IEC 61158-5-4

Edition 2.0 2014-08

INTERNATIONAL STANDARD

NORME INTERNATIONALE

colour inside

Industrial communication networks – Fieldbus specifications – Part 5-4: Application layer service definition – Type 4 elements

Réseaux de communication industriels – Spécifications des bus de terrain – Partie 5-4: Définition des services de la couche application – Eléments de type 4

IEC 61158-5-4:2014-08(en-fr)

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00
info@iec.ch
www.iec.ch

About the IEC
The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications
The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

IEC Catalogue - webstore.iec.ch/catalogue
The stand-alone application for consulting the entire bibliographical information on IEC International Standards, Technical Specifications, Technical Reports and other documents. Available for PC, Mac OS, Android Tablets and iPad.

IEC publications search - www.iec.ch/searchpub
The advanced search enables to find IEC publications by a variety of criteria (reference number, text, technical committee,…). It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished
Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and also once a month by email.

Electropedia - www.electropedia.org
The world's leading online dictionary of electronic and electrical terms containing more than 30 000 terms and definitions in English and French, with equivalent terms in 14 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC Glossary - std.iec.ch/glossary
More than 55 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

IEC Customer Service Centre - webstore.iec.ch/csc
If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: csc@iec.ch.

A propos de l'IEC
La Commission Electrotechnique Internationale (IEC) est la première organisation mondiale qui élabore et publie des Normes internationales pour tout ce qui a trait à l'électricité, à l'électronique et aux technologies apparentées.

A propos des publications IEC
Le contenu technique des publications IEC est constamment revu. Veuillez vous assurer que vous possédez l'édition la plus récente, un corrigendum ou amendement peut avoir été publié.

Catalogue IEC - webstore.iec.ch/catalogue
Application autonome pour consulter tous les renseignements bibliographiques sur les Normes internationales, Spécifications techniques, Rapports techniques et autres documents de l'IEC. Disponible pour PC, Mac OS, tablettes Android et iPad.

Recherche de publications IEC - www.iec.ch/searchpub
La recherche avancée permet de trouver des publications IEC en utilisant différents critères (numéro de référence, texte, comité d'études,…). Elle donne aussi des informations sur les projets et les publications remplacées ou retirées.

IEC Just Published - webstore.iec.ch/justpublished
Restez informé sur les nouvelles publications IEC. Just Published détaille les nouvelles publications parues. Disponible en ligne et aussi une fois par mois par email.

Electropedia - www.electropedia.org
Le premier dictionnaire en ligne de termes électroniques et électriques. Il contient plus de 30 000 termes et définitions en anglais et en français, ainsi que les termes équivalents dans 14 langues additionnelles. Egalement appelé Vocabulaire Electrotechnique International (IEV) en ligne.

Glossaire IEC - std.iec.ch/glossary
Plus de 55 000 entrées terminologiques électrotechniques, en anglais et en français, extraites des articles Termes et Définitions des publications IEC parues depuis 2002. Plus certaines entrées antérieures extraites des publications des CE 37, 77, 86 et CISPR de l'IEC.

Service Clients - webstore.iec.ch/csc
Si vous désirez nous donner des commentaires sur cette publication ou si vous avez des questions contactez-nous: csc@iec.ch.

IEC 61158-5-4

Edition 2.0 2014-08

INTERNATIONAL STANDARD

NORME INTERNATIONALE

colour inside

Industrial communication networks – Fieldbus specifications – Part 5-4: Application layer service definition – Type 4 elements

Réseaux de communication industriels – Spécifications des bus de terrain – Partie 5-4: Définition des services de la couche application – Eléments de type 4

INTERNATIONAL ELECTROTECHNICAL COMMISSION

COMMISSION ELECTROTECHNIQUE INTERNATIONALE

PRICE CODE CODE PRIX XB

ICS 25.040.40; 35.100.70; 35.110

ISBN 978-2-8322-1733-7

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 5-4: Application layer service definition – Type 4 elements

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
6) All users should ensure that they have the latest edition of this publication.
7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

Attention is drawn to the fact that the use of the associated protocol type is restricted by its intellectual-property-right holders. In all cases, the commitment to limited release of intellectual-property-rights made by the holders of those rights permits a layer protocol type to be used with other layer protocols of the same type, or in other type combinations explicitly authorized by its intellectual-property-right holders.

NOTE Combinations of protocol types are specified in IEC 61784-1 and IEC 61784-2.

International Standard IEC 61158-5-4 has been prepared by subcommittee 65C: Industrial networks, of IEC technical committee 65: Industrial-process measurement, control and automation.

This second edition cancels and replaces the first edition published in 2007. This edition constitutes an editorial revision with only minor editorial changes.

This edition includes the following significant changes with respect to the previous edition:

a) editorial improvements;

b) editorial corrections.

The text of this standard is based on the following documents:

FDIS	Report on voting
65C/763/FDIS	65C/773/RVD

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with ISO/IEC Directives, Part 2.

A list of all the parts of the IEC 61158 series, under the general title *Industrial communication networks – Fieldbus specifications*, can be found on the IEC web site.

The committee has decided that the contents of this publication will remain unchanged until the stability date indicated on the IEC web site under http://webstore.iec.ch in the data related to the specific publication . At this date, the publication will be

- reconfirmed;
- withdrawn;
- replaced by a revised edition, or
- amended.

IMPORTANT – The 'colour inside' logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.

INTRODUCTION

This part of IEC 61158 is one of a series produced to facilitate the interconnection of automation system components. It is related to other standards in the set as defined by the “three-layer” fieldbus reference model described in IEC 61158-1.

The application service is provided by the application protocol making use of the services available from the data-link or other immediately lower layer. This standard defines the application service characteristics that fieldbus applications and/or system management may exploit.

Throughout the set of fieldbus standards, the term “service” refers to the abstract capability provided by one layer of the OSI Basic Reference Model to the layer immediately above. Thus, the application layer service defined in this standard is a conceptual architectural service, independent of administrative and implementation divisions.

INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –

Part 5-4: Application layer service definition – Type 4 elements

1 Scope

1.1 General

The fieldbus application layer (FAL) provides user programs with a means to access the fieldbus communication environment. In this respect, the FAL can be viewed as a “window between corresponding application programs.”

This standard provides common elements for basic time-critical and non-time-critical messaging communications between application programs in an automation environment and material specific to Type 4 fieldbus. The term “time-critical” is used to represent the presence of a time-window, within which one or more specified actions are required to be completed with some defined level of certainty. Failure to complete specified actions within the time window risks failure of the applications requesting the actions, with attendant risk to equipment, plant and possibly human life.

This standard defines in an abstract way the externally visible service provided by the Type 4 fieldbus application layer in terms of

a) an abstract model for defining application resources (objects) capable of being manipulated by users via the use of the FAL service,

b) the primitive actions and events of the service;

c) the parameters associated with each primitive action and event, and the form which they take; and

d) the interrelationship between these actions and events, and their valid sequences.

The purpose of this standard is to define the services provided to

1) the FAL user at the boundary between the user and the application layer of the fieldbus reference model, and

2) Systems Management at the boundary between the application layer and Systems Management of the fieldbus reference model.

This standard specifies the structure and services of the Type 4 fieldbus application layer, in conformance with the OSI Basic Reference Model (ISO/IEC 7498-1) and the OSI application layer structure (ISO/IEC 9545).

FAL services and protocols are provided by FAL application-entities (AE) contained within the application processes. The FAL AE is composed of a set of object-oriented application service elements (ASEs) and a layer management entity (LME) that manages the AE. The ASEs provide communication services that operate on a set of related application process object (APO) classes. One of the FAL ASEs is a management ASE that provides a common set of services for the management of the instances of FAL classes.

Although these services specify, from the perspective of applications, how request and responses are issued and delivered, they do not include a specification of what the requesting and responding applications are to do with them. That is, the behavioral aspects of the applications are not specified; only a definition of what requests and responses they can

send/receive is specified. This permits greater flexibility to the FAL users in standardizing such object behavior. In addition to these services, some supporting services are also defined in this standard to provide access to the FAL to control certain aspects of its operation.

1.2 Specifications

The principal objective of this standard is to specify the characteristics of conceptual application layer services suitable for time-critical communications, and thus supplement the OSI Basic Reference Model in guiding the development of application layer protocols for time-critical communications.

A secondary objective is to provide migration paths from previously-existing industrial communications protocols. It is this latter objective which gives rise to the diversity of services standardized as the various Types of IEC 61158, and the corresponding protocols standardized in IEC 61158-6 series.

This specification may be used as the basis for formal application programming interfaces. Nevertheless, it is not a formal programming interface, and any such interface will need to address implementation issues not covered by this specification, including

a) the sizes and octet ordering of various multi-octet service parameters, and

b) the correlation of paired request and confirm, or indication and response, primitives.

1.3 Conformance

This standard does not specify individual implementations or products, nor does it constrain the implementations of application layer entities within industrial automation systems.

There is no conformance of equipment to this application layer service definition standard. Instead, conformance is achieved through implementation of conforming application layer protocols that fulfill the Type 2 application layer services as defined in this standard.

2 Normative references

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

NOTE All parts of the IEC 61158 series, as well as IEC 61784-1 and IEC 61784-2 are maintained simultaneously. Cross-references to these documents within the text therefore refer to the editions as dated in this list of normative references.

