INTERNATIONAL STANDARD

ISO 14229-2

Second edition

Road vehicles — Unified diagnostic services (UDS) —

Part 2: Session layer services

Véhicules routiers — Services de diagnostic unifiés (SDU) —

Partie 2: Services de la couche session

PROOF/ÉPREUVE

Reference number
ISO 14229-2:2021(E)

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

Published in Switzerland

Contents

Page

Foreword

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

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

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

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

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

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

This second edition cancels and replaces the first edition (ISO 14229-2:2013), which has been technically revised.

The main changes compared to the previous edition are as follows:

— restructuration of the document;

— introduction of requirement numbers and names;

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

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

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

Introduction

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

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

— application layer (layer 7) specified in ISO 14229-1;

— presentation layer (layer 6) specified in ISO 14229-1;

— session layer services (layer 5) specified in this document (ISO 14229-2).

Figure 1 illustrates the ISO 14229 series reference according to OSI model.

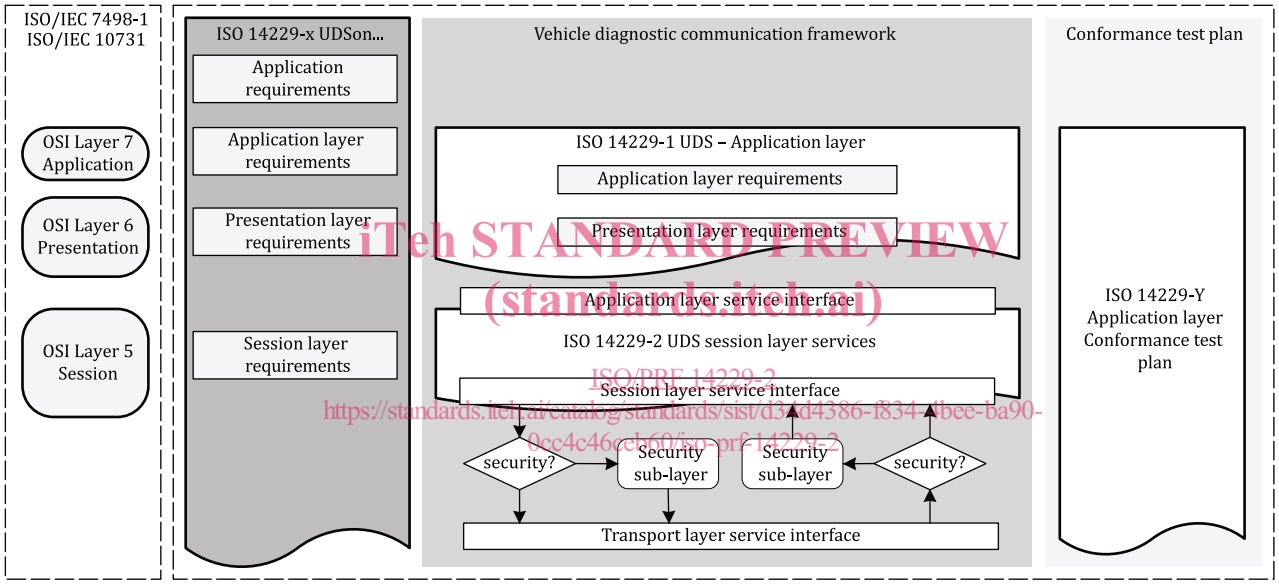


Figure 1 — ISO 14229 series reference according to OSI model

Road vehicles — Unified diagnostic services (UDS) —

Part 2:
Session layer services

1 Scope

This document specifies common session layer services and requirements to provide independence between unified diagnostic services (ISO 14229-1) and all transport protocols and network layer services (e.g. ISO 13400-2 DoIP, ISO 15765-2 DoCAN, ISO 10681-2 communication on FlexRay, ISO 14230-2 DoK-Line, and ISO 20794-3 CXPI).

This document specifies a common service primitive interface between OSI layer 5 (session) and layer 4 (transport) via so-called service request/indication/confirmation primitives.

