

Communication networks and systems in substations —

Part 7-2: Basic communication structure for substation and feeder equipment — Abstract communication service interface (ACSI)

The European Standard EN 61850-7-2:2003 has the status of a
British Standard

ICS 33.200

National foreword

This British Standard is the official English language version of EN 61850-7-2:2003. It is identical with IEC 61850-7-2:2003.

The UK participation in its preparation was entrusted to Technical Committee PEL/57, Power systems control, which has the responsibility to:

- aid enquirers to understand the text;
- present to the responsible international/European committee any enquiries on the interpretation, or proposals for change, and keep the UK interests informed;
- monitor related international and European developments and promulgate them in the UK.

A list of organizations represented on this committee can be obtained on request to its secretary.

Cross-references

The British Standards which implement international or European publications referred to in this document may be found in the *BSI Catalogue* under the section entitled “International Standards Correspondence Index”, or by using the “Search” facility of the *BSI Electronic Catalogue* or of British Standards Online.

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 does not of itself confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 4 July 2003

Summary of pages

This document comprises a front cover, an inside front cover, the EN title page, pages 2 to 172, an inside back cover and a back cover.

The BSI copyright date displayed in this document indicates when the document was last issued.

Amendments issued since publication

Amd. No.	Date	Comments

© BSI 4 July 2003

ISBN 0 580 42182 1

**Communication networks and systems in substations
Part 7-2: Basic communication structure for substation
and feeder equipment –
Abstract communication service interface (ACSI)
(IEC 61850-7-2:2003)**

Réseaux et systèmes de communication
dans les postes
Partie 7-2: Structure des communications
de base pour les postes électriques
et les équipements de lignes –
Interface abstraite des services
de communication (ACSI)
(CEI 61850-7-2:2003)

Kommunikationsnetze und -systeme
in Stationen
Teil 7-2: Grundlegende
Kommunikationsstruktur für stations- und
feldbezogene Ausrüstung –
Abstrakte Schnittstelle für
Kommunikationsdienste (ACSI)
(IEC 61850-7-2:2003)

This European Standard was approved by CENELEC on 2003-04-01. 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, Czech Republic, Denmark, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Luxembourg, Malta, Netherlands, Norway, Portugal, Slovakia, Spain, Sweden, Switzerland and 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 57/612/FDIS, future edition 1 of IEC 61850-7-2, prepared by IEC TC 57, Power system control and associated communications, was submitted to the IEC-CENELEC parallel vote and was approved by CENELEC as EN 61850-7-2 on 2003-04-01.

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) 2004-02-01
- latest date by which the national standards conflicting
with the EN have to be withdrawn (dow) 2006-04-01

Annexes designated "normative" are part of the body of the standard.
In this standard, annexes A and ZA are normative.
Annex ZA has been added by CENELEC.

Endorsement notice

The text of the International Standard IEC 61850-7-2:2003 was approved by CENELEC as a European Standard without any modification.

CONTENTS

INTRODUCTION.....	10
1 Scope.....	11
2 Normative references.....	11
3 Terms and definitions	12
4 Abbreviated terms.....	13
5 ACSI overview and basic concepts.....	14
5.1 General	14
5.2 Overview of basic information models	15
5.3 Overview of the other service models.....	16
5.4 Overview of ACSI services.....	19
5.5 Type definitions	20
5.5.1 Data attribute types.....	20
5.5.2 BasicTypes	21
5.5.3 Common ACSI types.....	22
6 SERVER class model	26
6.1 SERVER class definition.....	26
6.1.1 SERVER class syntax	26
6.1.2 SERVER class attributes.....	27
6.2 Server class services.....	27
6.2.1 Overview of directory and GetDefinition services	27
6.2.2 GetServerDirectory	28
7 Application association model	29
7.1 Introduction	29
7.2 Concept of application associations	29
7.3 Access control	29
7.4 TWO-PARTY-APPLICATION-ASSOCIATION (TPAA) class model.....	30
7.4.1 TWO-PARTY-APPLICATION-ASSOCIATION (TPAA) class definition	30
7.4.2 Two-party application association services	32
7.5 MULTICAST-APPLICATION-ASSOCIATION (MCAA) class	34
7.5.1 MULTICAST-APPLICATION-ASSOCIATION (MCAA) class definition.....	34
8 LOGICAL-DEVICE class model.....	36
8.1 LOGICAL-DEVICE class definition	36
8.1.1 LOGICAL-DEVICE class syntax	36
8.1.2 LOGICAL-DEVICE class attributes	36
8.2 LOGICAL-DEVICE class services	37
8.2.1 GetLogicalDeviceDirectory.....	37
9 LOGICAL-NODE class model.....	38
9.1 LOGICAL-NODE class definition	38
9.1.1 LOGICAL-NODE class syntax	38
9.1.2 LOGICAL-NODE class attributes.....	38
9.2 LOGICAL-NODE class services	40
9.2.1 Overview	40
9.2.2 GetLogicalNodeDirectory	40
9.2.3 GetAllDataValues.....	41

10	DATA class model	42
10.1	General	42
10.2	DATA class definition	42
10.2.1	DATA class syntax	42
10.2.2	DATA class attributes	45
10.3	Relation of DATA, common DATA, and compatible DATA classes	52
10.4	DATA class services	52
10.4.1	General definitions and overview	52
10.4.2	GetDataValues	53
10.4.3	SetDataValues	54
10.4.4	GetDataDirectory	55
10.4.5	GetDataDefinition	56
11	DATA-SET class model	56
11.1	General	56
11.2	DATA-SET class definition	58
11.2.1	DATA-SET class syntax	58
11.2.2	DATA-SET class attributes	58
11.3	DATA-SET class services	59
11.3.1	Overview	59
11.3.2	GetDataSetValues	59
11.3.3	SetDataSetValues	60
11.3.4	CreateDataSet	61
11.3.5	DeleteDataSet	61
11.3.6	GetDataSetDirectory	62
12	Substitution model	63
13	SETTING-GROUP-CONTROL-BLOCK class model	65
13.1	General	65
13.2	SGCB class definition	66
13.2.1	SGCB class syntax	66
13.2.2	SGCB class attributes	67
13.3	SGCB class services	68
13.3.1	Overview	68
13.3.2	SelectActiveSG	68
13.3.3	SelectEditSG	69
13.3.4	SetSGValues	70
13.3.5	ConfirmEditSGValues	71
13.3.6	GetSGValues	71
13.3.7	GetSGCBValues	72
14	REPORT-CONTROL-BLOCK and LOG-CONTROL-BLOCK class models	74
14.1	Overview	74
14.2	REPORT-CONTROL-BLOCK class model	76
14.2.1	Basic concepts	76
14.2.2	BUFFERED-REPORT-CONTROL-BLOCK (BRCB) class definition	76
14.2.3	BRCB class services	82
14.2.4	UNBUFFERED-REPORT-CONTROL-BLOCK (URCB) class definition	94
14.2.5	URCB class services	95
14.3	LOG-CONTROL-BLOCK class model	96
14.3.1	General	96
14.3.2	LCB class definition	97

14.3.3	LOG class definition.....	101
14.3.4	Procedures to generate the log entries	104
14.3.5	LOG services	104
15	Generic substation event class model (GSE).....	107
15.1	Overview	107
15.2	GOOSE-CONTROL-BLOCK (GoCB) class	109
15.2.1	GoCB definition	109
15.2.2	GOOSE service Definitions	111
15.2.3	Generic object oriented substation event (GOOSE) message	116
15.3	Generic substation state event (GSSE) control block (GsCB)	117
15.3.1	GsCB class definition.....	117
15.3.2	Generic substation state event (GSSE) control block class attributes.....	119
15.3.3	GSSE service definitions.....	120
15.3.4	Generic substation state event (GSSE) message	124
16	Transmission of sampled value class model.....	126
16.1	Overview	126
16.2	Transmission of sampled values using multicast	127
16.2.1	MSVCB class definition.....	127
16.2.2	Multicast sampled value class services	129
16.3	Transmission of sampled values using unicast.....	132
16.3.1	USVCB class definition	132
16.3.2	Unicast sampled value services	134
16.4	Sampled value format.....	137
16.4.1	MsvID or UsvID.....	138
16.4.2	OptFlds.....	138
16.4.3	DatSet	138
16.4.4	Sample [1..n]	138
16.4.5	SmpCnt	138
16.4.6	RefrTm	139
16.4.7	ConfRev	139
16.4.8	SmpSynch	139
16.4.9	SmpRate	139
17	CONTROL class model.....	139
17.1	Introduction	139
17.2	Control with normal security.....	140
17.2.1	Direct control with normal security.....	140
17.2.2	SBO control with normal security.....	141
17.3	Control with enhanced security	143
17.3.1	Introduction	143
17.3.2	Direct control with enhanced security	143
17.3.3	SBO control with enhanced security	144
17.4	Time-activated operate	146
17.5	CONTROL class service definitions	147
17.5.1	Overview	147
17.5.2	Service parameter definition.....	148
17.5.3	Service specification	150
18	Time and time-synchronization model	154
18.1	General	154
18.2	External information.....	154

19	Naming conventions	155
19.1	Class naming and class specializations.....	155
19.2	Referencing an instance of a class.....	156
19.3	Scope.....	158
20	File transfer	159
20.1	File transfer model.....	159
20.1.1	FileName	159
20.1.2	FileSize [0..1].....	159
20.1.3	LastModified	159
20.2	File services	160
20.2.1	GetFile	160
20.2.2	SetFile	160
20.2.3	DeleteFile	161
20.2.4	GetFileAttributeValues	162
	Annex A (normative) ACSI conformance statement.....	163
A.1	General	163
A.2	ACSI models conformance statement.....	163
A.3	ACSI service conformance statement.....	164
	Annex ZA (normative) Normative references to international publications with their corresponding European publications	169
	Bibliography.....	170
	Index	171
	Figure 1 – Excerpt of conceptual model.....	15
	Figure 2 – Basic conceptual class model of the ACSI.....	16
	Figure 3 – Conceptual service model of the ACSI	17
	Figure 4 – Data attribute type concept	20
	Figure 5 – Overview about GetDirectory and GetDefinition services	27
	Figure 6 – Access views of a server	30
	Figure 7 – Normal operation	31
	Figure 8 – Aborting association	31
	Figure 9 – Principle of multicast application association.....	35
	Figure 10 – Class diagram of DATA and DataAttributeType	43
	Figure 11 – Example of DATA	44
	Figure 12 – Relation of TrgOp and Reporting.....	50
	Figure 13 – Relation of DATA classes	52
	Figure 14 – Excerpt of data class services.....	53
	Figure 15 – Dynamic creation of data set instances	57
	Figure 16 – Principles of substitution	64
	Figure 17 – Basic model of the settings model.....	65
	Figure 18 – Setting group state machine.....	67

Figure 19 – Basic building blocks for reporting and logging	75
Figure 20 – BRCB state machine.....	78
Figure 21 – Buffer time.....	80
Figure 22 – Report example on the use of sequence number	84
Figure 23 – Data set members and reporting	86
Figure 24 – Report example	87
Figure 25 – Log model overview	96
Figure 26 – GoCB model	108
Figure 27 – Specifics for GsCB model	118
Figure 28 – Model for transmission of sampled values	126
Figure 29 – Principle of the control model.....	140
Figure 30 – State machine of direct control with normal security	141
Figure 31 – Direct control with normal security.....	141
Figure 32 – State machine of SBO control with normal security.....	142
Figure 33 – State machine of direct control with enhanced security.....	143
Figure 34 – State machine SBO control with enhanced security	144
Figure 35 – Select before operate with enhanced security – positive case	145
Figure 36 – Select before operate with enhanced security – negative case (no status change)	145
Figure 37 – Time-activated operate	147
Figure 38 – Time model and time synchronization (principle)	154
Figure 39 – Specializations.....	155
Figure 40 – Object names and object reference	158
 Table 1 – ACSI classes	 19
Table 2 – BasicTypes	21
Table 3 – ObjectName type	22
Table 4 – ObjectReference type	22
Table 5 – ServiceError type	23
Table 6 – PACKED-LIST type.....	23
Table 7 – TimeStamp type.....	24
Table 8 – TimeQuality definition	24
Table 9 – TimeAccuracy.....	25
Table 10 – TriggerConditions type	26
Table 11 – SERVER class definition	26
Table 12 – TWO-PARTY-APPLICATION-ASSOCIATION (TPAA) class definition	31
Table 13 – MULTICAST-APPLICATION-ASSOCIATION (MCAA) class definition.....	35
Table 14 – LOGICAL-DEVICE (LD) class definition.....	36
Table 15 – LOGICAL-NODE (LN) class definition.....	38
Table 16 – DATA class definition.....	45
Table 17 – DAType definition.....	46
Table 18 – Functional constraints	48
Table 19 – Trigger option	48

