

# Electricity metering — Data exchange for meter reading, tariff and load control —

Part 47: COSEM transport layers for  
IPv4 networks

The European Standard EN 62056-47:2007 has the status of a  
British Standard

ICS 91.140.50; 35.100.40





## National foreword

This British Standard was published by BSI. It is the UK implementation of EN 62056-47:2007. It is identical with IEC 62056-47:2006.

The UK participation in its preparation was entrusted to Technical Committee PEL/13, Electricity meters.

A list of organizations represented on PEL/13 can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 30 March 2007

© BSI 2007

ISBN 978 0 580 50394 8

### Amendments issued since publication

Amd. No.	Date	Comments

**Electricity metering -  
Data exchange for meter reading, tariff and load control -  
Part 47: COSEM transport layers for IPv4 networks  
(IEC 62056-47:2006)**

Equipements de mesure  
de l'énergie électrique -  
Echange des données pour la lecture  
des compteurs, le contrôle des tarifs  
et de la charge -  
Partie 47 : Couches de transport COSEM  
pour réseaux IPv4  
(CEI 62056-47:2006)

Messung der elektrischen Energie -  
Zählerstandsübertragung,  
Tarif- und Laststeuerung -  
Teil 47: COSEM Transportschichten  
für IPv4 Netzwerke  
(IEC 62056-47:2006)

This European Standard was approved by CENELEC on 2006-12-01. CENELEC members are bound to comply with the CEN/CENELEC Internal Regulations which stipulate the conditions for giving this European Standard the status of a national standard without any alteration.

Up-to-date lists and bibliographical references concerning such national standards may be obtained on application to the Central Secretariat or to any CENELEC member.

This European Standard exists in three official versions (English, French, German). A version in any other language made by translation under the responsibility of a CENELEC member into its own language and notified to the Central Secretariat has the same status as the official versions.

CENELEC members are the national electrotechnical committees of Austria, Belgium, Bulgaria, Cyprus, the Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, the Netherlands, Norway, Poland, Portugal, Romania, Slovakia, Slovenia, Spain, Sweden, Switzerland and the United Kingdom.

# CENELEC

European Committee for Electrotechnical Standardization  
Comité Européen de Normalisation Electrotechnique  
Europäisches Komitee für Elektrotechnische Normung

**Central Secretariat: rue de Stassart 35, B - 1050 Brussels**

## Foreword

The text of document 13/1386/FDIS, future edition 1 of IEC 62056-47, prepared by IEC TC 13, Electrical energy measurement, tariff- and load control, was submitted to the IEC-CENELEC parallel vote and was approved by CENELEC as EN 62056-47 on 2006-12-01.

The following dates were fixed:

- latest date by which the EN has to be implemented  
at national level by publication of an identical  
national standard or by endorsement (dop) 2007-09-01
- latest date by which the national standards conflicting  
with the EN have to be withdrawn (dow) 2009-12-01

The International Electrotechnical Commission (IEC) and CENELEC draw attention to the fact that it is claimed that compliance with this International Standard / European Standard may involve the use of a maintenance service concerning the stack of protocols on which the present standard IEC 62056-47 / EN 62056-47 is based.

The IEC and CENELEC take no position concerning the evidence, validity and scope of this maintenance service.

The provider of the maintenance service has assured the IEC that he is willing to provide services under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the provider of the maintenance service is registered with the IEC. Information may be obtained from:

DLMS <sup>1)</sup> User Association  
Geneva / Switzerland  
[www.dlms.ch](http://www.dlms.ch)

Annex ZA has been added by CENELEC.

---

## Endorsement notice

The text of the International Standard IEC 62056-47:2006 was approved by CENELEC as a European Standard without any modification.

---

---

<sup>1)</sup> Device Language Message Specification



# CONTENTS

1	Scope.....	4
2	Normative references .....	4
3	Terms, definitions and abbreviations .....	5
4	Overview .....	5
5	The COSEM connection-less, UDP-based transport layer.....	7
5.1	General .....	7
5.2	Service specification for the COSEM UDP-based transport layer.....	8
5.3	Protocol specification for the COSEM UDP-based transport layer.....	11
6	The COSEM connection-oriented, TCP-based transport layer.....	13
6.1	General .....	13
6.2	Service specification for the COSEM TCP-based transport layer .....	14
6.3	Protocol specification for the COSEM TCP-based transport layer .....	24
	Annex A (informative) Converting OSI-style transport layer services to and from RFC-style TCP function calls .....	31
	Annex ZA (normative) Normative references to international publications with their corresponding European publications.....	39
	Bibliography.....	37
	INDEX .....	38
	Figure 1 – COSEM as a standard Internet application protocol .....	6
	Figure 2 – Transport layers of the COSEM_on_IP profile .....	7
	Figure 3 – Services of the COSEM connection-less, UDP-based transport layer .....	8
	Figure 4 – The wrapper protocol data unit (WPDU) .....	12
	Figure 5 – The COSEM connection-less, UDP-based transport layer PDU (UDP-PDU) .....	12
	Figure 6 – Services of the COSEM connection-oriented, TCP-based transport layer .....	15
	Figure 7 – The TCP packet format .....	25
	Figure 8 – Figure TCP connection establishment .....	26
	Figure 9 – Disconnecting a TCP connection.....	27
	Figure 10 – Data communication using the COSEM TCP-based transport layer .....	29
	Figure 11 – High-level state transition diagram for the wrapper sub-layer .....	29
	Figure A.1 – TCP connection state diagram .....	31
	Figure A.2 – MSC and state transitions for establishing a transport layer and TCP connection .....	32
	Figure A.3 – MSC and state transitions for closing a transport layer and TCP connection .....	33
	Figure A.4 – Polling the TCP sub-layer for TCP abort indication .....	34
	Figure A.5 – Sending an APDU in three TCP packets .....	35
	Figure A.6 – Receiving the message in several packets.....	36
	Table 1 – Reserved wrapper Port numbers in the UDP-based COSEM profile.....	13
	Table 2 – Reserved wrapper port numbers in the TCP-based COSEM profile.....	26



# ELECTRICITY METERING – DATA EXCHANGE FOR METER READING, TARIFF AND LOAD CONTROL –

## Part 47: COSEM transport layers for IPv4 networks

### 1 Scope

This part of IEC 62056 specifies the transport layers for COSEM communication profiles for use on IPv4 networks.

These communication profiles contain a connection-less and a connection-oriented transport layer, providing OSI-style services to the service user COSEM application layer. The connection-less transport layer is based on the Internet standard User Datagram Protocol. The connection-oriented transport layer is based on the Internet standard Transmission Control Protocol.

Although the major part of the COSEM transport layers is the UDP and TCP as they are specified in the relevant Internet standards, they include an additional sub-layer, called wrapper.

Annex A shows how the OSI-style transport layer services can be converted to and from UDP and TCP function calls.

### 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.

IEC 60050-300:2001, *International Electrotechnical Vocabulary (IEV) – Electrical and electronic measurements and measuring instruments – Part 311: General terms relating to measurements – Part 312: General terms relating to electrical measurements – Part 313: Types of electrical measuring instruments – Part 314: Specific terms according to the type of instrument.*

IEC 62051:1999, *Electricity metering – Glossary of terms*

IEC 62051-1:2004, Ed.1., *Electricity metering – Data exchange for meter reading, tariff and load control – Glossary of terms – Part 1: Terms related to data exchange with metering equipment using DLMS/COSEM*

IEC 62056-53, *Electricity metering – Data exchange for meter reading, tariff and load control*  
– Part 53: COSEM application layer<sup>3</sup>

IEC 62056-62, *Electricity metering – Data exchange for meter reading, tariff and load control*  
– Part 62: Interface classes<sup>3</sup>

STD0005 – *Internet Protocol*

Author: J. Postel

Date: September 1981

Also: RFC0791, RFC0792, RFC0919, RFC0922, RFC0950, RFC1112



STD0006 – *User Datagram Protocol*

*Author: J. Postel*

*Date: 28 August 1980*

*Also: RFC0768*

STD0007 – *Transmission Control Protocol*

*Author: J. Postel*

*Date: September 1981*

*Also: RFC0793*

See also Bibliography for other related Internet RFCs.

### 3 Terms, definitions and abbreviations

#### 3.1 Terms and definitions

For the purposes of this document, the definitions given in IEC 60050-300, IEC 62051 and IEC 62051-1 apply.

#### 3.2 Abbreviations

APDU	Application Layer Protocol Data Unit
COSEM	COmpanion Specification for Energy Metering
COSEM_on_IP	The TCP-UDP/IP based COSEM communication profile
IP	Internet Protocol
PDU	Protocol Data Unit
PAR	Positive Acknowledgement with Retransmission
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
WPDU	Wrapper Protocol Data Unit

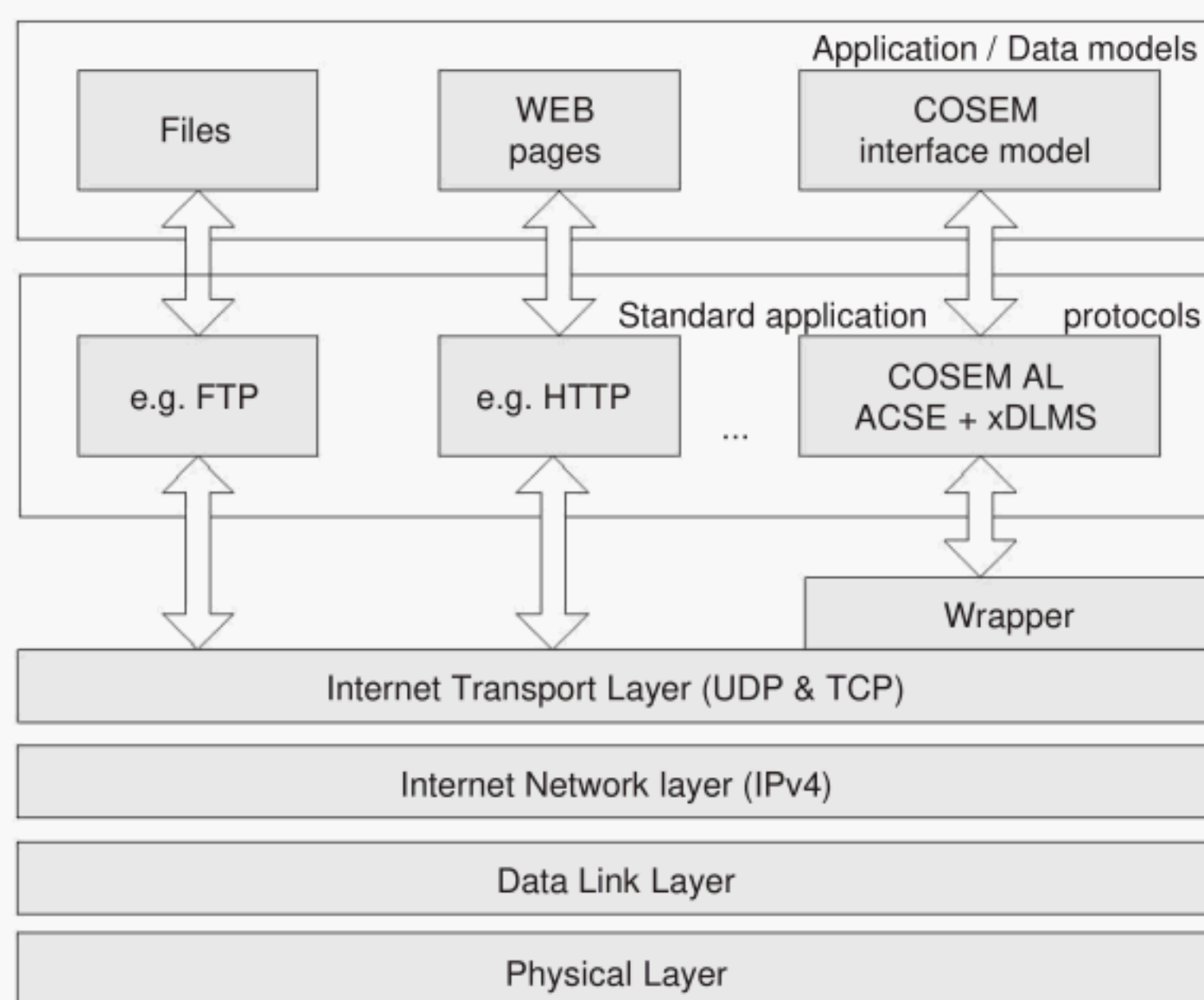
### 4 Overview

This standard specifies two transport layers for the COSEM\_on\_IP communication profiles: a connection-less transport layer, based on UDP, Internet standard STD0006 and a connection-oriented transport layer, based on TCP, Internet standard STD0007.

In these profiles, the COSEM application layer uses the services of one of these transport layers, which use then the services of the Internet Protocol (IPv4) network layer to communicate with other nodes connected to the abstract IPv4 network.

When used in these profiles, the COSEM application layer can be considered as another Internet standard application protocol (like the well-known HTTP, FTP or SNMP) and it may co-exist with other Internet application protocols, as shown in

Figure 1.



**Figure 1 – COSEM as a standard Internet application protocol**

As the COSEM application layer specified in IEC 62056-53 uses and provides OSI-style services, a wrapper has been introduced between the UDP/TCP layers and the COSEM application layer.

Therefore, the COSEM transport layers consist of a wrapper sub-layer and the UDP or TCP transport layer.

The wrapper sub-layer is a lightweight, nearly state-less entity: its main function is to adapt the OSI-style service set, provided by the COSEM transport layer, to UDP or TCP function calls and vice versa.

