. “PR_ISO_15031_5_on_ISO_15765_4” is covered by this part of ISO 22901. They capture the physical layer and transport layer specific protocol information.

EXAMPLE Name tags of protocol PR_ISO_15031_5_on_ISO_15765_4

```
<PROTOCOL ID="ID_PR_ISO_15031_5_on_ISO_15765_4">
  <SHORT-NAME>PR_ISO_15031_5_on_ISO_15765_4</SHORT-NAME>
  <LONG-NAME>ISO OBD on CAN</LONG-NAME>
  <COMPARAM-SPEC-REF DOCREF="ISO_OBD_on_ISO_15765_4" DOCTYPE="COMPARAM-SPEC" ID-
   REF="ID_ISO_OBD_on_ISO_15765_4"/>
</PROTOCOL>
```

Table 1 defines SHORT-NAME and LONG-NAME of the OBD protocols.

Table 1 — Definition of SHORT-NAME and LONG-NAME of OBD protocols

SHORT-NAME	LONG-NAME
PR_ISO_15031_5_on_ISO_15765_4	ISO OBD on CAN
PR_ISO_15031_5_on_SAE_J1850	ISO OBD on J1850 VPW and J1850 PWM
PR_ISO_15031_5_on_K_Line	ISO OBD on 9141-2 K-Line and KWP2000 K-Line

In order to identify and group OBD services effectively, all OBD Services are members of the Functional Class “OBD.PROTOCOL.OBDonCAN.FUNCT-CLASS.emissionRelatedDiagnosticServices”.

EXAMPLE FUNCT-CLASSS

```
<FUNCT-CLASSS>
  <FUNCT-CLASS ID="OBD.PROTOCOL.OBDonCAN.FUNCT-CLASS.emissionRelatedDiagnosticServices">
  <SHORT-NAME>ISO_15031_5</SHORT-NAME>
  <LONG-NAME>ISO 15031-5</LONG-NAME>
  </FUNCT-CLASS>
</FUNCT-CLASSS>
```

9.1.4 Authoring of emissions-related ECU-SHARED-DATA

Several named ECU-SHARED-DATA ODX containers capture OBDII relevant services as well as parameter encoding and decoding information.

“OBDII_DOPS” hold the encoding and decoding description of a response and request parameter as well as units and dimension specifications.

“OBDII_Common_Services” holds OBDII services definitions for modes 0x01-0x04, 0x06-0x09.

“OBDII_ModeA_Service” holds OBDII services definitions for mode 0x0A. Mode 0x0A service is separated out because it is not used for OBD other than on CAN.

“OBDII_Mode5_Service” holds OBDII service definitions for mode 0x05. Mode 0x05 service is separated out because it is not used for OBD on CAN.

EXAMPLE Ecu-Shared-Data

```
<ECU-SHARED-DATA>
  <SHORT-NAME>OBDII_DOPS</SHORT-NAME>
  <LONG-NAME>DTCS for the OBD protocol</LONG-NAME>
  <DIAG-DATA-DICTIONARY-SPEC>
    <DTC-DOPS>
      <DTC-DOP ID="DIAG-LAYER-CONTAINER.OBDTemplate.DTC-DOP.J2012DTC">
        <SHORT-NAME>J2012DTC</SHORT-NAME>
        <LONG-NAME>J2012DTC</LONG-NAME>
        <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
          <BIT-LENGTH>16</BIT-LENGTH>
        </DIAG-CODED-TYPE>
        <PHYSICAL-TYPE BASE-DATA-TYPE="A_UINT32" DISPLAY-RADIX="HEX"/>
        <COMPU-METHOD>
          <CATEGORY>IDENTICAL</CATEGORY>
        </COMPU-METHOD>
        <DTCS>
          <DTC ID="ID_OBD.DTC-DOP.J2012DTC.DTC.P0000">
            <SHORT-NAME>P0000</SHORT-NAME>
            <TROUBLE-CODE>0</TROUBLE-CODE>
            <TEXT>ISO/SAE reserved - Use Not Allowed</TEXT>
          </DTC>
                  …
        </DTCS>
      </DTC-DOP>
    </DTC-DOPS>
   …
  </DIAG-DATA-DICTIONARY-SPEC>
</ECU-SHARED-DATA>
```

9.1.5 Authoring of VEHICLE-INFO

The VEHICLE-INFO-SPECIFICATION “VI_OBDII” specifies the logical links to be supported by the Modular VCI (see ISO 22900).

All OBDII logical links refer to Functional Group “FG_OBD_II”. However, for different physical layers, the respective predefined protocol of 9.1.3 is referenced.

There shall be three Logical Links defined, named “LL_OBD_on_ISO_15765_4”, “LL_OBD_on_SAE_J1850” and “LL_OBD_on_K_Line”:

a) LL_OBD_on_ISO_15765_4 references FG_OBD_II and PR_OBD_on_ISO_15765_4;

b) LL_OBD_on_SAE_J1850 references FG_OBD_II and PR_OBD_on_SAE_J1850;

c) LL_OBD_on_K_Line references FG_OBD_II and PR_OBD_on_K_Line.

This technique ensures that selecting a logical link is sufficient to establish ECU communication using the correct protocol, communication parameters and services.

IMPORTANT — OBD communication initialization is specified in ISO 22900-2.

9.2 Service implementation in ODX

9.2.1 General

ISO 22901-1 provides at least two alternatives to author diagnostic data:

a) dedicated diagnostic service related data authoring, i.e. the same data (PIDS, INFOTYPES, ...) are used by all emissions-related OBD protocols and authored for each protocol redundantly;

b) diagnostic service independent related data authoring, also called “table based authoring”, i.e. the same data (PIDS, INFOTYPES, ...), are used by all emissions-related OBD protocols but authored only once and referenced by each protocol.

IMPORTANT — Emissions-related OBD data as specified in ISO 15031-5, ISO 15031-6 / SAE J1979 and SAE J1979-DA (Digital Annex) shall be authored according to b).

9.2.2 OBD Services authoring

9.2.2.1 ODX IDs of OBD services

The ID of the OBD services shall be generated by the following rule: <Layer-ID>.DS_SHORT-NAME

```
<DIAG-SERVICE
    ID="ES_OBDIICommonServices.DS_Service01RequestCurrentPowertrainDiagnosticDataPID13">
    <SHORT-NAME>Service01RequestCurrentPowertrainDiagnosticDataPID13</SHORT-NAME>
    <LONG-NAME>Service 0x01 - Request current powertrain diagnostic data (PID 0x13)</LONG-
NAME>
    …
</DIAG-SERVICE>
```

9.2.2.2 ODX DESCription of OBD services

The text of the ISO 15031-5 “Functional description” subclause of the particular OBD service shall be used.

9.2.2.3 ODX LONG-NAMEs of OBD services

The LONG-NAME of the DIAG-SERVICE shall use the complete wording of the service description in the headlines of ISO 15031-5. This document distinguishes between services for ISO 9141-2, ISO 14230-4, SAE J1850 and services for ISO 15765-4, but the descriptions of the headlines are the same as the services in both cases; this circumstance has to be considered in the LONG-NAME or later on in the description of the services itself. Request and POS-/NEG-RESPONSES have to be kept in mind in this case.

Table 2 defines the LONG-NAMEs for OBD services regarding ISO 15765-4.

Table 2 — LONG-NAMEs for OBD services regarding ISO 15765-4

Headline used in ISO 15031-5 and ISO 15765-4	LONG-NAME
Service 0x01 – Request current powertrain diagnostic data	Service 0x01 – Request current powertrain diagnostic data
Service 0x02 – Request powertrain freeze frame data	Service 0x02 – Request powertrain freeze frame data
Service 0x03 – Request emission-related diagnostic trouble codes	Service 0x03 – Request emission-related diagnostic trouble codes
Service 0x04 – Clear/reset emission-related diagnostic information	Service 0x04 – Clear/reset emission-related diagnostic information
Service 0x05 – Request oxygen sensor monitoring test results	Service 0x05 – Request oxygen sensor monitoring test results
Service 0x06 – Request on-board monitoring test results for specific monitored systems	Service 0x06 – Request on-board monitoring test results for specific monitored systems
Service 0x07 – Request emission-related diagnostic trouble codes detected during current or last completed driving cycle	Service 0x07 – Request emission-related diagnostic trouble codes detected during current or last completed driving cycle
Service 0x08 – Request control of on-board system, test or component	Service 0x08 – Request control of on-board system, test or component
Service 0x09 – Request vehicle information	Service 0x09 – Request vehicle information
Service 0x0A – Request emissions-related diagnostic trouble codes with permanent status	Service 0x0A – Request emissions-related diagnostic trouble codes with permanent status

It is necessary to consider the following exceptions. With service 0x01, the behaviour for PID 0x13 and PID 0x1D shall be distinguished. Therefore, it is necessary to define two services for service 0x01. The LONG-NAME of both services are extended by the marker (PID 0x13) and (PID 0x1D), respectively.

Table 3 defines the LONG-NAMEs for OBD service 0x01 distinguishing PID 0x13 and PID 0x1D.

Table 3 — LONG-NAMEs for OBD service 0x01 distinguishing PID 0x13 and PID 0x1D

Headline used in ISO 15031-5 and ISO 15765-4	LONG-NAME
Service 0x01 – Request current powertrain diagnostic data	Service 0x01 – Request current powertrain diagnostic data (PID 0x13)
Service 0x01 – Request current powertrain diagnostic data	Service 0x01 – Request current powertrain diagnostic data (PID 0x1D)

9.2.2.4 ODX SHORT-NAMEs of OBD services

Table 4 defines the SHORT-NAMEs for OBD services regarding ISO 15765-4.

Table 4 — SHORT-NAMEs for OBD services regarding ISO 15765-4

LONG-NAME	SHORT-NAME
Service 0x01 – Request current powertrain diagnostic data	Service01RequestCurrentPowertrainDiagnosticData
Service 0x02 – Request powertrain freeze frame data	Service02RequestPowertrainFreezeFrameData
Service 0x03 – Request emissions-related diagnostic trouble codes	Service03RequestEmissionRelatedDiagnosticTroubleCodes
Service 0x04 – Clear/reset emissions-related diagnostic information	Service04ClearResetEmissionRelatedDiagnosticInformation
Service 0x05 – Request oxygen sensor monitoring test results	Service05RequestOxygenSensorMonitoringTestResults
Service 0x06 – Request on-board monitoring test results for specific monitored systems	Service06RequestOnBoardMonitoringTestResults ForSpecificMonitoredSystems
Service 0x07 – Request emissions-related diagnostic trouble codes detected during current or last completed driving cycle	Service07RequestEmissionRelatedDiagnosticTroubleCodesD etectedDuringCurrentOrLastCompletedDrivingCycle
Service 0x08 – Request control of on-board system, test or component	Service08RequestControlOfOnBoardSystemTestOrComponent
Service 0x09 – Request vehicle information	Service09RequestVehicleInformation
Service 0x0A – Request emissions-related diagnostic trouble codes with permanent status	Service0ARequestEmissionRelatedDiagnosticTroubleCodes WithPermanentStatus

IMPORTANT — The addressing method shall be FUNCTIONAL.

Table 5 defines two different service 0x01 LONG-NAME and SHORT-NAME because the interpretation of the respective response impacts the interpretation of, for example, PID 0x1B.

Table 5 — SHORT-NAMEs for OBD service 0x01 distinguishing PID 0x13 and PID 0x1D

LONG-NAME	SHORT-NAME
Service 0x01 – Request current powertrain diagnostic data (PID 0x13)	Service01RequestCurrentPowertrainDiagnosticDataPID13
Service 0x01 – Request current powertrain diagnostic data (PID 0x1D)	Service01RequestCurrentPowertrainDiagnosticDataPID1D

9.2.3 ODX request implementation

9.2.3.1 ODX IDs of requests

For the ID of an OBD request the prefix REQ_ shall be combined with the SHORT-NAME of the request.

9.2.3.2 LONG-NAMEs and SHORT-NAMEs of requests

The LONG-NAME of the request shall be the same as the LONG-NAME of the DIAG-SERVICE that the request belongs to. The SHORT-NAME of the request shall be the same as the SHORT-NAME of the DIAG-SERVICE that the request belongs to.

Table 6 defines the LONG-NAMEs and SHORT-NAMEs of requests.

Table 6 — LONG-NAMEs and SHORT-NAMEs of requests

LONG-NAME of request	SHORT-NAME of request
Service 0x01 – Request current powertrain diagnostic data	Service01RequestCurrentPowertrainDiagnosticData
Service 0x02 – Request powertrain freeze frame data	Service02RequestPowertrainFreezeFrameData
Service 0x03 – Request emission-related diagnostic trouble codes	Service03RequestEmissionRelatedDiagnosticTroubleCodes
Service 0x04 – Clear/reset emission-related diagnostic information	Service04ClearResetEmissionRelatedDiagnosticInformation
Service 0x05 – Request oxygen sensor monitoring test results	Service05RequestOxygenSensorMonitoringTestResults
Service 0x06 – Request on-board monitoring test results for specific monitored systems	Service06RequestOnBoardMonitoringTestResultsForSpecific MonitoredSystems
Service 0x07 – Request emission-related diagnostic trouble codes detected during current or last completed driving cycle	Service07RequestEmissionRelatedDiagnosticTroubleCodes DetectedDuringCurrentOrLastCompletedDrivingCycle
Service 0x08 – Request control of on-board system, test or component	Service08RequestControlOfOnBoardSystemTestOr Component
Service 0x09 – Request vehicle information	Service09RequestVehicleInformation
Service 0x0A – Request emissions-related diagnostic trouble codes with permanent status	Service0ARequestEmissionRelatedDiagnosticTroubleCodes WithPermanentStatus

9.2.4 ODX POS-RESPONSE implementation

9.2.4.1 ODX IDs of POS-RESPONSES

For the ID of an OBD POS-RESPONSE the prefix PRE_ shall be combined with the SHORT-NAME of the POS-RESPONSE.

9.2.4.2 LONG-NAMEs of POS-RESPONSES

The LONG-NAME of the POS-RESPONSE shall be the same as the LONG-NAME of the DIAG-SERVICE that the POS-RESPONSE belongs to.

9.2.4.3 SHORT-NAMEs of POS-RESPONSES

The SHORT-NAME of the POS-RESPONSE shall be the same as the SHORT-NAME of the DIAG-SERVICE that the POS-RESPONSE belongs to.

Table 7 defines the LONG-NAMEs and SHORT-NAMEs of POS-RESPONSES.