Table 20 – COMMON-DATA class definition	51
Table 21 – DATA-SET (DS) class definition	58
Table 22 – SGCB class definition	66
Table 23 – BRCB class definition	77
Table 24 – Report format specification	83
Table 25 – URCB class definition	94
Table 26 – LCB class definition	97
Table 27 – LOG class definition.....	102
Table 28 – GOOSE control block class definition	109
Table 29 – GOOSE message definition.....	116
Table 30 – GSSE control block class definition	118
Table 31 – GSSE message definition.....	125
Table 32 – MSVCB class definition	127
Table 33 – USVCB class definition	132
Table 34 – Sampled value (SV) format definition	138
Table 35 – Control services	147
Table 36 – Control time-stamp definition.....	148
Table 37 – Test status definition.....	148
Table 38 – Check condition definition	148
Table 39 – Additional cause diagnosis definition	149
Table 40 – AddCause semantic	149
Table 41 – TimeActivatedOperate response definition	150
Table 42 – List of ObjectReferences	156
Table 43 – FILE class definition.....	159
Table A.1 – Basic conformance statement.....	163
Table A.2 – ACSI models conformance statement	164
Table A.3 – ACSI service conformance statement	165

INTRODUCTION

This document is part of a set of specifications which details a layered substation communication architecture. This architecture has been chosen to provide abstract definitions of classes and services such that the specifications are independent of specific protocol stacks, implementations, and operating systems.

The IEC 61850 series is intended to provide interoperability between a variety of substation and feeder devices. Communication between these devices is achieved by the definition of a hierarchical class model (for example, logical device, logical node, data, data set, report control, or log) and services provided by these classes (for example, get, set, report, define, delete) in parts IEC 61850-7-x.

This part of IEC 61850 defines the abstract communication service interface (ACSI) for use in the utility substation domain that require real-time cooperation of intelligent electronic devices. The ACSI has been defined so as to be independent of the underlying communication systems. Specific communication service mappings³ (SCSM) are specified in part 8-x and part 9-x of this standard.

This part of IEC 61850 defines the abstract communication service interface in terms of

- a hierarchical class model of all information that can be accessed via a communication network,
- services that operate on these classes, and
- parameters associated with each service.

The ACSI description technique abstracts away from all the different approaches to implement the cooperation of the various devices.

NOTE 1 Abstraction in ACSI has two meanings. First, only those aspects of a real device (for example, a breaker) or a real function that are visible and accessible over a communication network are modelled. This abstraction leads to the hierarchical class models and their behaviour defined in IEC 61850-7-2, IEC 61850-7-3, and IEC 61850-7-4. Second, the ACSI abstracts from the aspect of concrete definitions on how the devices exchange information; only a conceptual cooperation is defined. The concrete information exchange is defined in the SCSMs.

NOTE 2 This part of IEC 61850 does not provide comprehensive tutorial material. It is recommended that IEC 61850-5 and IEC 61850-7-1 be read first in conjunction with IEC 61850-7-2 and IEC 61850-7-3.

NOTE 3 Examples use names of classes (e.g. XCBR for a class of a logical node) defined in IEC 61850-7-4 and IEC 61850-7-3. The normative names are defined in IEC 61850-7-4 and IEC 61850-7-3 only.

³ The ACSI is independent of the specific mapping. Mappings to standard application layers or middle ware technologies are possible.

COMMUNICATION NETWORKS AND SYSTEMS IN SUBSTATIONS –

Part 7-2: Basic communication structure for substation and feeder equipment – Abstract communication service interface (ACSI)

1 Scope

This part of IEC 61850 applies to the ACSI communication in substations and feeder applications. The ACSI provides the following abstract interfaces.

- a) Abstract interface describing communications between a client and a remote server for
 - real-time data access and retrieval,
 - device control,
 - event reporting and logging,
 - publisher/subscriber,
 - self-description of devices (device data dictionary),
 - data typing and discovery of data types, and
 - file transfer.
- b) Abstract interface for fast and reliable system-wide event distribution between an application in one device and many remote applications in different devices (publisher/subscriber) and for transmission of sampled measured values (publisher/subscriber).

This part of IEC 61850 may also be applied to describe device models and functions for additional activities, such as:

- substation to substation information exchange,
- substation to control centre information exchange,
- power plant to control centre information exchange,
- information exchange for distributed generation, or
- information exchange for metering.

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 61850-2, *Communication networks and systems in substations – Part 2: Glossary*

IEC 61850-5, *Communication networks and systems in substations – Part 5: Communication requirements for functions and devices models*

IEC 61850-7-1, *Communication networks and systems in substations – Part 7-1: Basic communication structure for substation and feeder equipment – Principles and models*

IEC 61850-7-3, *Communication networks and systems in substations – Part 7-3: Basic communication structure for substation and feeder equipment – Common data classes*

IEC 61850-7-4, *Communication networks and systems in substations – Part 7-4: Basic communication structure for substation and feeder equipment – Compatible logical node classes and data classes*

IEC 61850-8-1: *Communication networks and systems in substations – Part 8-1: Specific communication service mapping (SCSM) – Mappings to MMS (ISO/IEC 9506-1 and ISO/IEC 9506-2) and to ISO/IEC 8802-3*

3 Terms and definitions

For the purpose of this document, the terms and definitions provided in IEC 61850-2 and the following definitions apply.

3.1

class

description of a set of objects that share the same attributes, services, relationships, and semantics

3.2

client

entity that requests a service from a server and that receives unsolicited messages from a server

3.3

device

entity that performs control, actuating and/or sensing functions and interfaces to other such entities within an automation system

NOTE Devices alone do not perform energy transport functions.

3.4

external equipment

entity that is stand-alone, or interfaces to an automation system, and that performs energy transport functions

EXAMPLE Transformer, circuit-breaker, line.

NOTE 1 Equipment can contain devices.

NOTE 2 Equipment cannot have a direct connection to the communication network – only devices can be directly connected to the communication network.

3.5

instance (of a class)

entity that has unique identity, to which a set of services can be applied, and which has a state that stores the effects of the services

NOTE Instance is a synonym for the term object.

3.6

Logical device

entity that represents a set of typical substation functions

3.7

Logical node

entity that represents a typical substation function

3.8

physical device

entity that represents the physical parts of a device (hardware and operating system, etc.)

NOTE Physical devices host logical devices.

4 Abbreviated terms

AA	APPLICATION-ASSOCIATION
ACSI	abstract communication service Interface
BRCB	BUFFERED-REPORT-CONTROL-BLOCK
CDC	common DATA class (IEC 61850-7-3)
CT	current transformer
DA	data attribute
DAType	data attribute type
DataRef	data reference
dchg	data change trigger option
DS	DATA-SET
dupd	data-update trigger option
FC	functional constraint
FCD	functionally constrained DATA
FCDA	functionally constrained DataAttribute
GI	general interrogation
GoCB	GOOSE-CONTROL-BLOCK
GOOSE	generic object oriented substation events
GSE	generic substation event
GsCB	GSSE-CONTROL-BLOCK
GSSE	generic substation status event
IED	intelligent electronic device
IntgPd	integrity period
LCB	LOG-CONTROL-BLOCK
LD	LOGICAL-DEVICE
LN	LOGICAL-NODE
MC	multicast
MCAA	multicast application association
MMS	manufacturing message specification
MSVCB	MULTICAST-SAMPLED-VALUE-CONTROL-BLOCK
PDU	protocol data unit
PICS	protocol implementation conformance statement
PIXIT	protocol Implementation extra information
qchg	quality change trigger option
SBO	select before operate

SCL	substation configuration language (IEC 61850-6)
SCSM	specific communication service mapping (defined in IEC 61850-8-x and IEC 61850-9-x)
SG	setting group
SGCB	SETTING-GROUP-CONTROL-BLOCK
SoE	sequence-of-events
SVC	sampled value control
TP	TWO-PARTY
TPAA	TWO-PARTY-APPLICATION-ASSOCIATION
TrgOp	trigger option
UCA™	Utility Communication Architecture
URCB	UNBUFFERED-REPORT-CONTROL-BLOCK
UTC	coordinated universal time
SV	sampled value
USVCB	UNICAST-SAMPLED-VALUE-CONTROL-BLOCK
VT	voltage transformer

5 ACSI overview and basic concepts

5.1 General

The models of the ACSI provide

- the specification of a basic model for the definition of the substation-specific information models contained in IEC 61850-7-3 (common **DATA** classes) and IEC 61850-7-4 (compatible **LOGICAL-NODE** classes and compatible **DATA** classes) and
- the specification of information exchange service models.

The information models and information exchange services are interwoven. From a descriptive point of view, the two aspects are separated to some degree (see the excerpt shown in Figure 1). The common models (for example, **LOGICAL-NODE** and **DATA** classes including their services) are applied in IEC 61850-7-3 and IEC 61850-7-4 to define many specialized information models – the substation automation models.

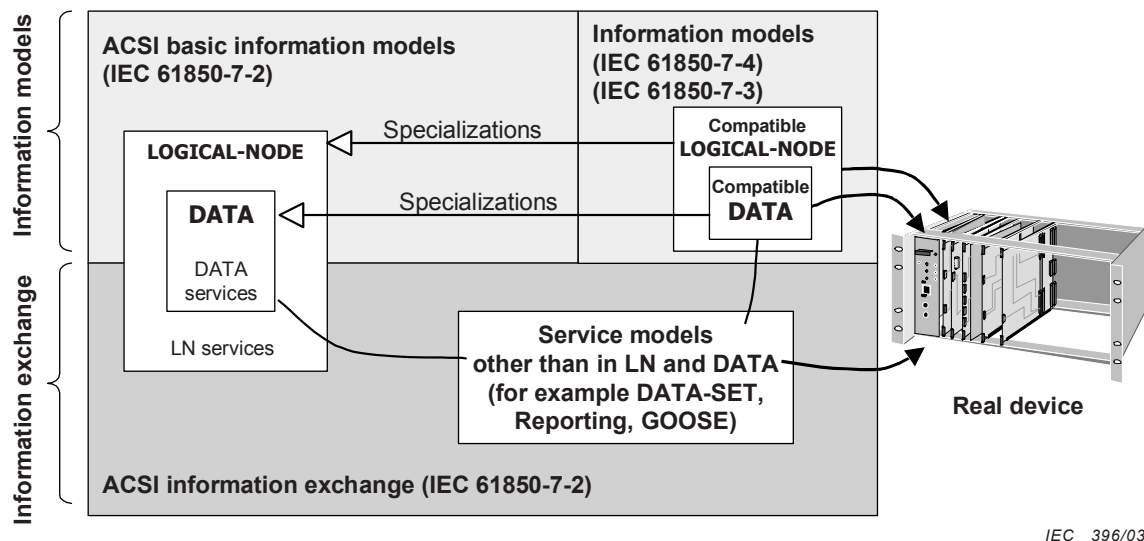


Figure 1 – Excerpt of conceptual model

Other service models required for substation automation systems (for example, **DATA-SET** and reporting provide specific information exchange services) are also defined in this part of the standard; these models are linked to **LOGICAL-NODEs** and **DATA**. The information exchange services are completely defined in the ACSI. The information models defined in IEC 61850-7-4 reference the services defined in the various models of the ACSI.