2 Normative references

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

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

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

3 Terms and definitions

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

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

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

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

3.1
gateway
networking device that transfers the PDU on different OSI layers

EXAMPLE A network device that enables communication between control module networks that uses different communication protocols, different communication rates, etc. and that includes, but is not limited to, gateway functionalities like bridge, *switch* (3.3), *router* (3.2) or application layer routing.

3.2
router
networking device that transfers the PDU on OSI layers 3 and 4

3.3
switch
networking device that transfers the PDU on OSI layer 2

4 Symbols and abbreviated terms

4.1 Symbols

—	empty cell/undefined
t	time

4.2 Abbreviated terms

Diag	diagnostics
ECU	electronic control unit
N/A	not applicable
OSI	open systems interconnection
RDiag	remote diagnostics
S_AE	session layer address extension
S_Data	session layer data transfer service name
S_Length	session layer length of data
S_Mtype	session layer message type
S_PDU	session layer protocol data unit
S_SA	session layer source address
S_TA	session layer target address
S_TAtype	session layer target address type
SecureDiag	secure diagnostics
SecureRDiag	secure remote diagnostics
SI	service identifier
SOM	start of message
SPP	service primitive parameter

5 Conventions

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

Annex B describes vehicle diagnostic OSI layer architecture examples.

6 Session layer services

6.1 Service interface

The service interface defines a set of services that are needed to access the functions offered by the session layer, i.e. transmission/reception of data and setting of protocol parameters.

The service primitives define how a service user (e.g. diagnostic application) cooperates with a service provider (e.g. session layer). To define the services, three types of service primitives are specified:

— a service request primitive `S_Data.request`, used by the higher application layer to pass control information or data required to be transmitted to the session layer (i.e. the service provider is being requested by the service user to process control information or to transmit data);

— a service indication primitive `S_Data.indication`, used by the session layer to pass status information and received data to the higher application layer (i.e. the service user is being informed by the service provider about an internal event of the session layer or the service request of a peer protocol layer entity service user);

— a service confirmation primitive `S_Data.confirm` used by the session layer to pass status information to the application layer (i.e. the service user is being informed by service provider about the result of a preceding service request of the service user).

6.2 Service interface parameters

The session layer services have the same general format. Service primitives are written in the form:

```
service_name.type  (
                    parameter A,
                    parameter B,
                    parameter C,
                    [parameter X],
                    ...
                    )
```

where

— “service_name” is the name of the service (e.g. S_Data),

— “type” indicates the type of the service primitive (e.g. request, indication, confirm),

— “parameter A, ...” is the S_PDU (session layer protocol data unit) as a list of values passed by the service primitive (e.g. addressing information, data, length, result),

— “parameter A, parameter B, parameter C” are mandatory parameters that are included in all service calls, “[parameter X]” is an optional parameter that is included if specific conditions are fulfilled.

6.3 Service interface primitives

Figure 2 shows the session layer service primitives of a message transmission with a `T_Data.ind` reception at the session layer of the receiver side from the lower OSI layer.