Table 7 — LONG-NAMEs and SHORT-NAMEs of POS-RESPONSES

LONG-NAME of POS-RESPONSE	SHORT-NAME of POS-RESPONSE
Service 0x01 – Request current powertrain diagnostic data	Service01RequestCurrentPowertrainDiagnosticData
Service 0x02 – Request powertrain freeze frame data	Service02RequestPowertrainFreezeFrameData
Service 0x03 – Request emission-related diagnostic trouble codes	Service03RequestEmissionRelatedDiagnosticTroubleCodes
Service 0x04 – Clear/reset emission-related diagnostic information	Service04ClearResetEmissionRelatedDiagnosticInformation
Service 0x05 – Request oxygen sensor monitoring test results	Service05RequestOxygenSensorMonitoringTestResults
Service 0x06 – Request on-board monitoring test results for specific monitored systems	Service06RequestOnBoardMonitoringTestResultsForSpecificMonitoredSystems
Service 0x07 – Request emission-related diagnostic trouble codes detected during current or last completed driving cycle	Service07RequestEmissionRelatedDiagnosticTroubleCodesDetectedDuringCurrentOrLastCompletedDrivingCycle
Service 0x08 – Request control of on-board system, test or component	Service08RequestControlOfOnBoardSystemTestOrComponent
Service 0x09 – Request vehicle information	Service09RequestVehicleInformation
Service 0x0A – Request emissions-related diagnostic trouble codes with permanent status	Service0ARequestEmissionRelatedDiagnosticTroubleCodesWithPermanentStatus

9.2.5 ODX NEG-RESPONSE implementation

A GLOBAL-NEG-RESPONSE in the ECU-SHARED-DATA “OBDII_Common_Services” defines the negative responses according to ISO 15031-5 for OBD communication based on ISO 15765-4, SAE J1850. The ID of the GLOBAL-NEG-RESPONSE is “GNR_OBDIIServicesNegativeResponse”. Its LONG-NAME is “OBDII Services – Negative Response”. Its SHORT-NAME is “OBDIIServicesNegativeResponse”.

No DIAG-SERVICE specific NEG-RESPONSEs are used for the OBD Modes 0x01-0x0A.

9.3 ODX PARAMs implementation

As a general rule, the ODX names of a parameter shall be derived from the “Description” of the parameter as specified in ISO 15031-5. The LONG-NAME shall be copied from the description in ISO 15031-5. The SHORT-NAME shall be generated by removing all characters from the LONG-NAME disallowed by the ODX specification.

In case a parameter in ISO 15031-5 has no description or when the description is not favourable for a specific parameter, the naming convention shall be as detailed in this part of ISO 22901.

The second parameter at the POS-RESPONSE shall describe the PID of the request. This shall be done with the use of MATCHING-REQUEST-PARAM. The LONG-NAME of this parameter is “Matching PID”. The SHORT-NAME shall be MatchingParameterID.

9.3.1 Service IDs (SID)

9.3.1.1 LONG-NAMEs and SHORT-NAMEs of request SIDs

The general rule shall be applied, however, the words “request SID” shall be appended to the LONG-NAME of the SID parameter. The SHORT-NAME is then derived from the LONG-NAME as described in the general rule.

Table 8 defines the LONG-NAMEs and SHORT-NAMEs of PARAMs of the request SID.

Table 8 — LONG-NAMEs and SHORT-NAMEs of PARAMs of the request SID

Request SID	LONG-NAME of PARAM	SHORT-NAME of PARAM
Service 0x01	Request current powertrain diagnostic data request SID	RequestCurrentPowertrainDiagnosticDataRequestSID
Service 0x02	Request powertrain freeze frame data request SID	RequestPowertrainFreezeFrameDataRequestSID
Service 0x03	Request emission-related diagnostic trouble codes request SID	RequestEmissionRelatedDiagnosticTroubleCodesRequestSID
Service 0x04	Clear/reset emission-related diagnostic information request SID	ClearResetEmissionRelatedDiagnosticInformationRequestSID
Service 0x05	Request oxygen sensor monitoring test results request SID	RequestOxygenSensorMonitoringTestResultsRequestSID
Service 0x06	Request on-board monitoring test results for specific monitored systems request SID	RequestOnBoardMonitoringTestResults ForSpecificMonitoredSystemsRequestSID
Service 0x07	Request emission-related diagnostic trouble codes detected during current or last completed driving cycle request SID	RequestEmissionRelatedDiagnosticTroubleCodesDetected DuringCurrentOrLastCompletedDrivingCycleRequestSID
Service 0x08	Request control of on-board system, test or component request SID	RequestControlOfOnBoardSystemTestOrComponent RequestSID
Service 0x09	Request vehicle information request SID	RequestVehicleInformationRequestSID
Service 0x0A	Request emissions-related diagnostic trouble codes with permanent status request SID	RequestEmissionsRelatedDiagnosticTroubleCodesWithPermanentStatusRequestSID

The request SID parameters shall be of type CODED-CONST defined as DIAG-CODED-TYPE, i.e. an 8 bit unsigned integer value. The SEMANTIC of the parameter shall be “SERVICE-ID”.

9.3.1.2 LONG-NAMEs and SHORT-NAMEs of response SIDs

The general rule shall be applied, however, the words “response SID” shall be appended to the LONG-NAME of the SID parameter. The SHORT-NAME is then derived from the LONG-NAME as described in the general rule.

Table 9 defines the LONG-NAMEs and SHORT-NAMEs of PARAMs at a service response.

Table 9 — LONG-NAMEs and SHORT-NAMEs of PARAMs at a service response

LONG-NAME of PARAM	SHORT-NAME of PARAM
Request current powertrain diagnostic data response SID	RequestCurrentPowertrainDiagnosticDataResponseSID
Request powertrain freeze frame data response SID	RequestPowertrainFreezeFrameDataResponseSID
Request emission-related diagnostic trouble codes response SID	RequestEmissionRelatedDiagnosticTroubleCodesResponseSID
Clear/reset emission-related diagnostic information response SID	ClearResetEmissionRelatedDiagnosticInformationResponseSID
Request oxygen sensor monitoring test results response SID	RequestOxygenSensorMonitoringTestResultsResponseSID
Request on-board monitoring test results for specific monitored systems response SID	RequestOnBoardMonitoringTestResults ForSpecificMonitoredSystemsResponseSID
Request emission-related diagnostic trouble codes detected during current or last completed driving cycle response SID	RequestEmissionRelatedDiagnosticTroubleCodes DetectedDuringCurrentOrLastCompletedDrivingCycleResponseSID
Request control of on-board system, test or component response SID	RequestControlOfOnBoardSystemTestOrComponentResponseSID
Request vehicle information response SID	RequestVehicleInformationResponseSID

The response SID parameters shall be of type CODED-CONST defined as DIAG-CODED-TYPE, i.e. an 8 bit unsigned integer value. The SEMANTIC of the parameter shall be “SERVICE-ID”.

9.3.2 Local Identifier implementation

9.3.2.1 General

As a general modelling principle, the services 0x01, 0x02, 0x06, 0x08 and 0x09, shall contain TABLE-KEY parameters. This enforces the definition of request and response message structures in TABLES. To further document the relation between DIAG-SERVICE and TABLE, the TABLEs are linked to the DIAG-SERVICES using TABLE-DIAG-COMM-CONNECTOR elements.

Each TABLE-ROW shall carry a specific semantic as specified in 9.3.2.2. This is required to determine which entries in the TABLE are used to inquire diagnostic capabilities and which return actual emissions-related OBD data.

9.3.2.2 SEMANTICs

ODX V2.2.0 supports the specification of TABLE-ROW.SEMANTIC values.

Table 10 defines the SEMANTIC values for DIAG-SERVICEs and TABLEs.

Table 10 — SEMANTIC values for DIAG-SERVICEs and TABLEs

ServiceId (PID/OBDMID/TID/ INFOTYPE)	DIAG-SERVICE.SEMANTIC/ TABLE-ROW.SEMANTIC	TABLE.SEMANTIC
Powertrain Diagnostic and Freeze Frame Data		
0x01 (0x00,0x20,..)	READ_DATA_PWRTRAIN_SUPPORTED	PWRTRAIN_CURR
0x01 (0x01,0x02,..)	READ_DATA_PWRTRAIN_CURR	
0x02 (0x00,0x00, 0x20,0x00, ..)	READ_DATA_PWRTRAIN_FREEZE_FRAME_SUPPORTED	PWRTRAIN_FREEZE_FRAME
0x02 (0x01,0x00, 0x02,0x00, ..)	READ_DATA_PWRTRAIN_FREEZE_FRAME	
Emission-related Trouble Codes		
0x03	READ_DTC_POWERTRAIN	no usage of TABLEs
0x07	READ_DTC_DETECTION_TIME_DEPENDENT	
0x0A	READ_DTC_PERMANENT_STATUS	
0x04	READ_DTC_CLEAR	
On-board Monitoring Test Results for Specific Monitored Systems		
0x06 (0x00,0x20,..)	READ_OBDMID_SUPPORTED	OBDMID
0x06 (0x01,0x02,..)	READ_OBDMID_DATA	
Control of Onboard Systems, Test or Control		
0x08 (0x00,0x020)	READ_TID_SUPPORTED	TID
0x08 (0x01,0x02)	READ_TID_DATA	
Vehicle Information		
0x09 (0x00,0x20)	READ_INFOTYPE_SUPPORTED	INFOTYPE
0x09 (0x02,0x04,..)	READ_INFOTYPE_DATA	

Table 11 defines in its first column the SEMANTIC of the TABLE-DIAG-COMM-CONNECTOR that connects the table with the semantic of the third column with the services named in the second column. As the same service is used to query the supported IDs and to actually request the value for a supported ID, that service is referenced through two TWO-DIAG-COMM-CONNECTORs with different SEMANTIC attributes.

Table 11 — SEMANTIC values for DIAG-SERVICEs and TABLEs

TABLE-DIAG-COMM-CONNECTOR.SEMANTIC	DIAG-SERVICE.SHORT-NAME	TABLE.SEMANTIC
Powertrain Diagnostic and Freeze Frame Data		
SUPPORT_QUERY	Service01RequestCurrentPowertrainDiagnosticData Service01RequestCurrentPowertrainDiagnosticDataPID13 Service01RequestCurrentPowertrainDiagnosticDataPID1D	PWRTRAIN_CURR
READ		
SUPPORT_QUERY	Service02RequestPowertrainFreezeFrameData	PWRTRAIN_FREEZE_FRAME
READ		
On-board Monitoring Test Results for Specific Monitored Systems		
SUPPORT_QUERY	Service06RequestOnBoardMonitoringTestResultsForSpecificMonitoredSystems	OBDMID
READ		
Control of Onboard Systems, Test or Control		
SUPPORT_QUERY	Service08RequestControlOfOnBoardSystemTestOrComponent	TID
READ		
Vehicle Information		
SUPPORT_QUERY	Service09RequestVehicleInformation	INFOTYPE
READ		

9.3.3 Service-specific parameters

9.3.3.1 Service-specific request parameters for services 0x01, 0x02, 0x06, 0x08 and 0x09

Parameters 2 and above are collects in an END-OF-PDU-FIELD Parameter. For services 0x01, 0x06, 0x08 and 0x09, an END-OF-PDU-FIELD with MAX–NUMBER-OF-ITEMS equal to 6 and MIN-NUMBER-OF-ITEMS equal to 1 shall be used. The SHORT-NAME of this END-OF-PDU-FIELD shall be “PIDsSupportedServices01060809RequestPID13” and “PIDsSupportedServices01060809RequestPID1D”, respectively. For service 0x02, an END-OF-PDU-FIELD with MAX-NUMBER-OF-ITEMS equal to 3 and MIN-NUMBER-OF-ITEMS equal to 1 shall be used. The SHORT-NAME of this END-OF-PDU-FIELD shall be “PIDsSupportedServices02Request”.

The END-OF-PDU-FIELD “PIDsSupportedServices01060809RequestPID13” references an ODX STRUCTURE with SHORT-NAME “PIDsSupportedRequestPID13”.

“PIDsSupportedRequestPID13” has a single PARAM of Type TABLE-KEY with SHORT-NAME “PIDRequest”. “PIDRequest” references an object of type TABLE with SHORT-NAME “PIDsSupportedForPID13”. The TABLE has one TABLE-ROW for each PID (0x00 – 0xFF) with LONG-NAME “OBD Supported ID <PIDnumber>” and SHORT-NAME “OBDSupportedID<PIDnumber>”. The corresponding TABLE-STRUCT shall also be used when decoding the response message (see below).

The value of the TABLE-KEY shall be the value of the PID.

END-OF-PDU-FIELD “PIDsSupportedServiceS02” reference an ODX STRUCTURE with SHORT-NAME “PIDsFramesSupported”. “PIDsFramesSupported” shall have two parameters: The first parameter is of type TABLE-KEY with SHORT-NAME “PIDs”. “PIDs” references an object of type TABLE with SHORT-NAME “PIDsSupported”. The second parameter is of type VALUE and its SHORT-NAME shall be “FrameNo”. The application using this information may, then, store the requested frame number in this parameter before sending the request.

Figure 8 shows a graphical representation of the ODX service definition structure taking service 0x01 as an example. The diagram also includes the data elements used in the response definition.

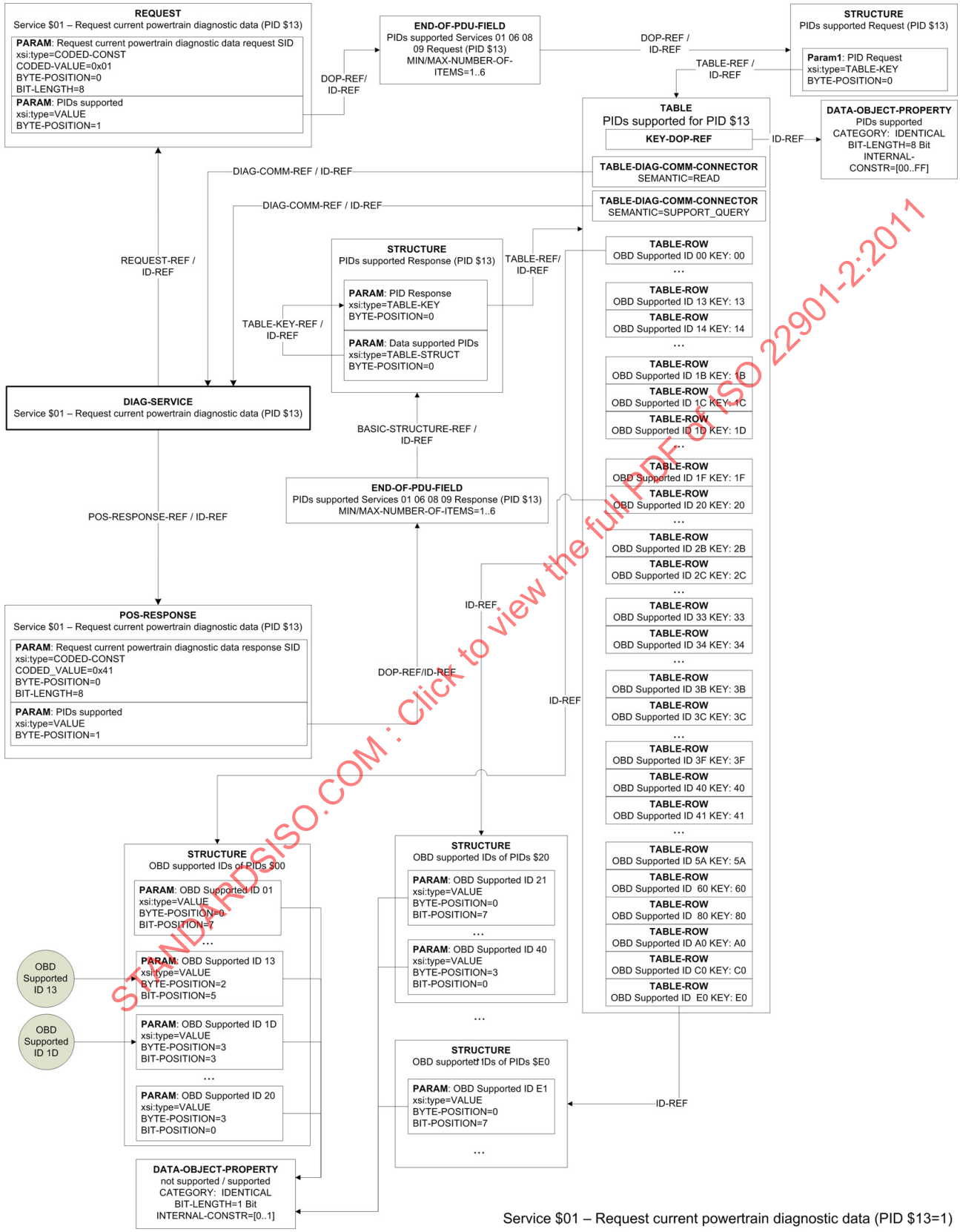


Figure 8 — Service 0x01 — Request current powertrain diagnostic data (PID 0x13=1)

In an ECU implementation, only either PID 0x13 or PID 0x1D is supported as mandated by ISO 15031-5. Different response messages are defined for each case. Therefore, Service 0x01 is described twice in the ODX data description. The two services differ in the POS-RESPONSE because different TABLEs are referenced. Figure 8 illustrates the situation for support of PID 0x13. The service 0x01 for support of PID 0x1D is built up accordingly. It is up to the diagnostic application to find out which of the two PIDs is supported by using any of the two DIAG-SERVICEs requesting PID 0x00. The ECU response contains the information regarding which of the two PIDs (0x13 or 0x1D) is supported. Depending on the response, it is necessary that either DIAG-SERVICE “Service 0x01 – Request current powertrain diagnostic data (PID 0x13)” or “Service 0x01 – Request current powertrain diagnostic data (PID 0x1D)” be used to request PIDs from the ECU.