5.2 Overview of basic information models

The conceptual models to build the domain-specific information models are:

- SERVER** – represents the external visible behaviour of a device. All other ACSI models are part of the server.
 NOTE 1 A server has two roles: to communicate with a client (most service models in IEC 61850 provide communication with client devices) and to send information to peer devices (for example, for sampled values).
- LOGICAL-DEVICE (LD)** – contains the information produced and consumed by a group of domain-specific application functions; functions are defined as **LOGICAL-NODEs**.
- LOGICAL-NODE (LN)** – contains the information produced and consumed by a domain-specific application function, for example, overvoltage protection or circuit-breaker.
- DATA** – provide means to specify typed information, for example, position of a switch with quality information and timestamp, contained in **LOGICAL-NODEs**.

Each of these information models is defined as a class. The classes comprise attributes and services. The conceptual class diagram of the ACSI is depicted in Figure 2.

NOTE 2 The classes are major building blocks that provide the framework for substation automation device models. Additional details on the modelling and relations between IEC 61850-7-4, IEC 61850-7-3, and this part of IEC 61850 can be found in IEC 61850-7-1.

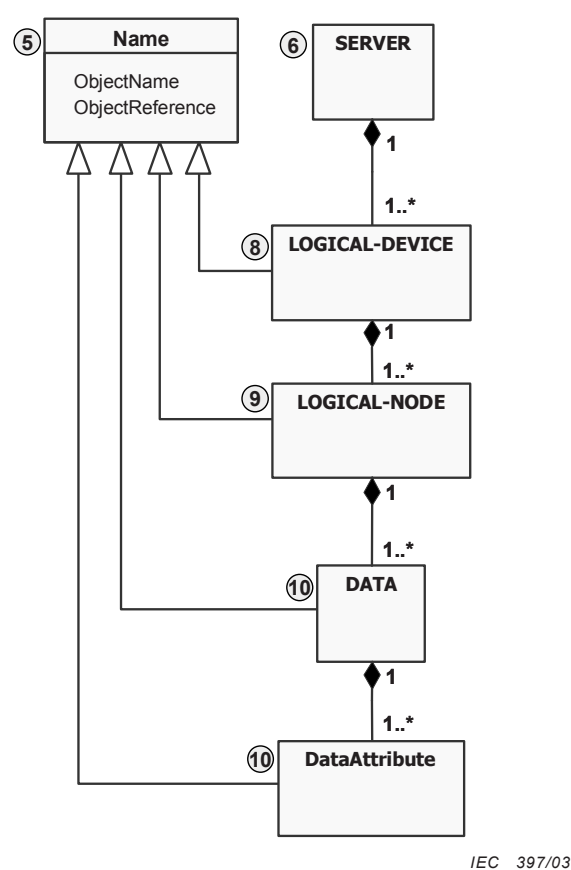


Figure 2 – Basic conceptual class model of the ACSI

NOTE 3 The numbers in the circles indicate the respective clauses in this part of IEC 61850.

The Name class is inherited by the classes **LOGICAL-DEVICE**, **LOGICAL-NODE**, **DATA**, and **DataAttribute**.

EXAMPLE In an implementation the logical device, logical node, data, and data attribute have each an object name (instance name) which is a unique name among classes of the same container to which they belong. In addition, each of the four has an ObjectReference (path name) which is a concatenation of all object names from each container. The four object names (one per column) can be concatenated.

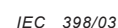
	Logical device	Logical node	Data	Data attribute
Object name	"Atlanta_HV5"	"XCBR1"	"Pos"	"stVal"
Description	High-voltage station 5	Circuit-breaker 1	Position	Status value

5.3 Overview of the other service models

In addition to the models listed above, the ACSI comprises the following models that provide services operating on data, data attributes, and data sets.

- a) **DATA-SET** – permits the grouping of data and data attributes. Used for direct access and for reporting and logging.
- b) **Substitution** – supports replacement of a process value by another value.
- c) **SETTING-GROUP-CONTROL-BLOCK** – defines how to switch from one set of setting values to another one and how to edit setting groups.

- An overview of the conceptual service model of the ACSI is shown in Figure 3.



AVA Documents
Document Sharing Hub

NOTE 1 The numbers in the circles indicate the respective clauses in this part of IEC 61850.

NOTE 2 The class diagrams are conceptual. Details are defined in the respective clauses. Comprehensive diagrams are contained in IEC 61850-7-1. The **DATA** class may be defined recursively. The operations for substitution and control are restricted to the lowest level in the **DATA** class. The DataAttributes may be defined recursively as well.

The logical node is one of the major building blocks that has associations to most of the other information exchange models, for example, report control, log control, and setting control.

Any other information exchange service model, for example, report control, log control, and setting control shall inherit the ObjectName and ObjectReference as depicted in Figure 2.

NOTE 3 The class models and services are defined using an object-oriented approach allowing for the mapping of class models and services to different application layer and middle ware solutions.

5.4 Overview of ACSI services

The complete list of ACSI classes and their services is shown in Table 1.

Table 1 – ACSI classes

<u>SERVER model (Clause 6)</u> GetServerDirectory <u>ASSOCIATION model (Clause 7)</u> Associate Abort Release <u>LOGICAL-DEVICE model (Clause 8)</u> GetLogicalDeviceDirectory <u>LOGICAL-NODE model (Clause 9)</u> GetLogicalNodeDirectory GetAllDataValues <u>DATA model (Clause 10)</u> GetDataValues SetDataValues GetDataDirectory GetDataDefinition <u>DATA-SET model (Clause 11)</u> GetDataSetValues SetDataSetValues CreateDataSet DeleteDataSet GetDataSetDirectory <u>Substitution model (Clause 12)</u> SetDataValues GetDataValues <u>SETTING-GROUP-CONTROL-BLOCK model (Clause 13)</u> SelectActiveSG SelectEditSG SetSGValues ConfirmEditSGValues GetSGValues GetSGCBValues <u>REPORT-CONTROL-BLOCK and LOG-CONTROL-BLOCK model (Clause 14)</u> BUFFERED-REPORT-CONTROL-BLOCK: Report GetBRCBValues SetBRCBValues UNBUFFERED-REPORT-CONTROL-BLOCK: Report GetURCBValues SetURCBValues	LOG-CONTROL-BLOCK model: GetLCBValues SetLCBValues QueryLogByTime QueryLogAfter GetLogStatusValues <u>Generic substation event model – GSE (Clause 15)</u> GOOSE SendGOOSEMessage GetGoReference GetGOOSEElementNumber GetGoCBValues SetGoCBValues GSSE SendGSSEMessage GetGsReference GetGSSEDataOffset GetGsCBValues SetGsCBValues <u>Transmission of sampled values model (Clause 16)</u> MULTICAST-SAMPLE-VALUE-CONTROL-BLOCK: SendMSVMessage GetMSVCBValues SetMSVCBValues UNICAST-SAMPLE-VALUE-CONTROL-BLOCK: SendUSVMessage GetUSVCBValues SetUSVCBValues <u>Control model (Clause 17)</u> Select SelectWithValue Cancel Operate CommandTermination TimeActivatedOperate <u>Time and time synchronization (Clause 18)</u> TimeSynchronization <u>FILE transfer model (Clause 20)</u> GetFile SetFile DeleteFile GetFileAttributeValues
--	---

5.5 Type definitions

5.5.1 Data attribute types

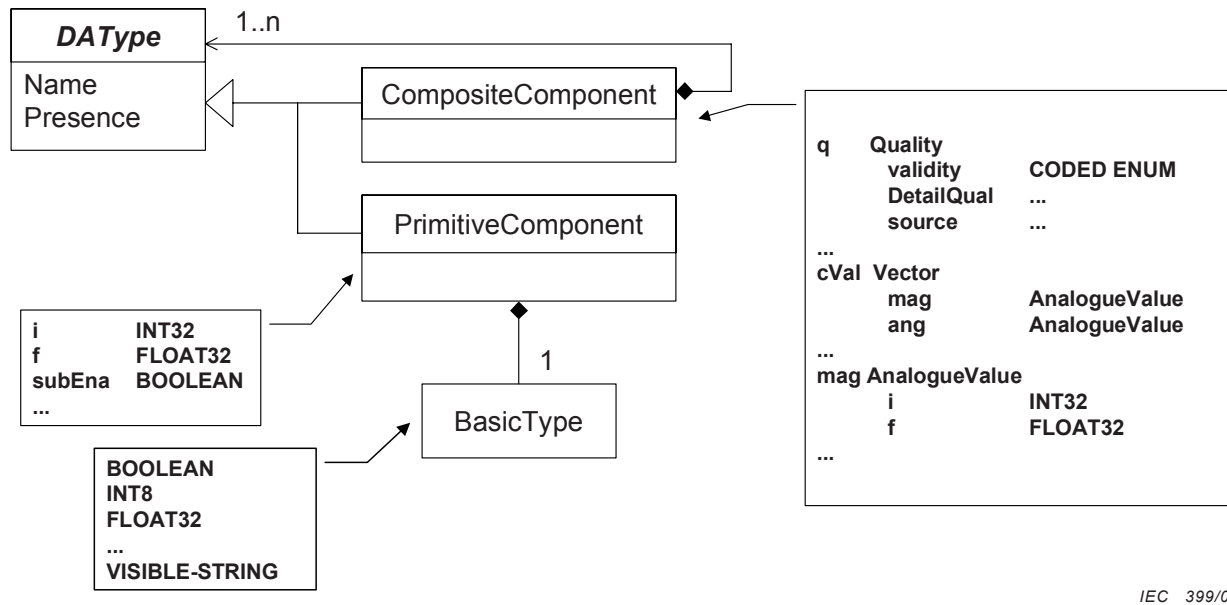
IEC 61850-7-2 and IEC 61850-7-3 shall use the types that are defined in the following subclauses in order to define the specific data for the application models in IEC 61850-7-4 and the control blocks in this part of IEC 61850 (for example, report control blocks).

The data attribute type concept is depicted in Figure 4. The data attribute type (**DAType**) is a class that has a Name, an indication (Presence) if the attribute is mandatory (present) or optional (possibly not-present), and BasicTypes.

- NOTE 1 The **DAType** class is an abstract class that is an auxiliary means to construct the primitive and composite components.
- NOTE 2 The formal specification of the **DAType** class and the use of **DATypes** to specify the types of data attributes can be found in Clause 10. The class diagram has been introduced in this subclause to depict the context in which the basic types are used.
- NOTE 3 A comprehensive example is provided in IEC 61850-7-1.

The **BasicTypes** (for example, **BOOLEAN** and **INT8**) are used to build **PrimitiveComponents** and **CompositeComponents**. **PrimitiveComponents** shall have a Name, a Presence, and a **BasicType** (for example, Name = **i**, Presence = Mandatory, and **BasicType** = **INT32**). The **Composite Component** is constructed by one or more **PrimitiveComponents** each of **BasicType** (for example, Name = **mag** of type AnalogueValue comprising two **PrimitiveComponents i** (of **INT32**) and **f** (of **FLOAT32**)).

Common **CompositComponents** and **PrimitiveComponents** are defined in the various common **DATA** classes of IEC 61850-7-3.



IEC 399/03

NOTE The attribute *Presence* is not shown in the examples.

Figure 4 – Data attribute type concept

5.5.2 BasicTypes

The **BasicTypes** shall be as listed in Table 2.

