
**Road vehicles — Communication on
FlexRay —**

**Part 2:
Communication layer services**

*Véhicules routiers — Communication par FlexRay —
Partie 2: Services de la couche de communication*



Reference number
ISO 10681-2:2010(E)

AVA Documents
Document Sharing Hub
© ISO 2010

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.



COPYRIGHT PROTECTED DOCUMENT

© ISO 2010

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

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	iv
Introduction	v
1 Scope	1
2 Normative references	1
3 Terms, definitions, symbols and abbreviated terms	2
4 Conventions	4
5 Communication layer overview	4
6 Communication layer services	10
7 Communication layer protocol	22
8 Data link layer usage	48
Annex A (informative) Implementation examples	58
Bibliography	64

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 10681-2 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 10681 consists of the following parts, under the general title *Road vehicles — Communication on FlexRay*:

- *Part 1: General information and use case definition*
- *Part 2: Communication layer services*

Introduction

This part of ISO 10681 is based on the Open Systems Interconnection (OSI) Basic Reference Model specified in ISO/IEC 7498-1 and ISO/IEC 10731, which structures communication systems into seven layers (see example in Table 1). When mapped on this model, this part of ISO 10681 incorporates the network layer (layer 3) and the transport layer (layer 4) services as communication layer services.

Table 1 — Example of enhanced diagnostic specifications according to the OSI layers

Applicability	OSI layers	Vehicle manufacturer enhanced diagnostics
Seven layers according to ISO/IEC 7498-1 and ISO/IEC 10731	Application layer	ISO 14229-1
	Presentation layer	N/A
	Session layer	ISO 14229-2
	Transport layer	ISO 10681-2
	Network layer	
	Data link layer	FlexRay Communications Systems Protocol Specification
	Physical layer	FlexRay Communications System Electrical Physical Layer Specification

Road vehicles — Communication on FlexRay —

Part 2: Communication layer services

1 Scope

This part of ISO 10681 specifies the requirements for a communication protocol tailored to meet the requirements of FlexRay-based vehicle network systems as specified in the FlexRay Communications Systems Protocol Specification. As the communication protocol combines the network layer and transport layer functionality (OSI layers 3 and 4), this part of ISO 10681 does not explicitly distinguish between these layer services.

The technical features of this communication protocol are as follows:

- transmit messages with known data length;
- transmit messages with unknown but finite data length;
- additional acknowledgement with retry mechanism;
- routing data on the fly;
- support of dynamic frame length.

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

ISO 7498-2, *Information processing systems — Open Systems Interconnection — Basic Reference Model — Part 2: Security Architecture*

ISO/IEC 7498-3, *Information technology — Open Systems Interconnection — Basic Reference Model: Naming and addressing*

ISO/IEC 7498-4, *Information processing systems — Open Systems Interconnection — Basic Reference Model — Part 4: Management framework*

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

3 Terms, definitions, symbols and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 7498-1, ISO 7498-2, ISO/IEC 7498-3 and ISO/IEC 7498-4 and the following apply.

3.1.1

communication layer

CL

layer that includes the network layer (layer 3) and the transport layer (layer 4)

3.1.2

protocol data unit

PDU

⟨layered system⟩ data unit that is specified in the protocol of a given layer

NOTE The protocol data unit contains user data of that layer and possible protocol control information. The protocol data unit of layer X is the service data unit of its lower layer (X – 1).

3.1.3

service data unit

SDU

⟨layered system⟩ set of data that is sent by a user of the service in a given layer

NOTE It is transmitted to a peer service user with no semantic change.

3.2 Abbreviated terms

For the purposes of this document, the following abbreviated terms apply.

ABT	abort
ACK	acknowledge
BC	bandwidth control
BfS	buffer size
BP	byte position
C_AI	communication address information
C_Ar	communication layer timing parameter A receiver
C_As	communication layer timing parameter A sender
C_Br	communication layer timing parameter B receiver
C_Bs	communication layer timing parameter B sender
C_Cr	communication layer timing parameter C receiver
C-Cs	communication layer timing parameter C sender
C_CT	communication type
C_Data	communication layer data transfer service name
C_PCI	communication protocol control information
C_SA	communication source address
C_TA	communication target address

C_TAType	communication target address type
CF	consecutive frame
CL	communication layer
COMM	communication
CTS	continue to send
C_PDU	communication layer protocol data unit
C_SDU	communication layer service data unit
DLL	data link layer
ECU	electronic control unit
EOB	end of block
FC	flow control
FIFO	first in first out
FPL	frame payload length
FR	FlexRay
FS	flow status
Ind	indication
LF	last frame
L_PDU	data link layer protocol data unit
max	maximum
ML	message length
MNPC	maximum numbers of PDUs per cycle
N/A	not applicable
N_PDU	network protocol data unit
NUM	number
OVFLW	overflow
PDU	protocol data unit
pLatestTx	latest point of transmission
Req	request
RET	retry
RX	receive
SCexp	separation cycle exponent
SDU	service data unit
SN	sequence number
STF	start frame
STFA	start frame acknowledged
STFU	start frame unacknowledged

UNEXP	unexpected
UNUM	unsigned numeric
TX	transmission
WFT	wait frame transmission
WT	wait

3.3 Symbols

Σ	summation
$\neq$	not equal

4 Conventions

ISO 10681 is based on the conventions discussed in the OSI Services Conventions (ISO/IEC 10731:1994) as they apply for diagnostic services.