9.3.3.2 Service-specific request parameters for services 0x03 0x04 and 0x07

For services 0x03, 0x04 and 0x07, no service-specific request parameters are defined because these services take no further parameters beside the SID.

9.3.3.3 Service-specific response parameters for services 0x01, 0x08 and 0x09

The response message shall use the same data structures as the corresponding request message, i.e. the TABLE with SHORT-NAME “PIDsSupported” (see above). The second response parameter returns the requested PID. In the ODX representation, parameters 2 and above are collects in an END-OF-PDU-FIELD “PIDsSupported01060809Response” parameter similar to the request message. The ODX STRUCTURE corresponding to “PIDsSupported01060809Response” shall reference the TABLE-KEY of TABLE “PIDsSupported” in its first parameter and TABLE-STRUCT of TABLE “PIDsSupported” in its second parameter.

9.3.3.4 Service-specific response parameters for services 0x02

The response message shall have two parameters. The naming of parameter one follows the rule for Service ID explained earlier. The second response parameter shall be an END-OF-PDU-FIELD.

9.3.3.5 Service-specific response parameters for services 0x03 and 0x07

The response message shall have three parameters. The naming of parameter one follows the rule for Service ID explained earlier. The second response parameter, called “NoDTCS” of type VALUE, returns the number of stored DTCs. The third parameter shall be of type END-OF-PDU-FIELD called “StoredDTCS”.

9.3.3.6 Service-specific response parameters for services 0x04

The response message shall have one parameter. The naming of parameter one follows the rule for Service ID explained earlier. For a positive response, there are no further parameters defined in ISO 15031-5.

9.3.3.7 Service-specific response parameters for services 0x06

The response message shall have three parameters. The naming of parameter one follows the rule for Service ID explained earlier. The second response parameter shall be of type END-OF-PDU-FIELD.

9.4 Conversion of PIDs to ODX

9.4.1 Properties of PIDs

Rules for authoring of PIDs:

— Rule 0: PIDs are described in ISO 15031-5 in tables with columns PID, Description, Data byte, Scaling/bit and “External test equipment SI (Metric) / English display” and optionally min. and max. value. A value is a line in a table where a bit or a bit range is specified in column “Data byte”.

— Rule 1: The value(s) of the same PID are mapped to one ODX STRUCTURE.

For the purpose of mapping the values of a PID to ODX, these values are categorized in a number of different types. See Table 12.

Table 12 — Value types

Value type	Description	Example value
Linear	The value describes a continuous numeric range. In ISO 15031-5 typically min. value, max. value, scaling and a unit are given.	PID 0x06 Byte A
Boolean	The value refers to a single bit, such as an attribute like ON/OFF or YES/NO.	PID 0x01 Byte A Bit 7
BitSelect	The values cover a range of bits of which at most one is set. The result depends on which bit is set.	PID 0x03 Byte A
BitSet	The values cover a range of bits. Each bit represents a property to be present or absent.	PID 0x13
Number2Text	The value covers a range of bits. The bits are interpreted as a number and each number translates to some text.	PID 0x1C
Number	The value covers a range of bits. The bits are directly interpreted as a number without conversion.	PID 0x01 Byte A Bits 0 - 6
DTC	The value covers 2 bytes and is interpreted as a diagnostic trouble code.	PID 0x02

9.4.2 Rules for type “Linear”

“Linear” rules:

— Rule L1: Each value is mapped to a parameter of type VALUE.

— Rule L2: The LONG-NAME of the parameter shall be the content of the column “Description” of the corresponding value.

— Rule L3: The attribute TI of the LONG-NAME shall be “OBD_PID<#>_<SHORT-NAME>”.

— Rule L4: The SHORT-NAME shall be generated from the LONG-NAME by the following algorithm: All characters disallowed in an ODX SHORT-NAME shall be removed. Wherever there is a blank in the LONG-NAME, the next character in the SHORT-NAME shall be converted to upper case.

— Rule L5: The BYTE-POSITION depends on the “Data byte” in the table. For byte A, B, … the BYTE-POSITION shall be 0, 1, …, respectively.

— Rule L6: The parameter shall reference a DOP with DIAG-CODED-TYPE of type STANDARD-LENGTH-TYPE. Its BIT-LENGTH shall be set according to the “Data byte” column, typically 8 or 16. The COMPU-METHOD shall be of CATEGORY “LINEAR”. The COMPU-RATIONAL-COEFFS shall be set according to the “Scaling/bit” column. A UNIT shall be referenced that describes the unit in the max value column. The physical type shall be represented as A_UINT32 if only positive integer values can occur, as A_INT32 if integers can occur and as A_FLOAT otherwise. The PRECISION shall be set according to the metric value in the column “External test equipment SI (Metric) / English display”.

9.4.3 Rules for type “Boolean”

“Boolean” rules:

— Rule O1: Each value is mapped to a parameter of type VALUE.

— Rule O2: The LONG-NAME of the parameter shall be the content of the column “Description” of the corresponding value.

— Rule O3: The attribute TI of the LONG-NAME shall be “OBD_PID<#>_<SHORT-NAME>”.

— Rule O4: The SHORT-NAME shall be generated from the LONG-NAME by the following algorithm: All characters disallowed in an ODX SHORT-NAME shall be removed. Wherever there is a blank in the LONG-NAME, the next character in the SHORT-NAME shall be converted to upper case.

— Rule O5: The BYTE-POSITION depends on the “Data byte” in the table. For byte A, B, … the BYTE-POSITION shall be 0, 1, …, respectively.

— Rule O6: The BIT-POSITION shall be determined by the value in “Data byte”.

— Rule O7: The parameter shall reference a DOP with DIAG-CODED-TYPE of type STANDARD-LENGTH-TYPE. Its BIT-LENGTH shall be set to 1. The COMPU-METHOD shall be of CATEGORY “TEXTTABLE”. The VT values of this TEXTTABLE shall be set according to the entries in column “External test equipment SI (Metric) / English display”. LOWER-LIMIT shall be set according to the entries in column “Scaling/bit”.

9.4.4 Rules for type “BitSelect”

“BitSelect” rules:

— Rule B1: All values are mapped to a single parameter of type VALUE.

— Rule B2: The LONG-NAME of the parameter shall be the content of the column “Description” of the corresponding Byte (e.g. “Fuel system 1 status:” for PID 0x03 Byte A).

— Rule B3: The attribute TI of the LONG-NAME shall be “OBD_PID<#>_<SHORT-NAME>”.

— Rule B4: The SHORT-NAME shall be generated from the LONG-NAME by the following algorithm: All characters disallowed in an ODX SHORT-NAME shall be removed. Wherever there is a blank in the LONG-NAME, the next character in the SHORT-NAME shall be converted to upper case.

— Rule B5: The BYTE-POSITION depends on the “Data byte” in the table. For byte A, B, … the BYTE-POSITION shall be 0, 1, …, respectively.

— Rule B6: The parameter shall reference a DOP with DIAG-CODED-TYPE of type STANDARD-LENGTH-TYPE. Its BIT-LENGTH shall be set to 8. The COMPU-METHOD shall be of CATEGORY “TEXTTABLE”. The TEXTTABLE shall contain one entry for each non-reserved value. LOWER-LIMIT shall be set to 2 to the power of bit position, i.e. 1, 2, 4, … The VT values of this TEXTTABLE shall be set according to the entries in column “External test equipment SI (Metric) / English display”. An additional entry shall be created with LOWER-LIMIT 0 and VT equal to “-” if it is valid that no bit is set.

9.4.5 Rules for type “BitSet”

“BitSet” rules:

— Rule S1: Each value is mapped to a parameter of type VALUE.

— Rule S2: The LONG-NAME of each parameter shall be the concatenation of the content of the column “Description” of the corresponding Byte (e.g. “Location of Oxygen Sensors” for PID 0x13 Bit 0 and thus Byte A) and the deterministic part of the content of the column “Scaling/bit” of the corresponding value (e.g. “Bank 1 Sensor 1” for PID 0x13 Bit 0) separated by space.

— Rule S3: The attribute TI of the LONG-NAME shall be “OBD_PID<#>_<SHORT-NAME>”.

— Rule S4: The SHORT-NAME shall be generated from the LONG-NAME by the following algorithm: All characters disallowed in an ODX SHORT-NAME shall be removed. Wherever there is a blank in the LONG-NAME, the next character in the SHORT-NAME shall be converted to upper case.

— Rule S5: The BYTE-POSITION depends on the “Data byte” in the table. For byte A, B, … the BYTE-POSITION shall be 0, 1, …, respectively.

— Rule S6: The BIT-POSITION shall be determined by the value in “Data byte”.

— Rule S7: The parameter shall reference a DOP with DIAG-CODED-TYPE of type STANDARD-LENGTH-TYPE. Its BIT-LENGTH shall be set to 1. The COMPU-METHOD shall be of CATEGORY “TEXTTABLE”. 1 shall be mapped to the text shown in column “External test equipment SI (Metric) / English display”. 0 shall be mapped to the empty text.

9.4.6 Rules for type “Number2Text”

“Number2Text” rules:

— Rule X1: The value is mapped to a parameter of type VALUE.

— Rule X2: The LONG-NAME of the parameter shall be the content of the column “Description” of the corresponding byte (e.g. “OBD requirements to which vehicle is designed”).

— Rule X3: The attribute TI of the LONG-NAME shall be “OBD_PID<#>_<SHORT-NAME>”.

— Rule X4: The SHORT-NAME shall be generated from the LONG-NAME by the following algorithm: All characters disallowed in an ODX SHORT-NAME shall be removed. Wherever there is a blank in the LONG-NAME, the next character in the SHORT-NAME shall be converted to upper case.

— Rule X5: The BYTE-POSITION depends on the referenced data byte. For byte A, B, … the BYTE-POSITION shall be 0, 1, …, respectively.

— Rule X6: The parameter shall reference a DOP with DIAG-CODED-TYPE of type STANDARD-LENGTH-TYPE. Its BIT-LENGTH shall be set to the number of bits covered by the value, typically 8. The COMPU-METHOD shall be of CATEGORY “TEXTTABLE”. The entries in column “Data byte” shall be mapped to the texts in column “External test equipment SI (Metric) / English display”.

NOTE ODX uses the decimal representation of numbers.

9.4.7 Rules for type “Number”

“Number” rules:

— Rule N1: The value is mapped to a parameter of type VALUE.

— Rule N2: The LONG-NAME of the parameter shall be the content of the column “Description” of the corresponding value.

— Rule N3: The attribute TI of the LONG-NAME shall be “OBD_PID<#>_<SHORT-NAME>”.

— Rule N4: The SHORT-NAME shall be generated from the LONG-NAME by the following algorithm: All characters disallowed in an ODX SHORT-NAME shall be removed. Wherever there is a blank in the LONG-NAME, the next character in the SHORT-NAME shall be converted to upper case.

— Rule N5: The BYTE-POSITION depends on the “Data byte” in the table. For byte A, B, … the BYTE-POSITION shall be 0, 1, …, respectively.

— Rule N6: If the column “Data byte” describes a bit range, the BIT-POSITION shall be set to the lower value. If zero, the BIT-POSITION may be omitted.

— Rule N7: The parameter shall reference a DOP with DIAG-CODED-TYPE of type STANDARD-LENGTH-TYPE. Its BIT-LENGTH shall be set according to the “Data byte” column. The COMPU-METHOD shall be of CATEGORY “IDENTICAL”. The DISPLAY-RADIX shall be set according to the entry in column “Scaling/bit”.

9.4.8 Rules for type “DTC”

“DTC” rules:

— Rule D1: The value is mapped to a parameter of type VALUE.

— Rule D2: The LONG-NAME of the parameter shall be the content of the column “Description” of the corresponding byte (e.g. “OBD requirements to which vehicle is designed”).

— Rule D3: The attribute TI of the LONG-NAME shall be “OBD_PID<#>_<SHORT-NAME>”.

— Rule D4: The SHORT-NAME shall be generated from the LONG-NAME by the following algorithm: All characters disallowed in an ODX SHORT-NAME shall be removed. Wherever there is a blank in the LONG-NAME, the next character in the SHORT-NAME shall be converted to upper case.

— Rule D5: The BYTE-POSITION depends on the referenced data byte. For byte A, B, … the BYTE-POSITION shall be 0, 1, …,respectively.

— Rule D6: The parameter shall reference a DTC-DOP with SHORT-NAME “ObdDtcs”.

9.5 Conversion of DTCs to ODX

The data for diagnostic trouble codes are defined in the digital annexes of ISO 15031-6. This part of ISO 22901 describes rules to derive concrete ODX data from these annexes.

All DTC elements are to be defined in the ECU-SHARED-DATA diagnostic layer with short name `ObdIIDopsDtcDeclarations` and ID `ES_ObdIIDopsDtcDeclarations`. It may contain any LONG-NAME. It shall contain a DIAG-DICTIONARY-SPEC. In addition, it may contain only ADMIN-DATA, COMPANY-DATAS, and SDGS. It shall not contain any other element.

The DIAG-DICTIONARY-SPEC shall contain DTC-DOPS and may contain ADMIN-DATA, and SDGS. It shall not contain any other element.