IEC 61158-3-4:2014, *Industrial communication networks – Fieldbus specifications – Part 3-4: Data-link layer service definition – Type 4 elements*

IEC 61158-4-4:2014, *Industrial communication networks – Fieldbus specifications – Part 4-4: Data-link layer protocol specification – Type 4 elements*

IEC 61158-6-4:2014, *Industrial communication networks – Fieldbus specifications – Part 6-4: Application layer protocol specification – Type 4 elements*

IEC 61158-6 (all subparts), *Industrial communication networks – Fieldbus specifications – Part 6: Application layer protocol specification*

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

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

ISO/IEC 8822, *Information technology – Open Systems Interconnection – Presentation service definition*

ISO/IEC 8824-1, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation*

ISO/IEC 9545, *Information technology – Open Systems Interconnection – Application Layer structure*

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

ISO/CEI/IEEE 60559, *Information technology – Microprocessor Systems – Floating-Point arithmetic*

3 Terms and definitions

For the purposes of this document, the following terms as defined in these publications apply:

3.1 ISO/IEC 7498-1 terms

a) application entity
b) application process
c) application protocol data unit
d) application service element
e) application entity invocation
f) application process invocation
g) application transaction
h) real open system
i) transfer syntax

3.2 ISO/IEC 8822 terms

For the purposes of this document, the following terms as defined in ISO/IEC 8822 apply:

a) abstract syntax
b) presentation context

3.3 ISO/IEC 9545 terms

For the purposes of this document, the following terms as defined in ISO/IEC 9545 apply:

a) application-association
b) application-context
c) application context name
d) application-entity-invocation
e) application-entity-type
f) application-process-invocation

g) application-process-type
h) application-service-element
i) application control service element

3.4 ISO/IEC 8824-1 terms

For the purposes of this document, the following terms as defined in ISO/IEC 8824-1 apply:

a) object identifier
b) type

3.5 Fieldbus data-link layer terms

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

a) DL-Time
b) DL-Scheduling-policy
c) DLCEP
d) DLC
e) DLPDU
f) DLSDU
g) DLSAP
h) fixed tag
i) generic tag
j) link
k) network address
l) node address
m) node
n) tag
o) scheduled
p) unscheduled

3.6 Fieldbus application layer specific definitions

For the purposes of this standard, the following terms and definitions apply.

3.6.1
application
function or data structure for which data is consumed or produced

3.6.2
application objects
multiple object classes that manage and provide a run time exchange of messages across the network and within the network device

3.6.3
application process
part of a distributed application on a network, which is located on one device and unambiguously addressed

3.6.4
application process identifier
distinguishes multiple application processes used in a device

3.6.5
application process object
component of an application process that is identifiable and accessible through an FAL application relationship

Note 1 to entry: Application process object definitions are composed of a set of values for the attributes of their class (see the definition for Application Process Object Class Definition). Application process object definitions may be accessed remotely using the services of the FAL Object Management ASE. FAL Object Management services can be used to load or update object definitions, to read object definitions, and to dynamically create and delete application objects and their corresponding definitions.

3.6.6
application process object class
class of application process objects defined in terms of the set of their network-accessible attributes and services

3.6.7
application relationship
cooperative association between two or more application-entity-invocations for the purpose of exchange of information and coordination of their joint operation

Note 1 to entry: This relationship is activated either by the exchange of application-protocol-data-units or as a result of preconfiguration activities.

3.6.8
application relationship application service element
application-service-element that provides the exclusive means for establishing and terminating all application relationships

3.6.9
application relationship endpoint
context and behavior of an application relationship as seen and maintained by one of the application processes involved in the application relationship

Note 1 to entry: Each application process involved in the application relationship maintains its own application relationship endpoint.

3.6.10
attribute
description of an externally visible characteristic or feature of an object

Note 1 to entry: The attributes of an object contain information about variable portions of an object. Typically, they provide status information or govern the operation of an object. Attributes may also affect the behaviour of an object. Attributes are divided into class attributes and instance attributes.

3.6.11
behaviour
indication of how an object responds to particular events

3.6.12
bit-no
designates the number of a bit in a bitstring or an octet

3.6.13
channel
single physical or logical link of an input or output application object of a server to the process

3.6.14
class
set of objects, all of which represent the same kind of system component

Note 1 to entry: A class is a generalisation of an object; a template for defining variables and methods. All objects in a class are identical in form and behaviour, but usually contain different data in their attributes.

3.6.15
class attributes
attribute that is shared by all objects within the same class

3.6.16
class code
unique identifier assigned to each object class

3.6.17
class specific service
service defined by a particular object class to perform a required function which is not performed by a common service

Note 1 to entry: A class specific object is unique to the object class which defines it.

3.6.18
client
a) object which uses the services of another (server) object to perform a task

b) initiator of a message to which a server reacts

3.6.19
communication objects
components that manage and provide a run time exchange of messages across the network

EXAMPLES: Connection Manager object, Unconnected Message Manager (UCMM) object, and Message Router object.

3.6.20
connection
logical binding between application objects that may be within the same or different devices

Note 1 to entry: Connections may be either point-to-point or multipoint.

3.6.21
conveyance path
unidirectional flow of APDUs across an application relationship

3.6.22
dedicated AR
AR used directly by the FAL User

Note 1 to entry: On Dedicated ARs, only the FAL Header and the user data are transferred.

3.6.23
default DL-address
value 126 as an initial value for DL-address, which has to be changed (e.g. by assignment of a DL-address via the fieldbus) before operation with a DP-master (class 1)

3.6.24
device
physical hardware connected to the link

Note 1 to entry: A device may contain more than one node.

3.6.25
dynamic AR
AR that requires the use of the AR establishment procedures to place it into an established state

3.6.26
endpoint
one of the communicating entities involved in a connection

3.6.27
error
discrepancy between a computed, observed or measured value or condition and the specified or theoretically correct value or condition

3.6.28
error class
general grouping for related error definitions and corresponding error codes

3.6.29
error code
identification of a specific type of error within an error class

3.6.30
event
instance of a change of conditions

3.6.31
FAL subnet
subnetworks composed of one or more data link segments, identified by a subset of the network address

Note 1 to entry: FAL subnets are permitted to contain bridges but not routers.

3.6.32
FIFO variable
Variable Object class, composed of a set of homogeneously typed elements, where the first written element is the first element that can be read

Note 1 to entry: On the fieldbus only one, complete element can be transferred as a result of one service invocation.

3.6.33
frame
denigrated synonym for DLPDU

3.6.34
interface
a) shared boundary between two functional units, defined by functional characteristics, signal characteristics, or other characteristics as appropriate

b) collection of FAL class attributes and services that represents a specific view on the FAL class

3.6.35
invocation
act of using a service or other resource of an application process

Note 1 to entry: Each invocation represents a separate thread of control that may be described by its context. Once the service completes, or use of the resource is released, the invocation ceases to exist. For service invocations, a service that has been initiated but not yet completed is referred to as an outstanding service invocation. Also for service invocations, an Invoke ID may be used to unambiguously identify the service invocation and differentiate it from other outstanding service invocations.

3.6.36
index
address of an object within an application process

3.6.37
instance
actual physical occurrence of an object within a class that identifies one of many objects within the same object class

EXAMPLE California is an instance of the object class state.

Note 1 to entry: The terms object, instance, and object instance are used to refer to a specific instance.

3.6.38
instance attributes
attribute that is unique to an object instance and not shared by the object class

3.6.39
instantiated
object that has been created in a device

3.6.40
logical device
certain FAL class that abstracts a software component or a firmware component as an autonomous self-contained facility of an automation device

3.6.41
manufacturer ID
identification of each product manufacturer by a unique number

3.6.42
management information
network-accessible information that supports managing the operation of the fieldbus system, including the application layer

Note 1 to entry: Managing includes functions such as controlling, monitoring, and diagnosing.

3.6.43
member
piece of an attribute that is structured as an element of an array

3.6.44
method
<object> a synonym for an operational service which is provided by the server ASE and invoked by a client

3.6.45
module
hardware or logical component of a physical device

3.6.46
network
set of nodes connected by some type of communication medium, including any intervening repeaters, bridges, routers and lower-layer gateways

3.6.47
object
abstract representation of a particular component within a device, usually a collection of related data (in the form of variables) and methods (procedures) for operating on that data that have clearly defined interface and behaviour

3.6.48
object specific service
service unique to the object class which defines it

3.6.49
peer
role of an AR endpoint in which it is capable of acting as both client and server

3.6.50
physical device
automation or other network device

3.6.51
property
general term for descriptive information about an object

3.6.52
provider
source of a data connection

3.6.53
publisher
role of an AR endpoint that transmits APDUs onto the fieldbus for consumption by one or more subscribers

Note 1 to entry: A publisher may not be aware of the identity or the number of subscribers and it may publish its APDUs using a dedicated AR.

3.6.54
publishing manager
role of an AR endpoint in which it issues one or more confirmed service request APDUs to a publisher to request the publisher to publish a specified object

Note 1 to entry: Two types of publishing managers are defined by this standard, pull publishing managers and push publishing managers, each of which is defined separately

3.6.55
pull subscriber
type of subscriber that recognizes received confirmed service response APDUs as published object data

3.6.56
resource
processing or information capability of a subsystem

3.6.57
route endpoint
object container containing Variable Objects of a variable class

3.6.58
server
a) role of an AREP in which it returns a confirmed service response APDU to the client that initiated the request

b) object which provides services to another (client) object

3.6.59
service
operation or function than an object and/or object class performs upon request from another object and/or object class

3.6.60
subscriber
role of an AREP in which it receives APDUs produced by a publisher

3.7 Abbreviations and symbols

AE	Application Entity
AL	Application Layer
ALME	Application Layer Management Entity
ALP	Application Layer Protocol
APO	Application Object
AP	Application Process
APDU	Application Protocol Data Unit
API	Application Process Identifier
AR	Application Relationship
AREP	Application Relationship End Point
ASCII	American Standard Code for Information Interchange
ASE	Application Service Element
Cnf	Confirmation
CR	Communication Relationship
CREP	Communication Relationship End Point
DL-	(as a prefix) Data Link-
DLC	Data Link Connection
DLCEP	Data Link Connection End Point
DLL	Data Link Layer
DLM	Data Link-management
DLSAP	Data Link Service Access Point
DLSDU	DL-service-data-unit
DNS	Domain Name Service
DP	Decentralised Peripherals
FAL	Fieldbus Application Layer

FIFO	First In First Out
HMI	Human-Machine Interface
ID	Identifier
IDL	Interface Definition Language
IEC	International Electrotechnical Commission
Ind	Indication
IP	Internet Protocol
ISO	International Organization for Standardization
LDev	Logical Device
LME	Layer Management Entity
OSI	Open Systems Interconnect
PDev	Physical Device
PDU	Protocol Data Unit
PL	Physical Layer
QoS	Quality of Service
REP	Route Endpoint
Req	Request
Rsp	Response
RT	Runtime
SAP	Service Access Point
SCL	Security Level
SDU	Service Data Unit
SEM	State event matrix
SMIB	System Management Information Base
SMK	System Management Kernel
STD	State transition diagram, used to describe object behaviour
VAO	Variable Object

3.8 Conventions

3.8.1 Overview

The FAL is defined as a set of object-oriented ASEs. Each ASE is specified in a separate subclause. Each ASE specification is composed of two parts, its class specification, and its service specification.