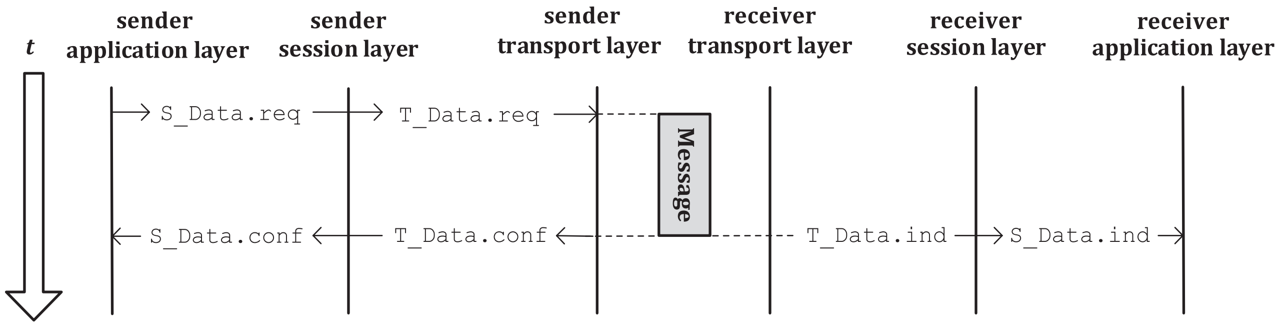
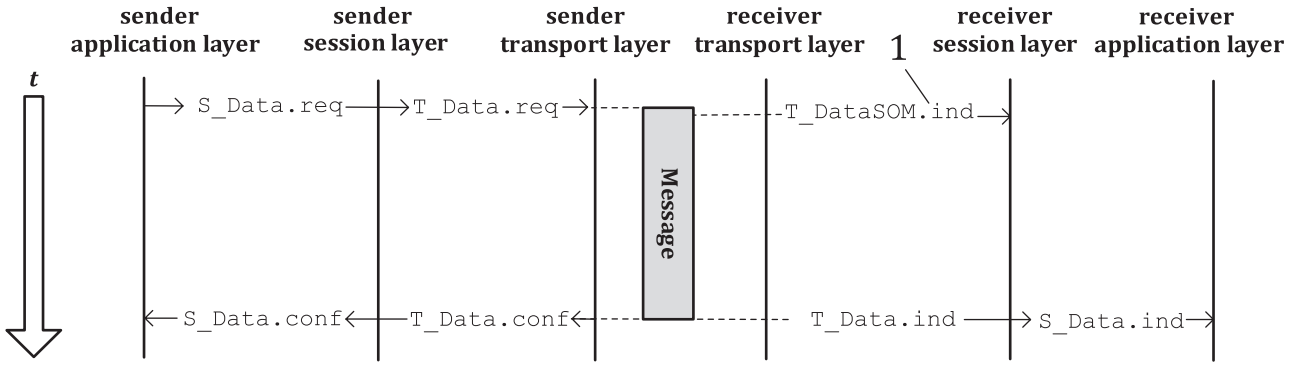


Figure 2 — Session layer service primitives – T_Data.ind message reception

Figure 3 shows the session layer service primitives of a message transmission with a `T_DataSOM.ind` reception at the beginning of the message and a `T_Data.ind` reception at the end of the message at the session layer of the receiver side from the lower OSI layer.



Key

1 transport layer StartOfMessage data indication, e.g. ISO 15765-2

Figure 3 — Session layer service primitives – T_DataSOM.ind and T_Data.ind reception

The following communication scenarios are distinguished:

a) physical communication during

   1) default session, and

   2) non-default session — session handling required;

b) functional communication during

   1) default session, and

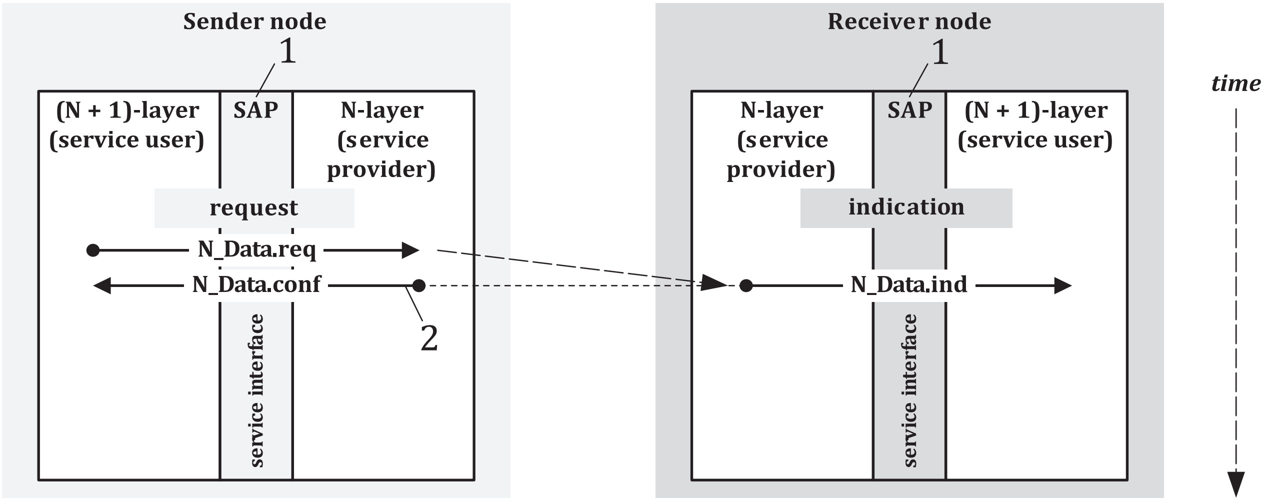
   2) non-default session — session handling required.