ISO 15031-6 defines four groups of DTCs: Powertrain (their names start with “P”), Chassis (their names start with “C”), Body (their names start with “B”), and Network (their name starts with “U”). The DIAG-DICTIONARY-SPEC contains a single DTC-DOP for each of these four groups. Each DTC-DOP lists all the DTCs defined for that group in ISO 15031-6.

Their IDs and SHORT-NAMEs shall be defined as in Table 13.

Table 13 — SHORT-NAME and ID of DTC-DOPs

Marker	Group	SHORT-NAME of DTC-DOP	ID attribute of DTC-DOP
P	Powertrain	ObdDtcsPowertrain	ES_ObdIIDopsDtcDeclarations.DOP_ObdDtcsPowertrain
C	Chassis	ObdDtcsChassis	ES_ObdIIDopsDtcDeclarations.DOP_ObdDtcsChassis
B	Body	ObdDtcsBody	ES_ObdIIDopsDtcDeclarations.DOP_ObdDtcsBody
U	Network	ObdDtcsNetwork	ES_ObdIIDopsDtcDeclarations.DOP_ObdDtcsNetwork

Each DTC-DOP shall contain the same DIAG-CODED-TYPE, PHYSICAL-TYPE, and COMPU-METHOD as shown below:

```
<DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
    <BIT-LENGTH>16</BIT-LENGTH>
</DIAG-CODED-TYPE>
<PHYSICAL-TYPE BASE-DATA-TYPE="A_UINT32" DISPLAY-RADIX="HEX"/>
<COMPU-METHOD>
    <CATEGORY>IDENTICAL</CATEGORY>
</COMPU-METHOD>
```

In addition, it shall contain a DTCS element that lists all the diagnostic trouble codes defined in the digital annex of ISO 15031-6 for the corresponding group (Powertrain, Chassis, Body, and Network). The digital annexes list the following fields for each DTC: Number (e.g. P000A), DTC Description (e.g. “A” Camshaft Position Slow Response), Location (e.g. Bank 1), and comment. Footnote marks like “a)” shall not be considered as content of the corresponding fields. These fields shall be transformed into ODX elements as follows.

— A DTC element shall exist for each DTC line in the annex. Its ID attribute shall have the prefix `ObdDtc_` immediately followed by the value of the Number field of the digital annex, for example: `<DTC ID="ObdDtc_P000A">`. It shall not contain the IS-TEMPORARY attribute but may contain the OID attribute with an arbitrary value. The element shall contain the elements SHORT-NAME, TROUBLE-CODE, DISPLAY-TROUBLE-CODE, and TEXT, in that order.

— The SHORT-NAME element shall contain the same value as the ID attribute of the DTC element itself, for example: `<SHORT-NAME>ObdDtc_P000A</SHORT-NAME>`.

— The TROUBLE-CODE element shall contain a decimal representation of the Number field's content. This number is defined by reverting the algorithm described in SAE J2012. The last four digits are interpreted as a hexadecimal number. Depending on the group, the hexadecimal value of Table 14 is added.

Table 14 — Value offset for different DTC groups

Group	Value to add
Powertrain	0x0000
Chassis	0x4000
Body	0x8000
Network	0xC000

The result as a decimal value is the content of the TROUBLE-CODE element. For example, P000A is converted as 0x000A+0x0000=0xA=10 and results in `<TROUBLE-CODE>10</TROUBLE-CODE>`, while B1001 is converted as 0x1001+0x8000=0x9001=36865 and, thus, `<TROUBLE-CODE>36865</TROUBLE-CODE>`.

— The DISPLAY-TROUBLE-CODE element shall contain the Number field's value, e.g. <DISPLAY-TROUBLE-CODE>P000A</DISPLAY-TROUBLE-CODE>.

— The TEXT element shall contain the concatenation of the Description field's and the Location field's value of the DTC separated by a space, e.g. <TEXT>“A” Camshaft Position Slow Response Bank 1</TEXT>. The value or existence of a TI attribute is not defined by this part of ISO 22901.

— If the value of the comment field is added, it shall be inserted as an XML comment after the TEXT element. If the comment contains “--”, it shall be replaced by a single dash “-”, as XML comments are not to contain “--”.

9.6 ODX samples of ISO 15031-5 services and authored data

9.6.1 General

In ODX, services with their request and their response(s) are defined with the ODX elements DIAG-SERVICE, request and POS-RESPONSE (more than one are allowed). Within the definition of the DIAG-SERVICE, the references to the request and the POS-RESPONSE are itemized.

The DIAG-SERVICE defines the LONG-NAME and the SHORT-NAME of the service itself as it is described in 9.2.2.3 and 9.2.2.4. The references to the appropriate request and to the appropriate response(s) is done with the ID-REF within the elements REQUEST-REF and POS-RESPONSE-REF. The ID-REFs are filled with the ID of the request and the POS-RESPONSE respectively.

The examples in 9.6.2 from ISO 15031-5 and ISO 22900-2 show the structure of these request and response messages.

9.6.2 Service 0x01 — Request current powertrain diagnostic data

9.6.2.1 ISO 22900-2 — DIAG-SERVICE 0x01 with LONG-NAME and SHORT-NAME sample

```
<DIAG-SERVICE
ID="ES_OBDIICommonServices.DS_Service01RequestCurrentPowertrainDiagnosticData
                ADDRESSING="FUNCTIONAL">
        <SHORT-NAME>Service01RequestCurrentPowertrainDiagnosticData</SHORT-NAME>
        <LONG-NAME>Service 0x01 - Request current powertrain diagnostic data</LONG-NAME>
        <REQUEST-REF ID-REF="REQ_Service01RequestCurrentPowertrainDiagnosticData"/>
        <POS-RESPONSE-REFS>
            <POS-RESPONSE-REF
                ID-REF="PRE_Service01RequestCurrentPowertrainDiagnosticData"/>
        </POS-RESPONSE-REFS>
</DIAG-SERVICE>
```

9.6.2.2 ISO 15031-5 — Read-supported PIDs

The purpose of this service is to allow access to current emission-related data values, including analogue inputs and outputs, digital inputs and outputs, and system status information. The request for information includes a parameter identification (PID) value that indicates to the on-board system the specific information requested.

Not all PIDs are applicable or supported by all systems. PID 0x00 is a bit-encoded value that indicates for each ECU which PIDs are supported. PID 0x00 indicates support for PIDs from 0x01 to 0x20. PID 0x20

indicates support for PIDs 0x21 through 0x40, etc. This is the same concept as for PIDs/OBD Monitor IDs/TIDs/InfoTypes support in Services 0x01, 0x02, 0x06, 0x08, 0x09.

The external test equipment requests supported PIDs (0x00, 0x20, 0x40, 0x60, 0x80, 0xA0) from the vehicle. ECU(s) shall respond to all supported ranges if requested. A range is defined as a block of 32 PIDs (e.g. range #1: PID 0x01-0x20). The ECU shall not respond to unsupported PID ranges unless subsequent ranges have a supported PID(s).

Table 15 defines the Request current powertrain supported PIDs request message.

Table 15 — Request current powertrain supported PIDs request message

Message direction:	External test equipment → All ECUs		
Message type:	Request		
Data byte	**Description (all PID values are in hexadecimal)**	**Byte value (Hex)**	**Mnemonic**
#1	Request current powertrain diagnostic data request SID	0x01	SIDRQ
#2	PID used to determine PID support for PIDs 0x01-0x20	0x00	PID
#3	PID used to determine PID support for PIDs 0x21-0x40	0x20	PID
#4	PID used to determine PID support for PIDs 0x41-0x60	0x40	PID
#5	PID used to determine PID support for PIDs 0x61-0x80	0x60	PID
#6	PID used to determine PID support for PIDs 0x81-0xA0	0x80	PID
#7	PID used to determine PID support for PIDs 0xA1-0xC0	0xA0	PID

Table 16 defines the ECU#1 response: Request current powertrain supported PIDs response message.

Table 16 — ECU#1 response: Request current powertrain supported PIDs response message

Message direction:	ECU#1 → External test equipment		
Message type:	Response		
Data byte	**Description (all PID values are in hexadecimal)**	**Byte value (Hex)**	**Mnemonic**
#1	Request current powertrain diagnostic data response SID	0x41	SIDPR
#2	PID requested	0x00	PID
#3	Data byte A, representing support for PIDs 0x01, 0x03-0x08	10111111b = 0xBF	DATA_A
#4	Data byte B, representing support for PIDs 0x09, 0x0B-0x10	10111111b = 0xBF	DATA_B
#5	Data byte C, representing support for PIDs 0x11, 0x13, 0x15	10101000b = 0xA8	DATA_C
#6	Data byte D, representing support for PIDs 0x19, 0x1C, 0x20	10010001b = 0x91	DATA_D
#7	PID requested	0x20	PID
#8	Data byte A, representing support for PID 0x21	10000000b = 0x80	DATA_A
#9	Data byte B, representing no support for PIDs 0x29-0x30	00000000b = 0x00	DATA_B
#10	Data byte C , representing no support for PIDs 0x31-0x38	00000000b = 0x00	DATA_C
#11	Data byte D, representing no support for PIDs 0x39-0x40	00000000b = 0x00	DATA_D

9.6.2.3 ISO 22900-2 — Read-supported PIDs ODX sample

The request and response message described in this subclause can be used within ISO 15765-4.

An ODX example of above request and response message is provided below.

```
<REQUEST ID="REQ_Service01RequestCurrentPowertrainDiagnosticData">
    <SHORT-NAME>Service01RequestCurrentPowertrainDiagnosticData</SHORT-NAME>
    <LONG-NAME>Service 0x01 - Request current powertrain diagnostic data</LONG-NAME>
    <PARAMS>
        <PARAM xsi:type="CODED-CONST">
            <SHORT-NAME>RequestCurrentPowertrainDiagnosticDataRequestSID</SHORT-NAME>
            <LONG-NAME>Request current powertrain diagnostic data request SID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <CODED-VALUE>1</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
                <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
        <PARAM xsi:type="VALUE">
            <SHORT-NAME>PIDsSupported</SHORT-NAME>
            <LONG-NAME>PIDs supported</LONG-NAME>
            <BYTE-POSITION>1</BYTE-POSITION>
            <DOP-REF ID-REF="EOPDUF_PIDsSupportedServices01060809Request"/>
        </PARAM>
    </PARAMS>
</REQUEST>
```

The PARAM “PIDsSupported” of the request “Service01RequestCurrentPowertrainDiagnosticData” with the ID=“REQ_Service01RequestCurrentPowertrainDiagnosticData” allows the user to define up to six requesting PIDs (0x00, 0x20, 0x40, .. 0xC0) to find out the appropriate supported PIDs belonging to the requesting PIDs.

The requesting PIDs are defined in the TABLE “PIDsSupported”. The reference to this TABLE is established via the PARAM “PIDsSupported”.

```
<TABLE ID="TAB_PIDsSupported">
    <SHORT-NAME>PIDsSupported</SHORT-NAME>
    <LONG-NAME>PIDs supported</LONG-NAME>
    <KEY-DOP-REF ID-REF="DOP_PIDsSupported"/>
    <TABLE-ROW ID="TABROW_OBDSupportedID00">
        <SHORT-NAME>OBDSupportedID00</SHORT-NAME>
        <LONG-NAME>OBD Supported ID 00</LONG-NAME>
        <KEY>0</KEY>
        <STRUCTURE-REF ID-REF="STRUC_OBDSupportedIDsOfPIDs00And20ToE0"/>
    </TABLE-ROW>

    …

    <TABLE-ROW ID="TABROW_OBDSupportedID20">
        <SHORT-NAME>ODXSupportedID20</SHORT-NAME>
        <LONG-NAME>ODX Supported ID 20</LONG-NAME>
        <KEY>32</KEY>
        <STRUCTURE-REF ID-REF="STRUC_OBDSupportedIDsOfPIDs00And20ToE0"/>
    </TABLE-ROW>

    …

    <TABLE-ROW ID="TABROW_OBDSupportedIDE0">
        <SHORT-NAME>ODXSupportedIDE0</SHORT-NAME>
        <LONG-NAME>ODX Supported ID E0</LONG-NAME>
        <KEY>224</KEY>
        <STRUCTURE-REF ID-REF="STRUC_OBDSupportedIDsOfPIDs00And20ToE0"/>
    </TABLE-ROW>
</TABLE>
```

The structure of the result of the above request is described with the following POS-RESPONSE.

```
<POS-RESPONSE ID="PRE_Service01RequestCurrentPowertrainDiagnosticData">
    <SHORT-NAME>Service01RequestCurrentPowertrainDiagnosticData</SHORT-NAME>
    <LONG-NAME>Service 0x01 – Request current powertrain diagnostic data</LONG-NAME>
    <PARAMS>
        <PARAM xsi:type="CODED-CONST" SEMANTIC="SERVICE-ID">
            <SHORT-NAME>RequestCurrentPowertrainDiagnosticDataResponseSID</SHORT-NAME>
            <LONG-NAME>Request current powertrain diagnostic data response SID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <CODED-VALUE>65</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
                <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
            <PARAM xsi:type="VALUE">
                <SHORT-NAME>PIDsSupported</SHORT-NAME>
            <LONG-NAME>PIDs supported</LONG-NAME>
            <BYTE-POSITION>1</BYTE-POSITION>
            <DOP-REF ID-REF="EOPDUF_PIDsSupportedServices01060809Response"/>
        </PARAM>
    </PARAMS>
</POS-RESPONSE>
```

The resulting PIDs are defined in the TABLE “PIDsSupported”. The reference to this TABLE is established via the PARAM “PIDsSupported”.

The STRUCTURE “OBD supported IDs of PIDs 0x00 and 0x20 to 0xE0” is used for interpreting which PIDs inside a PID-Range (0x00, 0x20, … 0xE0) are supported.

```
<STRUCTURE ID="STRUC_OBDSupportedIDsOfPIDs00And20ToE0">
    <SHORT-NAME>OBDSupportedIDsOfPIDs00And20ToE0</SHORT-NAME>
    <LONG-NAME>OBD supported IDs of PIDs 0x00 and 0x20 to 0xE0</LONG-NAME>
    <PARAMS>
        <PARAM xsi:type="VALUE">
            <SHORT-NAME>OBDSupportedID01</SHORT-NAME>
            <LONG-NAME>OBD Supported ID 01</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <BIT-POSITION>7</BIT-POSITION>
            <DOP-REF ID-REF="DOP_NotSupportedSupported"/>
        </PARAM>
        <PARAM xsi:type="VALUE">
            <SHORT-NAME>OBDSupportedID02</SHORT-NAME>
            <LONG-NAME>OBD Supported ID 02</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <BIT-POSITION>6</BIT-POSITION>
            <DOP-REF ID-REF="DOP_NotSupportedSupported"/>
        </PARAM>
        …
```

```
            <PARAM xsi:type="VALUE">
                <SHORT-NAME>OBDSupportedID1F</SHORT-NAME>
                <LONG-NAME>OBD Supported ID 1F</LONG-NAME>
                <BYTE-POSITION>3</BYTE-POSITION>
                <BIT-POSITION>1</BIT-POSITION>
                <DOP-REF ID-REF="DOP_NotSupportedSupported"/>
            </PARAM>
            <PARAM xsi:type="VALUE">
                <SHORT-NAME>OBDSupportedID20</SHORT-NAME>
                <LONG-NAME>OBD Supported ID 20</LONG-NAME>
                <BYTE-POSITION>3</BYTE-POSITION>
                <BIT-POSITION>0</BIT-POSITION>
                <DOP-REF ID-REF="DOP_NotSupportedSupported"/>
            </PARAM>
            ...
        </PARAMS>
    </STRUCTURE>
```

9.6.2.4 ISO 15031-5 — Request multiple PIDs from vehicle

The request and response message described in this subclause can be used within ISO 15765-4.