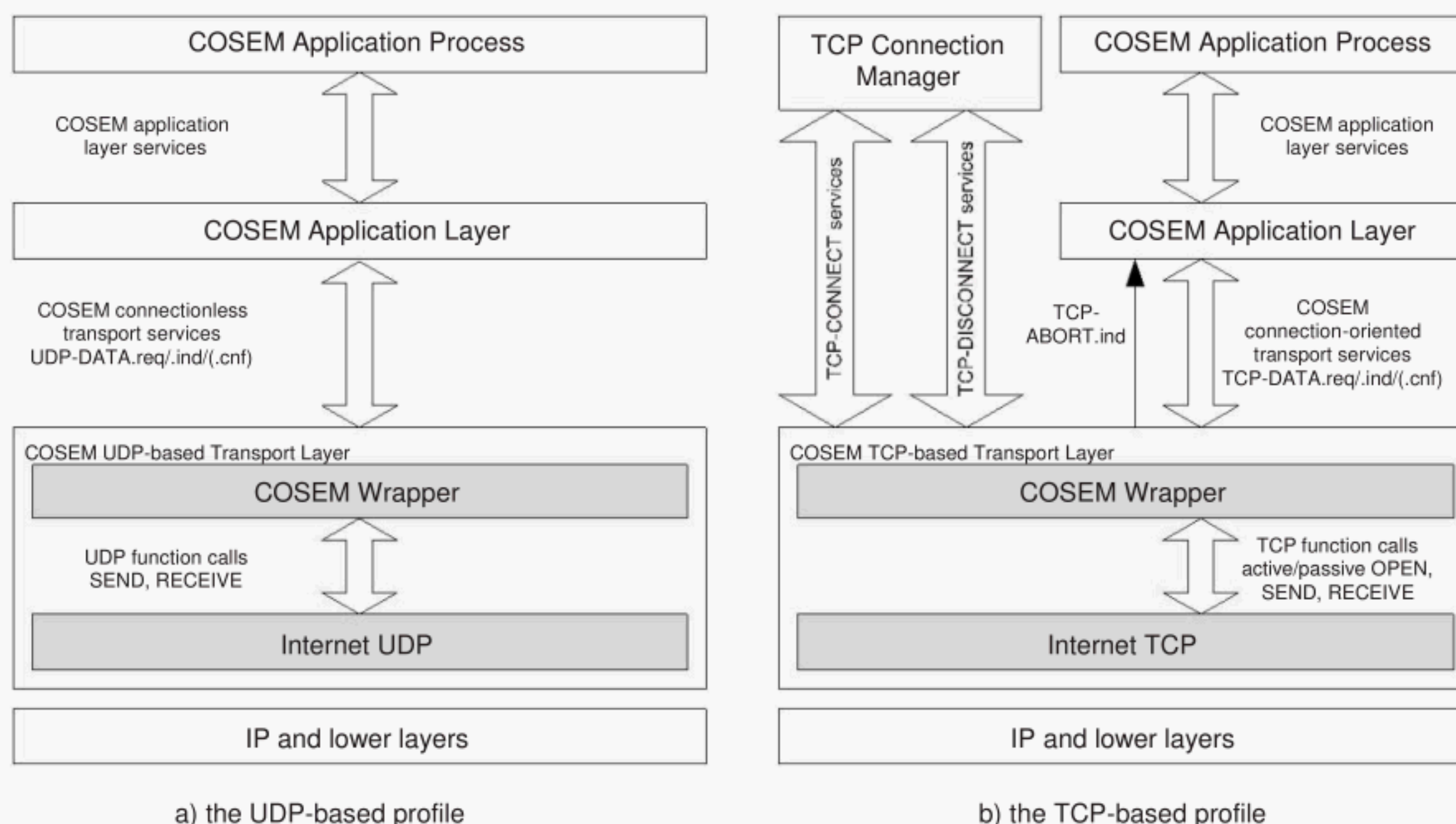
In addition, the wrapper sub-layer has the following functions:

- it provides an additional addressing capability (wPort) on top of the UDP/TCP port;
- it provides information about the length of the data transported. This feature helps the sender to send and the receiver to recognize the reception of a complete APDU, which may be sent and received in multiple TCP packets.

As specified in IEC 62056-53, B.3.3, the COSEM application layer is listening only on one UDP or TCP port. On the other hand, as defined in IEC 62056-62, a COSEM physical device may host several client application processes or server logical devices. The additional addressing capability provided by the wrapper sub-layer allows identifying these application processes.

The structure of the COSEM transport layer and their place in COSEM-on\_IP is shown in Figure 2.





**Figure 2 – Transport layers of the COSEM\_on\_IP profile**

The service user of the UDP-DATA and the TCP-DATA services is the COSEM application layer. On the other hand, the service user of the TCP-CONNECT and TCP-DISCONNECT services is the TCP Connection Manager Process. The COSEM TCP-based transport layer also provides a TCP-ABORT.indication service to the service user COSEM application layer.

## 5 The COSEM connection-less, UDP-based transport layer

### 5.1 General

The COSEM connection-less transport layer is based on the User Datagram Protocol (UDP) as specified in STD0006.

UDP provides a procedure for application programs to send messages to other programs with a minimum of protocol mechanism. On the one hand, the protocol is transaction oriented, and delivery and duplicate protection are not guaranteed. On the other hand, UDP is simple, it adds a minimum of overhead, it is efficient and easy to use. Several well-known Internet applications, like SNMP, DHCP, TFTP, etc. take advantage of these performance benefits, either because some datagram applications do not need to be reliable or because the required reliability mechanism is ensured by the application itself. Request/response type applications, like a confirmed COSEM application association established on the COSEM UDP-based transport layer, then invoking confirmed COSEM data communication services is a good example for this second category. Another advantage of UDP is that being connection-less, it is easily capable of multi- and broadcasting.

UDP basically provides an upper interface to the IP layer, with an additional identification capability, the UDP port number. This allows distinguishing between application processes, hosted in the same physical device and identified by its IPv4 address<sup>2</sup>.

<sup>2</sup> The addressing/identification scheme for the COSEM\_on\_IP profiles is defined in IEC 62056-53, B.3.3.



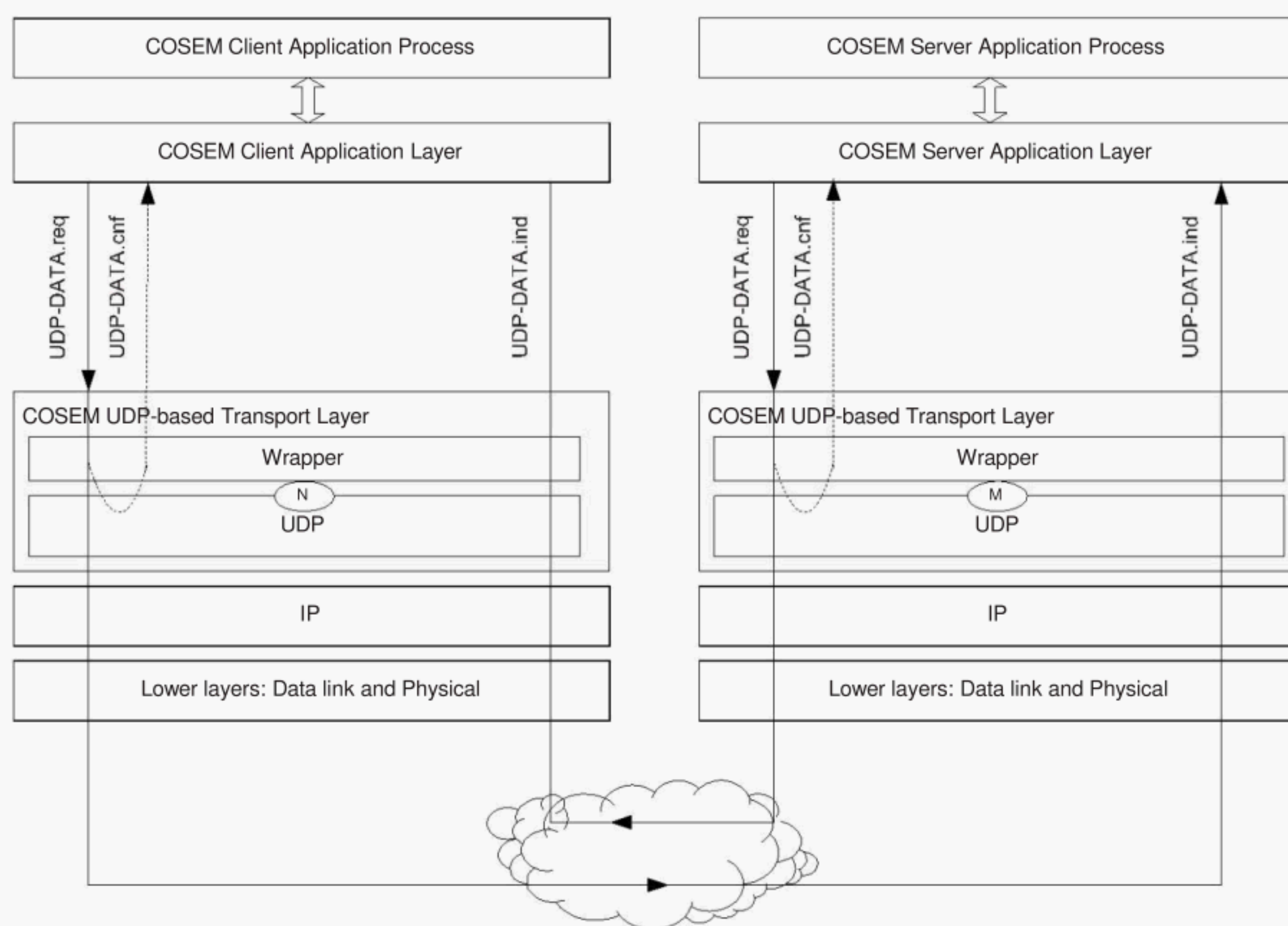
As already mentioned in Clause 4, the COSEM application layer is listening only on one UDP port. On the other hand, as defined in IEC 62056-62, a COSEM physical device may host several client application processes or server logical devices. The additional addressing capability provided by the wrapper sub-layer, using the wrapper port (wPort) numbers on top of the UDP/TCP port numbers allows identifying these application processes.

The wrapper also adds length information to the APDU to be transported.

## 5.2 Service specification for the COSEM UDP-based transport layer

### 5.2.1 General

The COSEM UDP-based transport layer provides the same set of services both at the Client and at the Server sides, as shown in Figure 3.



**Figure 3 – Services of the COSEM connection-less, UDP-based transport layer**

The COSEM UDP-based transport layer provides only data communication services: the connection-less UDP-DATA services. The service set for the UDP-DATA services is the same at both the client and server sides: consequently, the service specification for these services is the same for both the client and server transport layers.

The .request and .indication service primitives are mandatory. The implementation of the local .confirm service primitive is optional.

NOTE The APDU pre-fixed with the header by the wrapper sub-layer must fit in a single UDP datagram.

## 5.2.2 The UDP-DATA services

### 5.2.2.1 UDP-DATA.request

#### *Function*

This service primitive is invoked by the service user COSEM application layer to request the transmission of an APDU to the peer COSEM application layer.

#### *Service parameters*

The semantics of the primitive is as follows:

```

UDP-DATA.request
(
    Local_wPort,
    Remote_wPort,
    Local_UDP_Port,
    Remote_UDP_Port,
    Local_IP_Address,
    Remote_IP_Address,
    Data_Length,
    Data
)
    
```

The Local\_wPort, Local\_UDP\_Port and Local\_IP\_Address parameters indicate wrapper Port number, UDP Port number and IP Address parameters belonging to the device/application process requesting to send the Data.

The Remote\_wPort, Remote\_UDP\_Port and Remote\_IP\_Address parameters indicate the wrapper Port number, UDP Port number and IP Address parameters belonging to the device/application process to which the Data is to be transmitted.

The Local\_UDP\_Port and Remote\_UDP\_Port parameters identify the local and remote UDP ports respectively. Note, that as no well-known port number is reserved for COSEM communications, the value of these parameters must be in the non-privileged range (above 1024).

The Data\_Length parameter indicates the length of the Data parameter in bytes.

The Data parameter contains the COSEM APDU to be transferred to the peer application layer.

#### *Use*

The UDP-DATA.request primitive is invoked by either the client or the server COSEM application layer to request sending an APDU to a single peer application layer, or, in the case of multi- or broadcasting, to multiple peer application layers.

The reception of this service primitive shall cause the wrapper sub-layer to pre-fix the wrapper header to the APDU received, and then to call the SEND() function of the UDP sub-layer with the properly formed WPDU, see at 5.3.2, as DATA. The UDP sub-layer shall transmit the WPDU to the peer wrapper sub-layer as described in STD0006.



### 5.2.2.2 UDP-DATA.indication

#### *Function*

This service primitive is invoked by the COSEM transport layer to indicate to the service user COSEM application layer that an APDU has been received from a remote application layer.

#### *Service parameters*

The semantics of the primitive is as follows:

```

UDP-DATA.indication
(
    Local_wPort,
    Remote_wPort,
    Local_UDP_Port,
    Remote_UDP_Po
rt,
    Local_IP_Addres
s,
    Remote_IP_Addr
ess,
    Data_Length,
    Data
)

```

The Local\_wPort, Local\_UDP\_Port and Local\_IP\_Address parameters indicate wrapper Port number, UDP Port number and IP Address parameters belonging to the device/application process receiving the Data.

The Remote\_wPort, Remote\_UDP\_Port and Remote\_IP\_Address parameters indicate the wrapper Port number, UDP Port number and IP Address parameters belonging to the device/application process, which has sent the data.

The Local\_UDP\_Port and Remote\_UDP\_Port parameters identify the local and remote UDP ports respectively. Note, that as no well-known port number is reserved for COSEM communications, the value of these parameters must be in the non-privileged range (above 1024).

The Data\_Length parameter indicates the length of the Data parameter in bytes.

The Data parameter contains the COSEM APDU received from the peer application layer.

#### *Use*

The UDP-DATA.indication service primitive is used to indicate to the service user COSEM application layer that an APDU from the peer layer entity has been received.

The primitive is generated following the reception of an UDP Datagram by the UDP sub-layer, if both the Local\_UDP\_Port and Local\_wPort parameters of the message received contain valid wPort numbers, meaning that there is a COSEM application process in the receiving device bound to the given port numbers. Otherwise, the message received shall simply be discarded.

### 5.2.2.3 UDP-DATA.confirm

#### *Function*

This optional service primitive is invoked by the COSEM transport layer to confirm to the service user COSEM application layer the result of the previous UDP-DATA.request service. The service represents a local confirmation only.



### *Service parameters*

The semantics of the primitive is as follows:

```

UDP-DATA.confirm
(
    Local_wPort,
    Remote_wPort,
    Local_UDP_Port,
    Remote_UDP_Port,
    Local_IP_Address,
    Remote_IP_Address,
    Result
)
    
```

The Local\_wPort, Remote\_wPort, Local\_UDP\_Port, Remote\_UDP\_Port, Local\_IP\_Address and Remote\_IP\_Address parameters carry the same values as the corresponding UDP-DATA.request service being confirmed.

The value of the Result parameter indicates whether the COSEM UDP-based transport layer was able to send the requested UDP Datagram (OK) or not (NOK).

### *Use*

If implemented, this service primitive is used to confirm the result of a previous UDP-DATA.request service. It is locally generated and indicates only whether the Data in the .request primitive could be sent or not. In other words, an UDP-DATA.confirm with Result == OK means only that the Data has been sent, and does not mean that the Data has been (or will be) successfully delivered to the destination.

## **5.3 Protocol specification for the COSEM UDP-based transport layer**

### **5.3.1 General**

As it is shown on the left side of Figure 2, the COSEM UDP-based transport layer includes the Internet standard UDP layer, as specified in Internet standard STD0006, and the COSEM-specific light-weight wrapper sub-layer.