Table 2 – BasicTypes

BasicTypes			
Name	Value range	Remark	Used by
BOOLEAN			IEC 61850-7-3 IEC 61850-7-2
INT8	-128 to 127		IEC 61850-7-3 IEC 61850-7-2
INT16	-32 768 to 32 767		IEC 61850-7-3 IEC 61850-7-2
INT24	-8 388 608 to 8 388 607	for TimeStamp type	IEC 61850-7-2
INT32	-2 147 483 648 to 2 147 483 647		IEC 61850-7-3 IEC 61850-7-2
INT128	-2^{127} to $(2^{127})-1$	Required for counters	IEC 61850-7-3
INT8U	Unsigned integer, 0 to 255		IEC 61850-7-3 IEC 61850-7-2
INT16U	Unsigned integer, 0 to 65 535		IEC 61850-7-3 IEC 61850-7-2
INT24U	Unsigned integer, 0 to 16 777 215		IEC 61850-7-2
INT32U	Unsigned integer, 0 to 4 294 967 295		IEC 61850-7-3 IEC 61850-7-2
FLOAT32	Range of values and precision as specified by IEEE 754 single- precision floating point		IEC 61850-7-3
FLOAT64	Range of values and precision as specified by IEEE 754 double- precision floating point		IEC 61850-7-3
ENUMERATED	Ordered set of values, defined where type is used	Custom extensions are allowed	IEC 61850-7-3 IEC 61850-7-2
CODED ENUM	Ordered set of values, defined where type is used	Custom extensions shall not be allowed. Type shall be mapped to an efficient encoding in a SCSM	IEC 61850-7-3 IEC 61850-7-2
OCTET STRING	Max. length shall be defined where type is used ^a		IEC 61850-7-3 IEC 61850-7-2
VISIBLE STRING	Max. length shall be defined where type is used ^a		IEC 61850-7-3 IEC 61850-7-2
UNICODE STRING	Max. length shall be defined where type is used ^a		IEC 61850-7-3

^a The length suffix shall have the format "...STRINGnn" where "nn" is the length in characters.

5.5.3 Common ACSI types

5.5.3.1 General

The common ACSI types shall be used for the attribute definitions of the classes (for example, report control blocks) defined in this part of IEC 61850. The common ACSI types may also be used in the application models defined in IEC 61850-7-3 and IEC 61850-7-4.

5.5.3.2 ObjectName

The **ObjectName** shall specify a unique instance name among instances of a class owned by the same parent class with a type as specified in Table 3.

Table 3 – ObjectName type

ObjectName type			
Attribute name	Attribute type	Value/value range/explanation	Used by
ObjectName	VISIBLE STRING32	Name of an instance of a class of a single hierarchy level	IEC 61850-7-4 IEC 61850-7-3 IEC 61850-7-2
NOTE Clause 19 specifies constraints on the use of the type ObjectName.			

5.5.3.3 ObjectReference

Instances of classes in the hierarchical information model (ACSI class hierarchy of logical device, logical node, data, data attributes) shall be constructed by the concatenation of all instance names comprising the whole path-name of an instance of a class that identifies the instance uniquely. The type of the **ObjectReference** shall be as specified in Table 4.

Table 4 – ObjectReference type

ObjectReference type			
Attribute name	Attribute type	Value/value range/explanation	Used by
ObjectReference	VISIBLE STRING255	ObjectReference comprises the whole path-name of an instance of a class that identifies the instance uniquely	IEC 61850-7-2

The **ObjectReference** syntax shall be:

LDName/LNName[.Name[. ...]]

The “/” shall separate the instance name of a logical device (**LDName**) from the name of an instance of a logical node (**LNName**). The “.” shall separate the further names in the hierarchy. The “[]” shall indicate an option. The inner square bracket “[. ...]” shall indicate further names of recursively nested definitions.

NOTE 1 In any case where the context of the text provides sufficient information that an instance of a class is meant, the term “instance of” is not used.

NOTE 2 Clause 19 specifies constraints on the use of the type ObjectReference.

5.5.3.4 ServiceError type

The service error code for negative service responses (originated within the server) shall be as specified in Table 5.

Table 5 – ServiceError type

ServiceError type definition			
Attribute name	Attribute type	Value /value range/explanation	Used by
ServiceError	ENUMERATED	instance-not-available instance-in-use access-violation access-not-allowed-in-current-state parameter-value-inappropriate parameter-value-inconsistent class-not-supported instance-locked-by-other-client control-must-be-selected type-conflict failed-due-to-communications-constraint failed-due-to-server-constraint	IEC 61850-7-2

Additional **ServiceError** values for negative service responses (originated in the application, for example, additional cause diagnosis for control-related services) shall be as specified in the appropriate service models.

NOTE The **ServiceError** may be extended by an SCSM and the application layer referenced by an SCSM.

5.5.3.5 EntryID type

The type **EntryID** shall represent an arbitrary OCTET STRING used to identify an entry in a sequence of events such as a log or a buffered report as specified by an SCSM.

NOTE 1 The **EntryID** (handle) allows a client to re-synchronize, for example, with the sequence of the events stored in the IED. The syntax and semantic of the **EntryID** are outside the scope of this standard.

NOTE 2 The **EntryID** is used in this part of IEC 61850.

5.5.3.6 Packed list type

The **PACKED LIST** type shall be as defined in Table 6.

Table 6 – PACKED-LIST type

PACKED-LIST type definition			
Name	Value range	Remark	Used by
PACKED LIST	Ordered list of types; defined where type is used	Any value inside a PACKED LIST shall be mapped to an efficient encoding in a SCSM. No access to individual members of the list is required	IEC 61850-7-3 IEC 61850-7-2

5.5.3.7 TimeStamp type

5.5.3.7.1 General

The relation between a time stamp value, the synchronization of an internal time with an external time source (for example, UTC time), and other time-model-related information are defined in Clause 18.

NOTE 1 The TimeStamp type relies on requirements specified in Clause 18. The reader should first read that clause. The presentation of the TimeStamp is defined in the SCSMs.

NOTE 2 The TimeStamp is used in this part of IEC 61850 and in IEC 61850-7-3.

5.5.3.7.2 TimeStamp syntax

The **TimeStamp** type shall represent a UTC time with the epoch of midnight (00:00:00) of 1970-01-01 specified in Table 7.

Table 7 – TimeStamp type

TimeStamp type definition			
Attribute name	Attribute type	Value/value range/explanation	M/O
SecondSinceEpoch	INT32	(0...MAX)	M
FractionOfSecond	INT24U	Value = SUM from i=0 to 23 of $b_i \cdot 2^{23-(i+1)}$; Order = b0, b1, b2, b3, ...	M
TimeQuality	TimeQuality		M

5.5.3.7.3 TimeStamp attributes

5.5.3.7.3.1 SecondSinceEpoch

The **SecondSinceEpoch** shall be the interval in seconds continuously counted from the epoch 1970-01-01 00:00:00 UTC.

NOTE SecondSinceEpoch corresponds with the Unix epoch.

5.5.3.7.3.2 FractionOfSecond

The attribute **FractionOfSecond** shall be the fraction of the current second when the value of the **TimeStamp** has been determined. The fraction of second shall be calculated as (SUM from $l = 0$ to 23 of $b_l \cdot 2^{23-(l+1)}$ s).

NOTE 1 The resolution is the smallest unit by which the time stamp is updated. The 24 bits of the integer provides 1 out of 16777216 counts as the smallest unit; calculated by $1/2^{24}$ which equals approximately 60 ns.

NOTE 2 The resolution of a time stamp may be $1/2^{21}$ (= 0,5 s) if only the first bit is used; or may be $1/2^{22}$ (= 0,25 s) if the first two bits are used; or may be approximately 60 ns if all 24 bits are used. The resolution provided by an IED is outside the scope of this standard.

5.5.3.7.3.3 TimeQuality

The **TimeQuality** shall provide information about the time source of the sending IED as listed in Table 8.

Table 8 – TimeQuality definition

TimeQuality definition			
Attribute name	Attribute type	Value/Value range/explanation	M/O
	PACKED LIST		
LeapSecondsKnown	BOOLEAN		M
ClockFailure	BOOLEAN		M
ClockNotSynchronized	BOOLEAN		O
TimeAccuracy	CODED ENUM	Number of significant bits in the FractionOfSecond: Minimum time interval shall be: 2^{23-n}	M

LeapSecondsKnown: The value TRUE of the attribute **LeapSecondsKnown** shall indicate that the value for SecondSinceEpoch takes into account all leap seconds occurred. If it is FALSE then the value does not take into account the leap seconds that occurred before the initialization of the time source of the device.

ClockFailure: The attribute **clockFailure** shall indicate that the time source of the sending device is unreliable. The value of the TimeStamp shall be ignored.

ClockNotSynchronized: The attribute **clockNotSynchronized** shall indicate that the time source of the sending device is not synchronized with the external UTC time.

TimeAccuracy: The attribute **TimeAccuracy** shall represent the time accuracy class of the time source of the sending device relative to the external UTC time. The **timeAccuracy** classes shall represent the number of significant bits in the **FractionOfSecond**.

The values of n shall be as listed in Table 9.

NOTE 1 The **TimeAccuracy** meets the requirements specified in IEC 61850-5 for the selected values of n.

Table 9 – TimeAccuracy

n	Resulting TimeAccuracy (2 ⁿ –n)	Corresponding time performance class defined in IEC 61850-5
31	–	– unspecified
7	approx. 7,8 ms	10 ms (performance class T0)
10	approx. 0,9 ms	1 ms (performance class T1)
14	approx. 61 µs	100 µs (performance class T2)
16	approx. 15 µs	25 µs (performance class T3)
18	approx. 3,8 µs	4 µs (performance class T4)
20	approx. 0,9 µs	1 µs (performance class T5)

5.5.3.8 EntryTime type

The type **EntryTime** shall represent the time and date as applied internally for the communication, reporting, logging, and subsystem as specified by a SCSM.

NOTE 1 The TimeStamp type is used for common **DATA** classes in IEC 61850-7-3 and definition of compatible **DATA** classes in IEC 61850-7-4. The **EntryTime** type is used for all IEC 61850-7-2 class definitions. The **EntryTime** type may or may not be the same as **TimeStamp** in a SCSM.

NOTE 2 The EntryTime is used in this part of IEC 61850.

5.5.3.9 TriggerConditions type

The **TriggerConditions** type shall represent the trigger conditions used to trigger processing reports and logs (see Table 10).

NOTE 1 The TriggerConditions are used in this part of IEC 61850 and in IEC 61850-7-3.

Table 10 – TriggerConditions type

TriggerConditions type			
Attribute name	Attribute type	TriggerOption (TrgOp) for use in DataAttributes	Value/value range/explanation
	PACKED LIST		
data-change	BOOLEAN	dchg	Trigger used in DATA-Attributes determined by common DATA classes of IEC 61850-7-3
quality-change	BOOLEAN	qchg	Trigger used in DATA-Attributes determined by common DATA classes of IEC 61850-7-3
data-update	BOOLEAN	dupd	Trigger used in DATA-Attributes determined by common DATA classes of IEC 61850-7-3
integrity	BOOLEAN	–	Trigger whose value (time) can be set by a service or by configuration; independent of an instance of DATA
general-interrogation	BOOLEAN	–	Trigger whose value (initiate general interrogation) can be set by a service or by configuration; independent of an instance of DATA

The TriggerOption (**TrgOp**) shall be used in the specification of **DataAttributes** to indicate on which change/update the value of an instance of a **DataAttribute** may be reported or logged.

NOTE 2 Details on the use of **TriggerConditions** are defined in 10.2.2.4.3 and Clause 14.

6 SERVER class model

6.1 SERVER class definition

6.1.1 SERVER class syntax

The class **SERVER** shall represent the externally visible behaviour of a device. The **SERVER** shall be a composition as defined in Table 11.

NOTE 1 For simple devices the server may comprise just one logical device with the GOOSE control model with no other service.