The external test equipment requests a combination of a maximum of six (6) PIDs in one request message to gain best performance of displaying current data.

— PID 0x15: Bank 1 - Sensor 2 PID is supported by ECU #1;
— PID 0x01: Number of emission-related DTCs and MIL status PID is supported by ECU #1 and #2;
— PID 0x05: Engine coolant temperature PID is supported by ECU #1;
— PID 0x03: Fuel system 1 status PID is supported by ECU #1;
— PID 0x0C: Engine speed PID is supported by ECU #1; and
— PID 0x0D: Vehicle speed PID is supported by ECU #2.

Table 17 defines the Request current powertrain diagnostic data request message.

Table 17 — Request current powertrain diagnostic data request message

Message direction:	External test equipment → All ECUs		
Message type:	Request		
Data byte	**Description (all PID values are in hexadecimal)**	**Byte value (Hex)**	**Mnemonic**
#1	Request current powertrain diagnostic data request SID	01	SIDRQ
#2	PID: Bank 1 - Sensor 2	15	PID(15)
#3	PID: Number of emission-related DTCs and MIL status	01	PID(01)
#4	PID: Engine coolant temperature	05	PID(05)
#5	PID: Fuel system 1 status	03	PID(03)
#6	PID: Engine speed	0C	PID(0C)
#7	PID: Vehicle speed	0D	PID(0D)

Table 18 defines the ECU#1 response: Request current powertrain diagnostic data response message.

Table 18 — ECU#1 response: Request current powertrain diagnostic data response message

Message direction:	ECU#1 → External test equipment
Message type:	Response

Data byte	Description (all PID values are in hexadecimal)	Byte value (Hex)	Mnemonic
#1	Request current powertrain diagnostic data response SID	41	SIDPR
#2	PID: Engine coolant temperature	05	PID(05)
#3	Data byte A	6E	DATA(A)
#4	PID: Number of emission-related DTCs and MIL status	01	PID(01)
#5	MIL: ON; Number of emission-related DTCs: 03	83	DATA(A)
#6	Misfire -, Fuel system -, Comprehensive monitoring	33	DATA(B)
#7	Catalyst -, Heated catalyst -, …, monitoring supported	FF	DATA(C)
#8	Catalyst -, Heated catalyst -, …, monitoring test complete/not complete	63	DATA(D)
#9	PID: Bank 1 - Sensor 2	15	PID(15)
#10	Bank 2 - Sensor 2: 0,8 V	A0	DATA(A)
#11	Bank 2 - Sensor 2: 93,7 %	78	DATA(B)
#12	PID: Engine speed	0C	PID(0C)
#13	Data byte A: 667 r/min	0A	DATA(A)
#14	Data byte B: 667 r/min	6B	DATA(B)
#15	PID: Fuel system 1 status	03	PID(03)
#16	Data byte A: Closed loop - using oxygen sensor(s) as feedback for fuel control	02	DATA(A)
#17	Data byte B	00	DATA(B)

Table 19 defines the ECU#2 response: Request current powertrain diagnostic data response message.

Table 19 — ECU#2 response: Request current powertrain diagnostic data response message

Message direction:	ECU#2 → External test equipment
Message type:	Response

Data byte	Description (all PID values are in hexadecimal)	Byte value (Hex)	Mnemonic
#1	Request current powertrain diagnostic data response SID	41	SIDPR
#2	PID: Vehicle speed	0D	PID(0D)
#3	Data byte A	23	DATA(A)
#4	PID: Number of emission-related DTCs and MIL status	01	PID(01)
#5	MIL: OFF; Number of emission-related DTCs: 01	01	DATA(A)
#6	Comprehensive monitoring: supported, test complete	44	DATA(B)
#7	Catalyst -, Heated catalyst -, …, monitoring supported	00	DATA(C)
#8	Catalyst -, Heated catalyst -, …, monitoring test complete/not complete	00	DATA(D)

9.6.2.5 This part of ISO 22901 — Request multiple PIDs from vehicle ODX sample

If, for example, PID 0x13 is supported, the following service should be executed to request PIDs from the vehicle. The relationship between its request, response, table and table rows is defined as in 9.6.2.3.

```
<DIAG-SERVICE
ID="ES_OBDIICommonServices.DS_Service01RequestCurrentPowertrainDiagnosticDataPID13"
ADDRESSING="FUNCTIONAL">
    <SHORT-NAME>Service01RequestCurrentPowertrainDiagnosticDataPID13</SHORT-NAME>
    <LONG-NAME>Service $01 - Request current powertrain diagnostic data (PID $13)</LONG-
NAME>
    <REQUEST-REF ID-REF="REQ_Service01RequestCurrentPowertrainDiagnosticDataPID13"/>
    <POS-RESPONSE-REFS>
        <POS-RESPONSE-REF ID-
REF="PRE_Service01RequestCurrentPowertrainDiagnosticDataPID13"/>
    </POS-RESPONSE-REFS>
</DIAG-SERVICE>
```

Examples of the STRUCTUREs reference by the TABLE-ROW elements can be found in 9.6.10.

9.6.3 Service 0x02 — Request powertrain freeze frame data

9.6.3.1 ISO 15031-5 — Request powertrain freeze frame with PID 0x02

Now the external test equipment requests PID 0x02 of freeze frame 0x00 from the vehicle. Since the ECU#2 (TCM) doesn't store a freeze frame data record, only the ECU#1 (ECM) will send a response message. In this example, the freeze frame data are stored based on a DTC P0130 occurrence. The parameter value of PID 0x02 “DTC that caused required freeze frame data storage” is set to the DTC P0130.

Table 20 defines the Request powertrain freeze frame DTC that caused data storage request message.

Table 20 — Request powertrain freeze frame DTC that caused data storage request message

Message direction:	External test equipment → All ECUs
Message type:	Request

Data byte	Description (all values are in hexadecimal)	Byte value	Mnemonic
#1	Request powertrain freeze frame data request SID	0x02	SIDRQ
#2	PID: DTC that caused required freeze frame data storage	0x02	PID
#3	Frame #	0x00	FRNO

Table 21 defines the request powertrain freeze frame DTC that caused data storage response message.

Table 21 — Request powertrain freeze frame DTC that caused data storage response message

Message direction:	ECU#1 → External test equipment
Message type:	Response

Data byte	Description (all values are in hexadecimal)	Byte value	Mnemonic
#1	Request powertrain freeze frame data response SID	0x42	SIDRQ
#2	PID: DTC that caused required freeze frame data storage	0x02	PID
#3	Frame #	0x00	FRNO
#4	DTC High Byte of P0130	0x01	DATA_A
#5	DTC Low Byte of P0130	0x30	DATA_B

NOTE ECU#2 does not store freeze frame data and, therefore, does not send a response message.

9.6.3.2 ISO 22900-2 — Request powertrain freeze frame PID 0x02 in ODX sample

```
<DIAG-SERVICE ID="ES_OBDIICommonServices.DS_Service02RequestPowertrainFreezeFrameData"
              ADDRESSING="FUNCTIONAL">
    <SHORT-NAME>Service02RequestPowertrainFreezeFrameData</SHORT-NAME>
    <LONG-NAME>Service 0x02 – Request powertrain freeze frame data</LONG-NAME>
    <REQUEST-REF ID-REF="REQ_Service02RequestPowertrainFreezeFrameData"/>
    <POS-RESPONSE-REFS>
        <POS-RESPONSE-REF ID-REF="PRE_Service02RequestPowertrainFreezeFrameData"/>
    </POS-RESPONSE-REFS>
</DIAG-SERVICE>

<REQUEST ID="REQ_Service02RequestPowertrainFreezeFrameData">
    <SHORT-NAME>Service02RequestPowertrainFreezeFrameData</SHORT-NAME>
    <LONG-NAME>Service 0x02 – Request powertrain freeze frame data</LONG-NAME>
    <PARAMS>
        <PARAM xsi:type="CODED-CONST" SEMANTIC="SERVICE-ID">
            <SHORT-NAME>RequestPowertrainFreezeFrameDataRequestSID</SHORT-NAME>
            <LONG-NAME>Request powertrain freeze frame data request SID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <CODED-VALUE>2</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
                <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
        ... <!-- field with PIDs and frame numbers -->
    </PARAMS>
</REQUEST>
```

```
<POS-RESPONSE ID="PRE_Service02RequestPowertrainFreezeFrameData">
    <SHORT-NAME>Service02RequestPowertrainFreezeFrameData</SHORT-NAME>
    <LONG-NAME>Service 0x02 – Request powertrain freeze frame data</LONG-NAME>
    <PARAMS>
        <PARAM xsi:type="CODED-CONST" SEMANTIC="SERVICE-ID">
            <SHORT-NAME>RequestPowertrainFreezeFrameDataResponseSID</SHORT-NAME>
            <LONG-NAME>Request powertrain freeze frame data response SID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <CODED-VALUE>66</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
               <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
        <PARAM>
            <!-- Get the freeze frame data -->
        </PARAM>
    </PARAMS>
</POS-RESPONSE>
```

9.6.3.3 ISO 15031-5 — Request powertrain freeze frame with multiple PIDs

The external test equipment requests the parameter value of PID 0x0C “Engine Speed”, PID 0x05 “Engine coolant temperature”, and PID 0x04 “Load” stored in the freeze frame.

Table 22 defines the request powertrain freeze frame data request message.

Table 22 — Request powertrain freeze frame data request message

Message direction:	External test equipment → All ECUs
Message type:	Request

Data byte	Description (all values are in hexadecimal)	Byte value	Mnemonic
#1	Request powertrain freeze frame data request SID	0x02	SIDRQ
#2	PID: Engine Speed	0x0C	PID
#3	Frame #	0x00	FRNO
#4	PID: Engine coolant temperature	0x05	PID
#5	Frame #	0x00	FRNO
#4	PID: Load	0x04	PID
#5	Frame #	0x00	FRNO

Table 23 defines the Request powertrain freeze frame data response message.

Table 23 — Request powertrain freeze frame data response message

Message direction:	ECU#1 → External test equipment
Message type:	Response

Data byte	Description (all values are in hexadecimal)	Byte value	Mnemonic
#1	Request powertrain freeze frame data response SID	0x42	SIDRQ
#2	PID: Engine Speed	0x0C	PID
#3	Frame #	0x00	FRNO
#4	High Byte: Engine Speed: 2 080 r/min	0x20	DATA_A
#5	Low Byte: Engine Speed: 2 080 r/min	0x80	DATA_B
#6	PID: Load	0x04	PID
#7	Frame #	0x00	FRNO
#8	Load: 50,2 %	0x80	DATA_A
#9	PID: Engine coolant temperature	0x05	PID
#10	Frame #	0x00	FRNO
#11	Engine coolant temperature: 0 °C	0x28	DATA_A

9.6.3.4 ISO 22900-2 — Request powertrain freeze frame with multiple PIDs in ODX sample

Requesting a freeze frame for a single PID or multiple PIDs does not impact the ODX data. Therefore, the examples shown are identical to 9.6.3.2.

```
<DIAG-SERVICE ID="ES_OBDIICommonServices.DS_Service02RequestPowertrainFreezeFrameData"
              ADDRESSING="FUNCTIONAL">
    <SHORT-NAME>Service02RequestPowertrainFreezeFrameData</SHORT-NAME>
    <LONG-NAME>Service 0x02 - Request powertrain freeze frame data</LONG-NAME>
    <REQUEST-REF ID-REF="REQ_Service02RequestPowertrainFreezeFrameData"/>
    <POS-RESPONSE-REFS>
        <POS-RESPONSE-REF ID-REF="Pre_Service02RequestPowertrainFreezeFrameData"/>
    </POS-RESPONSE-REFS>
</DIAG-SERVICE>

<REQUEST ID="REQ_Service02RequestPowertrainFreezeFrameData">
    <SHORT-NAME>Service02RequestPowertrainFreezeFrameData</SHORT-NAME>
    <LONG-NAME>Service 0x02 - Request powertrain freeze frame data</LONG-NAME>
    <PARAMS>
        <PARAM xsi:type="CODED-CONST" SEMANTIC="SERVICE-ID">
            <SHORT-NAME>RequestPowertrainFreezeFrameDataRequestSID</SHORT-NAME>
            <LONG-NAME>Request powertrain freeze frame data request SID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <CODED-VALUE>2</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
                <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
        ... <!-- field with PIDs and frame numbers -->
    </PARAMS>
</REQUEST>
```

```
<POS-RESPONSE ID="PRE_Service02RequestPowertrainFreezeFrameData">
    <SHORT-NAME>Service02RequestPowertrainFreezeFrameData</SHORT-NAME>
    <LONG-NAME>Service 0x02 – Request powertrain freeze frame data</LONG-NAME>
    <PARAMS>
        <PARAM xsi:type="CODED-CONST" SEMANTIC="SERVICE-ID">
            <SHORT-NAME>RequestPowertrainFreezeFrameDataResponseSID</SHORT-NAME>
            <LONG-NAME>Request powertrain freeze frame data response SID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <CODED-VALUE>66</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
               <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
        <PARAM>
            <!-- Get the freeze frame data -->
        </PARAM>
    </PARAMS>
</POS-RESPONSE>
```

9.6.4 Service 0x03, 0x07 and 0x0A — Request emissions-related DTC

9.6.4.1 General

Service 0x03, 0x07, and 0x0A are completely identical with regard to their structure. They shall be modelled alike in ODX. In the following, the requested modelling structure is shown in detail for service 0x03. Services 0x07 and 0x0A shall be modelled by mainly replacing 0x03 with 0x07 or 0x0A and adopting other ODX elements according to the rules defined in 9.2.

9.6.4.2 ISO 22900-2 — DIAG-SERVICE 0x03 with LONG-NAME and SHORT-NAME sample

```
<DIAG-SERVICE
ID="ES_OBDIICommonServices.DS_Service03RequestEmissionRelatedDiagnosticTroubleCodes"
              ADDRESSING="FUNCTIONAL">
    <SHORT-NAME>Service03RequestEmissionRelatedDiagnosticTroubleCodes</SHORT-NAME>
    <LONG-NAME>Service 0x03 – Request emission-related diagnostic trouble codes</LONG-
NAME>
    <REQUEST-REF ID-REF="REQ_Service03RequestEmissionRelatedDiagnosticTroubleCodes"/>
    <POS-RESPONSE-REFS>
        <POS-RESPONSE-REF
            ID-REF="PRE_Service03RequestEmissionRelatedDiagnosticTroubleCodes"/>
        </POS-RESPONSE-REFS>
</DIAG-SERVICE>
```

9.6.4.3 ISO 15031-5 service 0x03, 0x07, 0x0A — Request emission-related DTCs example

The request and response message described in this subclause can be used within ISO 15765-4.