7 Service interface (SI) definition from application layer to session layer

7.1 SI — S_Data.req, S_Data.ind, and S_Data.conf service interface

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

Figure 4 shows the `S_Data.req`, `S_Data.ind`, and `S_Data.conf` service interface.



Key

1 service access point

2 read back from N-layer service provider

Figure 4 — S_Data.req and S_Data.ind service interface

7.2 SI — S_Data.req, S_Data.ind, and S_Data.conf service interface parameter mapping

This requirement specifies the application service interface and parameter mapping between the session layer and the lower OSI layers.

REQ	0.1 SI — S_Data.req, S_Data.ind, and S_Data.conf service interface parameter mapping
The `S_Data.req`, `S_Data.ind`, and `S_Data.conf` service interface parameter mapping shall be implemented as specified in Table 1.	

Table 1 — S_Data.req and S_Data.ind service interface parameter mapping

Application layer (service user)	Session layer (service provider)	A_Data.req and A_Data.ind parameter validity		
		.req	.ind	.conf
`A_Mtype`	`S_Mtype`	X	X	X
`A_AI[TAtype]`	`S_AI[TAtype]`	X	X	X
`A_AI[TA]`	`S_AI[TA]`	X	X	X
`A_AI[SA]`	`S_AI[SA]`	X	X	X
`A_AI[AE]`	`S_AI[AE]`	X	X	X
`A_Length`	`S_Length`	X	X	—
`A_Data`	`S_Data`	X	X	—
`A_Result`	`S_Result`	—	X	X

Key

X = supported

— = not supported

7.3 SI — S_PDU mapping onto T_PDU and vice versa for message transmission

The parameters of the session layer protocol data unit defined to request the transmission of a diagnostic service request/response are mapped onto the parameters of the transport layer protocol data unit for the transmission of a message in the client/server. Annex A specifies the T_PDU interface and shall be followed.

The parameters of the transport layer protocol data unit defined for the reception of a message are mapped as follows onto the parameters of the session layer protocol data unit for the confirmation/indication of the reception of a diagnostic response/request.

The transport layer confirmation of the successful transmission of the message (`T_Data.conf`) is forwarded to the application, because it is needed in the application for starting those actions, which shall be executed immediately after the transmission of the request/response message (e.g. ECUReset, bit rate change).

The transport layer indication for the reception of a StartOfMessage T_PDU (`T_DataSOM.ind`), e.g. ISO 15765-2 is not forwarded to the application layer, because it is only used within the session layer to perform the session layer timing (see Clause 9). Therefore, no mapping of the `T_DataSOM.ind` T_PDU onto an S_PDU is defined.

Table 2 defines the mapping of session layer S_PDU onto transport layer T_PDU and vice versa.