The class specification defines the attributes of the class. The attributes are accessible from instances of the class using the Object Management ASE services specified in Clause 5 of this standard. The service specification defines the services that are provided by the ASE.

3.8.2 General conventions

This standard uses the descriptive conventions given in ISO/IEC 10731.

3.8.3 Conventions for class definitions

Class definitions are described using templates. Each template consists of a list of attributes for the class. The general form of the template is shown below:

FAL ASE:			**ASE Name**
CLASS:	**Class Name**		
CLASS ID:			#
PARENT CLASS:			Parent Class Name
ATTRIBUTES:			
1	(o)	Key Attribute:	numeric identifier
2	(o)	Key Attribute:	name
3	(m)	Attribute:	attribute name(values)
4	(m)	Attribute:	attribute name(values)
4.1	(s)	Attribute:	attribute name(values)
4.2	(s)	Attribute:	attribute name(values)
4.3	(s)	Attribute:	attribute name(values)
5.	(c)	Constraint:	constraint expression
5.1	(m)	Attribute:	attribute name(values)
5.2	(o)	Attribute:	attribute name(values)
6	(m)	Attribute:	attribute name(values)
6.1	(s)	Attribute:	attribute name(values)
6.2	(s)	Attribute:	attribute name(values)
SERVICES:			
1	(o)	OpsService:	service name
2.	(c)	Constraint:	constraint expression
2.1	(o)	OpsService:	service name
3	(m)	MgtService:	service name

(1) The "FAL ASE:" entry is the name of the FAL ASE that provides the services for the class being specified.

(2) The "CLASS:" entry is the name of the class being specified. All objects defined using this template will be an instance of this class. The class may be specified by this standard, or by a user of this standard.

(3) The "CLASS ID:" entry is a number that identifies the class being specified. This number is unique within the FAL ASE that will provide the services for this class. When qualified by the identity of its FAL ASE, it unambiguously identifies the class within the scope of the FAL. The value "NULL" indicates that the class cannot be instantiated. Class IDs between 1 and 255 are reserved by this standard to identify standardized classes. They have been assigned to maintain compatibility with existing national standards. CLASS IDs between 256 and 2048 are allocated for identifying user defined classes.

(4) The "PARENT CLASS:" entry is the name of the parent class for the class being specified. All attributes defined for the parent class and inherited by it are inherited for the class being defined, and therefore do not have to be redefined in the template for this class.

NOTE The parent-class "TOP" indicates that the class being defined is an initial class definition. The parent class TOP is used as a starting point from which all other classes are defined. The use of TOP is reserved for classes defined by this standard.

(5) The "ATTRIBUTES" label indicate that the following entries are attributes defined for the class.

a) Each of the attribute entries contains a line number in column 1, a mandatory (m) / optional (o) / conditional (c) / selector (s) indicator in column 2, an attribute type label in column 3, a name or a conditional expression in column 4, and optionally a list of enumerated values in column 5. In the column following the list of values, the default value for the attribute may be specified.

b) Objects are normally identified by a numeric identifier or by an object name, or by both. In the class templates, these key attributes are defined under the key attribute.

c) The line number defines the sequence and the level of nesting of the line. Each nesting level is identified by period. Nesting is used to specify

i) fields of a structured attribute (4.1, 4.2, 4.3),

ii) attributes conditional on a constraint statement (5). Attributes may be mandatory (5.1) or optional (5.2) if the constraint is true. Not all optional attributes require constraint statements as does the attribute defined in (5.2).

iii) the selection fields of a choice type attribute (6.1 and 6.2).

(6) The "SERVICES" label indicates that the following entries are services defined for the class.

a) An (m) in column 2 indicates that the service is mandatory for the class, while an (o) indicates that it is optional. A (c) in this column indicates that the service is conditional. When all services defined for a class are defined as optional, at least one has to be selected when an instance of the class is defined.

b) The label "OpsService" designates an operational service (1).

c) The label "MgtService" designates an management service (2).

d) The line number defines the sequence and the level of nesting of the line. Each nesting level is identified by period. Nesting within the list of services is used to specify services conditional on a constraint statement.

3.8.4 Conventions for service definitions

3.8.4.1 General

The service model, service primitives, and time-sequence diagrams used are entirely abstract descriptions; they do not represent a specification for implementation.

3.8.4.2 Service parameters

Service primitives are used to represent service user/service provider interactions (ISO/IEC 10731). They convey parameters which indicate information available in the user/provider interaction. In any particular interface, not all parameters need be explicitly stated.

The service specifications of this standard uses a tabular format to describe the component parameters of the ASE service primitives. The parameters which apply to each group of service primitives are set out in tables. Each table consists of up to five columns for the

1) Parameter name,
2) request primitive,
3) indication primitive,
4) response primitive, and
5) confirm primitive.

One parameter (or component of it) is listed in each row of each table. Under the appropriate service primitive columns, a code is used to specify the type of usage of the parameter on the primitive specified in the column:

M parameter is mandatory for the primitive

U parameter is a User option, and may or may not be provided depending on dynamic usage of the service user. When not provided, a default value for the parameter is assumed.

C parameter is conditional upon other parameters or upon the environment of the service user.

— (blank) parameter is never present.

S parameter is a selected item.

Some entries are further qualified by items in brackets. These may be

a) a parameter-specific constraint:
"(=)" indicates that the parameter is semantically equivalent to the parameter in the service primitive to its immediate left in the table.

b) an indication that some note applies to the entry:
"(n)" indicates that the following note "n" contains additional information pertaining to the parameter and its use.

3.8.4.3 Service procedures

The procedures are defined in terms of

- the interactions between application entities through the exchange of fieldbus Application Protocol Data Units, and
- the interactions between an application layer service provider and an application layer service user in the same system through the invocation of application layer service primitives.

These procedures are applicable to instances of communication between systems which support time-constrained communications services within the fieldbus Application Layer.

4 Concepts

4.1 Overview

The fieldbus is intended to be used in factories and process plants to interconnect primary automation devices (e.g. sensors, actuators, local display devices, annunciators, programmable logic controllers, small single loop controllers, and stand-alone field controls) with control and monitoring equipment located in control rooms.

Primary automation devices are associated with the lowest levels of the industrial automation hierarchy and perform a limited set of functions within a definite time window. Some of these functions include diagnostics, data validation, and handling of multiple inputs and outputs.

These primary automation devices, also termed field devices, are located close to the process fluids, the fabricated part, the machine, the operator and the environment. This use positions the fieldbus at the lowest levels of the Computer Integrated Manufacturing (CIM) architecture.

Some of the expected benefits in using fieldbus are reduction in wiring, increase in amount of data exchanged, wider distribution of control between the primary automation devices and the control room equipment, and the satisfaction of time critical constraints.

Clause 4 describes fundamentals of the FAL. Detailed descriptive information about each of the FAL ASEs can be found in the “Overview” subclause of each of the Communication Model specifications.

4.2 Architectural relationships

4.2.1 Relationship to the Application Layer of the OSI basic reference model

The functions of the FAL have been described according to OSI layering principles. However, its architectural relationship to the lower layers is different, as shown in Figure 1.

– The FAL includes OSI functions together with extensions to cover time-critical requirements. The OSI Application Layer Structure standard (ISO/IEC 9545) was used as a basis for specifying the FAL.

– The FAL directly uses the services of the underlying layer. The underlying layer may be the data link layer or any layer in between. When using the underlying layer, the FAL may provide functions normally associated with the OSI Middle Layers for proper mapping onto the underlying layer.

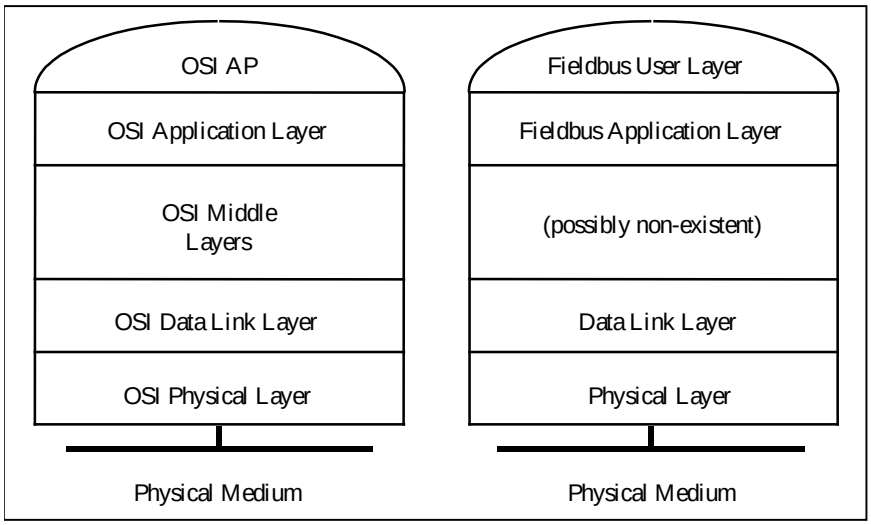


Figure 1 – Relationship to the OSI basic reference model

4.2.2 Relationships to other fieldbus entities

4.2.2.1 General

The fieldbus Application Layer (FAL) architectural relationships, as illustrated in Figure 2, have been designed to support the interoperability needs of time-critical systems distributed within the fieldbus environment.

Within this environment, the FAL provides communications services to time-critical and non-time-critical applications located in fieldbus devices.

In addition, the FAL directly uses the Data Link Layer to transfer its application layer protocol data units. It does this using a set of data transfer services and a set of supporting services used to control the operational aspects of the Data Link Layer.