In this communication profile, the wrapper sub-layer is a state-less entity: its only roles are to ensure source and destination COSEM application process identification using the wPort numbers and to provide conversion between the OSI-style UDP-DATA.xxx service invocations and the SEND() and RECEIVE() interface functions provided by the standard UDP.

Although it is not necessary in the UDP-based profile, in order to have the wrapper protocol control information – in other words the wrapper header – the same in both COSEM transport layers, the wrapper sub-layer shall also include the Data Length information in the wrapper protocol data unit.

### **5.3.2 The wrapper protocol data unit (WPDU)**

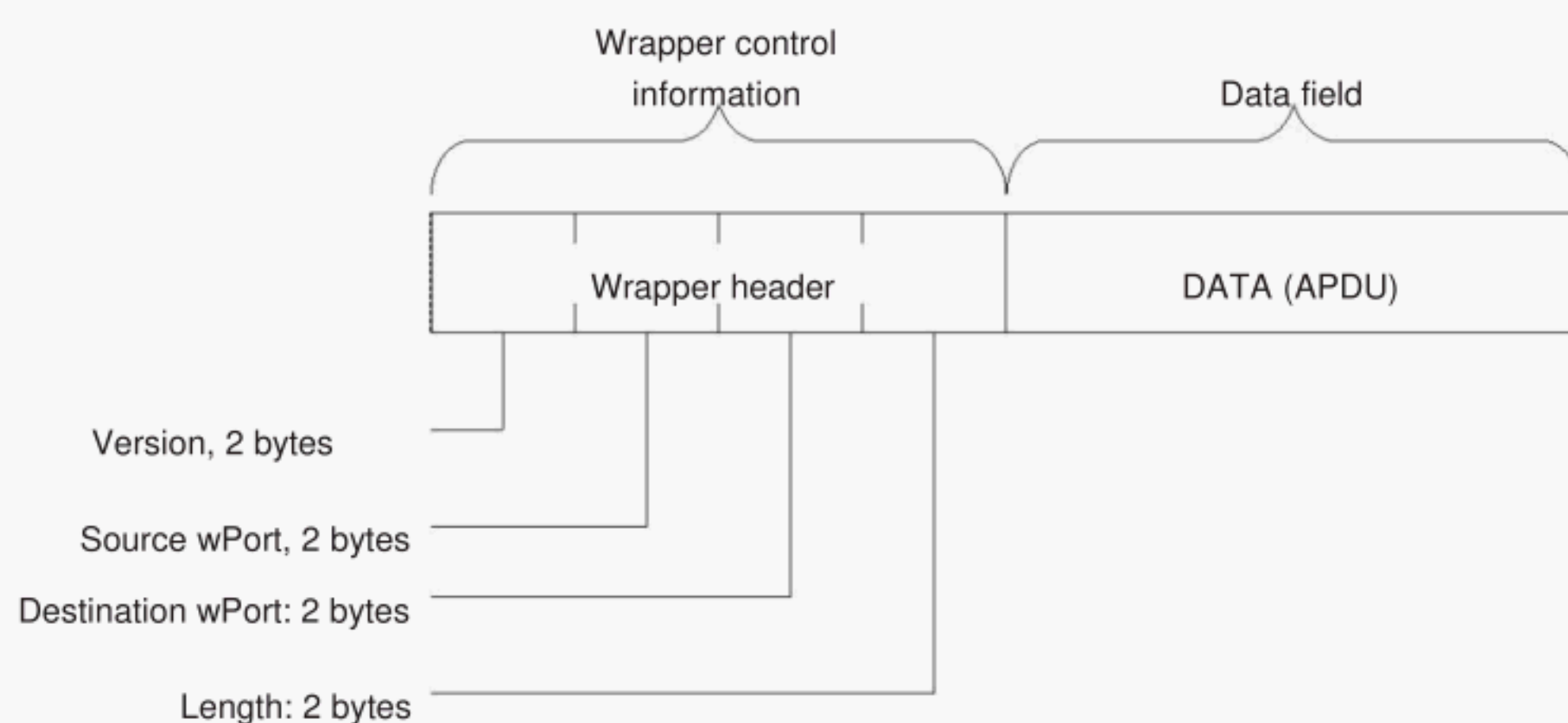
The WPDU consists of two parts:

- the wrapper header part, containing the wrapper protocol control information, and
- the Data part, containing the DATA parameter – the COSEM APDU – of the corresponding UDP-DATA.xxx service invocation.

The wrapper header includes four fields, each 16 bits long, as follows:

- Version: this 16 bit long unsigned integer value carries the version number identifying the version of the wrapper. Its value is controlled by the DLMS User Association. The current value is 0x0001. Note, that in later versions the wrapper header may have a different structure.
- Source wPort: this 16 bit long unsigned integer value carries the wPort number identifying the sender application process.
- Destination wPort: this 16 bit long unsigned integer value carries the wPort number identifying the destination application process.
- Data length: this 16 bit long, unsigned integer value indicates the length of the DATA field of the WPDU (the length of the APDU transported).

The Wrapper Protocol Data Unit (WPDU) is shown in Figure 4.

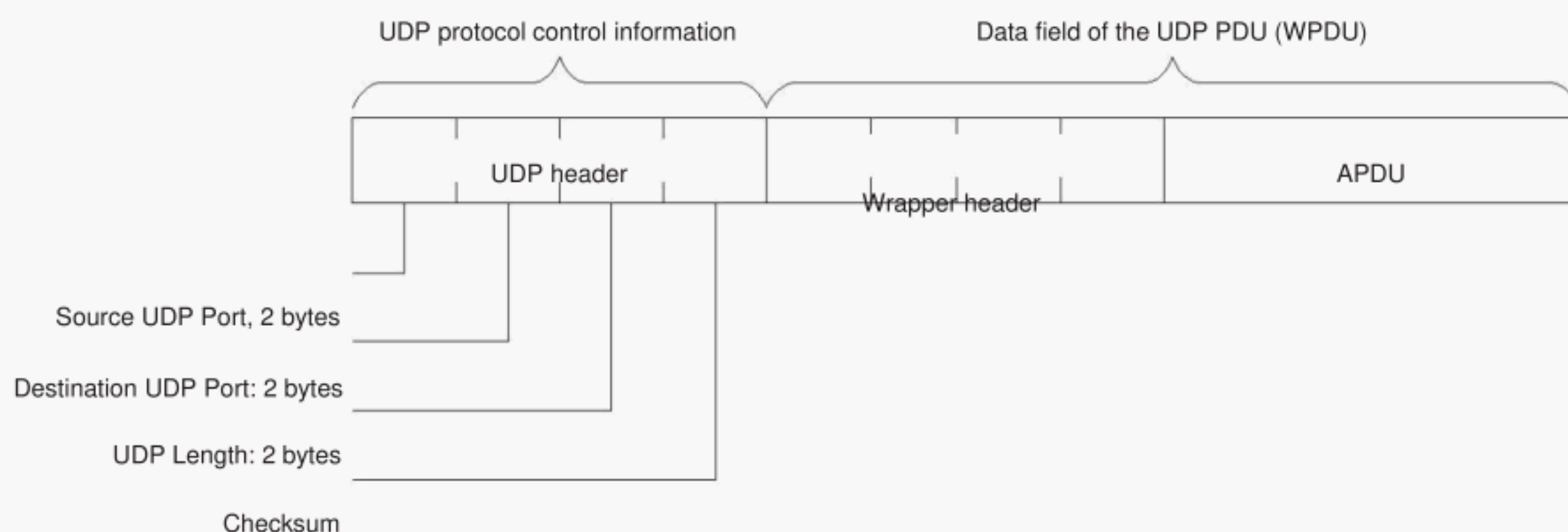


NOTE The maximum length of the APDU should be eight bytes less than the maximum length of the UDP datagram.

**Figure 4 – The wrapper protocol data unit (WPDU)**

### 5.3.3 The COSEM UDP-based transport layer protocol data unit

In this profile, WPDUs shall be transmitted in UDP Datagrams. The UDP Datagram is as it is specified in Internet standard STD0006, and it shall encapsulate the WPDU, as it is shown in Figure 5.



**Figure 5 – The COSEM connection-less, UDP-based transport layer PDU (UDP-PDU)**



From the external point of view, the COSEM connection-less transport layer PDU is an ordinary UDP Datagram: any COSEM specific element, including the wrapper-specific header is inside the UDP Data field. Consequently, standard UDP implementations can be (re-)used to easily implement this profile.

The Source and Destination UDP ports may refer to either local or remote UDP ports depending on the direction of the data transfer (i.e. from the point of view of the sending device, the Source UDP port in a Datagram corresponds to the Local\_UDP\_port, but from the point of view of the receiving device, the Source UDP port of a Datagram corresponds to the Remote\_UDP\_Port service parameter).

According to the UDP specification, filling the Source UDP Port and Checksum fields with real data is optional. A zero value – all bits are equal to zero – of these fields indicates that in the given UDP Datagram the field is not used. However, in the COSEM\_on\_IP profile, the Source UDP Port field shall always be filled with the Source UDP port number.

#### 5.3.4 Reserved wrapper port numbers (wPorts)

The following wPort Numbers are reserved:

**Table 1 – Reserved wrapper Port numbers in the UDP-based COSEM profile**

Client side reserved addresses	
	Wrapper Port Number
No-station	0x0000
Client Management Process	0x0001
Public Client	0x0010
Server side reserved addresses	
	Wrapper Port Number
No-station	0x0000
Management Logical Device	0x0001
Reserved	0x0002...0x000F
All-station (Broadcast)	0x007F

#### 5.3.5 Protocol state machine

As the wrapper sub-layer in this profile is state-less, for all other protocol related issues – protocol state machine, etc. – the governing rules are as they are specified in the Internet standard STD0006. The only supplementary rule is concerning discarding inappropriate messages: messages with no valid Destination wPort number – meaning that there are no COSEM application processes in the receiving device bound to this wPort number – shall be discarded by the wrapper sub-layer.

## 6 The COSEM connection-oriented, TCP-based transport layer

### 6.1 General

The COSEM connection-oriented transport layer is based on the connection-oriented Internet transport protocol, called Transmission Control Protocol or TCP.



TCP is an end-to-end reliable protocol. This reliability is ensured by a conceptual “virtual circuit”, using a method called “Positive Acknowledgement with Retransmission” or PAR. It provides acknowledged data delivery, error detection, data re-transmission after an acknowledgement time-out, etc., therefore is dealing with lost, delayed, duplicated or erroneous data packets. In addition, TCP offers an efficient flow control mechanism and full-duplex operation, too.

TCP, as a connection-oriented transfer protocol involves three phases: connection establishment, data exchange and connection release. Consequently, the COSEM TCP-based transport layer provides OSI-style services to the service user(s) for all three phases:

- for the connection establishment phase, TCP-CONNECT services are provided to the service user TCP connection manager process;
- for the data communication phase, TCP-DATA services are provided to the service user COSEM application layer;
- for the connection closing phase, TCP-DISCONNECT services are provided to the service user TCP connection manager process;
- in addition, a TCP-ABORT.indication service is provided to the service user COSEM application layer.

The COSEM connection-oriented, TCP-based transport layer contains the same wrapper sub-layer as the COSEM UDP-based transport layer. In addition to transforming OSI-style services to and from TCP function calls, this wrapper provides additional addressing and length information.

The COSEM connection-oriented, TCP-based transport layer is specified in terms of services and protocols.

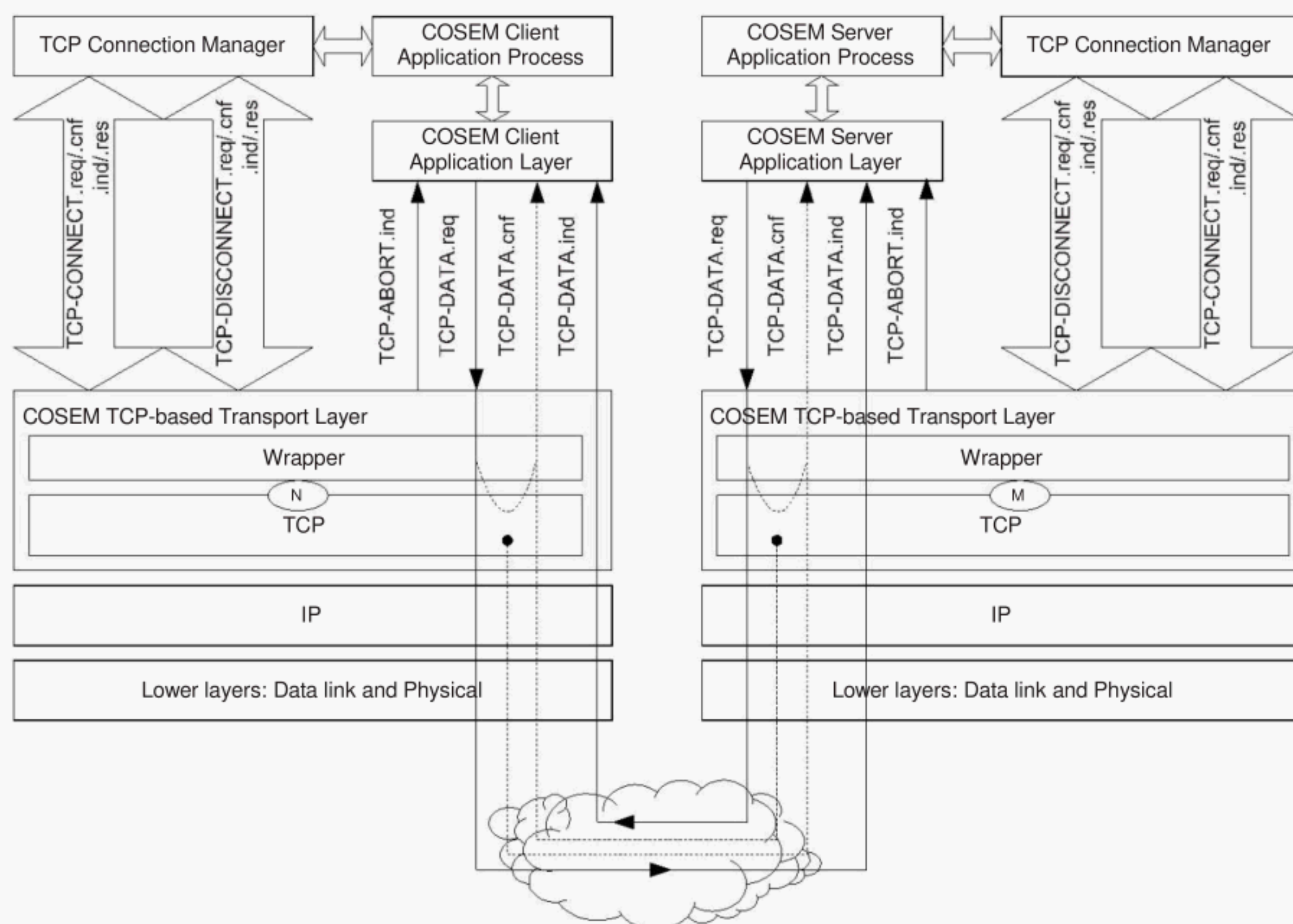
The conversion between OSI-style services and TCP function calls is presented in Annex A.