Table 2 — Mapping of session layer S_PDU onto transport layer T_PDU and vice versa

S_PDU parameter (session layer protocol data unit)	Description	T_PDU parameter (transport layer protocol data unit)	Description
`S_Mtype`	Session layer message type	`T_Ptype`	Transport layer segment type
`S_AI[TAtype]`	Session layer target address type	`T_AI[TAtype]`	Transport layer target address type
`S_AI[SA]`	Session layer source address	`T_AI[SA]`	Transport layer source address
`S_AI[TA]`	Session layer target address	`T_AI[TA]`	Transport layer target address
`S_AI[AE]`[a]	Session layer address extension	`T_AI[AE]`[a]	Transport layer address extension
`S_Data[1]` – `S_Data[n]`	Session layer data	`T_Data[1]` – `T_Data[n]`	Transport layer data
`S_Length`	Session layer data length	`T_Length`	Transport layer data length
`S_Result`	Session layer result	`T_Result`	Transport layer result

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

If `Mtype` = remote diagnostics/secure remote diagnostics, then the address information shall consist of the parameters `SA`, `TA`, `TAtype`, and `AE`.

7.4 SI — S_Data.req

The service primitive requests transmission of `S_Data` with `S_Length` number of bytes from the sender to the receiver peer entities identified by the address information in `S_AI[TAtype]`, `S_AI[SA]`, `S_AI[TA]`, and `S_AI[AE]`.

If the `S_Data.req` service is called, the session layer signals the completion (or failure) of the message transmission to the service user by means of the issuing of an `S_Data.conf` service call.

```
S_Data.req    (
              S_Mtype,
              S_AI[TAtype],
              S_AI[SA],
              S_AI[TA],
              [S_AI[AE]],
              S_Data[Data#1, Data#2, …, Data#n],
              S_Length
              )
```

7.5 SI — S_Data.ind

The `S_Data.indication` service is issued by the session layer. The service primitive shall indicate `S_Result` events and delivers `S_Data` with `S_Length` bytes received from a peer protocol entity identified

by the address information in S_AI[TAtype], S_AI[SA], S_AI[TA], and S_AI[AE] to the adjacent upper layer. The parameters S_Data and S_Length shall only be valid if S_Result equals S_OK.

```
S_Data.ind      (
                S_Mtype,
                S_AI[TAtype],
                S_AI[SA],
                S_AI[TA],
                [S_AI[AE]],
                S_Data[Data#1, Data#2, …, Data#n],
                S_Length,
                S_Result
               )
```

7.6 SI — S_Data.conf

The S_Data.conf service is issued by the session layer. The service primitive confirms the completion of an S_Data.req service identified by the address information in S_AI[TAtype], S_AI[SA], S_AI[TA], and S_AI[AE]. The parameter S_Result provides the status of the service request.

```
S_Data.conf     (
                S_Mtype,
                S_AI[TAtype],
                S_AI[SA],
                S_AI[TA],
                [S_AI[AE]],
                S_Result
                )
```


8 Service primitive parameters (SPP)

8.1 SPP – General

Clause 8 specifies the service primitive parameters and data types, which are used by the application layer services.

8.2 SPP – Data type definitions

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

8.3 SPP – S_Mtype, session layer message type

The parameter S_Mtype is used to identify the type and range of address information parameters included in a service call. This document specifies a range of two values for this parameter.

REQ	0.3 SPP – S_Mtype, session layer message type
The S_Mtype parameter shall be of data type Enum and shall be used to identify the message type and range of address information included in a service call.	

REQ	0.3 SPP – S_Mtype, session layer message type
—	If S_Mtype = diagnostics (Diag)/secure diagnostics (SecureDiag), then the address information shall consist of the parameters S_SA, S_TA and S_TAtype.
—	If S_Mtype = remote diagnostics (RDiag)/secure remote diagnostics (SecureRDiag), then the address information shall consist of the parameters S_SA, S_TA, S_TAtype and S_AE.
Range: [Diag, RDiag, SecureDiag, SecureRDiag]	

8.4 SPP – S_TAtype, session layer target address type

The parameter S_TAtype is a configuration attribute to the S_TA parameter. It is used to encode the communication model used by the communicating peer entities. Two communication models are specified: '1 to 1' communication, called physical addressing, and '1 to n' communication, called functional addressing.