5 Communication layer overview

5.1 General

This clause describes the overall functionality of the communication layer. This part of ISO 10681 specifies an unacknowledged and acknowledged communication layer protocol for the exchange of data with known or unknown length between network nodes, e.g. from ECU to ECU, or between a diagnostic tester equipment and an ECU. If the data to be transferred do not fit into a single L_PDU, a segmentation method is provided.

In order to describe the function of the communication layer, services provided to upper layers and the internal operation of the communication layer have to be considered.

5.2 Services provided by communication layer to upper layers

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

Four types of services are defined.

a) Communication services

These services, of which the following are defined, enable the transfer of up to 64 kbytes of data.

1) C_Data.request

This service is used to request the transfer of data. If necessary, the communication layer segments the data.

2) C_Data_STF.indication

This service is used to signal the beginning of a segmented message reception to the upper layer.

3) C_Data.indication

This service is used to provide received data to the upper layer.

4) C_Data.confirm

This service confirms to the upper layer that the requested service has been carried out (successfully or not).

b) Protocol parameter setting services

These services, of which the following are defined, enable the dynamic setting of protocol parameters.

1) C_ChangeParameter.request

This service is used to request the dynamic setting of specific internal parameters.

2) C_ChangeParameter.confirm

This service confirms to the upper layer that the request to change a specific parameter has been carried out (successfully or not).

3) C_GetParameter.request

This service is used to request the value of a communication layer parameter for a given connection.

4) C_GetParameter.confirm

This service is used to return the value of a communication layer parameter for a given connection.

c) Status services

1) C_GetStatus.request

This service is used to request the status of a transfer of data (transmit/receive).

2) C_GetStatus.confirm

This service confirms to the upper layer the status of the transfer of data (transmit/receive).

d) Transmission control services

C_Cancel.request

This service is used to request cancellation of an ongoing message transmission. This request can only be issued by the sender of a message, after the message transmission has been started via a C_Data.request service primitive.

5.3 Internal operation of communication layer

5.3.1 General

The internal operation of the communication layer provides the following methods for segmentation, transmission with flow control, and reassembly:

- transmission of a message with a known message length;
- transmission of a message with an unknown but finite message length;
- acknowledgement of the transmission with a retry mechanism.

The main purpose of the communication layer is to perform a transfer of a message that might or might not fit in a single FlexRay frame. Messages which do not fit into a single FlexRay frame are segmented into multiple parts, where each can be transmitted in a FlexRay frame.

5.3.2 Rules

The communication layer establishes certain rules for an ongoing transmission/reception, which are briefly described here for better understanding of the examples given in 5.3.3. For details on certain frames and their usage, see 7.4.

- Segmented messages are always concluded with a LastFrame. This frame might include the last bytes of the message if they did not fit completely into the previous ConsecutiveFrame.
- An acknowledged message transmission that is started with a StartFrame_ACK is always concluded by the receiver with a FlowControl_ACK after the reception of the last frame of the message, which can either be a single StartFrame or a LastFrame (in the case of a multiframe transmission).
- The request to acknowledge the reception of a block of ConsecutiveFrames via the ConsecutiveFrame_EOB shall always be confirmed by the receiver with a FlowControl frame which indicates the status of the reception (CTS, WAIT, etc.).
- In the case of an unknown message length transmission, the upper layer of the sender either asynchronously or synchronously provides data during the ongoing message transmission via the C_Data.request service primitive.
- In the case of a known message length transmission where the amount of data fits into a single STF and is completely available for transmission, a LastFrame will not be transmitted.
- Functional requests are transmitted as unsegmented and unacknowledged messages. Any other format will be ignored by the communication layer (no error will be communicated).
- If an STFU payload length is greater than the possible payload in an STFU, then the frame shall be ignored by the receiver.

5.3.3 Message sequence charts

All examples given assume a communication between the sender and the receiver on a single subnet, not taking into account communication over gateways.

Figure 1 shows an example of an unsegmented unacknowledged message transmission with a known message length.