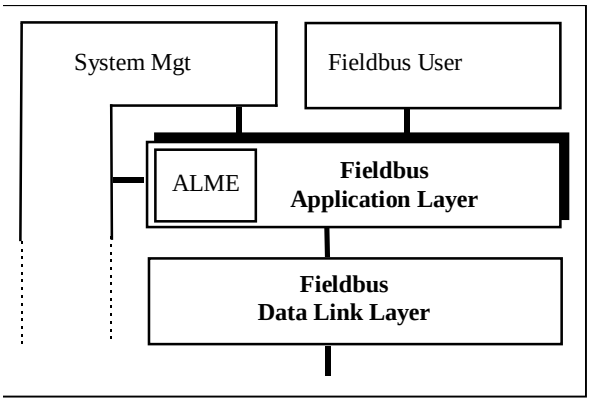


Figure 2 – Architectural positioning of the fieldbus Application Layer

4.2.2.2 Use of the fieldbus Data Link Layer

The fieldbus Application Layer (FAL) provides network access to fieldbus APs. It interfaces directly to the fieldbus Data Link Layer for transfer of its APDUs.

The Data Link Layer provides various types of services to the FAL for the transfer of data between Data Link endpoints (e.g. DLSAPs, DLCEPs).

4.2.2.3 Support for fieldbus applications

Fieldbus applications are represented to the network as application processes (APs). APs are the components of a distributed system that may be individually identified and addressed.

Each AP contains an FAL application entity (AE) that provides network access for the AP. That is, each AP communicates with other APs through its AE. In this sense, the AE provides a window of visibility into the AP.

APs contain identifiable components that are also visible across the network. These components are represented to the network as Application Process Objects (APO). They may be identified by one or more key attributes. They are located at the address of the application process that contains them.

The services used to access them are provided by APO-specific application service elements (ASEs) contained within the FAL. These ASEs are designed to support user, function block, and management applications.

4.2.2.4 Support for system management

The FAL services can be used to support various management operations, including management of fieldbus systems, applications, and the fieldbus network.

4.2.2.5 Access to FAL layer management entities

One layer management entity (LME) may be present in each FAL entity on the network. FALMEs provide access to the FAL for system management purposes.

The set of data accessible by the System Manager is referred to as the System Management Information Base (SMIB). Each fieldbus Application Layer Management Entity (FALME) provides the FAL portion of the SMIB. How the SMIB is implemented is beyond the scope of this standard.

4.3 Fieldbus Application Layer structure

4.3.1 Overview

The structure of the FAL is a refinement of the OSI Application Layer Structure (ISO/IEC 9545). As a result, the organization of 4.3 is similar to that of ISO/IEC 9545. Certain concepts presented here have been refined from ISO/IEC 9545 for the fieldbus environment.

The FAL differs from the other layers of OSI in two principal aspects.

- OSI defines a single type of application layer communications channel, the association, to connect APs to each other. The FAL defines the Application Relationship (AR), of which there are several types, to permit application processes (APs) to communicate with each other.
- The FAL uses the DLL to transfer its PDUs and not the presentation layer. Therefore, there is no explicit presentation context available to the FAL. Between the same pair (or set) of data link service access points the FAL protocol may not be used concurrently with other application layer protocols.

4.3.2 Fundamental concepts

The operation of time critical real open systems is modeled in terms of interactions between time-critical APs. The FAL permits these APs to pass commands and data between them.

Cooperation between APs requires that they share sufficient information to interact and carry out processing activities in a coordinated manner. Their activities may be restricted to a single fieldbus segment, or they may span multiple segments. The FAL has been designed using a modular architecture to support the messaging requirements of these applications.

Cooperation between APs also sometimes requires that they share a common sense of time. The FAL or the Data Link Layer (IEC 61158-3-4 and IEC 61158-4-4) may provide for the distribution of time to all devices. They also may define local device services that can be used by APs to access the distributed time.

The remainder of 4.3 describes each of the modular components of the architecture and their relationships with each other. The components of the FAL are modeled as objects, each of which provides a set of FAL communication services for use by applications. The FAL objects and their relationships are described below. The detailed specifications of FAL objects and their services are provided in the following clauses of this standard. IEC 61158-6-4 specifies the protocols necessary to convey these object services between applications.

4.3.3 Fieldbus application processes

4.3.3.1 Definition of the fieldbus AP

In the fieldbus environment, an application may be partitioned into a set of components and distributed across a number of devices on the network. Each of these components is referred to as a fieldbus Application Process (AP). A fieldbus AP is a variation of an Application Process as defined in ISO OSI Reference Model (ISO/IEC 7498). Fieldbus APs may be unambiguously addressed by at least one individual Data Link Layer service access point address. Unambiguously addressed, in this context, means that no other AP may simultaneously be located by the same address. This definition does not prohibit an AP from being located by more than one individual or group data link service access point address.

4.3.3.2 Communication services

Fieldbus APs communicate with each other using confirmed and unconfirmed services (ISO/IEC 10731). The services defined in this standard for the FAL specify the semantics of the services as seen by the requesting and responding APs. The syntax of the messages used to convey the service requests and responses is defined in IEC 61158-6-4. The AP behavior associated with the services is specified by the AP.

Confirmed services are used to define request/response exchanges between APs.

Unconfirmed services, in contrast, are used to define the unidirectional transfer of messages from one AP to one or more remote APs. From a communications perspective, there is no relationship between separate invocations of unconfirmed services as there is between the request and response of a confirmed service.

4.3.3.3 AP interactions

4.3.3.3.1 General

Within the fieldbus environment, APs may interact with other APs as necessary to achieve their functional objectives. No constraints are imposed by this standard on the organization of these interactions or the possible relationships that may exist between them.

For example, in the fieldbus environment, interactions may be based on request/response messages sent directly between APs, or on data/events sent by one AP for use by others. These two models of interactions between APs are referred to as client/server and publisher/subscriber interactions.

The services supported by an interaction model are conveyed by application relationship endpoints (AREPs) associated with the communicating APs. The role that the AREP plays in the interaction (e.g. client, server, peer, publisher, subscriber) is defined as an attribute of the AREP.

4.3.3.3.2 Client/server interactions

Client/server interactions are characterized by a bi-directional data flow between a client AP and one or more server APs. Figure 3 illustrates the interaction between a single client and a single server. In this type of interaction, the client may issue a confirmed or unconfirmed request to the server to perform some task. If the service is confirmed then the server will always return a response. If the service is unconfirmed, the server may return a response using an unconfirmed service defined for this purpose.

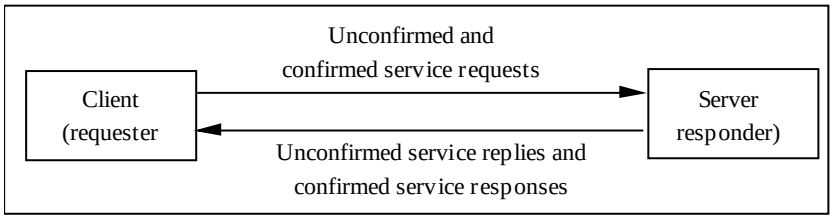


Figure 3 – Client/server interactions

4.3.3.3.3 Publisher/subscriber interactions

4.3.3.3.3.1 General

Publisher/subscriber interactions, on the other hand, involve a single publisher AP, and a group of one or more subscriber APs. This type of interaction has been defined to support variations of two models of interaction between APs, the "pull" model and the "push" model. In

both models, the setup of the publishing AP is performed by management and is outside the scope of this standard.

4.3.3.3.3.2 Pull model interactions

In the "pull" model, the publisher receives a request to publish from a remote publishing *manager*, and broadcasts (or multicasts) its response across the network. The publishing manager is responsible only for initiating publishing by sending a request to the publisher.

Subscribers wishing to receive the published data listen for responses transmitted by the publisher. In this fashion, data is "pulled" from the publisher by requests from the publishing manager.

Confirmed FAL services are used to support this type of interaction. Two characteristics of this type of interaction differentiate it from the other types of interaction. First, a typical confirmed request/response exchange is performed between publishing manager and the publisher. However, the underlying conveyance mechanism provided by the FAL returns the response not just to the publishing manager, but also to all subscribers wishing to receive the published information. This is accomplished by having the Data Link Layer transmit the response to a group address, rather than to the individual address of the publishing manager. Therefore, the response sent by the publisher contains the published data and is multicast to the publishing manager and to all subscribers.

The second difference occurs in the behavior of the subscribers. Pull model subscribers, referred to as pull subscribers, are capable of accepting published data in confirmed service responses without having issued the corresponding request. Figure 4 illustrates these concepts.

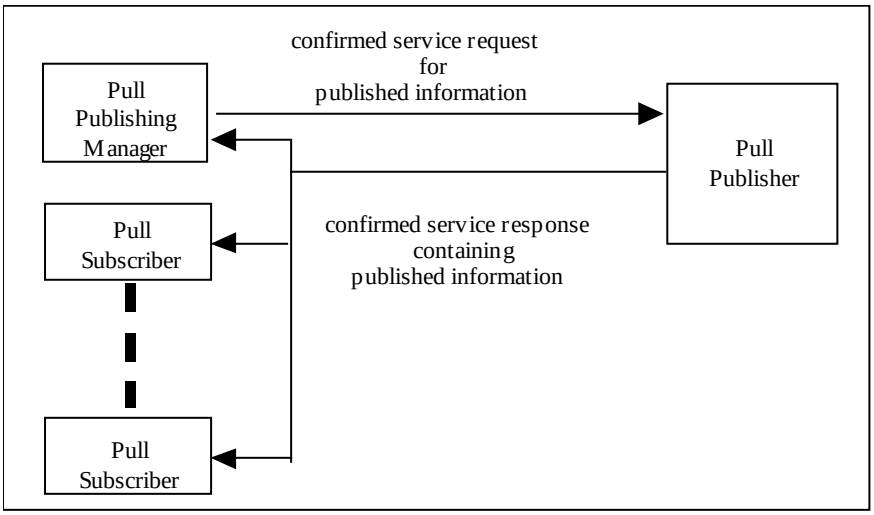


Figure 4 – Pull model interactions

4.3.3.3.3.3 Push model interactions

In the "push" model, two services may be used, one confirmed and one unconfirmed. The confirmed service is used by the subscriber to request to join the publishing. The response to this request is returned to the subscriber, following the client/server model of interaction. This exchange is only necessary when the subscriber and the publisher are located in different APs.

The unconfirmed service used in the Push Model is used by the publisher to distribute its information to subscribers. In this case, the publisher is responsible for invoking the correct unconfirmed service at the appropriate time and for supplying the appropriate information. In this fashion, it is configured to "push" its data onto the network.