The example below shows how the “Request emission-related DTCs” service shall be implemented. The external test equipment requests emission-related DTCs from the vehicle. The ECU#1 (ECM) has six (6) DTCs stored, the ECU #2 (TCM) has one (1) DTC stored, and the ECU #3 (ABS/Traction Control) has no DTC stored.

— ECU #1 (ECM): P0143, P0196, P0234, P02CD, P0357, P0A24

— ECU #2 (TCM): P0443

— ECU #3 (ABS/Traction Control): no emission-related DTC stored

Table 24 defines the request emission-related diagnostic trouble codes request message.

Table 24 — Request emission-related diagnostic trouble codes request message

Message direction:	External test equipment → All ECUs
Message type:	Request

Data byte	Description (all values are in hexadecimal)	Byte value (Hex)	Mnemonic
#1	Request emission-related DTCs request SID	03/07/0A	SIDRQ

Table 25 defines the request emission-related diagnostic trouble codes response message ECU#1.

Table 25 — Request emission-related diagnostic trouble codes response message ECU#1

Message direction:	ECU #1 → External test equipment
Message type:	Response

Data byte	Description (all values are in hexadecimal)	Byte value (Hex)	Mnemonic
#1	Request emission-related DTCs response SID	43/47/4A	SIDPR
#2	# of DTC {number of emission-related DTCs stored in this ECU}	06	#OFDTC
#3	DTC High Byte of P0143	01	DTC1HI
#4	DTC Low Byte of P0143	43	DTC1LO
#5	DTC High Byte of P0196	01	DTC2HI
#6	DTC Low Byte of P0196	96	DTC2LO
#7	DTC High Byte of P0234	02	DTC3HI
#8	DTC Low Byte of P0234	34	DTC3LO
#9	DTC High Byte of P02CD	02	DTC4HI
#10	DTC Low Byte of P02CD	CD	DTC4LO
#11	DTC High Byte of P0357	03	DTC5HI
#12	DTC Low Byte of P0357	57	DTC5LO
#13	DTC High Byte of P0A24	0A	DTC6HI
#14	DTC Low Byte of P0A24	24	DTC6LO

Table 26 defines the request emission-related diagnostic trouble codes response message ECU#3.

Table 26 — Request emission-related diagnostic trouble codes response message ECU#3

Message direction:	ECU #3 → External test equipment
Message type:	Response

Data byte	Description (all values are in hexadecimal)	Byte value (Hex)	Mnemonic
#1	Request emission-related DTCs response SID	43/47/4A	SIDPR
#2	# of DTC {number of emission-related DTCs stored in this ECU}	00	#OFDTC

Table 27 defines the request emission-related diagnostic trouble codes response message ECU#2.

Table 27 — Request emission-related diagnostic trouble codes response message ECU#2

Message direction:	ECU #2 → External test equipment
Message type:	Response

Data byte	Description (all values are in hexadecimal)	Byte value (Hex)	Mnemonic
#1	Request emission-related DTCs response SID	43/47/4A	SIDPR
#2	# of DTC {number of emission-related DTCs stored in this ECU}	01	#OFDTC
#3	DTC High Byte of P0443	04	DTC1HI
#4	DTC Low Byte of P0443	43	DTC1LO

9.6.4.4 This part of ISO 22901 — Request emission-related DTCs in ODX sample

```
<REQUEST ID="REQ_Service03RequestEmissionRelatedDiagnosticTroubleCodes">
  <SHORT-NAME>Service03RequestEmissionRelatedDiagnosticTroubleCodes</SHORT-NAME>
        <LONG-NAME>Service 0x03 - Request emission-related
                                          diagnostic trouble codes</LONG-NAME>
           <PARAMS>
               <PARAM xsi:type="CODED-CONST" SEMANTIC="SERVICE-ID">
                 <SHORT-NAME>RequestEmissionRelatedDiagnostic
                                          TroubleCodesRequestSID</SHORT-NAME>
                 <LONG-NAME>Request emission-related diagnostic
                                          trouble codes request SID</LONG-NAME>
                 <BYTE-POSITION>0</BYTE-POSITION>
                 <CODED-VALUE>3</CODED-VALUE>
                 <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32"
                                  xsi:type="STANDARD-LENGTH-TYPE">
                      <BIT-LENGTH>8</BIT-LENGTH>
                 </DIAG-CODED-TYPE>
               </PARAM>
            </PARAMS>
       </REQUEST>

<POS-RESPONSE ID="PRE_Service03RequestEmissionRelatedDiagnosticTroubleCodes">
    <SHORT-NAME>Service03RequestEmissionRelatedDiagnosticTroubleCodes</SHORT-NAME>
    <LONG-NAME>Service 0x03 - Request emission-related diagnostic trouble codes</LONG-
NAME>
    <PARAMS>
        <PARAM xsi:type="CODED-CONST" SEMANTIC="SERVICE-ID">
            <SHORT-NAME>RequestEmissionRelatedDiagnosticTrouble
                                                    CodesResponseSID</SHORT-NAME>
            <LONG-NAME>Request emission-related diagnostic trouble
                                                    codes response SID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <CODED-VALUE>67</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
                <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
        <PARAM SEMANTIC="DATA" xsi:type="VALUE">
```

```
            <SHORT-NAME>List_of_DTC</SHORT-NAME>
            <LONG-NAME>List of DTC</LONG-NAME>
            <BYTE-POSITION>1</BYTE-POSITION>
            <DOP-REF ID-REF="DLF List_of_DTCs"/>
        </PARAM>
    </PARAMS>
</POS-RESPONSE>

<DYNAMIC-LENGTH-FIELD ID="DLF_ListOfDTCs">
       <SHORT-NAME>ListOfDTCs</SHORT-NAME>
       <LONG-NAME>List of DTCs</LONG-NAME>
       <BASIC-STRUCTURE-REF ID-REF="STRUC_ListOfDTC" />
       <OFFSET>1</OFFSET>
       <DETERMINE-NUMBER-OF-ITEMS>
             <BYTE-POSITION>0</BYTE-POSITION>
             <DATA-OBJECT-PROP-REF ID-REF="DOP_NumberOfDTC" />
       </DETERMINE-NUMBER-OF-ITEMS>
</DYNAMIC-LENGTH-FIELD>

<STRUCTURE ID="STRUC_ListOfDTC">
    <SHORT-NAME>ListOfDTC</SHORT-NAME>
    <LONG-NAME>List of DTC</LONG-NAME>
    <PARAMS>
        <PARAM SEMANTIC="DATA" xsi:type="VALUE">
            <SHORT-NAME>TestDataObjectQual</SHORT-NAME>
            <LONG-NAME>TestDataObjectName</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <DOP-REF ID-REF="DTCDOP_DTCs"/>
        </PARAM>
    </PARAMS>
</STRUCTURE>

<DTC-DOP ID="DTCDOP_DTCs">
    <SHORT-NAME>RecordDataType</SHORT-NAME>
    <LONG-NAME>RecordDataType</LONG-NAME>
    <DIAG-CODED-TYPE BASE-TYPE-ENCODING="NONE" BASE-DATA-TYPE="A_UINT32"
                    xsi:type="STANDARD-LENGTH-TYPE">
        <BIT-LENGTH>16</BIT-LENGTH>
    </DIAG-CODED-TYPE>
    <PHYSICAL-TYPE BASE-DATA-TYPE="A_UINT32" DISPLAY-RADIX="HEX"/>
    <COMPU-METHOD>
        <CATEGORY>IDENTICAL</CATEGORY>
    </COMPU-METHOD>
    <DTCS>
        <DTC ID="OBD_ECU_78">
            <SHORT-NAME>DTC08</SHORT-NAME>
            <TROUBLE-CODE>8</TROUBLE-CODE>
            <TEXT>Engine Position System Performance</TEXT>
        </DTC>
    </DTCS>
</DTC-DOP>

<DATA-OBJECT-PROP ID="DOP_NumberOfDTC">
```

```
        <SHORT-NAME>NumberOfDTC</SHORT-NAME>
        <LONG-NAME>Number of DTC</LONG-NAME>
        <COMPU-METHOD>
              <CATEGORY>IDENTICAL</CATEGORY>
        </COMPU-METHOD>
        <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
              <BIT-LENGTH>8</BIT-LENGTH>
        </DIAG-CODED-TYPE>
        <PHYSICAL-TYPE BASE-DATA-TYPE="A_UINT32" />
</DATA-OBJECT-PROP>
```

9.6.5 Service 0x04 — Clear/Reset emission-related diagnostic information

9.6.5.1 ISO 15031-5 — Clear/Reset emission-related diagnostic information example

The example below shows how the “Clear/reset emission-related diagnostic information” service shall be implemented if ignition is ON and with the engine not running.

The external test equipment commands the vehicle to “Clear/reset emission-related diagnostic information”.

Table 28 defines the clear/reset emission-related diagnostic information request message.

Table 28 — Clear/reset emission-related diagnostic information request message

Message direction:	External test equipment → All ECUs		
Message type:	Request		
Data byte	**Description (all values are in hexadecimal)**	**Byte value (Hex)**	**Mnemonic**
#1	Clear/reset emission-related diagnostic information request SID	0x04	SIDRQ

Table 29 defines the clear/reset emission-related diagnostic information response message ECU#1.

Table 29 — Clear/reset emission-related diagnostic information response message ECU#1

Message direction:	ECU#1 → External test equipment		
Message type:	Response		
Data byte	**Description (all values are in hexadecimal)**	**Byte value (Hex)**	**Mnemonic**
#1	Clear/reset emission-related diagnostic information response SID	0x44	SIDPR

Table 30 defines the clear/reset emission-related diagnostic information response message ECU#2.

Table 30 — Clear/reset emission-related diagnostic information response message ECU#2

Message direction:	ECU#2 → External test equipment		
Message type:	Response		
Data byte	**Description (all values are in hexadecimal)**	**Byte value (Hex)**	**Mnemonic**
#1	Clear/reset emission-related diagnostic information response SID	0x44	SIDPR

Table 31 defines the negative response message — Clear/reset.

Table 31 — Negative response message — Clear/reset

Message direction:	ECU#3 → External test equipment
Message type:	Response

Data byte	Description (all values are in hexadecimal)	Byte value (Hex)	Mnemonic
#1	Negative Response Service Identifier	0x7F	SIDNR
#2	Clear/reset emission-related diagnostic information request SID	0x04	SIDRQ
#3	Negative Response Code: conditionsNotCorrect	0x22	NR_CNC

9.6.5.2 This part of ISO 22901 — Clear/Reset emission-related diagnostic information in ODX sample

```
<DIAG-SERVICE
ID="ES_OBDIICommonServices.DS_Service04ClearResetEmissionRelatedDiagnosticInformation"
              ADDRESSING="FUNCTIONAL">
    <SHORT-NAME>Service04ClearResetEmissionRelatedDiagnosticInformation</SHORT-NAME>
    <LONG-NAME>Service 0x04 - Clear/reset emission-related
                                          diagnostic information</LONG-NAME>
    <REQUEST-REF ID-REF="REQ_Service04ClearResetEmissionRelatedDiagnosticInformation"/>
    <POS-RESPONSE-REFS>
        <POS-RESPONSE-REF ID-REF="PRE_Service04ClearResetEmissionRelated
                                                    DiagnosticInformation"/>
    </POS-RESPONSE-REFS>
</DIAG-SERVICE>

<REQUEST ID="REQ_Service04ClearResetEmissionRelatedDiagnosticInformation">
    <SHORT-NAME>Service04ClearResetEmissionRelatedDiagnosticInformation</SHORT-NAME>
    <LONG-NAME>Service 0x04 - Clear/reset emission-related
                                          diagnostic information</LONG-NAME>
    <PARAMS>
        <PARAM xsi:type="CODED-CONST" SEMANTIC="SERVICE-ID">
            <SHORT-NAME>ClearResetEmissionRelated
                                DiagnosticInformationRequestSID</SHORT-NAME>
            <LONG-NAME>Clear/reset emission-related
                                diagnostic information request SID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <CODED-VALUE>4</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
                <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
    </PARAMS>
</REQUEST>
```

```
<POS-RESPONSE ID="PRE_Service04ClearResetEmissionRelatedDiagnosticInformation">
    <SHORT-NAME>Service04ClearResetEmissionRelatedDiagnosticInformation</SHORT-NAME>
    <LONG-NAME>Service 0x04 - Clear/reset emission-related
                                diagnostic information</LONG-NAME>
    <PARAMS>
        <PARAM xsi:type="CODED-CONST" SEMANTIC="SERVICE-ID">
            <SHORT-NAME>ClearResetEmissionRelated
                                DiagnosticInformationResponseSID</SHORT-NAME>
            <LONG-NAME>Clear/reset emission-related
                                diagnostic information response SID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <CODED-VALUE>68</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
                <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
    </PARAMS>
</POS-RESPONSE>
```

9.6.6 Service 0x06 — Request on-board monitoring test results

9.6.6.1 ISO 15031-5 — Request on-board monitoring test results example

The request and response message described in this subclause can be used within ISO 15765-4.

The external test equipment sends a “Request on-board monitoring test results for specific monitored systems” message with one supported OBDMID in the request message to the vehicle. In this example, the request message includes the following OBDMID:

— request message: OBDMID 0x01 - Oxygen Sensor Monitor Bank 1 - Sensor 1

Table 32 defines the Request oxygen sensor monitoring test results request message.

Table 32 — Request oxygen sensor monitoring test results request message

Message direction:	External test equipment → All ECUs		
Message type:	Request		
Data byte	**Description (all values are in hexadecimal)**	**Byte value (Hex)**	**Mnemonic**
#1	Request on-board monitoring test results for specific monitored systems request SID	06	SIDRQ
#2	OBDMID: 01 - Oxygen Sensor Monitor Bank 1 - Sensor 1	01	OBDMID

Table 33 defines the request oxygen sensor monitoring test results response message.

Table 33 — Request oxygen sensor monitoring test results response message

Message direction:	ECU #1 → External test equipment
Message type:	Response

Data byte	Description (all values are in hexadecimal)	Byte value (Hex)	Mnemonic
#1	Request on-board monitoring test results for specific monitored systems response SID	46	SIDPRQ
#2	OBDMID: 01 - Oxygen Sensor Monitor Bank 1 - Sensor 1	01	OBDMID
#3	Standardized Test ID: 01 - Rich to lean sensor threshold voltage (constant)	01	STID
#4	Unit And Scaling ID: Voltage	0A	UASID
#5	Test Value High Byte:	0B	TESTVAL
#6	Test Value Low Byte: 0,365 V	B0	TESTVAL
#7	Minimum Test Limit High Byte:	0B	MINLIMIT
#8	Minimum Test Limit Low Byte: 0,365 V	B0	MINLIMIT
#9	Maximum Test Limit High Byte:	0B	MAXLIMIT
#10	Maximum Test Limit Low Byte: 0,365 V	B0	MAXLIMIT
#11	OBDMID: 01 - Oxygen Sensor Monitor Bank 1 - Sensor 1	01	OBDMID
#12	Standardized Test ID: 05 - Rich to lean sensor switch time (calculated)	05	STID
#13	Unit And Scaling ID: Time	10	UASID
#14	Test Value High Byte	00	TESTVAL
#15	Test Value Low Byte: 0,072 s (0 min, 0 s)	48	TESTVAL
#16	Minimum Test Limit High Byte	00	MINLIMIT
#17	Minimum Test Limit Low Byte: 0,000 s (0 min, 0 s)	00	MINLIMIT
#18	Maximum Test Limit High Byte	00	MAXLIMIT
#19	Maximum Test Limit Low Byte: 0,100 s (0 min, 0 s)	64	MAXLIMIT
#20	OBDMID: 01 - Oxygen Sensor Monitor Bank 1 - Sensor 1	01	OBDMID
#21	Manufacturer Defined Test ID: 133 (The name of this Test ID shall be documented in the vehicle Service Information.)	85	MDTID
#22	Unit And Scaling ID: Counts	24	UASID
#23	Test Value High Byte	00	TESTVAL
#24	Test Value Low Byte: 150 counts	96	TESTVAL
#25	Minimum Test Limit High Byte	00	MINLIMIT
#26	Minimum Test Limit Low Byte: 75 counts	4B	MINLIMIT
#27	Maximum Test Limit High Byte	FF	MAXLIMIT
#28	Maximum Test Limit Low Byte: 65535 counts	FF	MAXLIMIT

NOTE ECU#2 does not support any Test IDs and therefore does not send a response message.