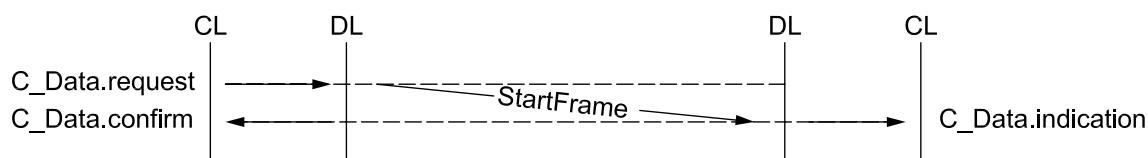


Figure 1 — Example of unsegmented unacknowledged message (known message length)

Figure 2 shows an example of an unsegmented acknowledged message transmission with a known message length.

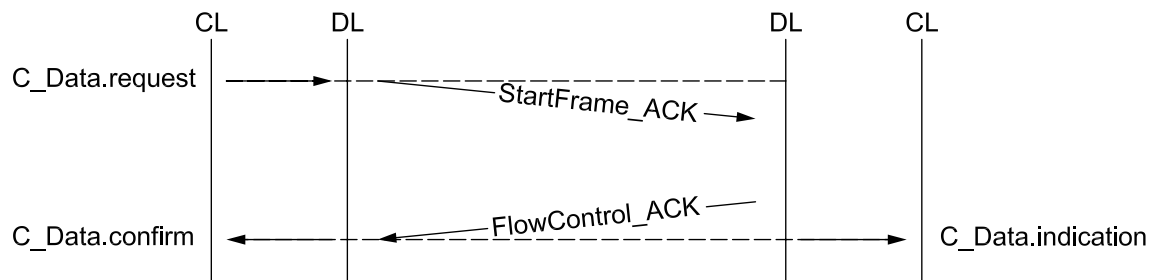


Figure 2 — Example of unsegmented acknowledged message (known message length)

Figure 3 shows an example of a segmented unacknowledged message transmission with a known message length.

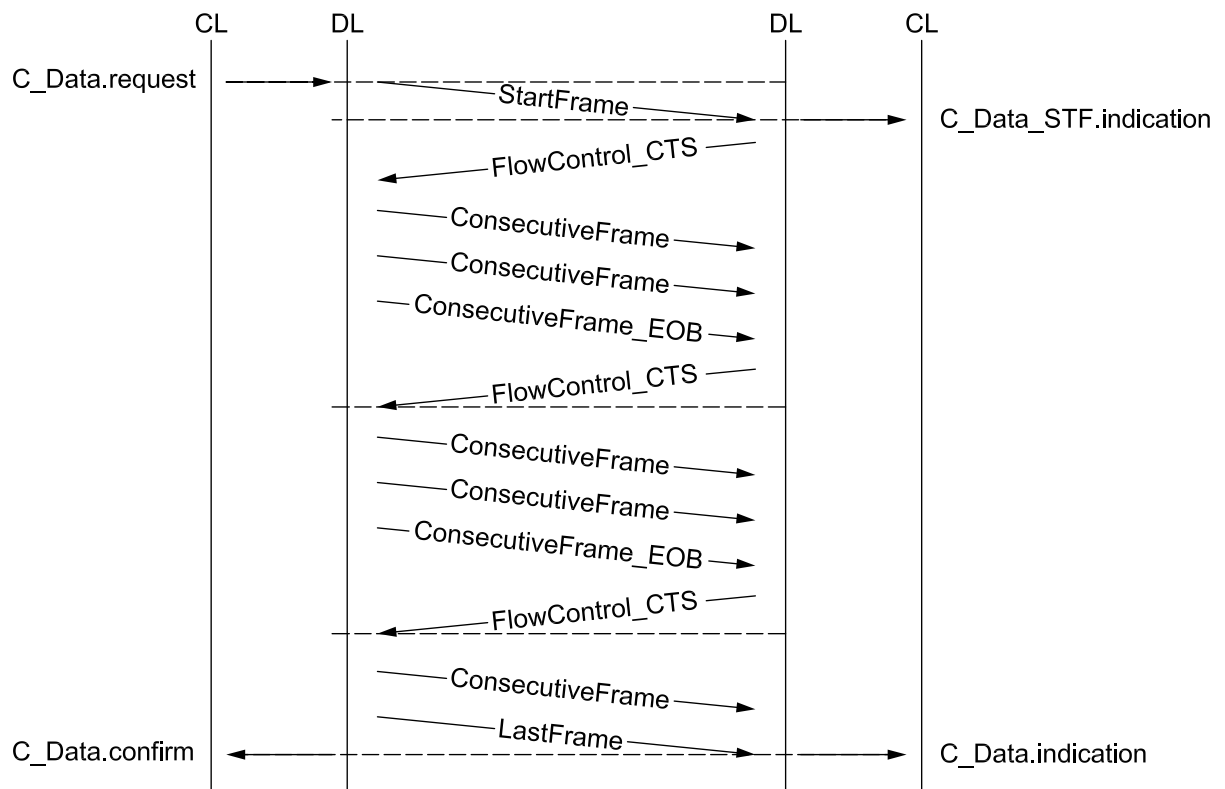


Figure 3 — Example of segmented unacknowledged message (known message length)

Figure 4 shows an example of a segmented acknowledged message transmission with a known message length.