Table 11 – SERVER class definition

SERVER class		
Attribute name	Attribute type	Value/value range/explanation
ServiceAccessPoint [1..n]	(*)	(*) Type is SCSCM specific
LogicalDevice [1..n]	LOGICAL-DEVICE	
File [0..n]	FILE	
TPAppAssociation [0..n]	TWO-PARTY-APPLICATION-ASSOCIATION	
MCAAppAssociation [0..n]	MULTICAST-APPLICATION-ASSOCIATION	
Services GetServerDirectory		

NOTE 2 The server's relationship to the underlying communication system and the concrete implementation depend on the SCSCM (specific communication service mapping, see IEC 61850-8-x and IEC 61850-9-x) used. Network management (as part of an SCSCM), device management, and system management are outside the scope of IEC 61850-7-2.

6.1.2 SERVER class attributes

6.1.2.1 ServiceAccessPoint

The attribute **ServiceAccessPoint** shall identify a **SERVER** within the scope of a system.

NOTE The **ServiceAccessPoint** is an abstraction of an address used to identify the server in the underlying SCSCM. The type depends on the SCSCM and should be defined there. A specific **ServiceAccessPoint** is required by most services to address a server. Nevertheless, it has not been included explicitly in the service parameter tables throughout this part of IEC 61850.

6.1.2.2 LogicalDevice [1..n]

The attribute **LogicalDevice** shall identify a **LogicalDevice** that is contained in a **SERVER**.

6.1.2.3 File [0..n]

The attribute **File** shall identify a **File** contained in a **SERVER**.

6.1.2.4 TAppAssociation [0..n] – two-party application association

The attribute **TAppAssociation** shall identify a client with which a **SERVER** maintains a two-party application association.

NOTE Details can be found in Clause 7.

6.1.2.5 MAppAssociation [0..n] – multicast application association

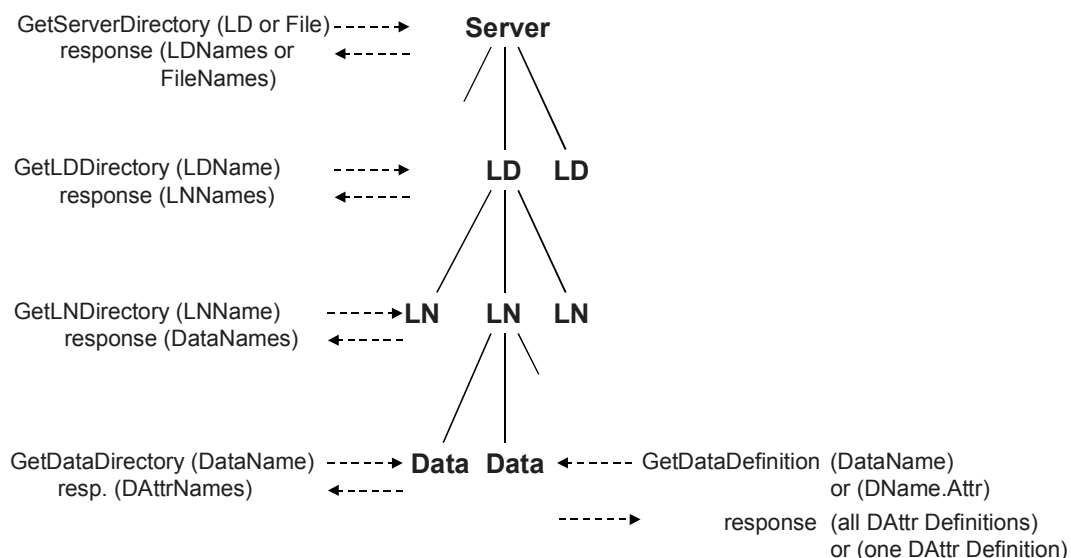
The attribute **MAppAssociation** shall identify a subscriber with which a **SERVER** (publisher) maintains a multicast application association.

NOTE Details can be found in Clause 7.

6.2 Server class services

6.2.1 Overview of directory and GetDefinition services

To support self-description of a device several GetXXDirectory and GetXXDefinition services as shown in Figure 5 are specified in this part of IEC 61850.



IEC 400/03

Figure 5 – Overview about GetDirectory and GetDefinition services

A client shall use these services to retrieve the definition of the complete hierarchy – as well as the definition of all accessible information – and of all instances of all underlying classes in a given server.

6.2.2 GetServerDirectory

6.2.2.1 GetServerDirectory parameter table

A client shall use the **GetServerDirectory** service to retrieve a list of the names of all **LOGICAL-DEVICES** or **Files** made visible and thus accessible to the requesting client by the addressed **SERVER**.

NOTE The visible instances are those that are defined within a given view (see Clause 7 for details on the view concept).

Parameter name
Request
ObjectClass
Response+
Reference [0..n]
Response–
ServiceError

6.2.2.2 Request

6.2.2.2.1 ObjectClass

The parameter **ObjectClass** shall contain the selected class. The client shall select one of the following classes:

- **LOGICAL-DEVICE**
- **FILE**

6.2.2.3 Response+

The parameter **Response+** shall indicate that the service request succeeded. A successful result shall return the following parameter.

6.2.2.3.1 Reference [0..n]

The parameter **Reference** shall contain the ObjectReference of the **LOGICAL-DEVICE** or the **FileName**.

NOTE The **FileName** type is VISIBLE STRING255.

6.2.2.4 Response–

The parameter **Response–** shall indicate that the service request failed. The appropriate ServiceError shall be returned.

7 Application association model

7.1 Introduction

The application association model consists of provisions on how the communication between the various types of devices is achieved. The model comprises

- class definitions of associations (two-party and multicast); and
- access control concepts (how to restrict access to instances in a server).

The security requirements for the restriction of access to the data in a server is defined in IEC 61850-5.

NOTE Security requirements are implemented by the SCSMs.

7.2 Concept of application associations

The application association model defines

- the services provided for managing associations between client and server (two-party application association); and
- the services provided for managing associations for multicast messaging (for example, GOOSE and transmission of sampled values).

The **two-party application association** class shall convey service requests and responses (thus transferring unconfirmed and confirmed services). The **multicast application association** class shall be capable of conveying unconfirmed services (in one direction only).

Application associations provide a mechanism for controlling the access to the instances of a device (access control).

NOTE The details of an application association model are defined in the SCSMs. The following descriptions provide a conceptual model of the application associations between devices.

7.3 Access control

The access control model provides the capability to restrict the access of a specific client to class instances, class instance attributes, and ACSI services acting upon class instances of a specific server. The ACSI server contains a set of, for example, **LOGICAL-DEVICES**, **LOGICAL-NODES**, **DATA**, or report controls. The set of instances visible (and therefore accessible) to a client is restricted on the basis of the identification of the client and the access control specification of the server. This restricted set is called a virtual access view. A virtual access view may not only restrict the visibility of instances or attributes but also the supported service. The concept of a virtual access view is illustrated in Figure 6.

NOTE 1 The virtual access view is the authentication's view of the IED's data model.

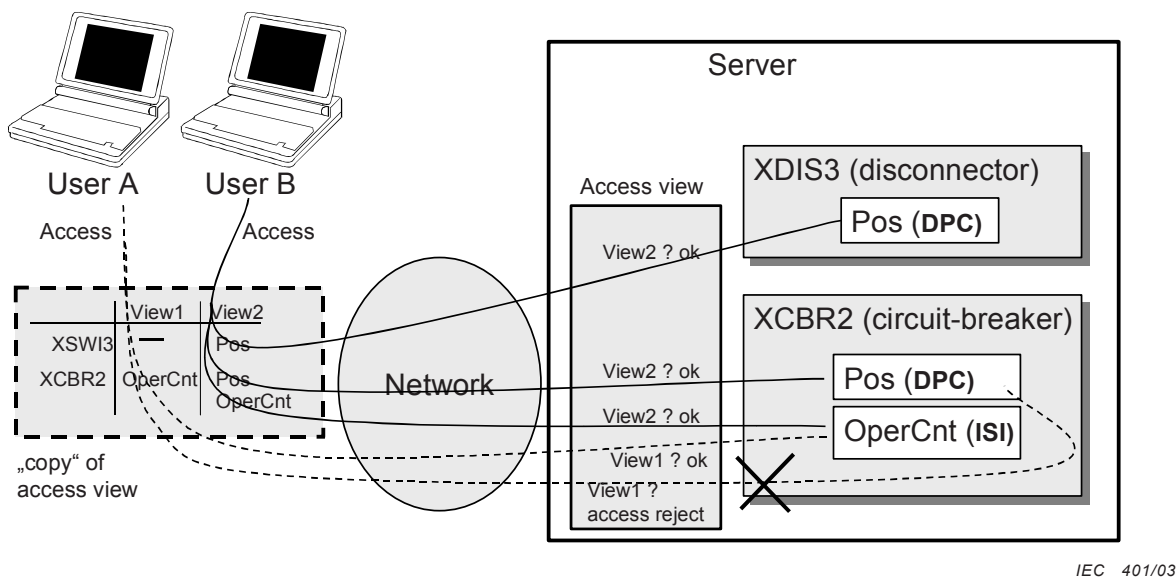


Figure 6 – Access views of a server

Two users (A and B) have different virtual access views (view1 and view2) of the server. View 1 allows just one **DATA** (**XCBR.OperCnt**) to be accessed remotely. View 2 allows all **DATA** to be accessed.

The intention of IEC 61850 is to implement the **virtual access view in the server** of a device, thus providing **access restriction to any user** who tries to access the instances. Independent of the implementation in the device, additional access restriction may be implemented at the user side, for example, local password or simply a key on the keyboard.

If a view hides a mandatory instance of an attribute of a **DATA** then this hidden attribute shall be implemented as required by the **DATA**.

NOTE 2 A view restricts the visibility to some users only.

A client (or a subscriber in the case of multicast application association) shall be identified by authentication parameters passed to the server when establishing the association with the server (two-party application association) or when sending information over multicast application associations.

NOTE 3 Mechanisms at the client side are outside the scope of this standard. A user may also use a "copy" of the access view to restrict the access at the client side.

NOTE 4 The details of access control including structure and content of authentication parameter are defined in the SCSMs.

7.4 TWO-PARTY-APPLICATION-ASSOCIATION (TPAA) class model

7.4.1 TWO-PARTY-APPLICATION-ASSOCIATION (TPAA) class definition

7.4.1.1 TWO-PARTY-APPLICATION-ASSOCIATION (TPAA) class syntax

A two-party application association type shall provide a bi-directional connection-oriented information exchange. The application associations shall be reliable and the information flow shall be controlled end to end. Reliable means that the connection on which the application association relies provides measures to notify reasons for non-deliverance of information in due time. End-to-end flow control means that sources of information do not send more information than the destination can buffer.

The services for associate, data exchange, and association release of the two-party application association class is depicted in Figure 7.

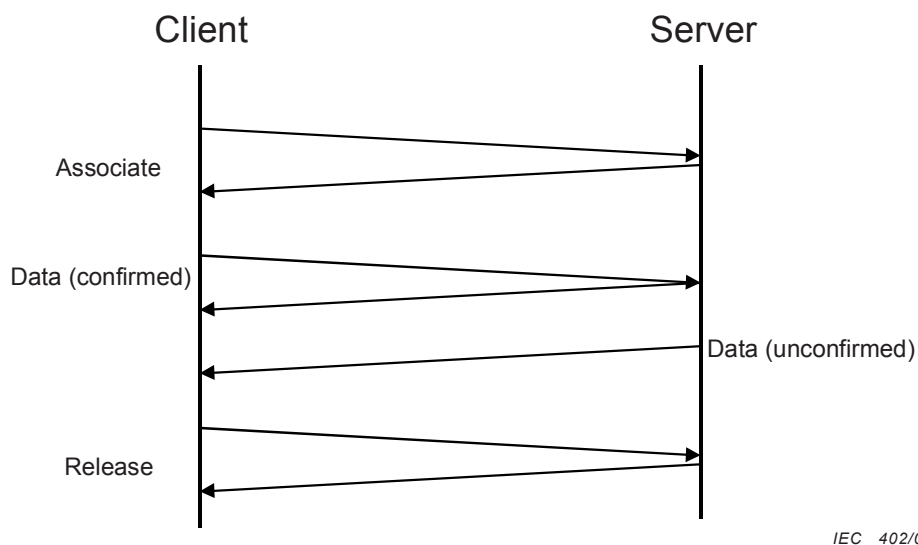


Figure 7 – Normal operation

The abort service for the two-party application association class is depicted in Figure 8.