9.6.6.2 ISO 22900-2 — Request on-board monitoring test results in ODX sample

```
<DIAG-SERVICE ID="ES_OBDIICommonServices.DS_Service06RequestOnBoardMonitoringTestResults
                                ForSpecificMonitoredSystems" ADDRESSING="FUNCTIONAL">
    <SHORT-NAME>Service06RequestOnBoardMonitoringTestResults
                                                ForSpecificMonitoredSystems</SHORT-NAME>
    <LONG-NAME>Service 0x06 - Request on-board monitoring test results
                                             for specific monitored systems</LONG-NAME>
    <REQUEST-REF ID-REF="REQ_Service06RequestOnBoardMonitoringTestResults
                                                        ForSpecificMonitoredSystems"/>
    <POS-RESPONSE-REFS>
        <POS-RESPONSE-REF ID-REF="PRE_Service06RequestOnBoardMonitoringTestResults
                                                            ForSpecificMonitoredSystems"/>
    </POS-RESPONSE-REFS>
</DIAG-SERVICE>
```

The following Request is used to determine which OBDMIDs are supported by the ECU.

```
<REQUEST ID="REQ_Service06RequestOnBoardMonitoringTestResultsForSpecificMonitoredSystems">
    <SHORT-NAME>Service06RequestOnBoardMonitoringTestResults
                                        ForSpecificMonitoredSystems</SHORT-NAME>
    <LONG-NAME>Service 0x06 - Request on-board monitoring test results
                                        for specific monitored systems</LONG-NAME>
    <PARAMS>
        <PARAM xsi:type="CODED-CONST" SEMANTIC="SERVICE-ID">
            <SHORT-NAME>RequestOnBoardMonitoringTestResults
                                ForSpecificMonitoredSystemsRequestSID</SHORT-NAME>
            <LONG-NAME>Request on-board monitoring test results
                                for specific monitored systems request SID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <CODED-VALUE>6</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
                <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
        <PARAM SEMANTIC="ID" ID="Param_OBDMID" xsi:type="TABLE-KEY">
            <SHORT-NAME>OBDMID</SHORT-NAME>
            <LONG-NAME>OBDMID</LONG-NAME>
            <BYTE-POSITION>1</BYTE-POSITION>
            <TABLE-REF ID-REF="ES_OBDIICommonServices.
                        TAB_OnboardMonitoringTestResultsReadSupportedOBDMIDs"/>
        </PARAM>
    </PARAMS>
</REQUEST>
```

```
<TABLE ID="ES_OBDIICommonServices.
            TAB_OnboardMonitoringTestResultsReadSupportedOBDMIDs" SEMANTIC="OBDMID">
    <SHORT-NAME>OnboardMonitoringTestResultsReadSupportedOBDMIDs</SHORT-NAME>
    <LONG-NAME>Onboard Monitoring Test Results Read supported OBDMIDs</LONG-NAME>
    <KEY-DOP-REF ID-REF="ES_OBDIICommonServices.DOP_OBDMIDsSupported"/>
    <TABLE-ROW ID="ES_OBDIICommonServices.
                        TAB_OnboardMonitoringTestResultsReadSupportedOBDMIDs.
                        TABROW_OBDMID00OBDMIDSupported01To20">
        <SHORT-NAME>OBDMID00OBDMIDSupported01To20</SHORT-NAME>
        <LONG-NAME>(OBDMID 0x00) OBDMID supported, 0x01 to 0x20</LONG-NAME>
        <KEY>0</KEY>
        <STRUCTURE-REF ID-REF="STRUC_ReadSupportedOBDMIDsOBDMID00OBDMIDSupported01To20"/>
    </TABLE-ROW>
</TABLE>
```

The Structure described below is used to interpret which OBDMIDs are supported.

```
<STRUCTURE ID="STRUC_ReadSupportedOBDMIDsOBDMID00OBDMIDSupported01To20">
    <SHORT-NAME>ReadSupportedOBDMIDsOBDMID00OBDMIDSupported01To20</SHORT-NAME>
    <LONG-NAME>Read supported OBDMIDs (OBDMID 0x00) OBDMID supported, 0x01to 0x20</LONG-
NAME>
    <PARAMS>
        <PARAM SEMANTIC="DATA" xsi:type="VALUE">
            <SHORT-NAME>SupportedOBDMIDs01To08</SHORT-NAME>
            <LONG-NAME>Supported OBDMIDs 0x01 to 0x08</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <DOP-REF ID-REF="STRUC_SupportedOBDMIDs01To08"/>
        </PARAM>
        …
    </PARAMS>
</STRUCTURE>

<STRUCTURE ID="STRUC_SupportedOBDMIDs01To08">
    <SHORT-NAME>SupportedOBDMIDs01To08</SHORT-NAME>
    <LONG-NAME>Supported OBDMIDs 0x01 to 0x08</LONG-NAME>
    <BYTE-SIZE>1</BYTE-SIZE>
    <PARAMS>
        <PARAM SEMANTIC="DATA" xsi:type="VALUE">
            <SHORT-NAME>OBDMID0x08Supported</SHORT-NAME>
            <LONG-NAME>OBDMID 0x08 supported</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <DOP-REF ID-REF="ES_OBDIICommonServices.DOP_OBDMIDsSupported"/>
        </PARAM>
        …
    </PARAMS>
</STRUCTURE>
```

To request the Data stored for a specific OBDMID, the following TABLE is used.

```
<TABLE ID="TAB_OnBoardMonitoringTestResultsReadOBDMIDData"
        SEMANTIC="OBDMID">
    <SHORT-NAME>OnBoardMonitoringTestResultsReadOBDMIDData</SHORT-NAME>
    <LONG-NAME>Onboard Monitoring Test Results Read OBDMID Data</LONG-NAME>
    <KEY-DOP-REF ID-REF="ES_OBDIICommonServices.DOP_OBDMIDsSupported"/>
    <TABLE-ROW ID="ES_OBDIICommonServices.
                    TAB_OnBoardMonitoringTestResultsReadOBDMIDData.
                    TABROW_OBDMID04OxygenSensorMonitorBank1ToSensor4">
        <SHORT-NAME>OBDMID04OxygenSensorMonitorBank1ToSensor4</SHORT-NAME>
        <LONG-NAME>(OBDMID 0x04) Oxygen Sensor Monitor Bank 1 to Sensor 4</LONG-NAME>
        <KEY>4</KEY>
        <STRUCTURE-REF ID-REF="STRUC_ReadOBDMIDDataPROBDMID04OxygenSensor
                                                    MonitorBank1ToSensor4"/>
    </TABLE-ROW>
</TABLE>

<STRUCTURE ID="STRUC_ReadOBDMIDDataPROBDMID04OxygenSensorMonitorBank1ToSensor4">
    <SHORT-NAME>ReadOBDMIDDataPROBDMID04OxygenSensorMonitorBank1ToSensor4</SHORT-NAME>
    <LONG-NAME>Read OBDMID Data PR (OBDMID 0x04) Oxygen Sensor
                                            Monitor Bank 1 to Sensor 4</LONG-NAME>
    <PARAMS>
        <PARAM SEMANTIC="DATA" xsi:type="VALUE">
            <SHORT-NAME>OBDMID04OxygenSensorMonitorBank1Sensor4Data</SHORT-NAME>
            <LONG-NAME>OBDMID 04 Oxygen Sensor Monitor Bank 1 Sensor 4 Data</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <DOP-REF ID-REF="EOPDUF_DataRecord"/>
        </PARAM>
    </PARAMS>
</STRUCTURE
```

For each OBDMID, the Interpretation is determined using Unit-and-Scaling-ID. This is represented by an END-OF-PDU-FIELD and a MUX.

```
<END-OF-PDU-FIELD ID="EOPDUF_DataRecord">
    <SHORT-NAME>DataRecord</SHORT-NAME>
    <LONG-NAME>Data Record</LONG-NAME>
    <BASIC-STRUCTURE-REF ID-REF="STRUC_DataRecord"/>
</END-OF-PDU-FIELD>

<STRUCTURE ID="STRUC_DataRecord">
    <SHORT-NAME>DataRecord</SHORT-NAME>
    <LONG-NAME>Data Record</LONG-NAME>
    <PARAMS>
        <PARAM SEMANTIC="DATA" xsi:type="VALUE">
            <SHORT-NAME>OBDMID</SHORT-NAME>
            <LONG-NAME>OBDMID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <DOP-REF ID-REF="DOP_UnitAndScaling"/>
        </PARAM>
        <PARAM SEMANTIC="DATA" xsi:type="VALUE">
            <SHORT-NAME>TID</SHORT-NAME>
            <LONG-NAME>TID</LONG-NAME>
```

```
            <BYTE-POSITION>1</BYTE-POSITION>
            <DOP-REF ID-REF="DOP_TID"/>
        </PARAM>
        <PARAM SEMANTIC="DATA" xsi:type="VALUE">
            <SHORT-NAME>UnitAndScalingID</SHORT-NAME>
            <LONG-NAME>Unit-And-Scaling-ID</LONG-NAME>
            <BYTE-POSITION>2</BYTE-POSITION>
            <DOP-REF ID-REF="DOP_UnitAndScaling"/>
        </PARAM>
        <PARAM SEMANTIC="DATA" xsi:type="VALUE">
            <SHORT-NAME>Multiplexer</SHORT-NAME>
            <LONG-NAME>Multiplexer</LONG-NAME>
            <BYTE-POSITION>2</BYTE-POSITION>
            <DOP-REF ID-REF="MUX_Multiplexer"/>
        </PARAM>
    </PARAMS>
</STRUCTURE>

<MUX ID="MUX_Multiplexer">
    <SHORT-NAME>Multiplexer</SHORT-NAME>
    <LONG-NAME>Multiplexer</LONG-NAME>
    <BYTE-POSITION>1</BYTE-POSITION>
    <SWITCH-KEY>
        <BYTE-POSITION>0</BYTE-POSITION>
        <DATA-OBJECT-PROP-REF ID-REF="DOP_UnitAndScaling"/>
    </SWITCH-KEY>
    <DEFAULT-CASE>
        <SHORT-NAME>Case_Default</SHORT-NAME>
        <STRUCTURE-REF ID-REF="STRUC_CaseDefault"/>
    </DEFAULT-CASE>
    <CASES>
        <CASE>
            <SHORT-NAME>Case_0x1</SHORT-NAME>
            <STRUCTURE-REF ID-REF="STRUC_Case_0x1"/>
            <LOWER-LIMIT>1</LOWER-LIMIT>
            <UPPER-LIMIT>1</UPPER-LIMIT>
        </CASE>
    </CASES>
</MUX>

<POS-RESPONSE ID="PRE_Service06RequestOnBoardMonitoringTestResults
                                        ForSpecificMonitoredSystems">
    <SHORT-NAME>Service06RequestOnBoardMonitoringTestResults
                                    ForSpecificMonitoredSystems</SHORT-NAME>
    <LONG-NAME>Service 0x06 - Request on-board monitoring test results
                                    for specific monitored systems</LONG-NAME>
    <PARAMS>
        <PARAM xsi:type="CODED-CONST" SEMANTIC="SERVICE-ID">
            <SHORT-NAME>RequestOnBoardMonitoringTestResults
                            ForSpecificMonitoredSystemsResponseSID</SHORT-NAME>
            <LONG-NAME>Request on-board monitoring test results
                        for specific monitored systems response SID</LONG-NAME>
```

```
            <BYTE-POSITION>0</BYTE-POSITION>
            <CODED-VALUE>70</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
                <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
        ...
    </PARAMS>
</POS-RESPONSE>
```

9.6.7 Service 0x08 — Request control of on-board device

9.6.7.1 ISO 15031-5 — Request control of on-board device example

The request and response message described in this subclause can be used within ISO 15765-4.

The external test equipment sends a “Request control of on-board device” message with one (1) supported Test ID 0x01 to the vehicle.

Table 34 defines the request control of on-board device request message.

Table 34 — Request control of on-board device request message

Message direction:	External test equipment → All ECUs
Message type:	Request

Data byte	Description (all values are in hexadecimal)	Byte value (Hex)	Mnemonic
#1	Request control of on-board device request SID	08	SIDRQ
#2	Test ID: 01 - Evaporative system leak test	01	TID

Table 35 defines the request control of on-board device response message.

Table 35 — Request control of on-board device response message

Message direction:	ECU #1 → External test equipment
Message type:	Response

Data byte	Description (all values are in hexadecimal)	Byte value (Hex)	Mnemonic
#1	Request control of on-board device response SID	48	SIDPR
#2	Test ID: 01 - Evaporative system leak test	01	TID

In the following example, the conditions of the system are not proper to run the Evaporative system leak test. Therefore, the ECM (ECU #1) responds with a negative response message with response code 0x22 - conditionsNotCorrect. The TCM (ECU #2) does not respond because it previously reported that it does not support the Evaporative system leak test.

Table 36 defines the request control of on-board device request message.

Table 36 — Request control of on-board device request message

Message direction:	External test equipment → All ECUs
Message type:	Request

Data byte	Description (all values are in hexadecimal)	Byte value (Hex)	Mnemonic
#1	Request control of on-board device request SID	08	SIDRQ
#2	Test ID: 01 - Evaporative system leak test	01	TID

Table 37 defines the Negative response message — Request control.