## **6.2 Service specification for the COSEM TCP-based transport layer**

### **6.2.1 General**

The COSEM connection oriented, TCP-based transport layer provides the same set of services both at the client and at the server sides, as it is shown in Figure 6.





**Figure 6 – Services of the COSEM connection-oriented, TCP-based transport layer**

In this communication profile, the full set of TCP connection management services (TCP-CONNECT and TCP-DISCONNECT) is provided both at the client and at the server sides. The purpose of this is to allow the server to initiate and to release a TCP connection, too <sup>3</sup>.

The service user of the TCP connection management services is not the COSEM application layer, but the TCP connection manager process. The specification of this process is out of the scope of this standard – however, the COSEM application layer sets some requirements concerning this. See B.3.4 of IEC 62056-53.

An additional COSEM-ABORT.indication service is provided to indicate to the COSEM application layer the disruption or disconnection of the supporting TCP connection.

Like in the COSEM UDP-based transport layer, the TCP-DATA.confirm service is also optional. However, the TCP-DATA.request service can be confirmed either locally or remotely.

## 6.2.2 The TCP-CONNECT services

### 6.2.2.1 TCP-CONNECT.request

#### Function

This service primitive is invoked by the service user TCP connection manager process to request the establishment of a TCP connection.

<sup>3</sup> Application association establishment is performed by the client application process.



*Service parameters*

The semantics of the primitive is as follows:

```
TCP-CONNECT.request
(
    Local_TCP_Port,
    Remote_TCP_Port,
    Local_IP_Address,
    Remote_IP_Address
)
```

The Local\_TCP\_Port and Remote\_TCP\_Port parameters identify the local and remote TCP ports respectively. As no well-known port number is reserved for COSEM communications, the value of these parameters must be in the non-privileged range (above 1024).

The Local\_IP\_Address and Remote\_IP\_Address parameters indicate the IP Address of the physical device requesting the TCP connection and of the target physical device, to which the TCP connection requested is to be established.

*Use*

The service user TCP connection manager process invokes this service primitive to initiate a TCP connection establishment with the peer COSEM TCP-based transport layer.

**6.2.2.2 TCP-CONNECT.indication***Function*

This service primitive is invoked by the COSEM TCP-based transport layer following the reception of a TCP packet, indicating to the TCP connection manager process that a remote device is requesting a new TCP connection.

*Service parameters*

The semantics of the primitive is as follows:

```
TCP-CONNECT.indication
(
    Local_TCP_Port,
    Remote_TCP_Port,
    Local_IP_Address,
    Remote_IP_Address
)
```

The Local\_TCP\_Port- and Remote\_TCP\_Port parameters indicate the two TCP ports between which the requested TCP connection is to be established.

The Local\_IP\_Address and Remote\_IP\_Address parameters indicate the IP addresses of the two devices participating in the TCP connection.

*Use*

When the COSEM TCP-based transport layer receives a TCP packet indicating that a remote TCP layer is requesting a new TCP connection, it shall indicate this to the service user TCP connection manager process using this service primitive.



### 6.2.2.3 TCP-CONNECT.response

#### *Function*

This service primitive is invoked by the TCP connection manager process to indicate to the COSEM TCP-based transport layer whether the previously requested TCP connection has been accepted. The TCP connection manager cannot reject a requested connection.

#### *Service parameters*

The semantics of the primitive is as follows:

```
TCP-CONNECT.response
(
    Local_TCP_Port,
    Remote_TCP_Port,
    Local_IP_Address,
    Remote_IP_Address,
    Result
)
```

The Local\_TCP\_Port and Remote\_TCP\_Port parameters indicate the two TCP ports between which the requested TCP connection is being established.

The Local\_IP\_Address and Remote\_IP\_Address parameters indicate the IP addresses of the two physical devices participating in the TCP connection.

The Result parameter indicates that the service user TCP connection manager has accepted the requested TCP connection. Its value must always be SUCCESS.

#### *Use*

The service user TCP connection manager process invokes this service to indicate to the COSEM TCP-based transport layer that it has accepted the previously requested TCP connection.

### 6.2.2.4 TCP-CONNECT.confirm

#### *Function*

This service primitive is invoked by the COSEM TCP-based transport layer to indicate to the service user TCP connection manager process the result of a previously received TCP-CONNECT.request service invocation.

#### *Service parameters*

The semantics of the primitive is as follows:

```
TCP-CONNECT.confirm
(
    Local_TCP_Port,
    Remote_TCP_Port,
    Local_IP_Address,
    Remote_IP_Address,
    Result,
    Reason_of_Failure
)
```

The Local\_TCP\_Port and Remote\_TCP\_Port parameters indicate the two TCP ports between which the TCP connection is being established.



The Local\_IP\_Address and Remote\_IP\_Address parameters indicate the IP addresses of the two physical devices participating in this TCP connection.

The Result parameter indicates, whether the requested TCP connection is established or not. Note, that this service primitive is normally the result of a remote confirmation – and as a TCP connection request cannot be rejected, the Result parameter shall always indicate SUCCESS.

However, the Result parameter may also indicate FAILURE, when it is locally confirmed. In this case the Reason\_of\_Failure parameter indicates the reason for the failure.

#### *Use*

The COSEM TCP-based transport layer shall indicate to the service user TCP connection manager the result of a previously received TCP-CONNECT.request service invocation with the help of this service.

### **6.2.3 The TCP-DISCONNECT services**

#### **6.2.3.1 TCP-DISCONNECT.request**

##### *Function*

This service primitive is invoked by the service user TCP connection manager process to request the disconnection of an existing TCP connection.

##### *Service parameters*

The semantics of the primitive is as follows:

```

TCP-DISCONNECT.request
(
    Local_TCP_Port,
    Remote_TCP_Port,
    Local_IP_Address,
    Remote_IP_Address
)

```

The service parameters are the identifiers of the TCP connection to be released. The Local\_TCP\_Port and Local\_IP\_Address parameters designate the local TCP port and IP Address of the requesting device and application, the Remote\_IP\_Address and Remote\_TCP\_Port parameters refer to the remote device and application.

#### *Use*

This service is used by the TCP connection manager process to request the disconnection of an existing TCP connection.

#### **6.2.3.2 TCP-DISCONNECT.indication**

##### *Function*

This service primitive is invoked by the COSEM TCP-based transport layer to the service user TCP connection manager process to indicate that the peer entity has requested the disconnection of an existing TCP connection.

The same service is used also to indicate if the transport layer detects a non-solicited disconnection of an existing TCP connection (for example, when the physical connection breaks down).



### *Service parameters*

The semantics of the primitive is as follows:

#### **TCP-DISCONNECT.indication**

```
(
    Local_TCP_Port,
    Remote_TCP_Port,
    Local_IP_Address,
    Remote_IP_Address,
    Reason
)
```

The Local\_TCP\_Port, Remote\_TCP\_Port, Local\_IP\_Address, Remote\_IP\_Address parameters identify the TCP connection, which is either requested to be released by the peer device, or has been aborted.

The Reason parameter indicates whether the service is invoked because of the peer device has requested a TCP disconnection (Reason == REMOTE\_REQ), or it is locally originated by detecting a kind of event, which must imply the disconnection of the TCP connection (Reason == ABORT<sup>4</sup>).

### *Use*

The COSEM TCP-based transport layer shall signal the reception of a TCP disconnection request or a detected TCP connection abort to the service user TCP connection manager with the help of this service primitive.

### **6.2.3.3 TCP-DISCONNECT.response**

#### *Function*

This service primitive is invoked by the TCP connection manager process to indicate to the COSEM TCP-based transport layer whether the previously requested TCP disconnection is accepted. Note, that the TCP connection manager process cannot reject the requested disconnection.

This service primitive is invoked only if the corresponding TCP-DISCONNECT.indication service indicated a remotely initiated disconnection request (Reason == REMOTE\_REQ).

### *Service parameters*

The semantics of the primitive is as follows:

#### **TCP-DISCONNECT.response**

```
(
    Local_TCP_Port,
    Remote_TCP_Port,
    Local_IP_Address,
    Remote_IP_Address,
    Result
)
```

The Local\_TCP\_Port and Remote\_TCP\_Port parameters identify the two TCP ports between which the TCP connection has to be disconnected.

---

<sup>4</sup> The COSEM transport layer may give more detailed information about the reason for the ABORT via layer management services. However, layer management services are out of the scope of this standard.



The Local\_IP\_Address and Remote\_IP\_Address parameters indicate the IP addresses of the two physical devices participating in the TCP connection to be disconnected.

The Result parameter indicates that the service user TCP connection manager process has accepted to disconnect the TCP connection referenced. The value of this parameter must always be SUCCESS.

#### *Use*

The TCP connection manager process invokes this service primitive to indicate to the service user COSEM TCP-based transport layer that it has accepted to disconnect the TCP connection referenced.

If the TCP-DISCONNECT.indication primitive has been invoked because the TCP connection has been aborted, the TCP-DISCONNECT.response primitive shall not be invoked.

### **6.2.3.4 TCP-DISCONNECT.confirm**

#### *Function*

This service primitive is invoked by the COSEM TCP-based transport layer to confirm to the service user TCP connection manager the result of a previous TCP-DISCONNECT.request service invocation.

#### *Service parameters*

The semantics of the primitive is as follows:

```

TCP-DISCONNECT.confirm
(
    Local_TCP_Port,
    Remote_TCP_Port,
    Local_IP_Address,
    Remote_IP_Address,
    Result,
    Reason_of_Failure
)

```

The Local\_TCP\_Port and Remote\_TCP\_Port parameters identify the two TCP ports between which the TCP connection has to be disconnected.

The Local\_IP\_Address and Remote\_IP\_Address parameters indicate the IP addresses of the two physical devices participating in the TCP connection to be disconnected.

The Result parameter indicates, whether the disconnection of the TCP connection referenced has succeeded or not. Normally, this service primitive is invoked as the result of a remote confirmation, and as a TCP disconnection request cannot be rejected, the value of the Result parameter is always SUCCESS.

However, the Result parameter may also indicate FAILURE, when it is locally confirmed. In this case, the Reason\_of\_Failure parameter indicates the reason of the failure.

#### *Use*

The COSEM TCP-based transport layer uses this service primitive to confirm to the service user TCP connection manager the result of a previously received TCP-DISCONNECT.request service invocation.



## 6.2.4 The TCP-ABORT service

### 6.2.4.1 General

The TCP-ABORT.indication service is provided to indicate to the COSEM application layer a non-solicited disruption of the TCP connection, supporting COSEM communications.

### 6.2.4.2 TCP-ABORT.indication

#### *Function*

This service primitive is invoked by the COSEM TCP-based transport layer to indicate to the service user COSEM application layer the disruption of the supporting TCP connection.

#### *Service parameters*

The semantics of the primitive is as follows:

```

TCP-ABORT.indication
(
    Local_TCP_Port,
    Remote_TCP_Port,
    Local_IP_Address,
    Remote_IP_Address,
    Reason
)
    
```

The Local\_TCP\_Port and Remote\_TCP\_Port parameters identify the two TCP ports the connection between which has aborted.

The Local\_IP\_Address and Remote\_IP\_Address parameters indicate the IP addresses of the two physical devices having been participated in the TCP connection aborted.

The Reason parameter indicates the reason of the TCP abort. This parameter is optional.

#### *Use*

The COSEM TCP-based transport layer shall indicate the disruption of the supporting TCP connection to the COSEM application layer. When this indication is received, the COSEM application layer shall release all application associations established using this TCP connection, and shall indicate this to the COSEM application process using the COSEM-ABORT.indication service primitive. See also 6.5.2.4 and 6.6.2.3 of IEC 62056-53.

## 6.2.5 The TCP-DATA services

### 6.2.5.1 TCP-DATA.request

#### *Function*

This service primitive is invoked by the service user COSEM application layer to request the transmission of an APDU to the peer COSEM application layer.

#### *Service parameters*

The semantics of the primitive is as follows:

```

TCP-DATA.request
(
    Local_wPort,
    Remote_wPort,
    Local_TCP_Port,
    Remote_TCP_Port,
    Local_IP_Address,
    Remote_IP_Address,
    Data_Length,
    Data
)

```

The Local\_wPort, Local\_TCP\_Port and Local\_IP\_Address parameters indicate wrapper Port number, TCP Port number and IP Address parameters of the device/application process requesting to send the Data.

The Remote\_wPort, Remote\_TCP\_Port and Remote\_IP\_Address parameters indicate the wrapper Port number, TCP Port number and IP Address parameters belonging to the device/application process to which the Data is to be transmitted.

The Local\_TCP\_Port and Remote\_TCP\_Port parameters identify the local and remote TCP ports respectively. As no well-known port number is reserved for COSEM communications, the value of these parameters must be in the non-privileged range (above 1024).

The Data\_Length parameter indicates the length of the Data parameter in bytes.

The Data parameter contains the COSEM APDU to be transferred to the peer application layer.

#### *Use*

The TCP-DATA.request primitive is invoked by either the client or the server COSEM application layer to request sending an APDU to a single peer application.

The reception of this primitive shall cause the wrapper sub-layer to pre-fix the wrapper-specific fields (Local\_wPort, Remote\_wPort and the Data\_Length) to the APDU received, and then to call the SEND() function of the TCP sub-layer with the properly formed WPDU, see 5.3.2, as DATA. The TCP sub-layer shall transmit the WPDU to the peer TCP sub-layer as described in STD0007.

### **6.2.5.2 TCP-DATA.indication**

#### *Function*

This service primitive is invoked by the COSEM transport layer to indicate to the service user COSEM application layer that an APDU has been received from a remote device.

#### *Service parameters*

The semantics of the primitive is as follows:

```

TCP-DATA.indication
(
    Local_wPort,
    Remote_wPort,
    Local_TCP_Port,
    Remote_TCP_Port,
    Local_IP_Address,
    Remote_IP_Address,
    Data_Length,
    Data
)

```



```
) Data_Length,  
Data
```

The Local\_wPort, Local\_TCP\_Port and Local\_IP\_Address parameters indicate wrapper Port number, TCP Port number and IP Address parameters belonging to the device/application process receiving the Data.