Subscribers for the Push Model receive the published unconfirmed services distributed by publishers. Figure 5 illustrates the concept of the Push Model.

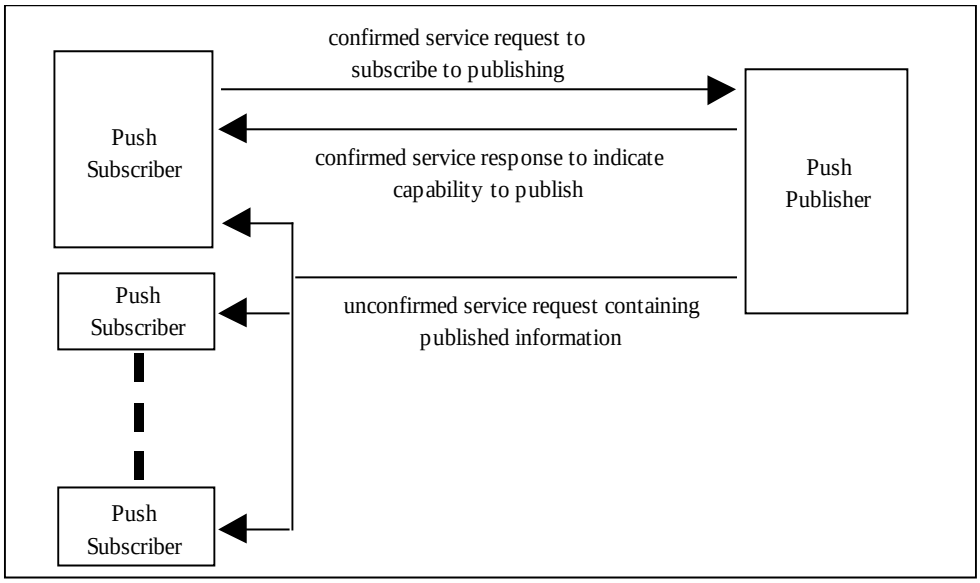


Figure 5 – Push model interactions

4.3.3.3.3.4 Timeliness of published information

To support the perishable nature of published information, the FAL may support four types of timeliness defined for publisher/subscriber interactions. Each make it possible for subscribers of published data to determine if the data they are receiving is up-to-date or "stale". These types are realized through mechanisms within the Data Link Layer (DLL). Each is described briefly below. For a more detailed description, refer to IEC 61158-3-4 and IEC 61158-4-4.

Type	Description
Transparent	This type of timeliness allows the user application process to determine the timeliness quality of data that it generates and have the timeliness quality accompany the information when it is transferred across the network. In this type of timeliness, the network provides no computation or measurement of timeliness. It merely conveys the timeliness quality provided with the data by the user application process.
Residence	When the FAL submits data from the publishing AP to the DLL for transmission, the DLL starts a timer. If the timer expires before the data has been transmitted, the DLL marks the buffer as "not timely" and conveys this timeliness information with the data.
Synchronized	This type of timeliness requires the coordination of two pieces of published information. One is the data to be published and the other is a special "sync mark". When the sync mark is received from the network a timer starts in each of the participating stations. Subsequently, when data is received for transmission by the DLL at the publishing station, or when the transmitted data is received from the network at a subscribing station, the DLL timeliness attribute for the data is set to TRUE. It remains TRUE until the reception of the next sync mark or until the timer expires. Data received after the timer expires but before the next sync mark does not cause the timeliness attribute to be reset to TRUE. It is only reset to TRUE if data is received within the time window after receipt of the sync mark. Data transmitted by the publisher station with the timeliness attribute set to FALSE maintains the setting of FALSE at each of the subscribers, regardless of their timer operation.
Update	This type of timeliness requires the coordination of the same two pieces of published information defined for *synchronized* timeliness. In this type, the sync mark also starts a timer in each of the participating stations. Like *synchronized* timeliness, expiration of the timer always causes the timeliness attribute to be set to FALSE. Unlike *synchronized* timeliness, receipt of new data at any time (not just within the time window started with the receipt of a sync mark) causes the timeliness attribute to be set to TRUE.

4.3.3.4 AP structure

The internals of APs may be represented by one, or more Application Process Objects (APOs) and accessed through one or more Application Entities (AEs). AEs provide the communication capabilities of the AP. For each fieldbus AP, there is one and only one FAL AE. APOs are the network representation of application specific capabilities (user application process objects) of an AP that are accessible through its FAL AE.

4.3.3.5 AP class

An AP class is a definition of the attributes and services of an AP. The standard class definition for APs is defined in this standard. User defined classes also may be specified. Class identifiers (described in 3.8.2) are assigned from a set reserved for this purpose.

4.3.3.6 AP type

As described above in the previous subclauses, APs are defined by instantiating an AP class. Each AP definition is composed of the attributes and services selected for the AP from those defined by its AP class. In addition, an AP definition contains values for one or more of the attributes selected for it. When two APs share the same definition, that definition is referred to as an AP type. Thus, an AP type is a generic specification of an AP that may used to define one or more APs.

4.3.4 Application process objects

4.3.4.1 Definition of APO

An application process object (APO) is a network representation of a specific aspect of an AP. Each APO represents a specific set of information and processing capabilities of an AP that are accessible through services of the FAL. APOs are used to represent these capabilities to other APs in a fieldbus system.

From the perspective of the FAL, an APO is modeled as a network accessible object contained within an AP or within another APO (APOs may contain other APOs). APOs provide the network definition for objects contained within an AP that are remotely accessible. The definition of an APO includes an identification of the FAL services that can be used by remote APs for remote access. The FAL services, as shown in Figure 6, are provided by the FAL communications entity of the AP, known as the FAL Applications Entity (FAL AE).

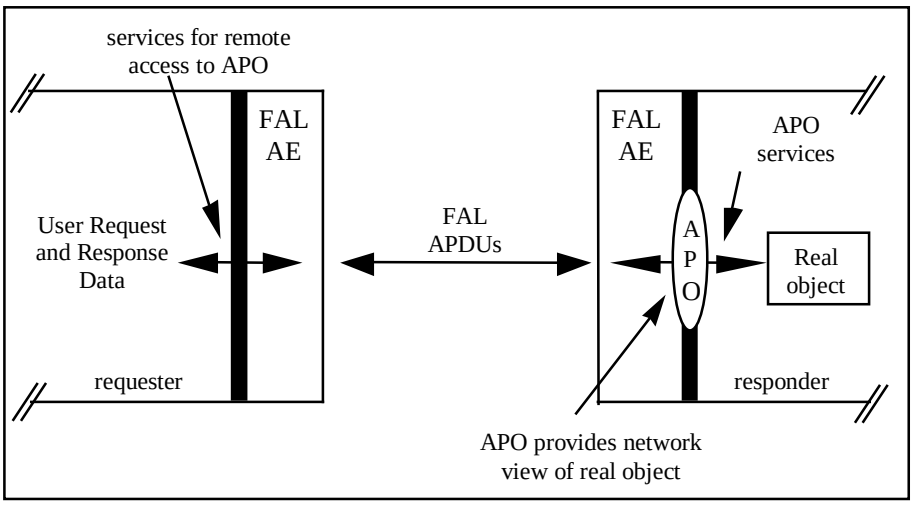


Figure 6 – APOs services conveyed by the FAL

In Figure 6, remote APs acting as clients may access the real object by sending requests through the APO that represents the real object. Local aspects of the AP convert between the network view (the APO) of the real object and the internal AP view of the real object.

To support the publisher/subscriber model of interaction, information about the real object can be published through its APO. Remote APs acting as subscribers see the APO view of the published information instead of having to know any of the real object specific details.

4.3.4.2 APO classes

An APO Class is a generic specification for a set of APOs, each of which is described by the same set of attributes and accessed using the same set of services.

APO Classes provide the mechanism for standardizing network visible aspects of APs. Each standard APO Class definition specifies a particular set of network accessible AP attributes and services. IEC 61158-6-4 specifies the syntax and the procedures used by the FAL protocol to provide remote access to the attributes and services of an APO Class.

Standard APO Classes are specified by this standard for the purpose of standardizing remote access to APs. User defined classes may also be specified.

User defined classes are defined as subclasses of standardized APO Classes or of other user-defined classes. They may be defined by identifying new attributes or by indicating that optional attributes for the parent class are mandatory for the subclass. The conventions for defining classes defined in 3.8.2 may be used for this purpose. The method for registering or otherwise making these new class definitions available for public use is beyond the scope of this standard.

4.3.4.3 APOs as instances of APO classes

APO Classes are defined in this standard using templates. These templates are used not only to define APO Classes, but also to specify the instances of a class.

Each APO defined for an AP is an instance of an APO Class. Each APO provides the network view of a real object contained in an AP. An APO is defined by

(1) selecting the attributes from its APO Class template that are to be accessible from the real object;

(2) assigning values to one or more attributes indicated as key in the template. Key attributes are used to identify the APO;

(3) assigning values to zero, one, or more non-key attributes for the APO. Non-key attributes are used to characterize the APO;

(4) selecting the services from the template that may be used by remote APs to access the real object.

Subclause 3.8.2 of this standard specifies the conventions for class templates. These conventions provide for the definition of mandatory, optional, and conditional attributes and services.

Mandatory attributes and services are required to be present in all APOs of the class. Optional attributes and services may be selected, on an APO by APO basis, for inclusion in an APO. Conditional attributes and services are defined with an accompanying constraint statement. Constraint statements specify the conditions that indicate whether or not the attribute is to be present in an APO.

4.3.4.4 APO types

APO types provide the mechanism for defining standard APOs.

As described above in the previous subclauses, APOs are defined by instantiating an APO class. Each APO definition is composed of the attributes and services selected for the APO from those defined by its APO class. In addition, an APO definition contain values for one or more of the attributes selected for it. When two APOs share the same definition, except for the key attribute settings, that definition is referred to as an APO type. Thus, an APO type is a generic specification of an APO that may be used to define one or more APOs.

4.3.5 Application entities

4.3.5.1 Definition of FAL AE

An application entity provides the communication capabilities for a single AP. An FAL AE provides a set of services and the supporting protocols to enable communications between APs in a fieldbus environment. The services provided by FAL AEs are grouped into Application Service Elements (ASE), such that the FAL services provided to an AP are defined by the ASEs its FAL AE contains. Figure 7 illustrates this concept.