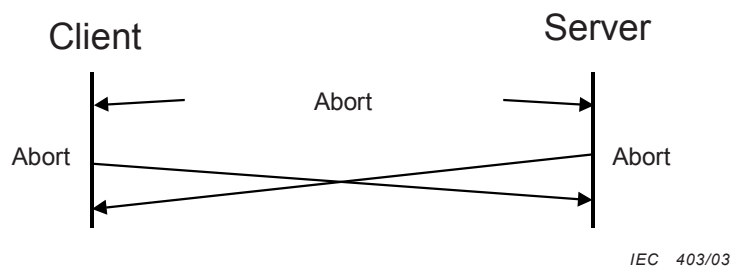


Figure 8 – Aborting association

The **TWO-PARTY-APPLICATION-ASSOCIATION** (TPAA) class shall be defined as in Table 12.

Table 12 – TWO-PARTY-APPLICATION-ASSOCIATION (TPAA) class definition

TWO-PARTY-APPLICATION-ASSOCIATION class		
Attribute name	Attribute type	Value/value range/explanation
AssociationId	(*)	(*) Type is SCSM specific
AuthenticationParameter	(*)	(*) Type is SCSM specific
Services Associate Abort Release Additional services that make use of the TWO-PARTY-APPLICATION-ASSOCIATION shall be as indicated in Table A.3 of Clause A.4 (in column Asso. marked as "TP")		

7.4.1.2 TWO-PARTY-APPLICATION-ASSOCIATION (TPAA) class attributes

7.4.1.2.1 AssociationId

The attribute **AssociationId** shall specify the identification used to identify the application associations.

NOTE The type of the **AssociationId** is defined in the SCSMs and it may be exchanged in an SCSM or be used locally only.

7.4.1.2.2 AuthenticationParameter

The attribute **authenticationParameter** shall represent the information required to grant permission to access instances of a specific access view to a server.

NOTE A minimum set of parameters is user identification, view and password. The details are defined in the SCSMs.

7.4.2 Two-party application association services

7.4.2.1 Overview

For **TWO-PARTY-APPLICATION-ASSOCIATION** the following services are defined.

Service	Description
Associate	Establish an association
Abort	Abort an association
Release	Release an association

7.4.2.2 Associate

7.4.2.2.1 Associate parameter

A client shall use the **Associate** service to establish an application association of type two-party with a specific server.

Parameter name
Request
ServerAccessPointReference
AuthenticationParameter
Response+
AssociationId
Result
Response-
ServiceError

7.4.2.2.2 Request

7.4.2.2.2.1 ServerAccessPointReference

The parameter **ServeAccessPointReference** shall identify the server, with which the application association shall be established.

7.4.2.2.2 AuthenticationParameter

This parameter **AuthenticationParameter** shall specify the **authenticationParameter** for this application association to be opened. If an **authenticationParameter** does not match with a valid parameter, the service request shall be rejected and an appropriate reason shall be returned.

NOTE The type of the **authenticationParameter** is defined in the SCSM.

7.4.2.2.3 Response+

AssociationId

The parameter **AssociationId** may be used to differentiate the application associations.

NOTE The **AssociationId** may be exchanged in a response+ message of an SCSM or be used locally only.

7.4.2.2.4 Result

The parameter **Result** shall indicate, if the establishment of the application association was successful or not.

7.4.2.2.5 Response–

The parameter **Response–** shall indicate that the service request failed. The appropriate **ServiceError** shall be returned.

7.4.2.3 Abort

7.4.2.3.1 Abort parameter

The service **Abort** shall be used to abruptly disconnect a specific application association between a client and a server. Abrupt means that all service requests issued shall be discarded – no further service shall be processed.

Parameter name
Request
AssociationId
Reason
Indication
AssociationId
Reason

7.4.2.3.2 Request

7.4.2.3.2.1 AssociationId

The parameter **AssociationId** shall specify the association to be aborted. The indication may be issued by the underlying layer (locally or remotely) or it may be sent from remote user of the association.

7.4.2.3.2.2 Reason

The parameter **Reason** shall specify the reason why the association has been aborted. The reason may be provided by the underlying layer (locally or remotely) or it may be sent from remote user of the association.

7.4.2.3.3 Indication

7.4.2.3.3.1 AssociationId

The parameter **AssociationId** shall specify the association that has been aborted.

7.4.2.3.3.2 Reason

The parameter **Reason** shall specify the reason for abrupt termination the application association.

7.4.2.4 Release

7.4.2.4.1 Release parameter

The service **Release** shall be used to gracefully disconnect a specific application association between a client and a server. Graceful means that all service requests issued shall be completed before termination. New request shall not be issued after disconnect initiation.

Parameter name
Request
AssociationId
Response+
AssociationId
Result
Response-
ServiceError

7.4.2.4.2 Request

7.4.2.4.3 AssociationId

The parameter **AssociationId** shall specify the association to be terminated.

7.4.2.4.4 Response+

7.4.2.4.5 Result

The parameter **Result** shall indicate, if the termination of the application association was successful or not.

7.4.2.4.6 Response-

The parameter **Response-** shall indicate that the service request failed. The appropriate ServiceError shall be returned.

7.5 MULTICAST-APPLICATION-ASSOCIATION (MCAA) class

7.5.1 MULTICAST-APPLICATION-ASSOCIATION (MCAA) class definition

7.5.1.1 MULTICAST-APPLICATION-ASSOCIATION (MCAA) class syntax

A multicast application association type shall provide a unidirectional information exchange. Multicast information exchange shall be provided between one source (publisher) and one or many destinations (subscriber). Unidirectional information exchange shall provide sufficient information for the receivers to uniquely interpret the context in which the exchange shall be processed.

The subscriber shall be capable to detect loss and duplication of information received. The receiver shall notify the loss of information to its user and shall discard duplicated information.

NOTE The possible restriction of multicast messages to be exchanged on a single subnet or sent through routers is an issue to be defined in an SCSM.

The multicast application association class is depicted in Figure 9.

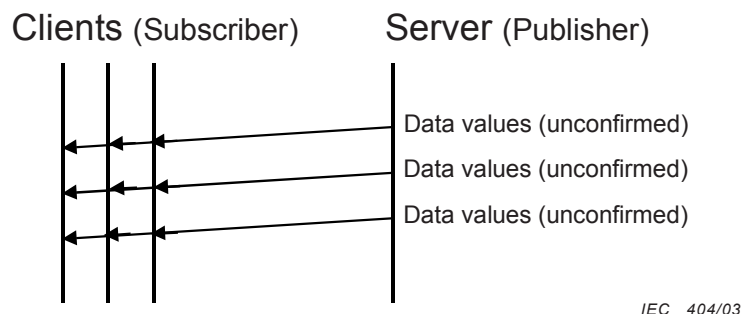


Figure 9 – Principle of multicast application association

The **MULTICAST-APPLICATION-ASSOCIATION (MCAA)** shall be as defined in Table 13.

Table 13 – MULTICAST-APPLICATION-ASSOCIATION (MCAA) class definition

MULTICAST-APPLICATION-ASSOCIATION class		
Attribute name	Attribute type	Value/value range/explanation
AuthenticationParameter	(*)	(*) Type is SCSM specific
Services Services that make use of the MULTICAST-APPLICATION-ASSOCIATION shall be as indicated in Table A.3 of Clause A.4 (in column Asso. marked as "MC")		

7.5.1.2 MULTICAST-APPLICATION-ASSOCIATION (MCAA) class attributes

7.5.1.2.1 AuthenticationParameter

The **authenticationParameter** shall represent the information required to grant permission to access instances of a specific access view to a client.

Each multicast service shall provide a service parameter that specifies the **authenticationParameter** for this data exchange. If an **authenticationParameter** does not match with a valid parameter, the service request shall be rejected by the receiving device.

NOTE 1 The type of the **authenticationParameter** is defined in the SCSM.

NOTE 2 Each exchange of information using multicast services can be understood as an "associate message" that carries association parameters and data. The "application association" ceases as soon as the service has been processed.

8 LOGICAL-DEVICE class model

8.1 LOGICAL-DEVICE class definition

8.1.1 LOGICAL-DEVICE class syntax

The **LOGICAL-DEVICE (LD)** shall be a composition of **LOGICAL-NODE** as defined in Table 14.

NOTE A **LOGICAL-DEVICE** can be used simply as a container of a group of **LOGICAL-NODEs** or as a device that functions as a gateway or proxy. Details on the use of **LOGICAL-DEVICE** can be found in IEC 61850-7-1.

Table 14 – LOGICAL-DEVICE (LD) class definition

LOGICAL-DEVICE class		
Attribute name	Attribute type	Value/value range/explanation
LDName	ObjectName	Instance name of an instance of LOGICAL-DEVICE
LDRef	ObjectReference	Path-name of an instance of LOGICAL-DEVICE
LogicalNode [3..n]	LOGICAL-NODE	IEC 61850-7-4 specifies specialized classes of LOGICAL-NODE
Services GetLogicalDeviceDirectory		

8.1.2 LOGICAL-DEVICE class attributes

8.1.2.1 LDName – logical device name

The attribute **LDName** shall unambiguously identify a **LOGICAL-DEVICE** within the scope of a system.

8.1.2.2 LDRef – logical device ObjectReference

The attribute **LDRef** shall be the unique path-name of a **LOGICAL-DEVICE**:



NOTE The **LOGICAL-DEVICE** is the root of tree. Therefore the **LDName** and **LDRef** are identical. For conceptual reasons they are both included in the table.

8.1.2.3 LogicalNode [3..n]

The attribute **LogicalNode** shall identify a **LOGICAL-NODE** that is contained in a **LOGICAL-DEVICE**.

Each **LOGICAL-DEVICE** shall have one and only one **LOGICAL-NODE-ZERO (LLNO)**, one and only one **LOGICAL-NODE-PHYSICAL-DEVICE (LPHD)**, and at least one other **LOGICAL-NODE**.

NOTE The substation-automation-specific **LLNO**, **LPHD**, and other logical nodes are defined in IEC 61850-7-4.

8.2 LOGICAL-DEVICE class services

8.2.1 GetLogicalDeviceDirectory

8.2.1.1 GetLogicalDeviceDirectory parameter table

A client shall use the **GetLogicalDeviceDirectory** service to retrieve the list of the ObjectReferences of all **LOGICAL-NODEs** made visible and thus accessible to the requesting client by the referenced **LOGICAL-DEVICE**.

NOTE The visible instances are those that are defined within a given view (see Clause 7 for details on the view concept).

Parameter Name
Request
LDReference
Response+
LNReference [3..n]
Response–
ServiceError

8.2.1.2 Request

8.2.1.2.1 LDReference – logical device ObjectReference

The parameter **LDReference** shall contain the ObjectReference **LDRef** of a **LOGICAL-DEVICE**.

8.2.1.3 Response+

The parameter **Response+** shall indicate that the service request succeeded. A successful result shall return the following parameter.

8.2.1.3.1 LNReference [3..n] – logical node ObjectReference

The parameter **LNReference** shall contain the ObjectReference **LNRef** of a **LOGICAL-NODE** from the referenced **LOGICAL-DEVICE**.

8.2.1.4 Response–

The parameter **Response–** shall indicate that the service request failed. The appropriate ServiceError shall be returned.

9 LOGICAL-NODE class model

9.1 LOGICAL-NODE class definition

9.1.1 LOGICAL-NODE class syntax

The **LOGICAL-NODE** shall be a composition of **DATA**, **DATA-SET**, **BRCB**, **URCB**, **LCB**, **LOG**, **SGCB**, **GoCB**, **GsCB**, **MSVCB**, and **USVCB** as defined in Table 15.