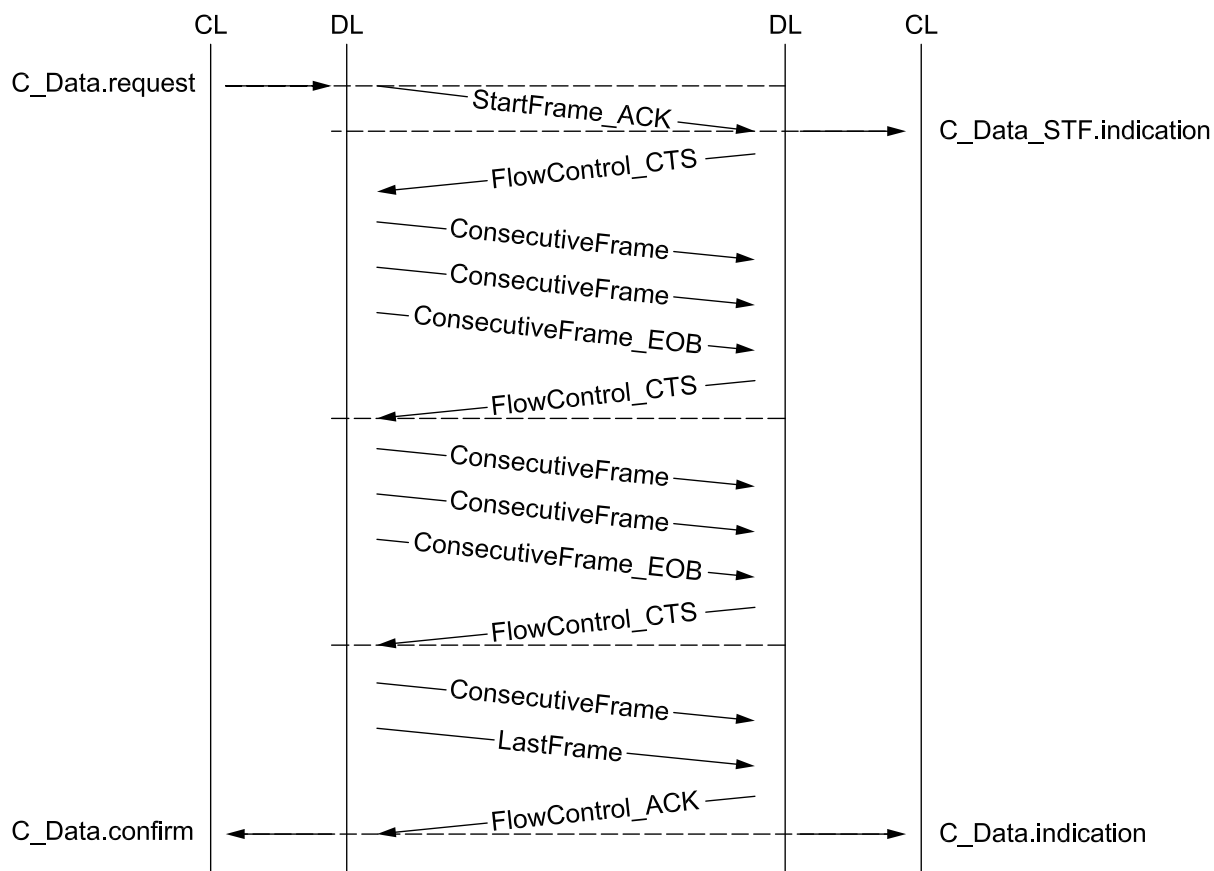


Figure 4 — Example of segmented acknowledged message (known message length)

Figure 5 shows an example of a segmented unacknowledged message transmission with an unknown message length.