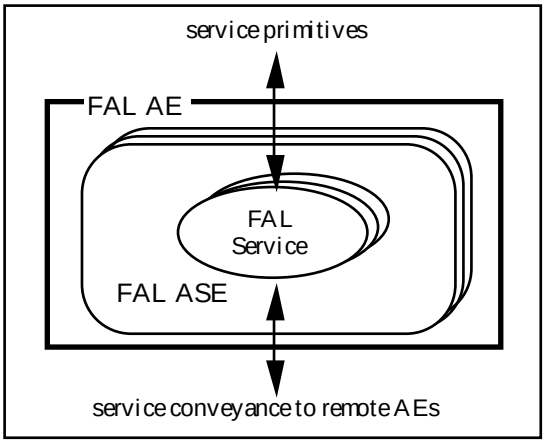


Figure 7 – Application entity structure

4.3.5.2 AE type

Application entities that provide the same set of ASEs are of the same AE-type. Two AEs that share a common set of ASEs are capable of communicating with each other.

4.3.6 Fieldbus application service elements

4.3.6.1 General

An application service element (ASE), as defined in ISO/IEC 9545, is a set of application functions that provide a capability for the interworking of application-entity-invocations for a specific purpose. ASEs provide a set of services for conveying requests and responses to and from application processes and their objects. AEs, as defined above, are represented by a collection of ASE invocations within the AE.

4.3.6.2 FAL services

FAL Services convey functional requests/responses between APs. Each FAL service is defined to convey requests and responses for access to a real object modeled as an FAL accessible object.

The FAL defines both confirmed and unconfirmed services. Confirmed service requests are sent to the AP containing the real object. An invocation of a confirmed service request may be identified by a user supplied Invoke ID. This Invoke ID is returned in the response by the AP containing the real object. When present, it is used by the requesting AP and its FAL AE to associate the response with the appropriate request.

Unconfirmed services may be sent from the AP containing the real object to send information about the object. They also may be sent to the AP containing the real object to access the real object. Both types of unconfirmed services may be defined for the FAL.

4.3.6.3 Definition of FAL ASEs

4.3.6.3.1 General

A modular approach has been taken in the definition of FAL ASEs. The ASEs defined for the FAL are also object-oriented. In general, ASEs provide a set of services designed for one specific object class or for a related set of classes. Common object management ASEs, when present, provide a common set of management services applicable to all classes of objects.

To support remote access to the AP, the Application Relationship ASE is defined. It provides services to the AP for defining and establishing communication relationships with other APs, and it provides services to the other ASEs for conveying their service requests and responses.

Each FAL ASE defines a set of services, APDUs, and procedures that operate on the classes that it represents. Only a subset of the ASE services may be provided to meet the needs of an application. Profiles may be used to define such subsets. Definition of profiles is beyond the scope of this standard.

APDUs are sent and received between FAL ASEs that support the same services. Each FAL AE contains, at a minimum, the AR ASE and at least one other ASE. Figure 8 illustrates an example set of the FAL ASEs and their architectural relationships. All APO ASEs follow this example.

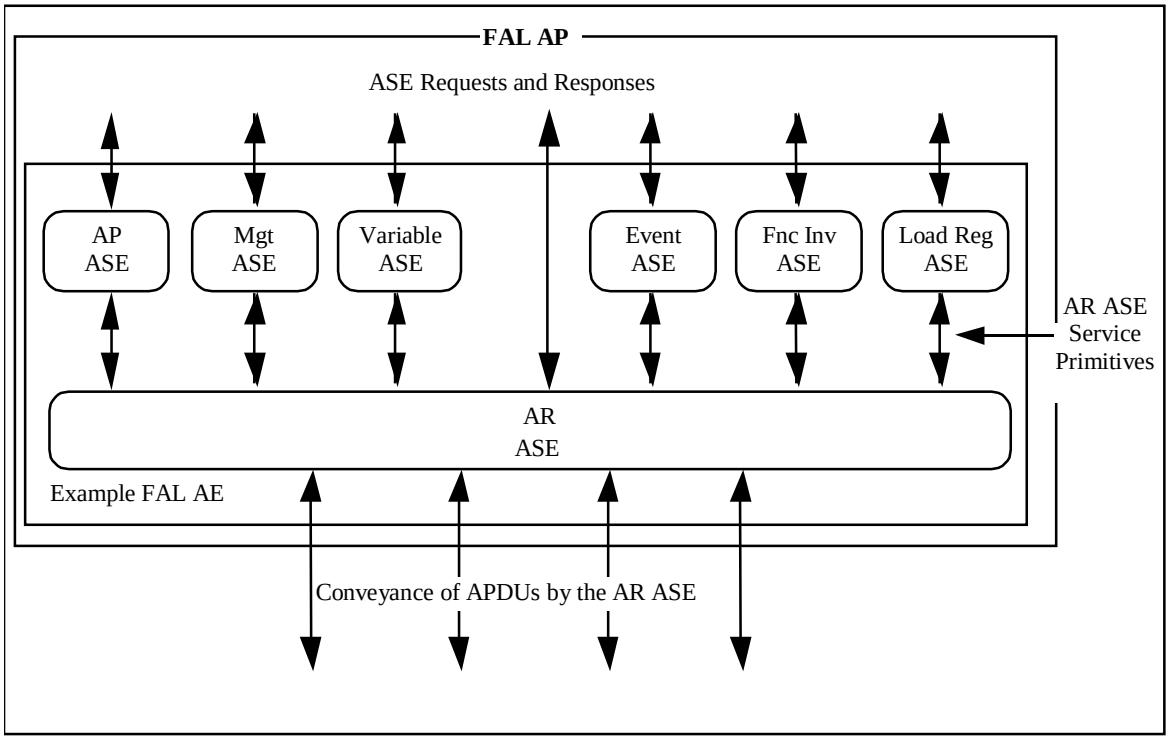


Figure 8 – Example FAL ASEs

4.3.6.3.2 Object management ASE

A special object management ASE may be specified for the FAL to provide services for the management of objects. Its services are used to access object attributes, and create and

delete object instances. These services are used to manage network visible AP objects accessed through the FAL. The specific operational services that apply to each object type are specified in the definition of the ASE for the object type. Figure 9 illustrates the integration of management and operational services for an object within an AP.

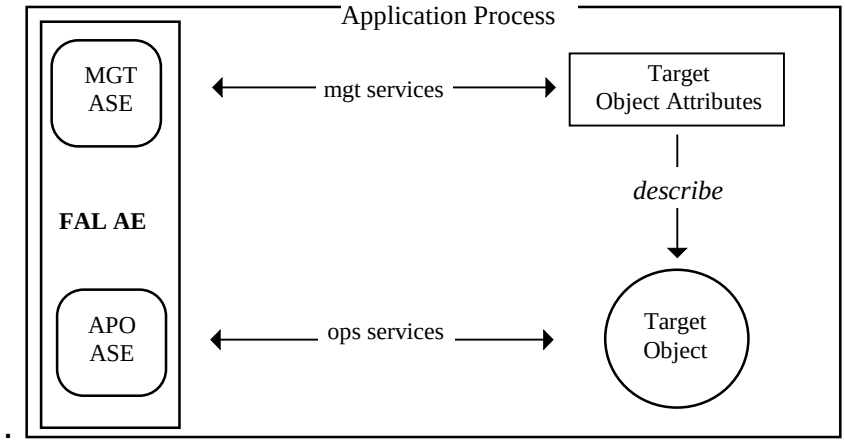


Figure 9 – FAL management of objects

4.3.6.3.3 AP ASE

An AP ASE may be specified for the identification and control of FAL APs. The attributes defined by the AP ASE specify characteristics of the AP about its manufacturer, and list its contents and capabilities.

4.3.6.3.4 APO ASEs

The FAL specifies a set of ASEs with services defined for accessing the APOs of an AP. The APO ASEs defined for the FAL are defined by each Communication Model.

4.3.6.3.5 AR ASE

An AR ASE is specified to establish and maintain application relationships (ARs) that are used to convey FAL APDUs between/among APs. ARs represent application layer communication channels between APs. AR ASEs are responsible for providing services at the endpoints of ARs. AR ASE services may be defined for establishing, terminating, and aborting ARs, for conveying APDUs for the AE, and for indicating the local status of the AR to the user. In addition, local services may be defined for accessing certain aspects of AR endpoints.

4.3.6.4 FAL service conveyance

FAL APO ASEs provide services to convey requests and responses between service users and real objects.

To accomplish the task of conveying service requests and responses, three types of activities for the sending user and three corresponding types for the receiving user are defined. At the sending user, they accept service requests and responses to be conveyed. Second, they select the type of FAL APDU that will be used to convey the request or response and encode the service parameters into its body portion. Then they submit the encoded APDU body to the AR ASE for conveyance.

At the receiving user, they receive encoded APDU bodies from the AR ASE. They decode the APDU bodies and extract the service parameters conveyed by them. To conclude the conveyance, they deliver the service request or response to the user. Figure 10 illustrates these concepts.

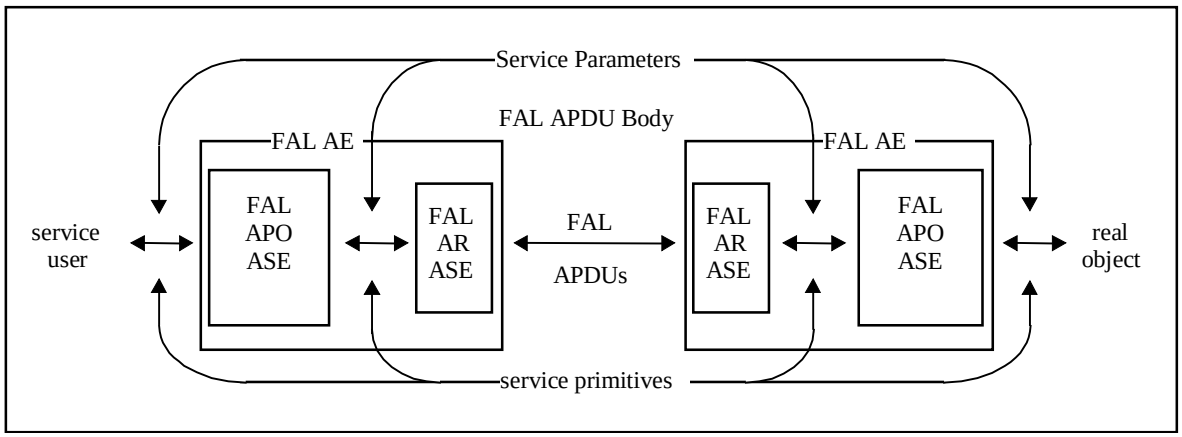


Figure 10 – ASE service conveyance

4.3.6.5 FAL presentation context

The presentation context in the OSI environment is used to distinguish the APDUs of one ASE from another, and to identify the transfer syntax rules used to encode each APDU. However, the fieldbus communications architecture does not include the presentation layer. Therefore, an alternate mechanism is provided for the FAL by each of the specific types of Communication Models.