Table 15 – LOGICAL-NODE (LN) class definition

LOGICAL-NODE class		
Attribute name	Attribute type	Explanation
LNName	ObjectName	Instance name of an instance of LOGICAL-NODE
LNRef	ObjectReference	Path-name of an instance of LOGICAL-NODE
Data [1..n]	DATA	
DataSet [0..n]	DATA-SET	
BufferedReportControlBlock [0..n]	BRCB	
UnbufferedReportControlBlock [0..n]	URCB	
LogControlBlock [0..n]	LCB	
IF compatible LN class defined in IEC 61850-7-4 equals LLNO		
SettingGroupControlBlock [0..1]	SGCB	
Log [0..1]	LOG	
GOOSEControlBlock [0..n]	GoCB	
GSSEControlBlock [0..n]	GsCB	
MulticastSampledValueControlBlock [0..n]	MSVCB	
UnicastSampledValueControlBlock [0..n]	USVCB	
Services GetLogicalNodeDirectory GetAllDataValues		
NOTE 1 IEC 61850-7-4 defines specialized logical node classes – the compatible logical node classes, for example, XCBR representing circuit-breakers.		

The definition of **LOGICAL-NODEs** for the substation-application domain is refined by the definition of specific **DATA** in IEC 61850-7-4. The definitions in IEC 61850-7-4 (and IEC 61850-7-3 for the common **DATA** classes) shall be taken into account to get the comprehensive definition of substation-domain-specific **LOGICAL-NODEs**.

NOTE 2 IEC 61850-7-4 defines further attributes for **LOGICAL-NODEs**; for example,, the mode (behaviour: ON, BLOCKED, TEST, etc.) of the substation-specific **LOGICAL-NODE** is defined in IEC 61850-7-4. The state model of a **LOGICAL-NODE** is modelled as a specific **DATA** (named **Mod**).

9.1.2 LOGICAL-NODE class attributes

9.1.2.1 LNName – Logical node name

The attribute **LNName** shall unambiguously identify **LOGICAL-NODE** within the scope of **LOGICAL-DEVICE**.

9.1.2.2 LNRef – Logical node ObjectReference

The attribute **LNRef** shall be the unique path-name of a **LOGICAL-NODE**.

The ObjectReference LNRef shall be:

LDName/LNName

9.1.2.3 Data [1..n]

The attribute **Data** shall identify **DATA** (see Clause 10) that is contained in the **LOGICAL-NODE**.

NOTE IEC 61850-7-4 defines standardized **DATA** called compatible **DATA** classes.

9.1.2.4 DataSet [0..n]

The attribute **DataSet** shall identify a **DATA-SET** (see Clause 11) that is contained in the **LOGICAL-NODE**.

9.1.2.5 BufferedReportControlBlock [0..n]

The attribute **BufferedReportControlBlock** shall identify a **BRCB** (see 14.2) that is contained in the **LOGICAL-NODE**.

9.1.2.6 UnbufferedReportControlBlock [0..n]

The attribute **UnbufferedReportControlBlock** shall identify an **URCB** (see 14.2) that is contained in the **LOGICAL-NODE**.

9.1.2.7 LogControlBlock [0..n]

The attribute **LogControlBlock** shall identify a **LCB** (see 14.3) that is contained in the **LOGICAL-NODE**.

9.1.2.8 SettingGroupControlBlock [0..1]

The attribute **SettingGroupControl** shall identify the **SGCB** (see Clause 13) that is contained in a **LLNO**.

9.1.2.9 Log [0..1]

The attribute **Log** shall identify the **LOG** (see 14.3.3) that is contained in the **LLNO**.

9.1.2.10 GOOSEControlBlock [0..n]

The attribute **GOOSEControlBlock** shall identify a **GoCB** (see 15.2) that is contained in the **LLNO**.

9.1.2.11 GSSEControlBlock [0..n]

The attribute **GSSEControl** shall identify the **GsCB** (see 15.3) that is contained in the **LLNO**.

9.1.2.12 MulticastSampledValueControlBlock [0..n]

The attribute **MulticastSampledValueControlBlock** shall identify a **MSVCB** (see 16.2) that is contained the **LLNO**.

9.1.2.13 UnicastSampledValueControlBlock [0..n]

The attribute **UnicastSampledValueControlBlock** shall identify a **USVCB** (see 16.3) that is contained in the **LLNO**.

9.2 LOGICAL-NODE class services

9.2.1 Overview

For **LOGICAL-NODE** the following services are defined:

Service	Description
GetLogicalNodeDirectory	Retrieve ObjectReferences of a specific ACSI class contained in the LOGICAL-NODE
GetAllDataValues	Retrieve all DataAttribute values of all DATA contained in the LOGICAL-NODE

9.2.2 GetLogicalNodeDirectory

9.2.2.1 GetLogicalNodeDirectory parameter table

A client shall use the **GetLogicalNodeDirectory** service to retrieve a list of the ObjectReferences of all instances of a requested class made visible and thus accessible to the requesting client by the referenced **LOGICAL-NODE**.

NOTE The visible instances are those that are defined within a given view (see Clause 7 for details on the view concept).

Parameter name
Request
LNReference
ACSIClass
Response+
InstanceName [0..n]
Response–
ServiceError

9.2.2.2 Request

9.2.2.2.1 LNReference

The parameter **LNReference** shall contain the ObjectReference **LNRef** of the **LOGICAL-NODE**.

9.2.2.2.2 ACSIClass

The parameter **ACSIClass** shall contain the selected ACSI class model for which the ObjectReferences of all ACSI class models shall be returned.

The client shall select one of the following ACSI class models:

9.2.2.3 Response+

The parameter **Response+** shall indicate that the service request succeeded. A successful result shall return the following parameter.

InstanceName [0..n]

The parameter **InstanceName** shall contain an ObjectName of one requested ACSI class model. In the case where the referenced **LOGICAL-NODE** does not contain the requested ACSI class, the server shall indicate that no ACSI class model exists in this **LOGICAL-NODE**.

9.2.2.4 Response–

The parameter **Response–** shall indicate that the service request failed. The appropriate ServiceError shall be returned.

9.2.3 GetAllDataValues

9.2.3.1 GetAllDataValues parameter table

A client shall use the **GetAllDataValues** service to retrieve all **DataAttribute** values (having the same **FunctionalConstraint**) of all **DATA** made visible and thus accessible to the requesting client by the referenced **LOGICAL-NODE**.

NOTE The visible instances are those that are defined within a given view (see Clause 7 for details on the view concept).

Parameter name
Request
LNReference
FunctionalConstraint [0..1]
Response+
LNReference
DataAttributeReference [1..n]
DataAttributeValue [1..n]
Response–
ServiceError

9.2.3.2 Request

9.2.3.2.1 LNReference

The parameter **LNReference** shall contain the ObjectReference **LNRef** of the **LOGICAL-NODE**.

9.2.3.2.2 FunctionalConstraint [0..1]

The parameter **FunctionalConstraint** shall contain the functional constraint parameter (**FC**) to filter the respective **DataAttributes** of all **DATA** contained in the **LOGICAL-NODE**. The **FC** shall be as defined in 10.2.2.4.2.

9.2.3.3 Response+

The parameter **Response+** shall indicate that the service request succeeded. A successful result shall return the following parameters.

9.2.3.3.1 DataAttributeReference [1..n]

The parameter **DataAttributeReference** shall contain the ObjectReference of a **DataAttribute** contained in the **LOGICAL-NODE** that shall be returned according to the value of the **FunctionalConstraint** received in the request.

NOTE The ObjectReference **DataAttributeReference** is defined in 10.2.2.4.

9.2.3.3.2 DataAttributeValue [1..n]

The parameter **DataAttributeValue** shall contain the value of a **DataAttribute** of the **DATA** contained in the referenced **LOGICAL-NODE**. Only values of those **DataAttributes** that have the functional constrained equal to the value of the parameter **FunctionalConstraint** in the service request shall be returned.

9.2.3.4 Response–

The parameter **Response–** shall indicate that the service request failed. The appropriate **ServiceError** shall be returned.

10 DATA class model

10.1 General

DATA classes represent meaningful information of applications located in an automation devices. The values of **DATA** instances can, for example, be written (**SetDataValues**) and read (**GetDataValues**). IEC 61850-7-4 specifies a list of common and substation-domain-specific – simple and complex – **DATA**, for example, **Pos** for position, **OilFil** for oil filtration. The composition of **DATA** in IEC 61850-7-4 is based on common templates (the common **DATA** classes, **CDC**) specified in IEC 61850-7-3. The concept of **DATA** classes is introduced in this clause. Any set of **DATA** (or parts of **DATA**) instances may be grouped to build **DATA-SET** instances applying the **CreateDataSet** service. **DATA-SET** instances can, for example, be written (**SetDataSetValues**) or read (**GetDataSetValues**)

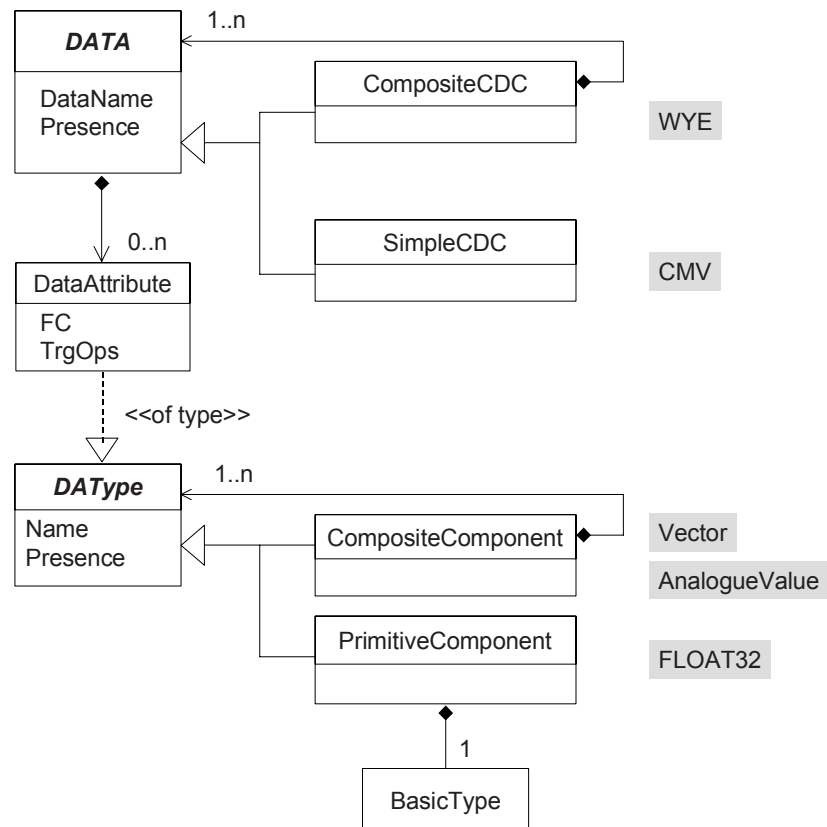
NOTE 1 The consequences of setting values to instances of **DATA** is outside this part of IEC 61850. IEC 61850-7-3 and IEC 61850-7-4 specify many substation-domain-specific **DATA** classes. These definitions provide information on the actions to be taken by the receiving application, for example, changing the **DATA Mode** from ON to TEST changes the state of the respective instance to test mode behaviour as defined in IEC 61850-7-4.

NOTE 2 The client queries values of **DATA** (**DATA-SET**) from a server using the service **GetDataValues** (**GetDataSetValues**). Services for unsolicited/spontaneous transmission of values of **DATA** from a server to clients (sometimes known as information report, traps, or spontaneous transmission) require a careful design. Uncontrolled spontaneous transmission may congest the network. Services for a controlled reporting are specified in Clause 14.