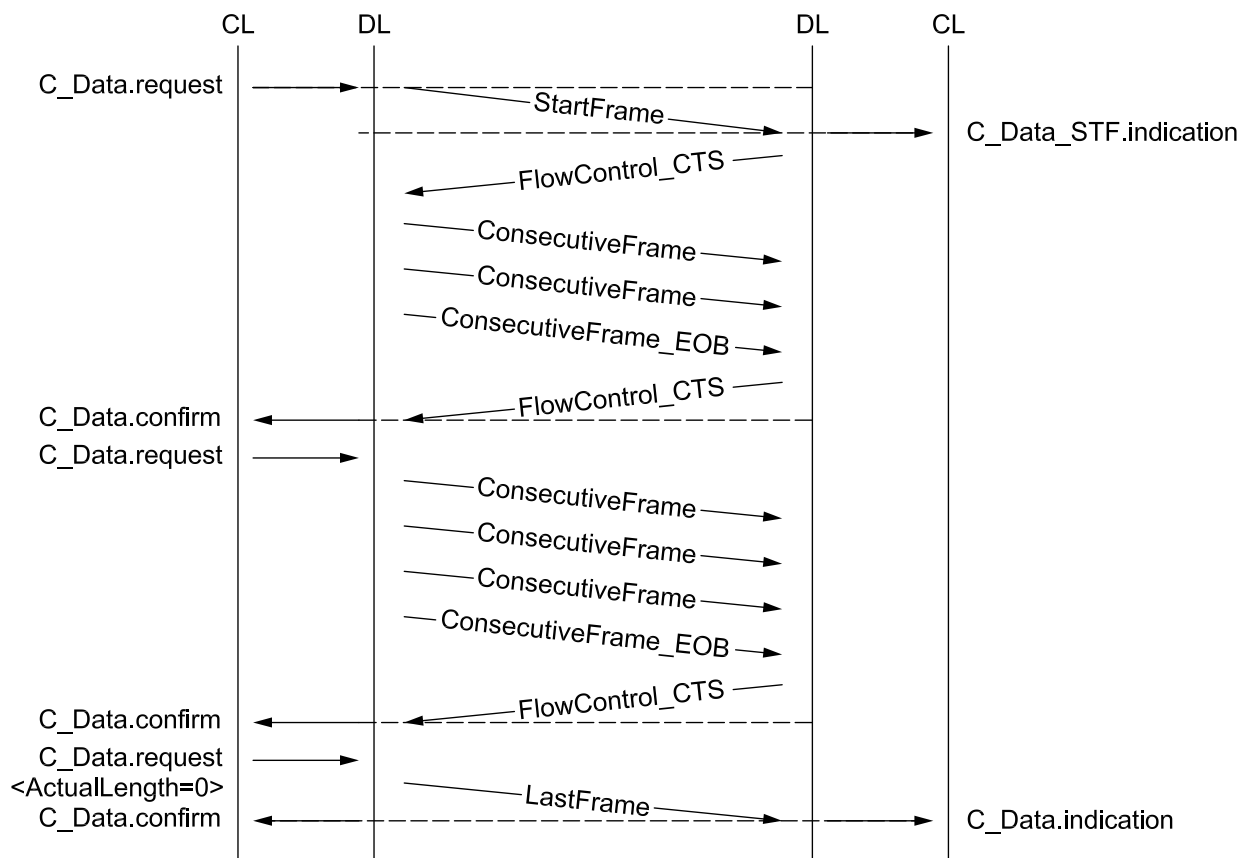


Figure 5 — Example of segmented unacknowledged message (unknown message length)

Figure 6 shows an example of a segmented acknowledged message transmission with an unknown message length.