The Remote\_wPort, Remote\_TCP\_Port and Remote\_IP\_Address parameters indicate the wrapper Port number, TCP Port number and IP Address parameters belonging to the device/application process which has sent the Data.

The Local\_TCP\_Port and Remote\_TCP\_Port parameters identify the local and remote TCP ports respectively. As no well-known port number is reserved for COSEM communications, the value of these parameters must be in the non-privileged range (above 1024).

The Data\_Length parameter indicates the length of the Data parameter in bytes.

The Data parameter contains the COSEM APDU received from the peer application layer.

#### *Use*

The TCP-DATA.indication service primitive is used to indicate to the service user COSEM application layer, that an APDU has been received from the peer layer entity.

The primitive is generated following the reception of a complete APDU (in one or more TCP packets) by the COSEM TCP-based transport layer, if both the Local\_TCP\_Port and Local\_wPort parameters in the TCP packet(s) carrying the APDU contain valid wPort numbers, meaning that there is a COSEM application process in the receiver device bound to the given port numbers. Otherwise, the message received shall simply be discarded.

### **6.2.5.3 TCP-DATA.confirm**

#### *Function*

This optional service primitive is invoked by the COSEM transport layer to confirm to the service user COSEM application layer the result of the execution of the previous TCP-DATA.request service. This service may represent either a local or a remote confirmation, depending on the implementation.

#### *Service parameters*

The semantics of the primitive is as follows:

```

TCP-DATA.confirm
(
    Local_wPort,
    Remote_wPort,
    Local_TCP_Port,
    Remote_TCP_Port,
    Local_IP_Address,
    Remote_IP_Address,
    Confirmation_Type,
    Result
)
    
```

The Local\_wPort, Remote\_wPort, Local\_TCP\_Port, Remote\_TCP\_Port, Local\_IP\_Address and Remote\_IP\_Address parameters carry the same values as the corresponding TCP-DATA.request service being confirmed.

The Confirmation\_Type parameter indicates whether the confirmation service is a LOCAL or a REMOTE confirmation.



The value of the Result parameter indicates the result of the previous TCP-DATA.request service. However, its interpretation is implementation dependent: it can be either local or remote confirmation. In the former case, it indicates either the result of sending out the requested data, or at least that the protocol stack took it in charge (see 6.3.5.4). When this service is implemented as a remote confirmation, it indicates whether the requested data has been delivered to the destination. Its value is either OK or NOK.

#### *Use*

If implemented, this service is used to confirm the result of a previously issued TCP-DATA.request service. The result is either OK or NOK, but the meaning of this depends on the implementation of this service. When it is implemented as a local confirmation, the result indicates whether the COSEM transport layer was able to buffer for sending or to send out the previously requested APDU or not. When this service is implemented as a remote confirmation, the result indicates whether the requested APDU has been successfully delivered to the destination or not.

### **6.3 Protocol specification for the COSEM TCP-based transport layer**

#### **6.3.1 General**

As it is shown on the right side of Figure 2, the COSEM connection-oriented, TCP-based transport layer includes the Internet standard TCP layer as specified in STD0007 and the COSEM specific wrapper sub-layer.

In this communication profile, the wrapper sub-layer is a little bit more complex, than it is in the COSEM UDP-based transport layer.

On the one hand, similarly to the UDP-based transport layer, its main role is also to ensure source and destination application process identification using the wPort numbers, and to convert OSI-style TCP-DATA.xxx service invocations to and from the SEND() and RECEIVE() interface functions provided by the standard TCP.

On the other hand, the wrapper sub-layer in the TCP-based transport layer has also the task to help the service user COSEM application layers to exchange complete APDUs, by making the “streaming” nature of the TCP protocol transparent to the COSEM application layer.

“Streaming” means that TCP does not preserve data boundaries. Without entering into the details here (see Clause A.4) this means, that the SEND() and RECEIVE() function calls of the TCP sub-layer return with success even if the number of the actually sent/received bytes is less than the number of bytes requested to send/receive. It is the responsibility of the wrapper sub-layer to know how much data had to be sent/received, to keep track of how much has been actually sent/received, and repeat the operation until the complete APDU is transmitted.

Consequently, the wrapper sub-layer in the TCP-based COSEM transport layer is not a stateless entity: it is doing the above-described track-keeping – re-trying procedure in order to make the “streaming” nature of the TCP transparent to the service user COSEM application layer.

#### **6.3.2 The wrapper protocol data unit (WPDU)**

The wrapper protocol data unit is as specified in 5.3.2.



### 6.3.3 The COSEM TCP-based transport layer protocol data unit

In this profile, WPDUs shall be transmitted in one or more TCP packets. The TCP packet is specified in STD0007, and shall encapsulate a part of the WPDU in its Data Field, as shown in Figure 7.

0	4	8	10	16	24	31
Source TCP Port				Destination TCP Port		
Sequence Number						
Acknowledgement Number						
Data Offset	Reserved		Flags	Window		
Checksum				Urgent Pointer		
TCP Options					Padding	
Data (part of the WPDU)						
...Data (part of the WPDU)...						

**Figure 7 – The TCP packet format**

For the specification of the various fields of the TCP packet, refer to STD0007.

The reason for having only a part of the WPDU in a TCP packet is the “streaming” nature of the TCP already mentioned.

From the external point of view the COSEM TCP-based transport layer PDU is an ordinary TCP packet: any COSEM specific element, including the wrapper-specific header in the first TCP packet, is inside the packet’s Data field. Consequently, standard TCP implementations can be (re-) used to easily implement this transport layer.

The source and destination TCP ports may refer to either local or remote TCP ports, depending on the direction of the data transfer (e.g. from the point of view of the Sender device the Source TCP port in a TCP packet corresponds to the Local\_TCP\_port, but from the point of view of the Receiver device, the Source TCP port of a Datagram corresponds to the Remote\_TCP\_Port service parameter).

As no well-known port number is reserved for COSEM communications, the value of these parameters must be in the non-privileged range (above 1024).

### 6.3.4 Reserved wrapper port numbers

The following wPort numbers are reserved:



**Table 2 – Reserved wrapper port numbers in the TCP-based COSEM profiles<sup>5</sup>**

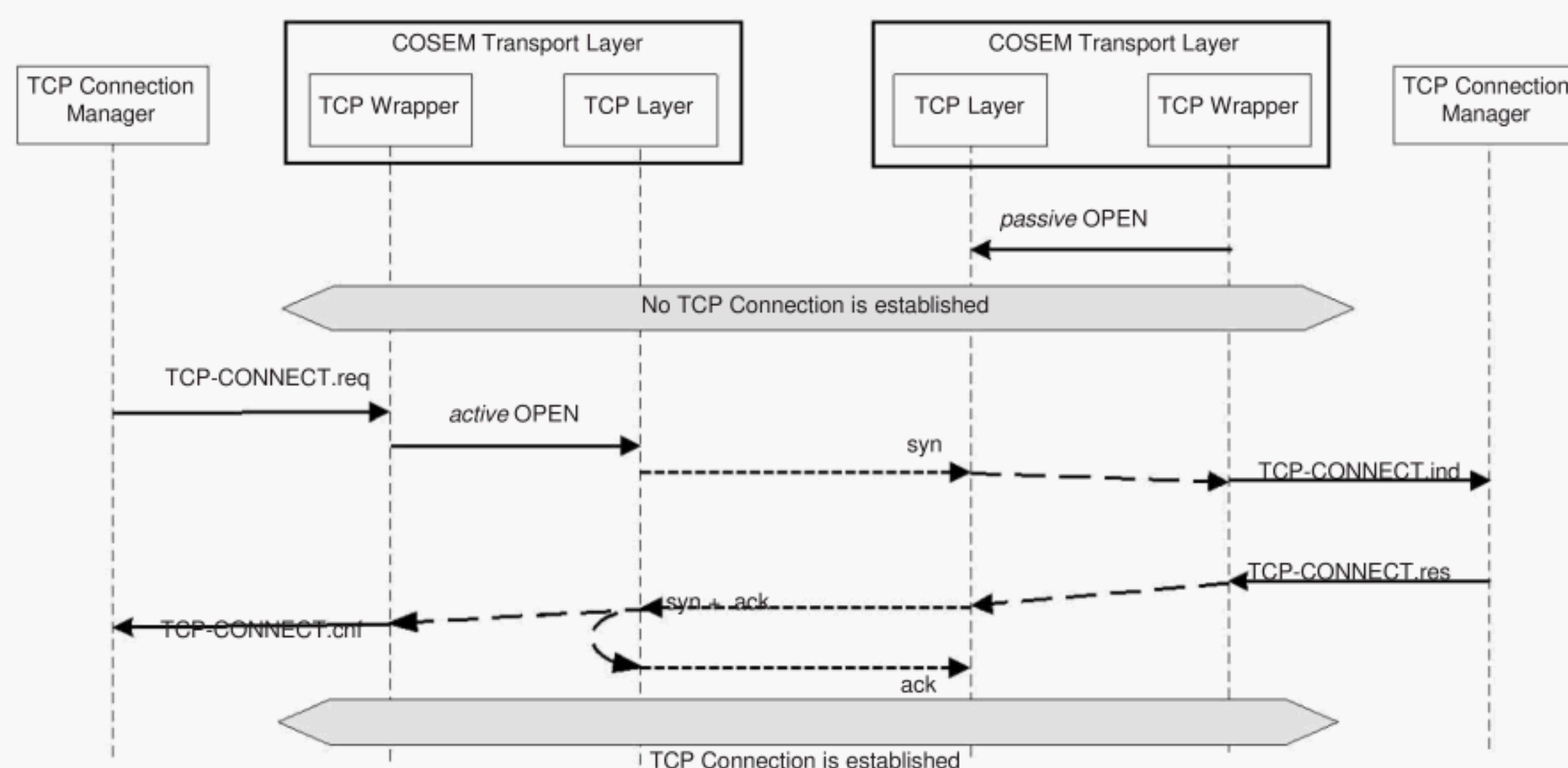
Client side reserved addresses	
	Wrapper Port Number
No-station	0x0000
Client Management Process	0x0001
Public Client	0x0010
Server side reserved addresses	
	Wrapper Port Number
No-station	0x0000
Management Logical Device	0x0001
Reserved	0x0002...0x000F
All-station (Broadcast)	0x007F

### 6.3.5 Definition of the procedures

#### 6.3.5.1 Setting up the TCP connection

Establishment of a TCP connection is initiated by the TCP-CONNECT.request service invocation. Although this service – as all COSEM transport layer services – is provided to the service user entity by the wrapper sub-layer, the TCP connection is established between the two (local and remote) TCP sub-layers. The role of the wrapper in this procedure is just to convert the four TCP-CONNECT service primitives (.request, .indication, .response and .confirm) to and from TCP function calls.

From the service user point of view, only the TCP-CONNECT services are visible: according to this, the TCP connection establishment takes place as shown in Figure 8.

**Figure 8 – Figure TCP connection establishment**

<sup>5</sup> These wPort numbers are the same as the reserved wPort numbers for the UDP-based profile.



The TCP connection is established using a three-way handshake mechanism, as described in STD0007. This requires three message exchanges as shown above and guarantees that both sides know that the other side is ready to transmit and also that the two sides are synchronized: the initial sequence numbers are agreed upon.

Both the client and server side TCP connection manager processes are allowed to initiate the TCP connection. To establish the connection, one of them shall play the role of the initiator, and the other the responder.

In order to be able to respond, the responder has to perform a ‘passive’ opening before receiving the first, SYN packet. To do this, it has to contact the local operating system (OS) to indicate, that it is ready to accept incoming connection requests. As the result of this contact, the OS shall assign<sup>6</sup> a TCP port number to that end-point of the connection and shall reserve the resources required for a future connection – but no message shall be sent out.

In the case of the COSEM TCP-based transport layer, the wrapper sub-layer shall initiate this passive opening autonomously during system initialisation. In other words, as this passive opening is the responsibility of the wrapper sub-layer, no service is provided to an external entity to initiate the passive opening.

As both the client and the server side TCP connection manager processes are allowed to play the role of the “Responder” application, the transport layers on both sides shall perform a passive opening during the system initialisation.

More details about TCP connection establishment is provided in Clause A.1.

### 6.3.5.2 Disconnecting the TCP connection

The TCP is disconnected using the TCP-DISCONNECT services, as shown in Figure 9.

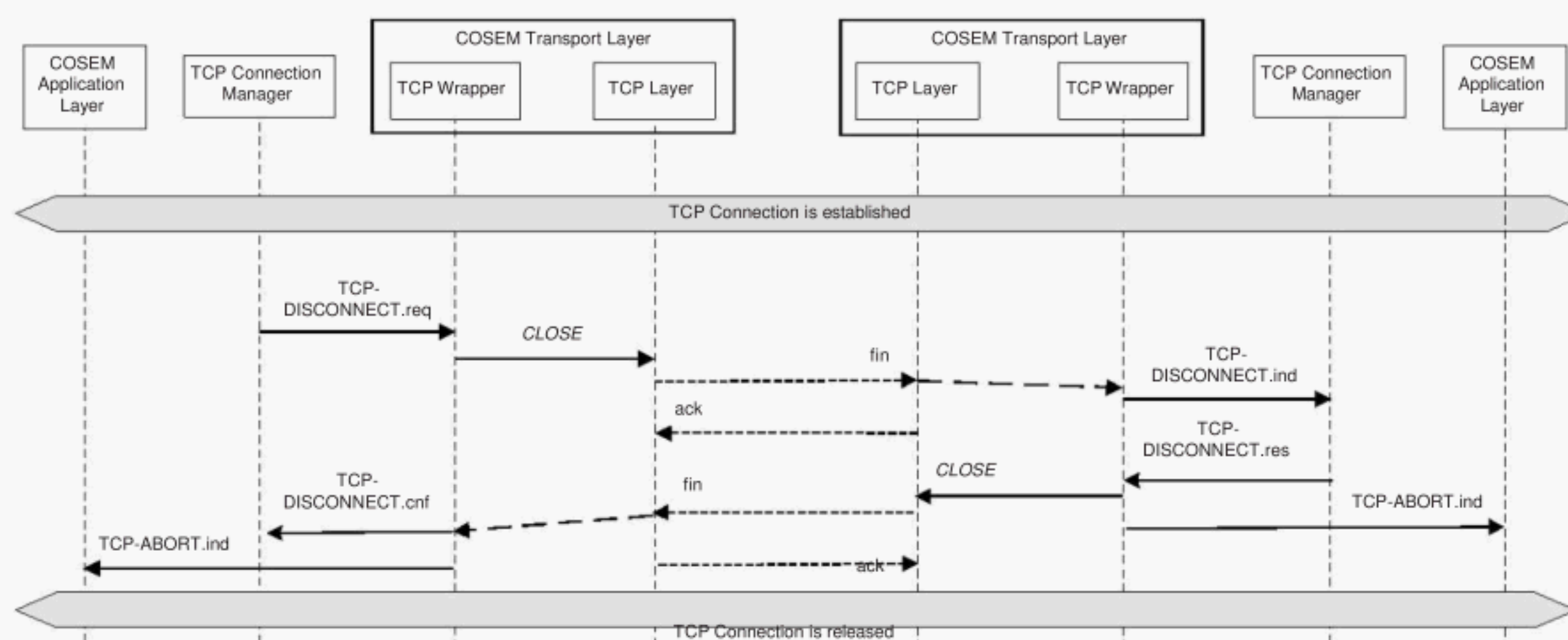


Figure 9 – Disconnecting a TCP connection

The procedure can be initiated either by the client or the server side TCP connection manager process, invoking the TCP-DISCONNECT.request service. This request shall be transformed by the “wrapper” to a CLOSE () function call to the TCP interface.