10.2 DATA class definition

10.2.1 DATA class syntax

The **DATA** class is a key element in IEC 61850. The class diagram in Figure 10 is intended as an introduction to the formal **DATA** class specification.



IEC 405/03

Figure 10 – Class diagram of **DATA** and **DataAttributeType**

NOTE 1 The example in Figure 10 uses definitions (for example, **WYE**, **CMV**, **Vector**, and **AnalogueValue** common **DATA** classes) found in IEC 61850-7-3. A comprehensive introduction to the modelling of **DATA** can be found in IEC 61850-7-1.

The **DATA** is a class that has a **DataName**, an indication (**Presence**) if the **DATA** is mandatory (present) or optional (not-present), and **DataAttributes**.

NOTE 2 The **DATA** class is an abstract class that is an auxiliary means to construct the primitive and composite common data classes.

NOTE 3 The following examples used in the text refer to Figure 11.

The **DataAttributes** (for example, **cVal** – complex value) are used to build a **SimpleCDC** (simple common data class) and **CompositeCDC** (composite common data class). **SimpleCDC** shall have a **DataName**, a **Presence**, and **DataAttributes** (for example, **DataName** = **phsA**, **Presence** = **Mandatory**, and **DataAttribute** = **cVal**). The **CompositeCDC** is constructed by one or more **SimpleCDC** and/or **DataAttributes** (for example, **CDC WYE** comprising a **SimpleCDC CMV**, etc.).

The **DAType** has already been explained in 5.5.1.

Figure 11 depicts an excerpt of a **DATA** instance (contained in a **LOGICAL-NODE** **MMXU1**). The instance of the **LOGICAL-NODE** with the name **MMXU1** (instantiated from **MMXU**) is composed of the instance of the **DATA** phase voltage named **PhV** (instantiated from **WYE**), which is composed of phase A voltage **phsA** (instantiated from **CMV**), which is composed of complex value **cVal** (of type **Vector**), which is composed of voltage **mag** (of type **AnalogueValue**), which is composed of floating-point value **f** (of type **FLOAT32**). The **DataAttribute** has additionally the functional constraint **FC** = **MX** (measurand) and the trigger option **TrgOp** = **dchg** (data-change).



Figure 11 – Example of DATA

NOTE 4 The explanation of the **DATA** class refers to the example shown in Figure 11. The example uses some definitions from IEC 61850-7-3 just to demonstrate the formal definition of the **DATA** class. A complete definition of the compatible classes is defined in IEC 61850-7-3.

The references for the various levels are listed at the bottom of the figure.

The **DATA** shall have the structure defined in Table 16.

The inheritance and relations between the classes **DATA**, **CompositeCDC**, **SimpleCDC**, and **DAType** shall be as shown in Figure 10.

The table notation does not easily show the inheritance. Therefore the class diagram in Figure 10 shall be normative. The tables and the class diagrams shall be used together.

Table 16 – DATA class definition

DATA class		
Attribute name	Attribute type	Value/value range/explanation
DataName	ObjectName	Instance name of an instance of DATA, for example, PhV (1st level), phsA (2nd level)
DataRef	ObjectReference	Path-name of an instance of DATA, for example, MMXU1.PhV or for example, MMXU1.PhV.PhsA
Presence	BOOLEAN	Indicates mandatory/optional
DataAttribute [0..n] DataAttributeType FunctionalConstraint TrgOp [0..n]	DAType FC TriggerConditions	For example, Vector class of IEC 61850-7-3 for example, MX for example, dchg
Specializations of DATA		
CompositeCDC [0..n]	DATA	For example, WYE class of IEC 61850-7-3
SimpleCDC [0..n]	COMMON-DATA	For example, CMV class of IEC 61850-7-3
Services GetDataValues SetDataValues GetDataDirectory GetDataDefinition		

An instance of a **DATA** class may contain zero or more instances of a **CompositeCDC**, **SimpleCDC** or a **DataAttribute**. However, they cannot all be absent, so at least one of these elements shall be present.

NOTE 5 The structure of a **DATA class** is recursive since a **CompositeCDC** is also of type **DATA class**. The level of recursion may be restricted by a SCSM, so the number of levels of recursion of **CompositeCDCs** is normally no greater than 1.

NOTE 6 **DATA** or part of a **DATA** may be referenced in a **DATA-SET**. The persistent existence of **DATA** is expected as long as they are referenced as members of a **DATA-SET**. A system has to take special measures to ensure their existence.

10.2.2 DATA class attributes

10.2.2.1 DataName

The attribute **DataName** shall unambiguously identify a **DATA** within the scope of a **LOGICAL-NODE**.

10.2.2.2 DataRef – data ObjectReference

The attribute **DataRef** shall be the unique path-name of a **DATA**.

The ObjectReference **DataRef** shall be:

LDName/LNName.DataName[.DataName[. ...]]

NOTE Nesting depends on the concrete definition of a **DATA** class.

10.2.2.3 Presence

The attribute **Presence** of type **BOOLEAN** shall specify if a **DATA** within a **compositeCDC** or a **LOGICAL-NODE** is mandatory (**Presence = TRUE**) or optional (**Presence = FALSE**).

10.2.2.4 DataAttribute
10.2.2.4.1 DataAttributeType
10.2.2.4.1.1 General

The attribute **DataAttributeType** of type **DAType** shall specify a data attribute.

10.2.2.4.1.2 DAType syntax

The **DAType** shall be as defined in Table 17.

Table 17 – DAType definition

DAType		
Attribute name	Attribute type	Value/value range/explanation
DATName	ObjectName	Instance name of an instance of DAType , for example, cVal (1st level), mag (2nd level), f (3rd level)
DATRef	ObjectReference	Path-name of an instance of DAType for example, MMXU1.PhV.phsA.cVal for example, MMXU1.PhV.phsA.cVal.mag or for example, MMXU1.PhV.phsA.cVal.mag.f
Presence	BOOLEAN	Indicates mandatory/optional
Specializations of DAType		
CompositeComponent [0..n]	DAType	For example, mag in Vector class of IEC 61850-7-3 for example, f in AnalogueValue of IEC 61850-7-3
PrimitiveComponent [0..1]	BasicType	For example, FLOAT32 class of IEC 61850-7-3 for f
NOTE 1 An instance of a DAType may contain 0 or more instances of a CompositeComponent or a PrimitveDAT. However, they cannot both be absent, so at least one of these elements must be present. NOTE 2 The structure of a DAType is recursive since a CompositeComponent is also of type DAType. The level of recursion may be restricted by a SCSM, so the number of levels of recursion of CompositeComponents is normally no greater than 2.		

DATName – data attribute type name

The attribute **DATName** shall unambiguously identify a **DAType** within the scope of a **DataAttribute** or a nested **DataAttribute**.

The **DATName** (if **DataAttribute** is not nested) or the **DATName** of the first level (if Data Attribute is nested) shall be called the **DataAttributeName**.

For the second and any deeper nesting levels the **DATName** shall be called **DAComponentName**.

The **ObjectReference** from the top (**LD**) down to the **DataAttributeName** shall be called **DataAttributeReference**.

EXAMPLE As shown in Figure 11, the **cVal** (derived from a common data attribute type – **Vector**) is the DataAttribute. The **mag** (also derived from a common data attribute type – **AnalogueValue**) is a DataAttributeComponent.

DATRef – data attribute type ObjectReference

The attribute **DATRef** shall be the unique path-name of a **DAType**.

The ObjectReference **DATRef** shall be:

LDName/LNName.
DataName[.DataName[. ...]].DataAttributeName[.DAComponentName[. ...]]

The ObjectReference **DataAttributeReference** shall be:

LDName/LNName.DataName[.DataName[. ...]].DataAttributeName

NOTE 3 Nesting depends on the concrete definition of a **DATA** class and **DAType** class.

NOTE 4 In each path within a **DATA** there is one and only one **DataAttribute** (level).

Presence

The attribute **Presence** of type **BOOLEAN** shall specify if a **DataAttribute** is mandatory (**Presence** = **TRUE**) or optional (**Presence** = **FALSE**).

CompositeComponent [0..n] – composite component

The attribute **CompositeComponent** shall be a specialization of **DAType**.

PrimitiveComponent [0..n] – primitive component

The attribute **PrimitiveComponent** shall be a specialization of **DAType**.

10.2.2.4.2 FC [0..1] – functional constraint

From an application point of view, the **DataAttributes** are classified according to their specific use; for example, some attributes are used for **controlling** purposes, other attributes are used for **reporting** and **logging**, **configuration**, others indicate **measurements** or **setting groups**, or some identify the **description** of a specific **DataAttribute**.

The **functional constraint** (**FC**) shall be a property of the **DataAttribute** characterizing the specific use of the **DataAttribute**. The **functional constraint** (**FC**) is used in the definition of **DATA** (contained in **LOGICAL-NODES**) and in the various control blocks (for example, **BRCB**). Most attributes of control blocks have a **functional constraint** (**FC**) property.

NOTE The functional constraint could be understood as a filter of the **DataAttributes**. The common data classes in IEC 61850-7-3 use the functional constraint values defined in this subclause.

The **functional constraint** is used in various definitions in this part of IEC 61850. The **functional constraint** (**FC**) shall indicate the services that are allowed to be operated on a specific **DataAttribute**. The **functional constraints** shall be as specified in Table 18.

Table 18 – Functional constraints

Functional constraint (FC)					
	Semantic	Services allowed	Initial values/storage/explanation	D ^a	CB ^b
ST	Status information	DataAttribute shall represent a status information whose value may be read, substituted, reported, and logged but shall not be written	Initial value of the DataAttribute shall be taken from the process	X	
MX	Measurands (analogue values)	DataAttribute shall represent a measurand information whose value may be read, substituted, reported, and logged but shall not be written	Initial value of the DataAttribute shall be taken from the process	X	
CO	Control	DataAttribute shall represent a control information whose value may be operated (control model) and read	N.a.	X	
SP	Setpoint	DataAttribute shall represent a set-point information whose value may be controlled (control model) and read. Values controlled shall become effective immediately	Initial value of the DataAttribute shall be as configured; value shall be non-volatile	X	X
SV	Substitution	DataAttribute shall represent a substitution information whose value may be written to substitute the value attribute and read	If the value of the DataAttribute is volatile then the initial value shall be FALSE, else the value should be as set or configured	X	
CF	Configuration	DataAttribute shall represent a configuration information whose value may be written and read. Values written may become effective immediately or deferred by reasons outside the scope of this standard	Initial value of the DataAttribute shall be as configured; value shall be non-volatile	X	
DC	Description	DataAttribute shall represent a description information whose value may be written and read	Initial value of the DataAttribute shall be as configured; value shall be non-volatile	X	
SG	Setting group	Logical devices that implement the SGCB class maintain multiple grouped values of all instances of DataAttributes with functional constraint SG. Each group contains one value for each DataAttribute with functional constraint SG which shall be the current active value (for details see 13). Values the of DataAttributes with FC=SG shall not be writeable	Initial value of the DataAttribute shall be as configured; value shall be non-volatile	X	
SE	Setting group editable	DataAttribute which can be edited by SGCB services	Value of the DataAttribute shall be as available after SelectEditSG service has been processed	X	
EX	Extended definition	DataAttribute shall represent an extension information providing a reference to a name space. Extensions are used in conjunction with extended definitions of LN s, DATA , and DataAttributes in 61850-7-3 and IEC 61850-7-4. Values the of DataAttributes with FC=EX shall not be writeable	Value of the DataAttribute shall be as configured; value shall be non-volatile	X	
BR	Buffered report ^c	Attribute shall represent a report control information of a BRCB whose value may be written and read	Initial value of the Attribute shall be as configured; value shall be non-volatile		X
RP	Unbuffered report ^c	Attribute shall represent a report control information of a URCB whose value may be written and read	Initial value of the Attribute shall be as configured; value shall be non-volatile		X