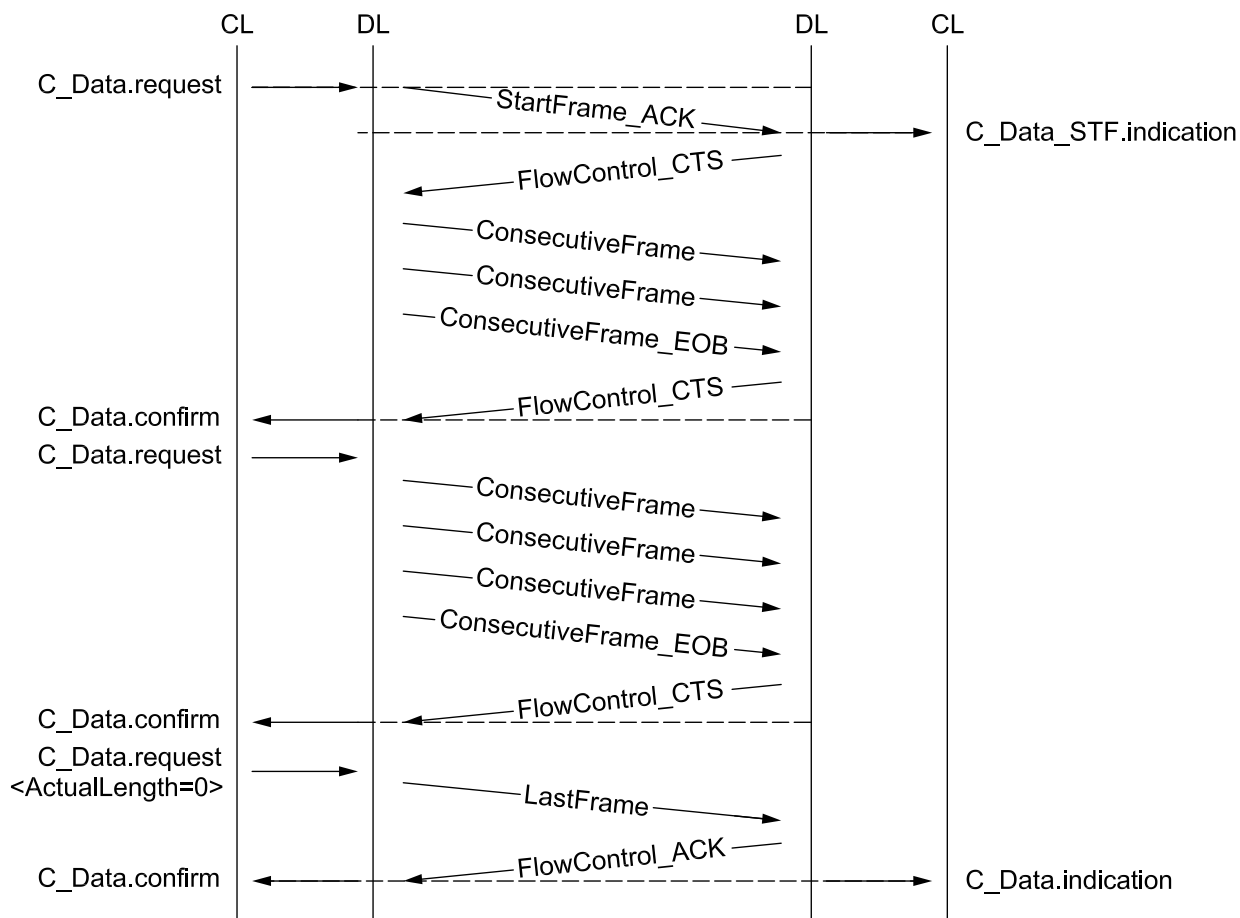


Figure 6 — Example of segmented acknowledged message (unknown message length)

6 Communication layer services

6.1 General

All communication layer services have the same general structure. To define the services, three types of service primitives are specified:

- a service *request* primitive, used by upper layers to pass control information and the data required to be transmitted to the communication layer;
- a service *indication* primitive, used by the communication layer to pass status information and the received data to upper layers;
- a service *confirmation* primitive used by the communication layer to pass status information to upper layers.

This service specification only specifies a set of service primitives that are independent of any implementation¹⁾. It does not specify an application programming interface.

All communication layer services have the same general format. Service primitives are written in the form:

```

service_name.type (
    parameter A,
    parameter B,
    parameter C,
    ...
)

```

where “service_name” is the name of the service. For C_Data, “type” indicates the type of the service primitive, and “parameter A, parameter B, parameter C, ...” are the C_SDU presented as a list of values passed by the service primitive.

6.2 Specification of communication layer service primitives

6.2.1 C_Data.request

The service primitive requests transmission of <MessageData> with <Length> number of bytes from the sender to the receiver peer entities identified by the address information in C_SA, C_TA, C_TAtype (see 6.3 for parameter definitions). Furthermore, <C_CT> is supported in order to distinguish between an acknowledged and an unacknowledged data transfer. For the support of an unknown message length transmission, an indication of the <ActualLength> number of bytes is provided.

Each time the C_Data.request service is called, the communication layer shall signal the completion (or failure) of the message transmission to the service user by means of the issuing of a C_Data.confirm service call:

```

C_Data.request (
    C_SA,
    C_TA,
    C_TAtype,
    C_CT,
    <Length>,
    <ActualLength>,
    <MessageData>
)

```

6.2.2 C_Data.confirm

The C_Data.confirm service is issued by the communication layer. The service primitive confirms the completion of a C_Data.request service identified by the address information in C_SA, C_TA and C_TAtype. The parameter <C_Result> provides the status of the service request (see 6.3 for parameter definitions).

```

C_Data.confirm (
    C_SA,
    C_TA,
    C_TAtype,
    <C_Result>
)

```

1) Annex A includes a detailed example on how the different scenarios of buffer resources in the sender and receiver are handled. Those examples make use of the service primitives as they would be API functions to describe the required behavior.

6.2.3 C_Data_STF.indication

The C_Data_STF.indication service is issued by the communication layer. The service primitive indicates to the upper layer the arrival of a start frame (STF) of a segmented message received from a peer protocol entity, identified by the address information in C_SA, C_TA and C_TAtype (see 6.3 for parameter definitions). This indication shall take place upon reception of the start frame (STF) of a segmented message.

```
C_Data_STF.indication (
    C_SA,
    C_TA,
    C_TAtype,
    <Length>
)
```

The C_Data_STF.indication service shall always be followed by a C_Data.indication service call from the communication layer, indicating the completion (or failure) of the message reception.