4.3.7 Application relationships

4.3.7.1 Definition of AR

ARs represent communication channels between APs. They define how information is communicated between APs. Each AR is characterized by how it conveys ASE service requests and responses from one AP to another. These characteristics are described below.

4.3.7.2 AR-endpoints

ARs are defined as a set of cooperating APs. The AR ASE in each AP manages an endpoint of the AR, and maintains its local context. The local context of an AR endpoint is used by the AR ASE to control the conveyance of APDUs on the AR.

4.3.7.3 AR-endpoint classes

ARs are composed of a set of endpoints of compatible classes. AR endpoint classes are used to represent AR endpoints that convey APDUs in the same way. Through the standardization of endpoint classes, ARs for different models of interaction can be defined.

4.3.7.4 AR cardinality

ARs characterize communications between APs. One of the characteristics of an AR is the number of AR endpoints in the AR. ARs that convey services between two APs have a cardinality of 1-to-1. Those that convey services from one AP to a number of APs have a cardinality of 1-to-many. Those that convey services from/to multiple APs have a cardinality of many-to-many.

4.3.7.5 Accessing objects through ARs

ARs provide access to APs and the objects within them through the services of one or more ASEs. Therefore, one characteristic is the set of ASE services that may be conveyed to and from these objects by the AR. The list of services that can be conveyed by the AR are selected from those defined for the AE.

4.3.7.6 AR conveyance paths

ARs are modeled as one or two conveyance paths between AR endpoints. Each conveyance path conveys APDUs in one direction between one or more AR endpoints. Each receiving AR endpoint for a conveyance path receives all APDUs transmitting on the AR by the sending AR endpoint.

4.3.7.7 AREP roles

Because APs interact with each other through endpoints, a basic determinant of their compatibility is the role that they play in the AR. The role defines how an AREP interacts with other AREPs in the AR.

For example, an AREP may operate as a client, a server, a publisher, or a subscriber. When an AREP interacts with another AREP on a single AR as both a client and a server, it is defined to have the role of “peer”.

Certain roles may be capable of initiating service requests, while others may be capable only of responding to service requests. This part of the definition of a role identifies the requirement for an AR to be capable of conveying requests in either direction, or only in one direction.

4.3.7.8 AREP buffers and queues

AREPs may be modeled as a queue or as a buffer. APDUs transferred over a queued AREP are delivered in the order received for conveyance. The transfer of APDUs over a buffered AREP is different. In this case, an APDU to be conveyed by the AR ASE is placed in a buffer for transfer. When the Data Link Layer gains access to the network, it transmits the contents of the buffer.

When the AR ASE receives another conveyance request, it replaces the previous contents of the buffer whether or not they were transmitted. Once an APDU is written into a buffer for transfer, it is preserved in the buffer until the next APDU to be transmitted replaces it. While in the buffer, an APDU may be read more than once without deleting it from the buffer or changing its contents.

At the receiving end, the operation is similar. The receiving endpoint places a received APDU into a buffer for access by the AR ASE. When a subsequent APDU is received, it overwrites the previous APDU in the buffer whether or not it was read. Reading the APDU from the buffer is not destructive — it does not destroy or change the contents of the buffer, allowing the contents to be read from the buffer one or more times.

4.3.7.9 User-triggered and scheduled conveyance

Another characteristic of an AREP is when they convey service requests and responses. AREPs that convey them upon submission by the user are called user-triggered. Their conveyance is asynchronous with respect to network operation.

AREPs that convey requests and responses at predefined intervals, regardless of when they are received for transfer are termed scheduled. Scheduled AREPs may be capable of indicating when transferred data was submitted late for transmission, or when it was submitted on time, but transmitted late.

4.3.7.10 AREP timeliness

AREPs convey APDUs between applications using the services of the Data Link Layer. When the timeliness capabilities are defined for an AREP and supported by the Data Link Layer, the AREP forwards the timeliness indicators provided by the Data Link Layer. These timeliness indicators make it possible for subscribers of published data to determine if the data they are receiving is up-to-date or “stale”.

To support these types of timeliness, the publishing AREP establishes a publisher data link connection reflecting the type of timeliness configured for it by management. After connection establishment, the AREP receives user data and submits it to the DLL for transmission, where timeliness procedures are performed. When the Data Link Layer has the opportunity to transmit the data, it transmits the current timeliness status with the data.

At the subscriber AREP, a data link connection is opened to receive published data that reflects the type of timeliness configured for it by management. The Data Link Layer computes the timeliness of received data and then delivers it to the AREP. The data is then delivered to the user AP through the appropriate ASE.

4.3.7.11 Definition and creation of AREPs

AREP definitions specify instances of AREP classes. AREPs may be predefined or they may be defined using a “create” service if their AE supports this capability.

AREPs may be pre-defined and pre-established, or they may be pre-defined and dynamically established. Figure 11 depicts these two cases. AREPs also may require both dynamic definition and establishment or they may be dynamically defined such that they may be used without any establishment (they are defined in an established state).

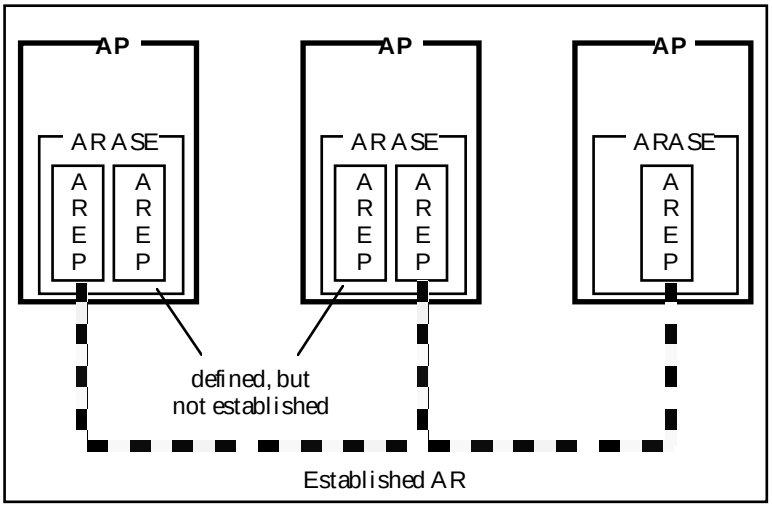


Figure 11 – Defined and established AREPs

4.3.7.12 AR establishment and termination

ARs may be established either before the operational phase of the AP or during its operation. When established during the operation of an AP, the AR is established through the exchange of AR APDUs.

Once an AR has been established, an AR may be terminated gracefully or it may be aborted, depending on the capabilities of the AR.

4.4 Fieldbus Application Layer naming and addressing

4.4.1 General

Subclause 4.4 refines the principles defined in ISO 7498-3 that involve the identification (naming) and location (addressing) of APOs referenced through the fieldbus Application Layer.

Subclause 4.4 defines how names and numeric identifiers are used to identify APOs accessible through the FAL. Subclause 4.4 also indicates how addresses from underlying layers are used to locate APs in the fieldbus environment.

4.4.2 Identifying objects accessed through the FAL

4.4.2.1 General

APOs accessed through the FAL are identified independent of their location. That is, if the location of the AP that contains the APO changes, the APO may still be referenced using the same set of identifiers.

Identifiers for APs and APOs within the FAL are defined as key attributes in the class definitions for APOs. Within these APO definitions, two types of key attributes are commonly used, names and numeric identifiers.

4.4.2.2 Names

Names are string-oriented identifiers. They are defined to permit APs and APOs to be named within the system where they are used. Therefore, although the scope of the name of an APO is specific to the AP in which it resides, the assignment of the name is administered within the system in which it is configured.

Names may be descriptive, although they do not have to be. Descriptive names make it possible to provide meaningful information, such as its use, about the object they name.

Names may also be coded. Coded names make it possible to identify an object using a short, compressed form of a name. They are typically simpler to transfer and process, but not as easy to understand as descriptive names.

4.4.2.3 Numeric identifiers

Numeric identifiers are identifiers whose values are numbers. They are designed for efficient use within the fieldbus system, and may be assigned for efficient access to APOs by their AP.

4.4.3 Addressing APs accessed through the FAL

Fieldbus addresses represent the network locations of APs. Addresses relevant to the FAL are the addresses of the underlying layers that are used to locate the AREPs of an AP.

4.5 Architecture summary

The summary of the FAL architecture is presented in 4.5. Figure 12 illustrates the major components of the FAL architecture and how they relate to each other.