6 In the case of the COSEM transport layer, the implementation must force the OS to assign the requested TCP/UDP port number to the local end point of the connection.



The TCP shall send a fin segment, which is acknowledged by the peer TCP.<sup>7</sup> At the same time, through the wrapper, the TCP-DISCONNECT.indication service is invoked, informing the user TCP connection manager that the connection is closing. The connection manager – in order to gracefully release the connection – responds with a TCP-DISCONNECT.response service primitive. The TCP wrapper calls the CLOSE function and the TCP sends out its fin segment. At the same time, it shall indicate the closing of the TCP connection to the COSEM application layer with the help of the TCP-ABORT.indication service.

On the requesting side, the TCP sends an acknowledgement and upon the reception of this by the peer the TCP connection is deleted. At the same time, the wrapper invokes the TCP-DISCONNECTION.confirm service primitive informing the connection manager process that the disconnection request has been accepted. Similarly to the peer, the TCP disconnection shall also be indicated to the COSEM application layer with the help of the COSEM-ABORT.indication service.

More details about TCP disconnection are provided in Clause A.2.

#### 6.3.5.3 TCP connection abort

The COSEM TCP-based transport layer shall indicate the disruption or disconnection of the supporting TCP connection to the COSEM application layer with the help of the TCP-ABORT.indication service primitive. Note, that this is the only TCP connection management service provided to the COSEM application layer.

The service shall be invoked either when the TCP connection is disconnected by the TCP connection manager process – the case of graceful disconnection – or when the TCP disconnection occurs in a non-solicited manner, for example the TCP sub-layer is detecting a non-resolvable error or the physical connection is shut down.

The purpose of this service is to inform the COSEM application layer about the disruption of the TCP connection, so that it could release all existing application associations.

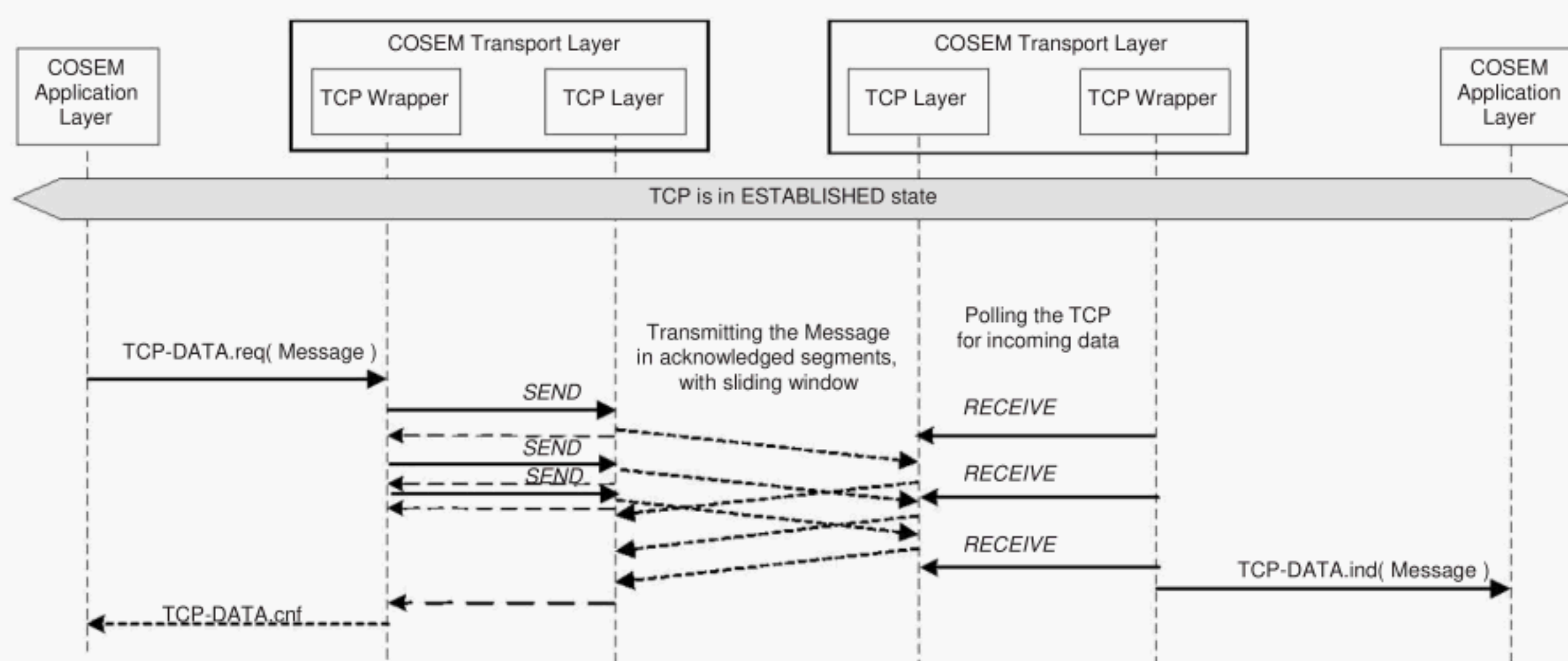
#### 6.3.5.4 Data communication – using the TCP-DATA services

Once the TCP connection is established, reliable data communication can be performed via this connection. Although providing this reliable data communications is a quite complex operation involving reliability mechanisms such as *positive acknowledgement with retransmission* (PAR) or flow control with sliding windows – provided by TCP and specified in STD0007 – the COSEM TCP-based layer provides only data communications service, the TCP-DATA service, as shown in Figure 10.

---

<sup>7</sup> TCP uses an improved 3-way handshake to release a connection, ensuring that possible duplication and delay – introduced by the non-reliable IP layer – do not pose problems. More about this procedure can be found in STD0007.





**Figure 10 – Data communication using the COSEM TCP-based transport layer**

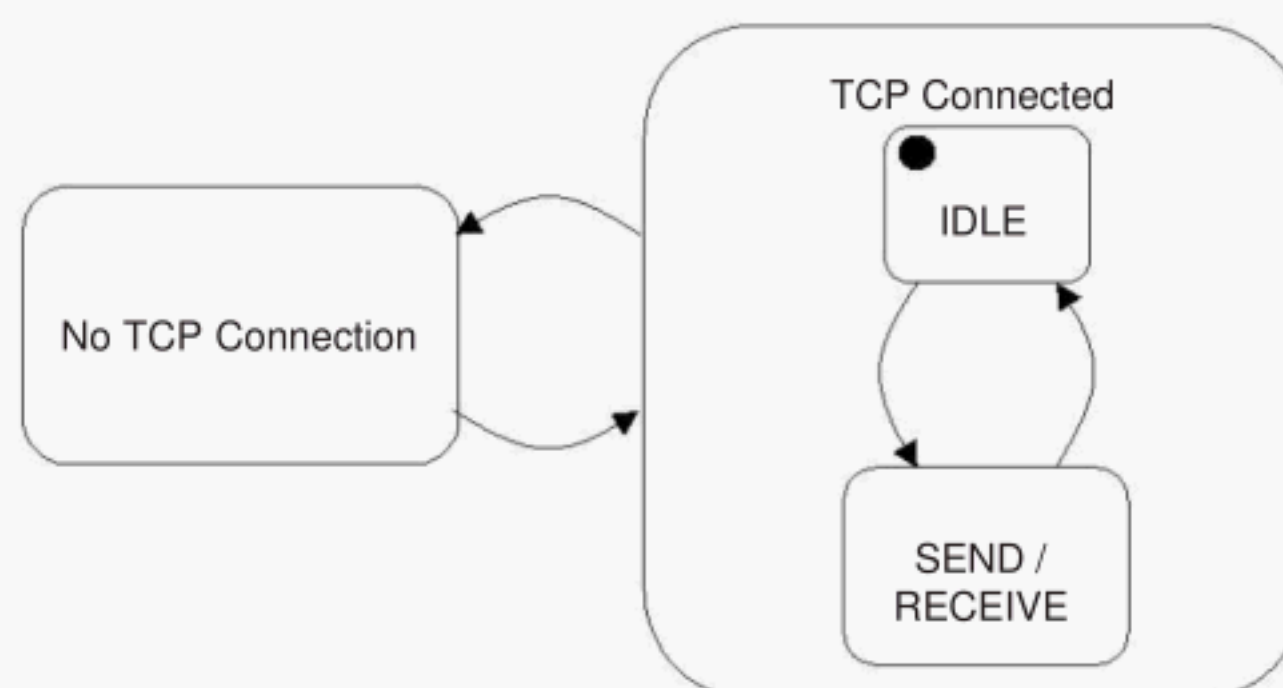
The use of the TCP-DATA services is the same both on the client and at the server side.

The optional TCP-DATA.confirm indicates the result of the previously invoked TCP-DATA.request service, which is either OK or NOK. However, the meaning of this result is implementation dependent: when it is implemented as a local confirmation, it means only whether the transport layer was able to process the request or not. (An OK may even not mean that the message has been sent to the network: it may mean only that the transport layer has buffered the message and will send it when it is possible.)

As shown in Figure 10, the message (a WPDU) may be transported (sent/received) in more than one TCP packet. It is because TCP sends data as a stream of octets, without preserving data boundaries. It is the responsibility of the wrapper sub-layer to hide this property of the TCP sub-layer from the service user COSEM application layer. The sender side wrapper shall keep track about the amount of data sent with one SEND() function call and repeat the operation until the whole WPDU is sent. The receiver side wrapper shall continue to receive incoming TCP packets until a complete WPDU is received. For more details, see Clause A.4.

#### 6.3.5.5 High-level state transition diagram of the wrapper sub-layer

The high-level state-diagram of the wrapper sub-layer is shown in Figure 11.



**Figure 11 – High-level state transition diagram for the wrapper sub-layer**



In both macro-states, (No TCP Connection and TCP Connected) the wrapper is polling the TCP layer for its connection status, and transits into the other macro-state if the status has changed.

The wrapper enters always into the IDLE sub-state of the TCP Connected state, and transits to the composite SEND/RECEIVE state either on a TCP-DATA.request or on the reception of a TCP packet.

In this state, the wrapper shall send and/or<sup>8</sup> receive WPDUs, as described in Annex A.

---

<sup>8</sup> TCP on the top of a full-duplex lower layer protocol stack may simultaneously send and receive.

## Annex A (informative)

### Converting OSI-style transport layer services to and from RFC-style TCP function calls

#### A.1 Transport layer and TCP connection establishment

As specified in STD0007, a TCP connection is established by calling the OPEN function. This function can be called in *active* or *passive* manner.

According to the TCP connection state diagram (Figure A.1) a *passive* OPEN takes the caller device to the LISTEN state, waiting for a connection request from any remote TCP and port.