A C_Data_STF.indication service call shall only be issued by the communication layer if a correct start frame (STF) message segment has been received.

If the communication layer detects an error in a start frame (STF), then the message is ignored by the communication layer and no C_Data_STF.indication shall be issued to the upper layer.

6.2.4 C_Data.indication

The C_Data.indication service is issued by the communication layer. The service primitive indicates <C_Result> events and delivers <MessageData> with <Length> bytes received from a peer protocol entity identified by the address information in C_SA, C_TA and C_TAtype to the upper layer (see 6.3 for parameter definitions).

The parameters <MessageData> and <Length> are only valid if <C_Result> equals C_OK.

```
C_Data.indication (
    C_SA,
    C_TA,
    C_TAtype,
    <C_Result>
    <Length>,
    <MessageData>)

```

The C_Data.indication service call is issued after the reception of a single start frame (STF) that includes the complete message (unsegmented message) or as an indication of the completion (or failure) of a segmented message reception.

If the communication layer detects any type of error in a single start frame (STF), then the message shall be ignored by the communication layer and no C_Data.indication shall be issued to the adjacent upper layer.

6.2.5 C_ChangeParameter.request

The service primitive is used to request the change of an internal parameter's value on the local protocol entity identified by the address information in C_SA, C_TA and C_TAtype. The <Parameter_Value> is assigned to the <Parameter> (see 6.3 for parameter definitions).

A parameter change is always possible, except after reception of the start frame (C_Data_STF.indication) and until the end of the reception of the corresponding message (C_Data.indication).

```

C_ChangeParameter.request (
    C_SA,
    C_TA,
    C_TAtype,
    <Parameter>,
    <Parameter_Value>
)

```

This is an optional service that can be replaced by implementation of fixed parameter values.

6.2.6 C_ChangeParameter.confirm

The service primitive confirms the completion of a C_ChangeParameter.request service applying to a message identified by the address information in C_SA, C_TA and C_TAtype (see 6.3 for parameter definitions).

```

C_ChangeParameter.confirm (
    C_SA,
    C_TA,
    C_TAtype
    <Parameter>
    <Result_ChangeParameter>
)

```

6.2.7 C_GetParameter.request

The service primitive is used to request the configured internal parameter's value on the local protocol entity identified by the address information in C_SA, C_TA and C_TAtype (see 6.3 for parameter definitions). A parameter read is always possible.

```

C_GetParameter.request (
    C_SA,
    C_TA,
    C_TAtype
    <Parameter>
)

```

This is an optional service that can be replaced by implementation of fixed parameter values.

6.2.8 C_GetParameter.confirm

The service primitive confirms the completion of a C_GetParameter.request service applying to a message identified by the address information in C_SA, C_TA and C_TAtype (see 6.3 for parameter definitions).

```

C_GetParameter.confirm (
    C_SA,
    C_TA,
    C_TAtype
    <Parameter>
    <Result_GetParameter>
    <Parameter_Value>
)

```

6.2.9 C_GetStatus.request

The service primitive requests status of an ongoing communication is identified by the address information in C_SA, C_TA and C_TAtype (see 6.3 for parameter definitions).

```
C_GetStatus.request (
    C_SA,
    C_TA,
    C_TAtype
    <StatusParameter>)
```

6.2.10 C_GetStatus.confirm

The C_GetStatus.confirm service is issued by the communication layer. The service primitive confirms the completion of a C_GetStatus.request service identified by the address information in C_SA, C_TA and C_TAtype. The parameter <Status_Value> provides the status of the ongoing communication (see 6.3 for parameter definitions).

```
C_GetStatus.confirm (
    C_SA,
    C_TA,
    C_TAtype
    <StatusParameter>
    <Result_GetStatus>
    <Status_Value>
)
```

6.2.11 C_Cancel.request

The service primitive requests cancellation of an ongoing transmission is identified by the address information in C_SA, C_TA and C_TAtype (see 6.3 for parameter definitions).

```
C_Cancel.request (
    C_SA,
    C_TA,
    C_TAtype)
```

There is no confirmation service primitive defined. If a new transmission is issued via C_Data.request before the C_Cancel.request has been served completely, the upper layer will receive a C_Data.confirm with C_Result = C_TX_ON.

6.3 Service data unit specification

6.3.1 C_AI, Address Information

6.3.1.1 C_AI description

These parameters refer to addressing information. As a whole, the C_AI parameters are used to identify the source address (C_SA) and target address (C_TA) of message senders and recipients. See Table 2.

Table 2 — C_AI format

Byte 1	Byte 2	Byte 3	Byte 4
15	0	15	0
C_TA		C_SA	

6.3.1.2 C_SA, Communication Source Address

Type: 16 bit unsigned integer value

Range: 0000 to FFFF hex

Description: The C_SA parameter shall be used to encode the sending communication layer protocol entity.