Table 37 — Negative response message — Request control

Message direction:	ECU#1 → External test equipment
Message type:	Response

Data byte	Description (all values are in hexadecimal)	Byte value (Hex)	Mnemonic
#1	Negative Response Service Identifier	7F	SIDNR
#2	Request control of on-board device request SID	08	SIDRQ
#3	Negative Response Code: conditionsNotCorrect	22	NR_CNC

9.6.7.2 ISO 22900-2 — Request control of on-board device in ODX sample

```
<DIAG-SERVICE
ID="ES_OBDIICommonServices.DS_Service08RequestControlOfOnBoardSystemTestOrComponent"
              ADDRESSING="FUNCTIONAL">
    <SHORT-NAME>Service08RequestControlOfOnBoardSystemTestOrComponent</SHORT-NAME>
    <LONG-NAME>Service 0x08 - Request control of on-board system, test or
              component</LONG-NAME>
    <REQUEST-REF ID-REF="REQ_Service08RequestControlOfOnBoardSystemTestOrComponent"/>
    <POS-RESPONSE-REFS>
        <POS-RESPONSE-REF ID-REF="PRE_Service08RequestControlOfOnBoardSystem
                                                              TestOrComponent"/>
    </POS-RESPONSE-REFS>
</DIAG-SERVICE>
```

The following request is used to determine the supported TIDs of the ECU.

```
<REQUEST ID="REQ_Service08RequestControlOfOnBoardSystemTestOrComponent">
    <SHORT-NAME>Service08RequestControlOfOnBoardSystemTestOrComponent</SHORT-NAME>
    <LONG-NAME>Service 0x08 - Request control of on-board system,
                                          test or component</LONG-NAME>
```

```
    <PARAMS>
        <PARAM xsi:type="CODED-CONST" SEMANTIC="SERVICE-ID">
            <SHORT-NAME>RequestControlOfOnBoardSystem
                        TestOrComponentRequestSID</SHORT-NAME>
            <LONG-NAME>Request control of on-board system,
                           test or component request SID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <CODED-VALUE>8</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
                <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
        <PARAM SEMANTIC="ID" ID="Param_TID" xsi:type="TABLE-KEY">
            <SHORT-NAME>TID</SHORT-NAME>
            <LONG-NAME>TID</LONG-NAME>
            <BYTE-POSITION>1</BYTE-POSITION>
            <TABLE-REF ID-REF="TAB_ControlOfOnboardSystemsReadSupportedTIDs"/>
        </PARAM>
    </PARAMS>
</REQUEST>


<TABLE ID="TAB_ControlOfOnboardSystemsReadSupportedTIDs">
    <SHORT-NAME>ControlOfOnboardSystemsReadSupportedTIDs</SHORT-NAME>
    <LONG-NAME>Control of Onboard Systems Read Supported TIDs</LONG-NAME>
    <KEY-DOP-REF ID-REF="DOP_TIDsSupported"/>
    <TABLE-ROW ID="TABROW_TID00TIDSupported01To20">
        <SHORT-NAME>TID00TIDSupported01To20</SHORT-NAME>
        <LONG-NAME>(TID 0x00) TID supported, 0x01 to 0x20</LONG-NAME>
        <KEY>0</KEY>
        <STRUCTURE-REF ID-REF="STRUC_ControlOfOnboardSystemsTIDSupported01To20"/>
    </TABLE-ROW>
</TABLE>

<STRUCTURE ID="STRUC_ControlOfOnboardSystemsTIDSupported01To20">
    <SHORT-NAME>ControlOfOnboardSystemsTIDSupported01To20</SHORT-NAME>
    <LONG-NAME>Control of Onboard Systems, TID supported, 0x01 to 0x20</LONG-NAME>
    <PARAMS>
        <PARAM SEMANTIC="DATA" xsi:type="VALUE">
            <SHORT-NAME>SupportedTIDs01To08</SHORT-NAME>
            <LONG-NAME>Supported TIDs0x01 to 0x08</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <DOP-REF ID-REF="DOP_SupportedTIDs01To08178"/>
        </PARAM>
        …
    </PARAMS>
</STRUCTURE>
```

```
<POS-RESPONSE ID="PRE_Service08RequestControlOfOnBoardSystemTestOrComponent">
    <SHORT-NAME>Service08RequestControlOfOnBoardSystemTestOrComponent</SHORT-NAME>
    <LONG-NAME>Service 0x08 – Request control of on-board system,
                                          test or component</LONG-NAME>
    <PARAMS>
        <PARAM xsi:type="CODED-CONST" SEMANTIC="SERVICE-ID">
            <SHORT-NAME>RequestControlOfOnBoardSystem
                                    TestOrComponentResponseSID</SHORT-NAME>
            <LONG-NAME>Request control of on-board system,
                                    test or component response SID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <CODED-VALUE>72</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
                <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
    </PARAMS>
</POS-RESPONSE>
```

9.6.8 Service 0x09 — Request vehicle information

9.6.8.1 ISO 15031-5 — Request vehicle information example

The request and response message described in this subclause can be used within ISO 15765-4.

Now the external test equipment requests the following InfoType:

— InfoType 0x02: VIN = [1G1JC5444R7252367] supported by ECU #1.

Table 38 defines the request vehicle information request message.

Table 38 — Request vehicle information request message

Message direction:	External test equipment → All ECUs		
Message type:	Request		
Data byte	**Description (all values are in hexadecimal)**	**Byte value (Hex)**	**Mnemonic**
#1	Request vehicle information request SID	09	SIDRQ
#2	InfoType: 02 - VIN (Vehicle Identification Number)	02	INFTYP

Table 39 defines the request vehicle information response message.

Table 39 — Request vehicle information response message

Message direction:	ECU #1 → External test equipment
Message type:	Response

Data byte	Description (all values are in hexadecimal)	Byte value (Hex)	Mnemonic
#1	Request vehicle information response SID	49	SIDPR
#2	InfoType: 02 - VIN (Vehicle Information Number)	02	INFTYP
#3	Number of data items: 01	01	NODI
#4	1st ASCII character of VIN: '1'	31	VIN
#5	2nd ASCII character of VIN: 'G'	47	VIN
#6	3rd ASCII character of VIN: '1'	31	VIN
#7	4th ASCII character of VIN: 'J'	4A	VIN
#8	5th ASCII character of VIN: 'C'	43	VIN
#9	6th ASCII character of VIN: '5'	35	VIN
#10	7th ASCII character of VIN: '4'	34	VIN
#11	8th ASCII character of VIN: '4'	34	VIN
#12	9th ASCII character of VIN: '4'	34	VIN
#13	10th ASCII character of VIN: 'R'	52	VIN
#14	11th ASCII character of VIN: '7'	37	VIN
#15	12th ASCII character of VIN: '2'	32	VIN
#16	13th ASCII character of VIN: '5'	35	VIN
#17	14th ASCII character of VIN: '2'	32	VIN
#18	15th ASCII character of VIN: '3'	33	VIN
#19	16th ASCII character of VIN: '6'	36	VIN
#20	17th ASCII character of VIN: '7'	37	VIN

9.6.8.2 This part of ISO 22901 — Request vehicle information in ODX sample

```
<DIAG-SERVICE ID="ES_OBDIICommonServices.DS_Service09RequestVehicleInformation"
ADDRESSING="FUNCTIONAL">
    <SHORT-NAME>Service09RequestVehicleInformation</SHORT-NAME>
    <LONG-NAME>Service 0x09 - Request vehicle information</LONG-NAME>
    <REQUEST-REF ID-REF="REQ_Service09RequestVehicleInformation"/>
    <POS-RESPONSE-REFS>
        <POS-RESPONSE-REF ID-REF="PRE_Service09RequestVehicleInformation"/>
    </POS-RESPONSE-REFS>
</DIAG-SERVICE>
```

The following request is used to determine the supported Vehicle Information of an ECU.

```
<REQUEST ID="REQ_Service09RequestVehicleInformation">
    <SHORT-NAME>Service09RequestVehicleInformation</SHORT-NAME>
    <LONG-NAME>Service 0x09 - Request vehicle information</LONG-NAME>
    <PARAMS>
        <PARAM xsi:type="CODED-CONST" SEMANTIC="SERVICE-ID">
```

```
            <SHORT-NAME>RequestVehicleInformationRequestSID</SHORT-NAME>
            <LONG-NAME>Request vehicle informationt request SID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <CODED-VALUE>9</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
                <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
        <PARAM SEMANTIC="ID" ID="Param_InfoType" xsi:type="TABLE-KEY">
            <SHORT-NAME>InfoType</SHORT-NAME>
            <LONG-NAME>InfoType</LONG-NAME>
            <BYTE-POSITION>1</BYTE-POSITION>
            <TABLE-REF ID-REF="TAB_VehicleInformationReadSupportedInfoTypes"/>
        </PARAM>
    </PARAMS>
</REQUEST>

<TABLE ID="TAB_VehicleInformationReadSupportedInfoTypes">
    <SHORT-NAME>VehicleInformationReadSupportedInfoTypes</SHORT-NAME>
    <LONG-NAME>Vehicle Information Read supported InfoTypes</LONG-NAME>
    <KEY-DOP-REF ID-REF="DOP_SupportedInfoTypes"/>
    <TABLE-ROW ID="TABROW_InfoType00InfoTypeSupported01To20">
        <SHORT-NAME>InfoType00InfoTypeSupported01To20</SHORT-NAME>
        <LONG-NAME>(InfoType 0x00) InfoType supported, 0x01 to 0x20</LONG-NAME>
        <KEY>0</KEY>
        <STRUCTURE-REF ID-REF="STRUC_InfoType00InfoTypeSupported01To20"/>
    </TABLE-ROW>
</TABLE>
```

The referenced STRUCTURE is then used to determine which information is supported.

```
<STRUCTURE ID="STRUC_SupportedInfoTypesInfoTypeSupported01To20">
    <SHORT-NAME>SupportedInfoTypesInfoTypeSupported01To20</SHORT-NAME>
    <LONG-NAME>Supported InfoTypes InfoType supported, 0x01 to 0x20</LONG-NAME>
    <PARAMS>
        <PARAM SEMANTIC="DATA" xsi:type="VALUE">
            <SHORT-NAME>_InfoType_00_InfoType_supported_01_20</SHORT-NAME>
            <LONG-NAME>Supported InfoTypes in 0x01 - 0x08</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <DOP-REF ID-REF="STRUC_SupportedInfoTypes01To08"/>
        </PARAM>
        ...
    </PARAMS>
</STRUCTURE>

<STRUCTURE ID="STRUC_SupportedInfoTypes01To08">
    <SHORT-NAME>SupportedInfoTypes01To08</SHORT-NAME>
    <LONG-NAME>Supported InfoTypes 0x01 to 0x08</LONG-NAME>
    <BYTE-SIZE>1</BYTE-SIZE>
    <PARAMS>
```

```
        <PARAM SEMANTIC="DATA" xsi:type="VALUE">
            <SHORT-NAME>InfoType0x08Supported</SHORT-NAME>
            <LONG-NAME>InfoType 0x08 supported</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <PHYSICAL-DEFAULT-VALUE>supported</PHYSICAL-DEFAULT-VALUE>
            <DOP-REF ID-REF="DOP_InfoType0x08Supported"/>
        </PARAM>
    </PARAMS>
</STRUCTURE>
```

When the supported Vehicle Information elements are determined, the following TABLE is used to determine the content.

```
<TABLE ID="TAB_VehicleInformationReadInfoTypeData"
SEMANTIC="INFOTYPE">
    <SHORT-NAME>VehicleInformationReadInfoTypeData</SHORT-NAME>
    <LONG-NAME>Vehicle Information Read InfoType Data</LONG-NAME>
    <KEY-DOP-REF ID-REF="ES_OBDIICommonServices.DOP_INFOTYPEsSupported"/>
    <TABLE-ROW ID="TABROW_InfoType02VehicleIdentificationNumber">
        <SHORT-NAME>InfoType02VehicleIdentificationNumber</SHORT-NAME>
        <LONG-NAME>(InfoType 0x02) Vehicle Identification Number</LONG-NAME>
        <KEY>2</KEY>
        <STRUCTURE-REF ID-REF="STRUC_ InfoType02VehicleIdentificationNumber"/>
    </TABLE-ROW>
    …
    <TABLE-ROW ID="TABROW_InfoType06CalibrationVerificationNumbers">
        <SHORT-NAME>InfoType06CalibrationVerificationNumbers</SHORT-NAME>
        <LONG-NAME>(InfoType 0x06) Calibration Verification Numbers</LONG-NAME>
        <KEY>6</KEY>
        <STRUCTURE-REF ID-REF="STRUC_ InfoType06CalibrationVerificationNumbers"/>
    </TABLE-ROW>
    …
</TABLE>

<POS-RESPONSE ID="PRE_Service09RequestVehicleInformation">
    <SHORT-NAME>Service09RequestVehicleInformation</SHORT-NAME>
    <LONG-NAME>Service 0x09 - Request vehicle information</LONG-NAME>
    <PARAMS>
        <PARAM xsi:type="CODED-CONST">
            <SHORT-NAME>RequestVehicleInformationResponseSID</SHORT-NAME>
            <LONG-NAME>Request vehicle information response SID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <CODED-VALUE>73</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
                <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
        …
    </PARAMS>
</POS-RESPONSE>
```

9.6.9 GLOBAL-NEG-RESPONSE

The GLOBAL-NEG-RESPONSE according to 9.2.5 is specified as follows.

```
<GLOBAL-NEG-RESPONSE ID="GNR_OBDIIServicesNegativeResponse">
    <SHORT-NAME>OBDIIServicesNegative_Response</SHORT-NAME>
    <LONG-NAME>OBDII Services – Negative Response</LONG-NAME>
    <PARAMS>
        <PARAM SEMANTIC="SERVICE-ID" xsi:type="CODED-CONST">
            <SHORT-NAME>SID</SHORT-NAME>
            <LONG-NAME>SID</LONG-NAME>
            <BYTE-POSITION>0</BYTE-POSITION>
            <BIT-POSITION>0</BIT-POSITION>
            <CODED-VALUE>0</CODED-VALUE>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
            <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
            </PARAM>
        <PARAM SEMANTIC="SERVICE-ID" xsi:type="MATCHING-REQUEST-PARAM">
            <SHORT-NAME>SID_of_Request</SHORT-NAME>
            <LONG-NAME>SID of Request</LONG-NAME>
            <BYTE-POSITION>1</BYTE-POSITION>
            <BIT-POSITION>0</BIT-POSITION>
            <REQUEST-BYTE-POS>0</REQUEST-BYTE-POS>
            <BYTE-LENGTH>1</BYTE-LENGTH>
        </PARAM>
        <PARAM SEMANTIC="DATA" xsi:type="NRC-CONST">
            <SHORT-NAME>NRC</SHORT-NAME>
            <LONG-NAME>NRC</LONG-NAME>
            <BYTE-POSITION>2</BYTE-POSITION>
            <BIT-POSITION>0</BIT-POSITION>
            <CODED-VALUES>
            <CODED-VALUE>17</CODED-VALUE>
            <CODED-VALUE>18</CODED-VALUE>
            <CODED-VALUE>34</CODED-VALUE>
            <CODED-VALUE>120</CODED-VALUE>
            </CODED-VALUES>
            <DIAG-CODED-TYPE BASE-DATA-TYPE="A_UINT32" xsi:type="STANDARD-LENGTH-TYPE">
            <BIT-LENGTH>8</BIT-LENGTH>
            </DIAG-CODED-TYPE>
        </PARAM>
        <PARAM xsi:type="VALUE">
            <SHORT-NAME>NRC_Values</SHORT-NAME>
            <LONG-NAME>NRC Values</LONG-NAME>
            <BYTE-POSITION>2</BYTE-POSITION>
            <DOP-REF ID-REF="DOP_NRCs" />
        </PARAM>
    </PARAMS>
</GLOBAL-NEG-RESPONSE>
```