REQ	0.4 SPP – S_TAtype, session layer target address type
The S_TAtype parameter shall be of data type Enum and shall be used to identify the target address type to be used with the request address.	
Range: [physical, functional]	

8.5 SPP – S_TA, session layer target address

S_TA parameter is used to encode the receiving session layer protocol entity. The parameter S_TA is used to encode client and server identifiers.

REQ	0.5 SPP – S_TA, session layer target address
The S_TA parameter shall be of data type Unsigned Word and shall contain the target address of the node.	
Range: [0000_{16} to $FFFF_{16}$]	

8.6 SPP – S_SA, session layer source address

S_SA parameter is used to encode the sending session layer protocol entity. The parameter S_SA is used to encode client and server identifiers.

REQ	0.6 SPP – S_SA, session layer source address
The S_SA parameter shall be of data type Unsigned Word and shall contain the source address of the node.	
Range: [0000_{16} to $FFFF_{16}$]	

8.7 SPP – S_AE, session layer address extension

S_AE parameter is used to encode the sending session layer protocol entity. The parameter S_AE is used to encode client and server identifiers.

REQ	0.7 SPP – S_AE, session layer address extension
The S_AE parameter shall be of data type Unsigned Word and shall contain the extended address of the node.	
Range: [0000_{16} to $FFFF_{16}$]	

8.8 SPP – S_Length, session layer length of S_Data

This parameter includes the length of data to be transmitted/received.

<table>
<tr><th>REQ</th><th>0.8 SPP – S_Length, session layer length of S_Data</th></tr>
<tr><td colspan="2">The S_Length parameter shall be of data type Unsigned Long and shall contain the length of the S_Data to be transmitted/received.
Range: [0000 0000$_{16}$ to FFFF FFFF$_{16}$]</td></tr>
</table>

8.9 SPP – S_Data, session layer data of PDU

This parameter includes data to be exchanged by the higher OSI layer entities.

<table>
<tr><th>REQ</th><th>0.9 SPP – S_Data, session layer data of PDU</th></tr>
<tr><td colspan="2">The S_Data parameter shall be of data type Byte Array and shall contain the message data content of the request or response message to be transmitted/received.
Range: [00$_{16}$ to FF$_{16}$]</td></tr>
</table>

8.10 SPP – S_Result, session layer result

This parameter contains the status related to the outcome of a service execution.

<table>
<tr><th>REQ</th><th>0.10 SPP – S_Result, session layer result</th></tr>
<tr><td colspan="2">The S_Result parameter shall be of data type Enum and shall contain the status relating to the outcome of a service execution (request field and response field sequence). If two or more errors are discovered at the same time, then the application layer entity shall set the appropriate error bit in the Result parameter.
Range: [OK, ERR_...]
The result OK shall be issued to the service user when the service execution is successfully completed. The OK shall be issued to a service user on both, the sender and receiver side.
The ERR_... shall be issued to the service user when an error is detected by a lower layer (provider). The ERR_... shall be issued to the service user on both, the sender and receiver side.</td></tr>
</table>

9 Timing parameter definition

9.1 General application timing considerations

9.1.1 Server

<table>
<tr><th>REQ</th><th>5.1 Timing parameter definition – Server – $t_{\text{P2 Server}}$</th></tr>
<tr><td colspan="2">A server shall use a single application timer ($t_{\text{P2\_Server}}$) implementation, which is triggered (started and stopped) by the T_Data service primitive interface (T_Data.req, T_Data.conf, T_DataSOM.ind, T_Data.ind).</td></tr>
</table>

<table>
<tr><th>REQ</th><th>5.2 Timing parameter definition – Server – $t_{\text{P2 Server Max}}$/$t_{\text{P2* Server Max}}$</th></tr>
<tr><td colspan="2">The $t_{\text{P2\_Server}}$ application timer shall be loaded with a $t_{\text{P2\_Server\_Max}}$/$t_{\text{P2*\_Server\_Max}}$ parameter value. Both parameters and values are specified in Table 3 and in Table 4.</td></tr>
</table>