NOTE The communication layer uses this parameter to identify corresponding C_PDUs, but neither modifies it nor defines values for the C_SA.

6.3.1.3 C_TA, Communication Target Address

Type: 16 bit unsigned integer value

Range: 0000 to FFFF hex

Description: The C_TA parameter shall be used to encode the receiving communication layer protocol entity.

NOTE The communication layer uses this parameter to identify corresponding C_PDUs, but does neither modify it nor define values for the C_TA.

6.3.1.4 C_TAtype, Communication Target Address type

Type: enumeration

Range: physical, functional

Description: The parameter C_TAtype is a configuration attribute to the C_TA parameter and implicitly part of C_TA. It shall be used to encode the communication model used by the communicating peer entities of the communication layer. Two communication models are specified: 1-to-1 communication, called “physical addressing” (unicast), and 1-to-*n* communication, called “functional addressing” (multicast/broadcast).

- Physical addressing (1-to-1 communication – unicast) shall be supported for all types of communication layer messages (unsegmented and segmented).
- Functional addressing (1-to-*n* communication – multicast/broadcast) shall only be supported for unsegmented, unacknowledged, known message length communication messages.

6.3.1.5 C_CT, Communication Type

Type: enumeration

Range: unacknowledged, acknowledged

Description: The C_CT parameter shall be used to specify the type of communication to be performed from a sender perspective. The parameter is implicitly configured within C_SA and C_TA.

6.3.2 <C_Result>

Type: enumeration

Range: C_OK, C_MOREDATA, C_TIMEOUT_A, C_TIMEOUT_Bs, C_TIMEOUT_Cr, C_WRONG_SN, C_ML_MISMATCH, C_INVALID_FS, C_UNEXP_PDU, C_WFT_OVRN, C_ABORT, C_CANCEL, C_WRONG_BP, C_BUFFER_OVFLW, C_ERROR

Description: This parameter contains the status relating to the outcome of a service execution. If two or more errors are discovered at the same time, then the communication layer entity shall use the first parameter value found in this list in the error indication to the upper layers.

NOTE Table 3 implicitly defines the priority of the result value evaluation by the order of the result values (the first entry has the highest priority).

Table 3 — C_Result values

C_Result	Sender	Receiver	Description
C_TIMEOUT_A	X		This value is issued to the protocol user when the timer C_As has passed its time-out value C_As _{max} .
		X	This value is issued to the protocol user when the timer C_Ar has passed its time-out value C_Ar _{max} .
C_TIMEOUT_Bs	X		This value is issued to the service user when the timer C_Bs has passed its time-out value C_Bs _{max} .
C_TIMEOUT_Cr		X	This value is issued to the service user when the timer C_Cr has passed its time-out value C_Cr _{max} .
C_ML_MISMATCH		X	This value indicates a mismatch of the message length information (ML) within the STF PDU, the sum of the FPL information and the LF C_PDU (check of the ML = 0 in the case an unknown message length is supported).
C_WRONG_SN		X	This value is issued to the service user upon reception of an unexpected sequence number (PCI.SN) value. For more details, see 7.5.7.2 <i>SN Error handling</i> .
C_INVALID_FS	X		This value is issued to the service user when an invalid or unknown FlowStatus value has been received in a flow control (FC) C_PDU.
C_UNEXP_PDU		X	This value is issued to the service user upon reception of an unexpected protocol data unit.
C_WFT_OVRN		X	The receiver shall wait until the transmission of the (WFT _{max} + 1) FC_WT C_PDU would occur, but does not transmit this FC C_PDU and instead indicates the error C_WFT_OVRN to the upper layer. The sender detects a C_Bs timeout and terminates the transmission.
C_ABORT	X		This value is issued to the service user upon reception of a flow control (FC) C_PDU with FlowStatus = ABT. It indicates that the receiver is currently busy and cannot take the request at that point in time; therefore, the transmission of the segmented message was aborted.
C_CANCEL	X		This value is issued to the service user upon the cancellation of an ongoing transmission as requested by the upper application via the cancel service primitive.
C_BUFFER_OVFLW	X		This value is issued to the service user upon reception of a flow control (FC) C_PDU with FlowStatus = OVFLW. It indicates that the buffer on the receiver side of a segmented message transmission cannot store the number of bytes specified by the StartFrame MessageLength (STF_ML) parameter in the StartFrame; therefore, the transmission of the segmented message was aborted.
C_WRONG_BP	X		This value is issued to the service user upon reception of a flow control (FC) C_PDU with FlowStatus = ACK_RET, where BP points to a position outside the buffer of the sender. Therefore, the sender cannot provide the requested data anymore. The segmented message was aborted.
C_ERROR	X	X	This is the general error value. It shall be issued to the service user when an error has been detected by the communication layer and no other parameter value can be used to better describe the error.