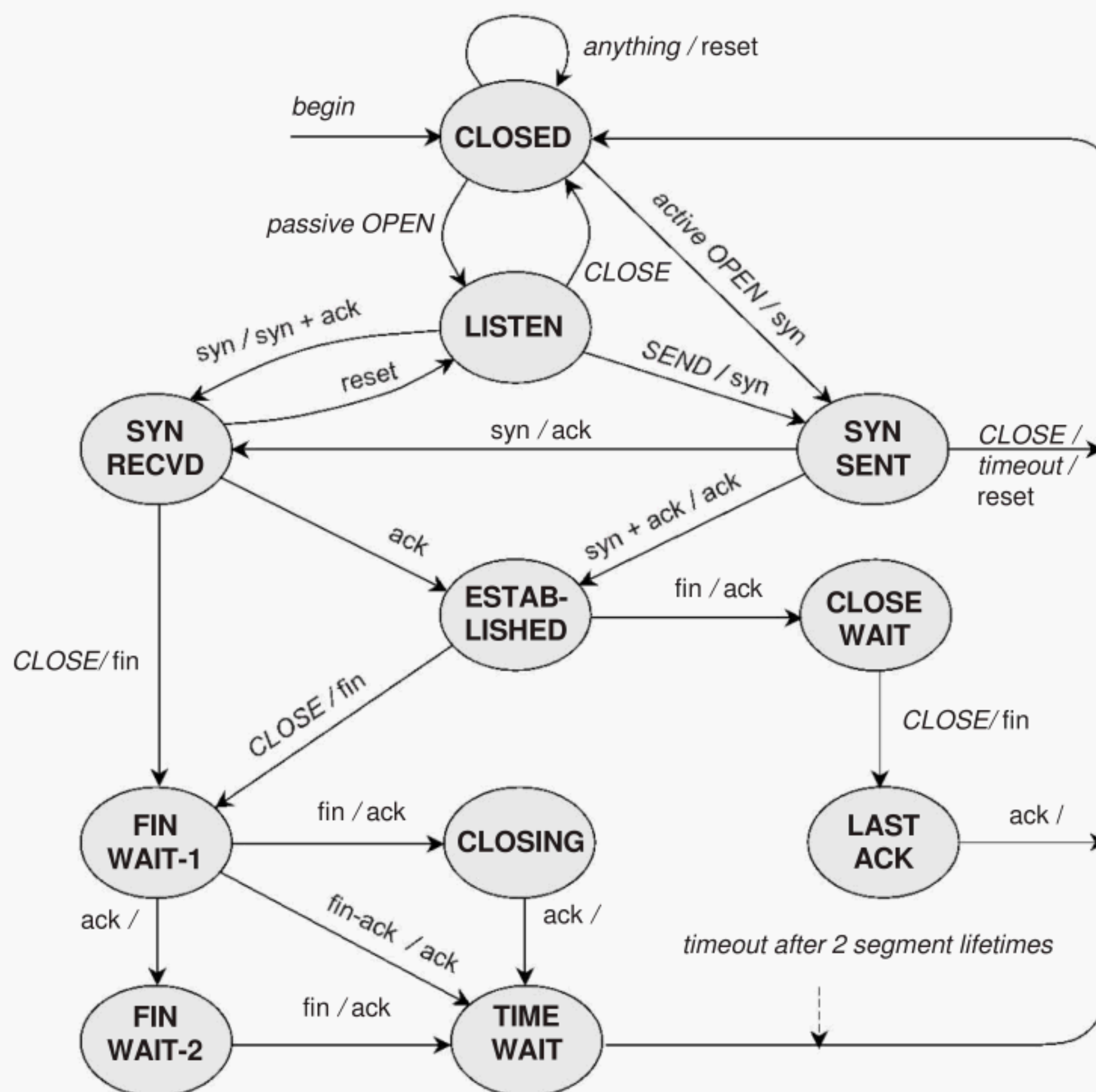


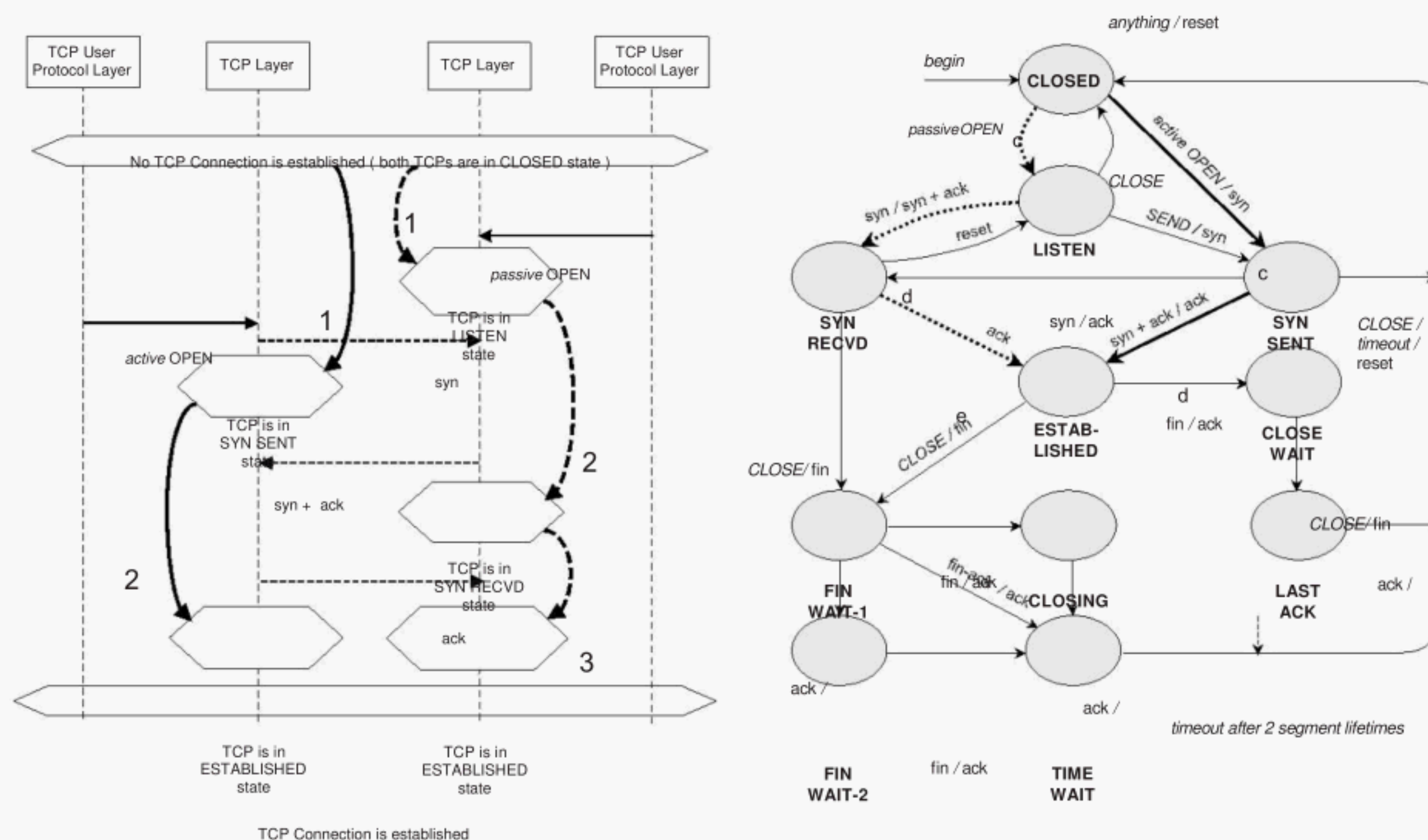
Figure A.1 – TCP connection state diagram

An *active* OPEN call shall make the TCP to establish the connection to a remote TCP.



The establishment of a TCP Connection is performed by using the so-called “three-way handshake” procedure. This is initiated by one TCP calling an *active OPEN* and responded by another TCP, the one, which has already been called a *passive OPEN* and consequently is in the LISTEN state.

The message sequence – and the state transitions corresponding to that message exchange – for this “three-way handshake” procedure are shown in Figure A.2.



NOTE In the case of the COSEM transport layer, the TCP user protocol layer is the wrapper sub-layer.

**Figure A.2 – MSC and state transitions for establishing a transport layer and TCP connection**

This process, consisting of three messages, establishes the TCP connection and “synchronizes” the initial sequence numbers<sup>9</sup> at both sides. This mechanism has been carefully designed to guarantee, that both sides are ready to transmit data and know that the other side is ready to transmit as well. Note, that the procedure also works if two TCPs simultaneously initiate the procedure.

## A.2 Closing a transport layer and a TCP connection

Closing a TCP connection is done by calling the CLOSE function, generally when there is no more data to be sent.

Upon the invocation of the TCP-DISCONNECT.request service primitive by the TCP connection manager process, the wrapper sub-layer invokes the CLOSE function of the TCP sub-layer.

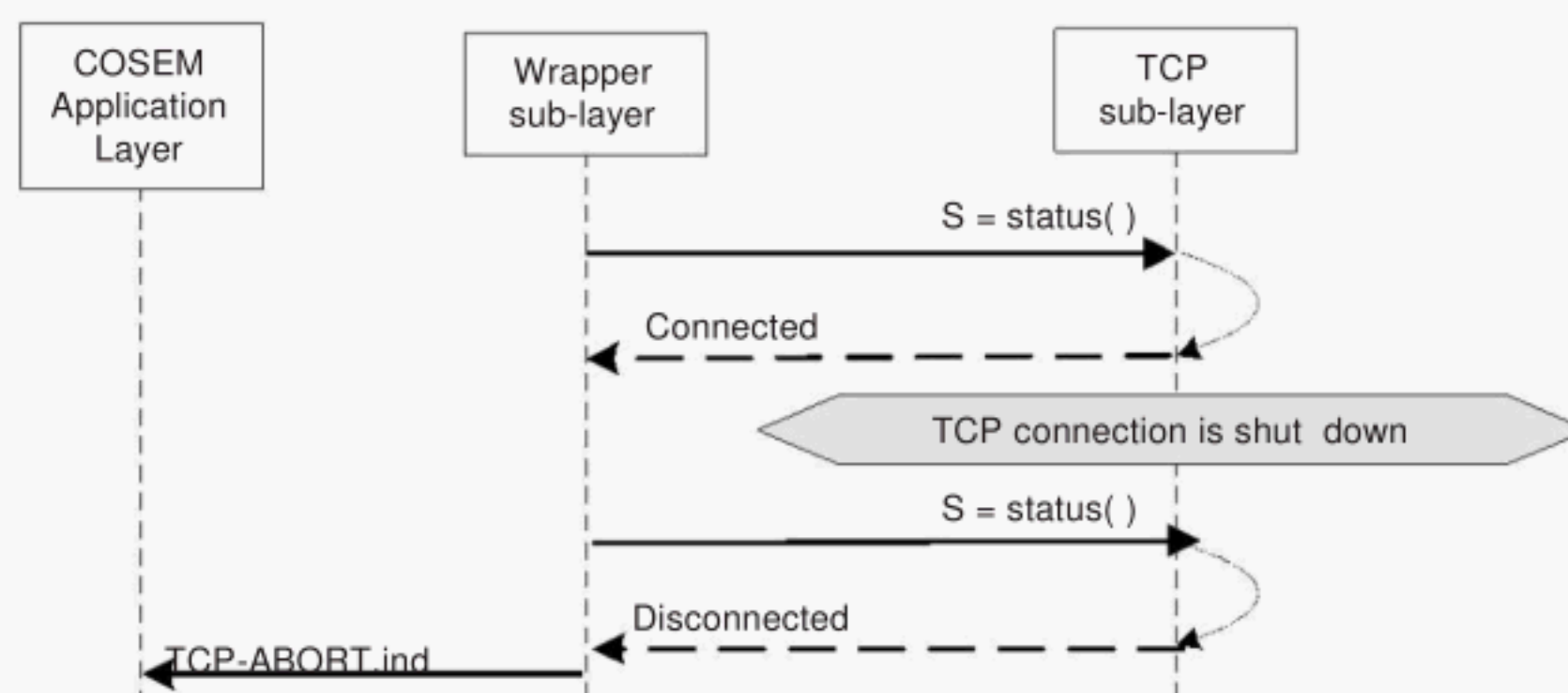
<sup>9</sup> Sequence numbers are part of the TCP packet, and are fundamental to reliable data transfer. For more details about sequence numbers ( or other TCP related issues ), please refer to STD0007.





### A.3 TCP connection abort

STD0007 does not specify a standard function to indicate an unexpected abort at TCP level. However, it can be detected by the TCP user entity by polling the status of the TCP with the



STATUS() function, as shown in Figure A.4.

**Figure A.4 – Polling the TCP sub-layer for TCP abort indication**

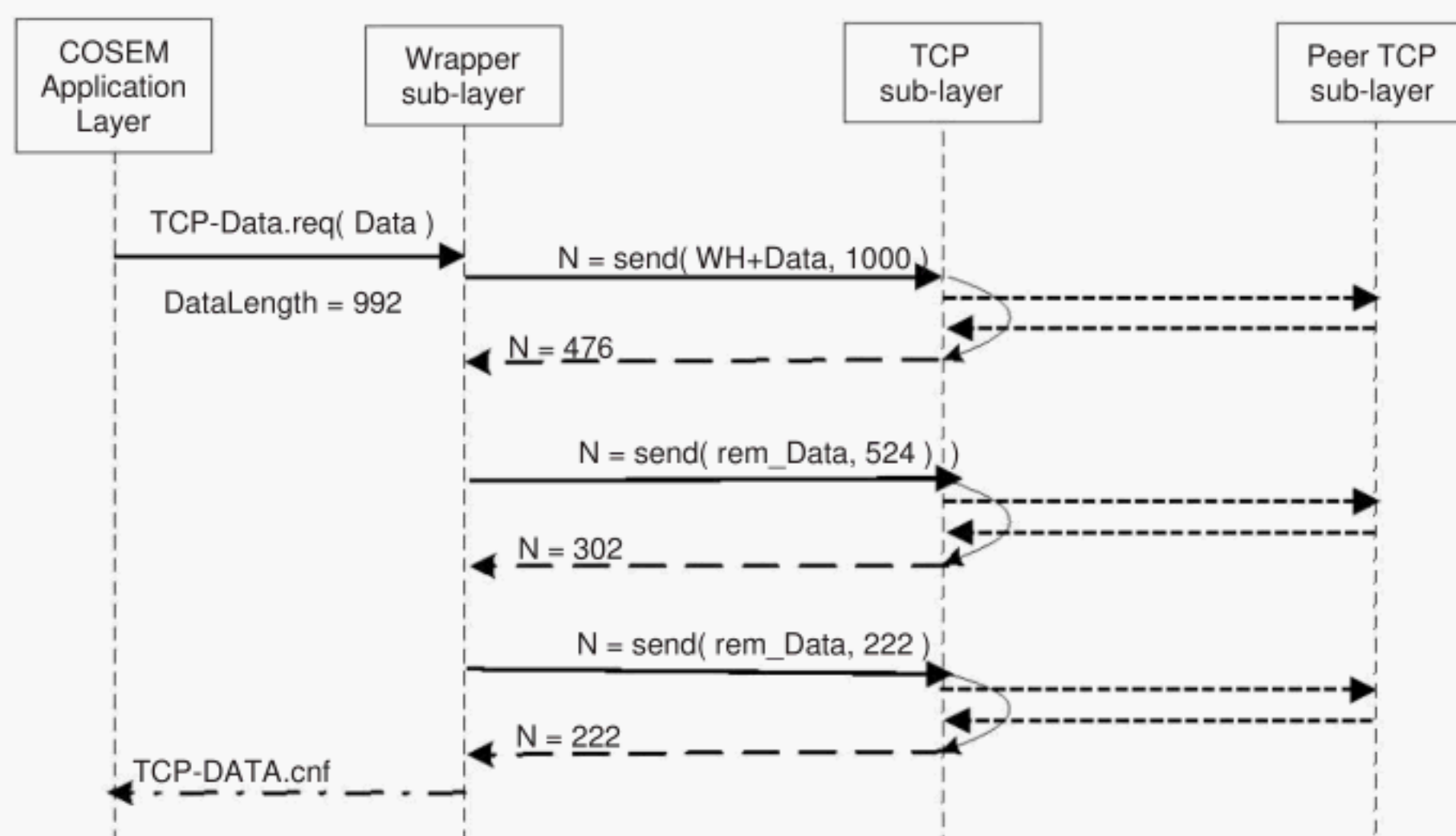
### A.4 Data communication – the TCP-DATA service

To send an APDU to the peer, the COSEM application layer shall simply invoke the TCP-DATA.request service of the COSEM TCP-based transport layer. Also, when a complete APDU is received, this shall be indicated to the COSEM application layer with the help of the TCP-DATA.indication service. Thus, for the application layer, the transport layer behaves as if it would transport the whole APDU in one piece.

However, nothing ensures that an APDU is actually transmitted in one TCP packet. The reason for that is that TCP is a streaming protocol – in other words, TCP does not preserve data boundaries. As it is already mentioned in 6.3.5.4, in the COSEM TCP-based transport layer it is the responsibility of the wrapper sub-layer to “hide” the streaming nature of the TCP sub-layer. The following example illustrates how the wrapper sub-layer accomplishes this task.

Let's suppose, that an application layer<sup>10</sup> entity wants to send an APDU containing 992 bytes via the COSEM TCP-based transport layer. It shall invoke the TCP-DATA.request service, with this APDU as the DATA, service parameter as shown in Figure A.5.

<sup>10</sup> Both the client- and server side application layers can be either sender or receiver.



**Figure A.5 – Sending an APDU in three TCP packets**

Upon the reception of this service invocation, the wrapper sub-layer shall construct the WPDU: it shall pre-fix to the APDU the wrapper header (WH), including the local and remote wPort numbers and the APDU length. It shall then call the SEND() function of the TCP sub-layer, requesting to send the WPDU, which is now 1000 bytes long: 8 bytes of wrapper header plus 992 bytes of APDU.

The SEND() function returns with the number of bytes sent or an error (a negative value). Let's suppose, that no error occurs, and the SEND() function successfully returns – with the value 476.

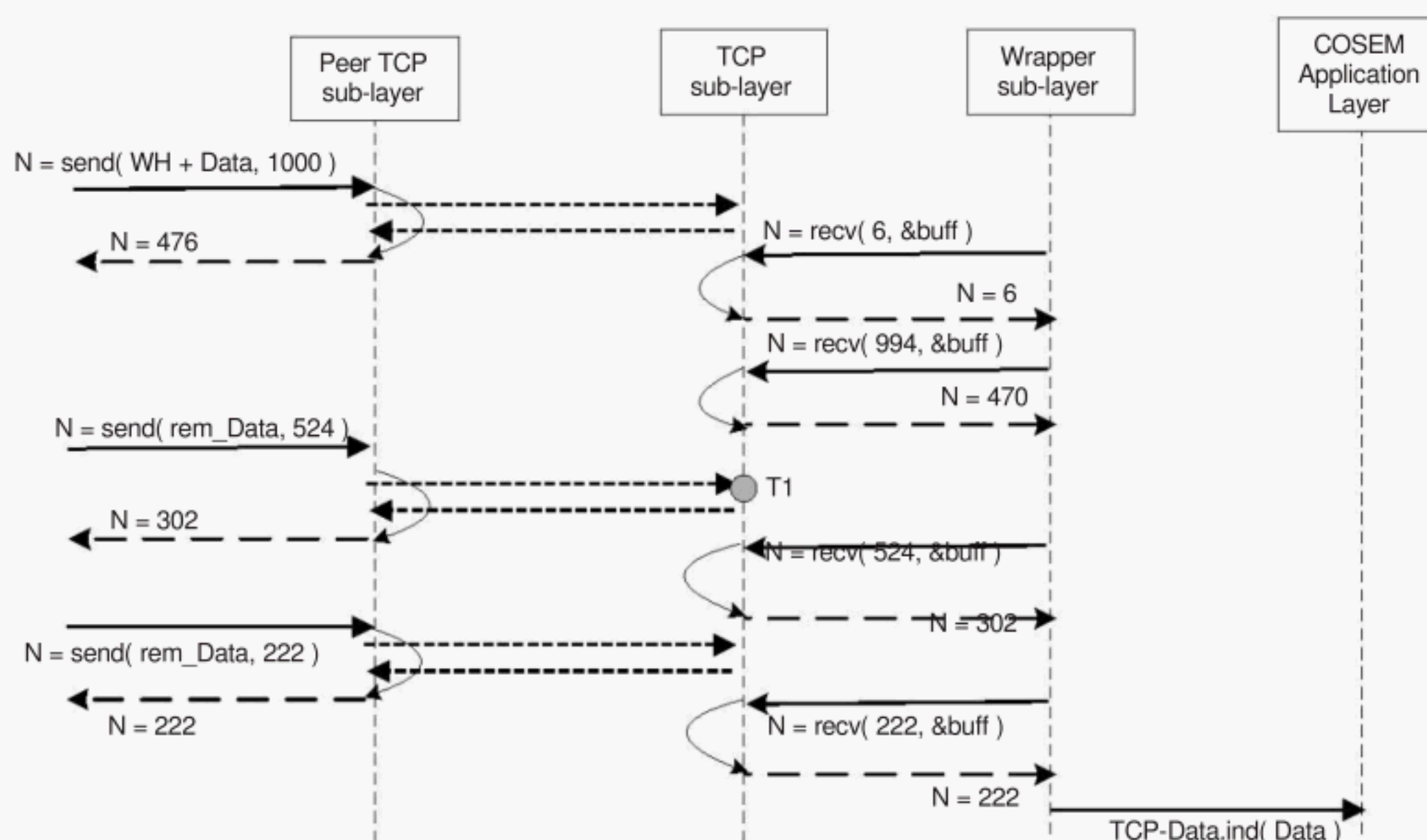
The number 476 means the number of bytes sent – and also illustrates the meaning of the “streaming” nature of the TCP: in fact, the SEND() function returns with success even if the number of bytes sent is less than the number of bytes requested to be sent.

From the value returned, the wrapper knows, that not the whole WPDU has been sent, and it shall call the SEND() function again, with the remaining part of the WPDU – and so on, until the complete WPDU is sent.

As it is already mentioned in 6.3.5.4, depending on the implementation, the successful return of the SEND() function may even not mean that something has really been sent to the network. It may mean only that the protocol implementation took and buffered the data. It may happen that the protocol implementation delays the transmission to comply with protocol conventions or network traffic related algorithms.

On the receiving side, it is also the responsibility of the wrapper sub-layer to assemble the complete APDU before invoking the TCP-DATA.indication service. This is possible by using the length bytes of the WPDU header. The wrapper shall repeat RECEIVE() calls until the number of bytes, indicated in the WPDU header is received. This is shown in Figure A.6.





NOTE 1 As calling the RECEIVE() function is asynchronous with regard to the TCP communications, it is perfectly possible, that the receiver calls the RECEIVE() function at a moment, when the reception of a TCP packet is in progress ( T1. in the Figure above) – or even if when no characters have been received since the last RECEIVE() call. It does not lead to erroneous reception: it increases only the number of necessary RECEIVE() function calls to get the complete message.

NOTE 2 It is also possible that one or more SEND() calls result in sending more than one TCP packets. It does not lead to erroneous reception either: sooner or later the receiver gets the whole message.

**Figure A.6 – Receiving the message in several packets**

All these SEND() and RECEIVE() calls are internal to the COSEM transport layer. The service user COSEM application layer simply uses the TCP-DATA services, and observes a reliable data transfer service preserving the data boundaries of the APDUs.

## Bibliography

IEC 62056-46:2002, *Electricity metering – Data exchange for meter reading, tariff and load control – Part 46: Data link layer using HDLC protocol*  
Amendment 1 <sup>11</sup>

NOTE Harmonized as EN 62056-46:2002 (not modified).

RFC 0768 – *User Datagram Protocol*  
Author: J. Postel Date: Aug-28-1980  
Also: STD0006

RFC 0791 – *Internet Protocol*  
Author: J. Postel Date: Sep-01-1981  
Also: STD0005 Updated by:  
RFC1349  
Obsoletes: RFC0760

RFC 0792 – *Internet Control Message Protocol*  
Author: J. Postel Date: Sep-01-1981 Also:  
STD0005 Updated by: RFC0950  
Obsoletes: RFC0777

RFC 0793 – *Transmission Control Protocol*  
Author: J. Postel Date: Sep-01-1981 Also:  
STD0007 Updated by: RFC3168

RFC 0919 – *Broadcasting Internet Datagrams*  
Author: J.C. Mogul Date: Oct-01-1984 Also:  
STD0005

RFC 0922 – *Broadcasting Internet datagrams in the presence of subnets*  
Author: J.C. Mogul  
Date: Oct-01-1984  
Also: STD0005

RFC 0950 – *Internet Standard Subnetting Procedure*  
Authors: J.C. Mogul, J. Postel  
Date: Aug-01-1985  
Also: STD0005  
Updates: RFC0792

RFC 1112 – *Host extensions for IP multicasting*  
Author: S.E. Deering Date: Aug-01-1989 Also:  
STD0005 Updated by: RFC2236 Obsoletes:  
RFC0988, RFC1054

---

<sup>11</sup> To be published.



## INDEX

- Aborting a TCP connection, 35
- Active OPEN, 33
- Addressing capability (wPort), 8
- Application process identification, 26
- Closing a transport layer and a TCP connection, 34
- Confirmation\_Type, 26
- Connection closing phase, 16
- Connection establishment phase, 16
- Connection-less transport layer, 7
- Connection-oriented transport layer, 7
- COSEM application layer, 7
- COSEM connection-less, UDP-based transport layer, 9
- COSEM connection-oriented, TCP-based transport layer, 15
- COSEM\_on\_IP, 7
- Data communication phase, 16
- Data length, 8, 11, 14
- Destination wPort, 14
- Disconnecting the TCP connection, 29
- Error detection, 16
- fin segment, 30
- Flow control, 16
- IDLE sub-state, 32
- Initiator, 29
- Internet Protocol, 7
- Local confirmation, 31
- Local\_IP\_Address, 11
- Local\_TCP\_Port, 18
- Local\_UDP\_Port, 11
- Local\_wPort, 11
- Multi- or broadcasting, 11
- No TCP Connection, 32
- OSI-style services, 8
- Passive opening, 29
- Positive acknowledgement with retransmission, 30
- Positive Acknowledgement with Retransmission, 16
- Protocol specification, 13, 26
- Protocol state machine, 15
- Remote\_IP\_Address, 11
- Remote\_TCP\_Port, 18
- Remote\_UDP\_Port, 11
- Remote\_wPort, 11
- Reserved wrapper port numbers, 15, 27
- Responder, 29
- SEND() function, 11
- SEND/RECEIVE state, 32
- Sequence numbers, 34
- Service specification, 16
- Setting up the TCP connection, 28
- Source UDP, 15
- Source wPort, 14
- State transition diagram, 31
- TCP Connected, 32
- TCP connection abort, 30
- TCP connection manager process, 9, 17
- TCP packets, 27
- TCP-ABORT.indication, 23
- TCP-CONNECT, 17
- TCP-CONNECT.confirm, 19
- TCP-CONNECT.indication, 18
- TCP-CONNECT.request, 17
- TCP-CONNECT.response, 19
- TCP-DATA service, 23, 36
- TCP-DATA.confirm, 25
- TCP-DATA.indication, 24
- TCP-DATA.request, 23
- TCP-DISCONNECT services, 20
- TCP-DISCONNECT.confirm, 22
- TCP-DISCONNECT.indication, 20
- TCP-DISCONNECT.request, 20
- TCP-DISCONNECT.response, 21
- Three-way handshake mechanism, 29
- Transmission Control Protocol, 15
- Transport layer and TCP connection establishment, 33
- UDP Datagram, 12, 14
- UDP-DATA service, 10
- UDP-DATA.confirm, 12
- UDP-DATA.indication, 12
- UDP-DATA.request, 11
- User Datagram Protocol, 9
- Valid wPort numbers, 12
- Virtual circuit, 16
- Well-known port number, 12
- Wrapper, 8
- Wrapper header, 13
- Wrapper protocol data unit, 13, 14, 27
- Wrapper sub-layer, 8, 13, 26

---

## Annex ZA (normative)

### Normative references to international publications with their corresponding European publications

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.

NOTE When an international publication has been modified by common modifications, indicated by (mod), the relevant EN/HD applies.

<u>Publication</u>	<u>Year</u>	<u>Title</u>	<u>EN/HD</u>	<u>Year</u>
IEC 60050-300	2001	International Electrotechnical Vocabulary - Electrical and electronic measurements and measuring instruments - Part 311: General terms relating to measurements - Part 312: General terms relating to electrical measurements - Part 313: Types of electrical measuring instruments - Part 314: Specific terms according to the type of instrument	-	-
IEC/TR 62051	1999	Electricity metering - Glossary of terms	-	-
IEC/TR 62051-1	2004	Electricity metering - Data exchange for meter - reading, tariff and load control - Glossary of terms - Part 1: Terms related to data exchange with metering using DLMS/COSEM	-	-
IEC 62056-53	2006	Electricity metering - Data exchange for meter reading, tariff and load control - Part 53: COSEM application layer	EN 62056-53	2007
IEC 62056-62	2006	Electricity metering - Data exchange for meter reading, tariff and load control - Part 62: Interface classes	EN 62056-62	2007
STD 0005	1981	Internet Protocol	-	-
STD 0006	1980	User Datagram Protocol	-	-
STD 0007	1981	Transmission Control Protocol	-	-



## BSI — British Standards Institution

BSI is the independent national body responsible for preparing British Standards. It presents the UK view on standards in Europe and at the international level. It is incorporated by Royal Charter.

### Revisions

British Standards are updated by amendment or revision. Users of British Standards should make sure that they possess the latest amendments or editions.

It is the constant aim of BSI to improve the quality of our products and services. We would be grateful if anyone finding an inaccuracy or ambiguity while using this British Standard would inform the Secretary of the technical committee responsible, the identity of which can be found on the inside front cover. Tel: +44 (0)20 8996 9000. Fax: +44 (0)20 8996 7400.

BSI offers members an individual updating service called PLUS which ensures that subscribers automatically receive the latest editions of standards.

### Buying standards

Orders for all BSI, international and foreign standards publications should be addressed to Customer Services. Tel: +44 (0)20 8996 9001. Fax: +44 (0)20 8996 7001. Email: [orders@bsi-global.com](mailto:orders@bsi-global.com). Standards are also available from the BSI website at <http://www.bsi-global.com>.

In response to orders for international standards, it is BSI policy to supply the BSI implementation of those that have been published as British Standards, unless otherwise requested.

### Information on standards

BSI provides a wide range of information on national, European and international standards through its Library and its Technical Help to Exporters Service. Various BSI electronic information services are also available which give details on all its products and services. Contact the Information Centre. Tel: +44 (0)20 8996 7111. Fax: +44 (0)20 8996 7048. Email: [info@bsi-global.com](mailto:info@bsi-global.com).

Subscribing members of BSI are kept up to date with standards developments and receive substantial discounts on the purchase price of standards. For details of these and other benefits contact Membership Administration. Tel: +44 (0)20 8996 7002. Fax: +44 (0)20 8996 7001. Email: [membership@bsi-global.com](mailto:membership@bsi-global.com).

Information regarding online access to British Standards via British Standards Online can be found at <http://www.bsi-global.com/bsonline>.

Further information about BSI is available on the BSI website at <http://www.bsi-global.com>.

### Copyright

Copyright subsists in all BSI publications. BSI also holds the copyright, in the UK, of the publications of the international standardization bodies. Except as permitted under the Copyright, Designs and Patents Act 1988 no extract may be reproduced, stored in a retrieval system or transmitted in any form or by any means – electronic, photocopying, recording or otherwise – without prior written permission from BSI.

This does not preclude the free use, in the course of implementing the standard, of necessary details such as symbols, and size, type or grade designations. If these details are to be used for any other purpose than implementation then the prior written permission of BSI must be obtained.

Details and advice can be obtained from the Copyright & Licensing Manager. Tel: +44 (0)20 8996 7070. Fax: +44 (0)20 8996 7553. Email: [copyright@bsi-global.com](mailto:copyright@bsi-global.com).