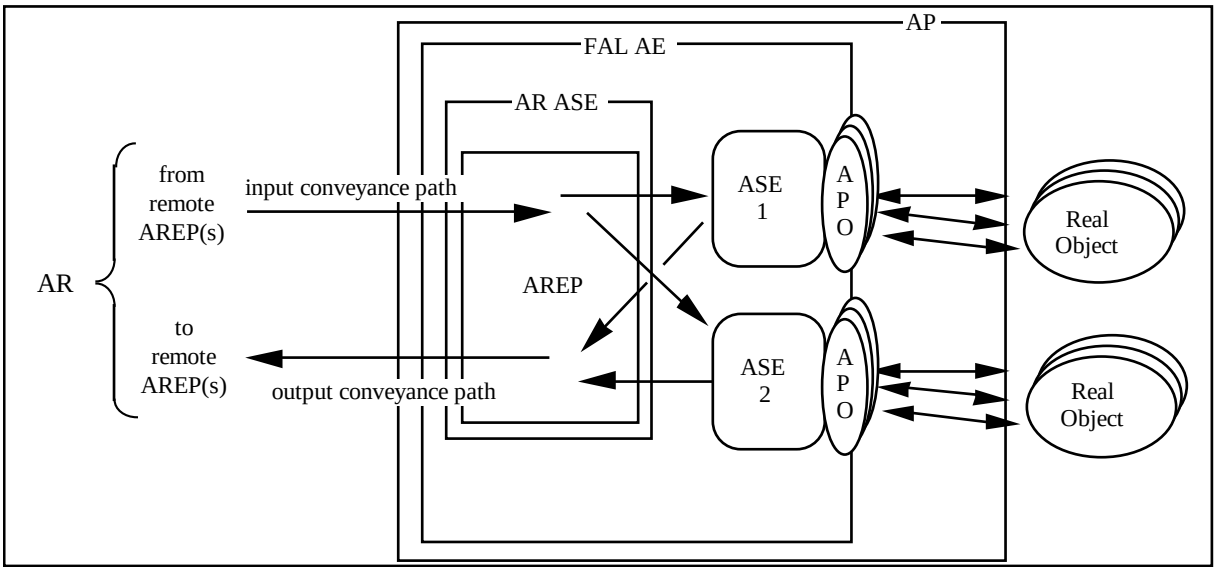


Figure 12 – FAL architectural components

Figure 12 depicts an AP that communicates through the FAL AE. The AP represents its internal real objects as APOs for remote access to them. Two ASEs are shown that provide the remote access services to their related APOs. The AR ASE contains a single AREP that conveys service requests and responses for the ASEs to one or more remote AREPs located in remote APs.

4.6 FAL service procedures

4.6.1 FAL confirmed service procedures

The requesting user submits a confirmed service request primitive to its FAL. The appropriate FAL ASE builds the related confirmed service request APDU Body and conveys it on the specified AR.

Upon receipt of the confirmed service request APDU Body, the responding ASE decodes it. If a protocol error did not occur, the ASE delivers a confirmed service indication primitive to its user.

If the responding user is able to successfully process the request, the user returns a confirmed service response (+) primitive.

If the responding user is unable to successfully process the request, the service fails and the user issues a confirmed service response (-) primitive indicating the reason.

The responding ASE builds a confirmed service response APDU Body for a confirmed service response (+) primitive or a confirmed service error APDU Body for a confirmed service response (–) primitive and conveys it on the specified AR.

Upon receipt of the returned APDU Body the initiating ASE delivers a confirmed service confirmation primitive to its user which specifies success or failure, and the reason for failure if a failure occurred.

4.6.2 FAL unconfirmed service procedures

The requesting user submits an unconfirmed service request primitive to its FAL AE. The appropriate FAL ASE builds the related unconfirmed service request APDU Body and conveys it on the specified AR.

Upon receipt of the unconfirmed request APDU Body, the receiving ASE(s) participating in the AR delivers the appropriate unconfirmed service indication primitive to its user. Timeliness parameters are included in the indication primitive if the AR that conveyed the APDU Body supports timeliness.

4.7 Common FAL attributes

In the specifications of the FAL classes that follow, many classes use the following attributes. Therefore, these attributes are defined here instead of with the other attributes for each of the classes, except for the Data Type class.

ATTRIBUTES:

1	(o)	Key Attribute:	Numeric Identifier
2	(o)	Key Attribute:	Name
3	(o)	Attribute:	User Description
4	(o)	Attribute:	Object Revision

1) Numeric Identifier

   This optional key attribute specifies the numeric id of the object. It is used as a shorthand reference by the FAL protocol to identify the object. There are three possibilities for identification purposes: numeric identifier or name or both. This attribute is required for the Data Type model.

2) Name

   This optional key attribute specifies the name of the object. There are three possibilities for identification purposes: numeric identifier or name or both.

3) User description

   This optional attribute contains user defined descriptive information about the object.

4) Object revision

   This optional attribute specifies the revision level of the object. It is a structured attribute composed of major and minor revision numbers. If Object Revision is supported, it contains both a Major Revision and a Minor Revision with a value range 0 to 15 for each. The use of major/minor fields is intended to provide the following features:

   – Major revision

     The Major Revision field contains the major revision value for the object. A change to the major revision indicates that interoperability is affected by the change.

   – Minor revision

     The Minor Revision field contains the minor revision value for the object. A change to the minor revision indicates that interoperability was not affected by the change -- that is users of the object will continue to be capable of interoperating with the object when its minor revision is changed, provided that the major revision remains the same.

4.8 Common FAL service parameters

In the specifications of the FAL services that follow, many services use the following parameters. Therefore, they are defined here instead of with the other parameters for each of the services.

– AREP

  This parameter contains sufficient information to locally identify the AREP to be used to convey the service. This parameter may use a key attribute of the AREP to identify the application relationship. When an AREP supports multiple contexts (established using the Initiate service) at the same time, the AREP parameter is extended to identify the context as well as the AREP.

– Invoke ID

This parameter identifies this invocation of the service. It is used to associate service requests with responses. Therefore, no two outstanding service invocations may be identified by the same Invoke ID value.

– FAL ASE/FAL class

This parameter specifies the FAL ASE (e.g. AP, AR, Variable, Data Type, Event, Function Invocation, Load Region) and the FAL Class within the ASE (e.g. AREP, Variable List, Notifier, Action).

– Numeric ID

This parameter is the numeric identifier of the object.

– Error info

This parameter provides error information for service errors. It is returned in confirmed service response(-) primitives. It is composed of the following elements.

- Error class

  This parameter indicates the general class of error. Valid values are specified in the definition of Error Code parameter, below.

- Error code

  This parameter identifies the specific service error.

- Additional code

  This optional parameter identifies the error encountered when processing the request specific to the object being accessed. When used, the value submitted in the response primitive is delivered unchanged in the confirmation primitive.

- Additional detail

  This optional parameter contains user data that accompanies the negative response. When used, the value submitted in the response primitive is delivered unchanged in the confirmation primitive.

4.9 APDU size

APDU size is Communication Model dependent.

5 Type 4 communication model specification

5.1 Concepts

5.1.1 Overview

The concepts of this model basically follow the common concepts with a few exceptions. The basic principle described in 4.2.2.1 through 4.2.2.3 are applicable for Clause 5. This Type supports Gateway functionality within the application layer. Part of this can also be recognised from the common concepts. However, to promote readability, some elements from Clause 4 are repeated in Clause 5.

5.1.2 Application entities

5.1.2.1 Definition of FAL AE

An FAL AE provides a set of services to enable communications between APs in a fieldbus environment. These services enable a user application in a Client to access user objects in a server.

The functionality of the FAL AE can be divided into two groups.

Application Relation (AR). The AR functionality is responsible for arranging the transportation of APDUs via the network.

Variable Access (VA). The Variable Access functionality is responsible for coding and encoding of APDUs. In a server the VA also convey data directly to / from user objects. How this is done is a local matter and out of scope for this standard.

5.1.2.2 FAL services

The services offered by the FAL AE to the user application are called FAL Services. Services for the AR are called AR ASE services and services for the Variable Access are called Route Endpoint ASE services (REP ASE services).

The FAL services provide access to local AR objects, and provide conveyance of requests and responses for access to real objects modelled as FAL accessible objects. The FAL provides as well confirmed as unconfirmed services. Service requests are conveyed to the AP containing the real object.

Figure 13 illustrates the FAL AE and the architectural relationships.

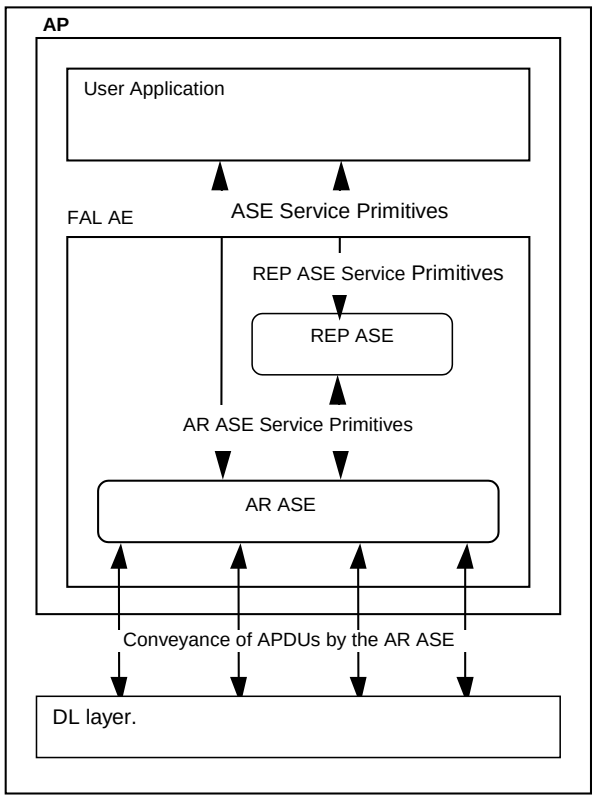


Figure 13 – FAL AE

5.1.2.3 AR ASE

To support access to the remote AP, the Application Relationship ASE is defined. It provides services to the AP for accessing communication-related parameters, and it provides services to the REP ASE for conveying service requests and responses.

The AR ASE provides services at the endpoints of ARs (AREPs).

5.1.2.4 REP ASE

The REP ASE provides a set of services used to read and write network visible attributes of Variable Objects and their application data. As only one ASE type is used remotely, the type is implicit and not transferred.

5.1.2.5 FAL service conveyance and gateway

The FAL AE provides services to convey requests and responses between User Applications and Variable Objects including Gateway functionality.

In a distributed system, application processes communicate with each other by exchanging application layer messages. The AR is defined to support the on-demand exchange of confirmed and unconfirmed services between two application processes. Connectionless data link services are used for the exchanges. The data link layer transfers may be either acknowledged or unacknowledged.

5.1.2.6 Application relationships

5.1.2.6.1 Definition of AR

The AR is responsible for conveying messages between applications. The messages that are conveyed are FAL service requests and responses. Each service request and response is submitted to the AR ASE for transfer. Within the AR only one object type exist, the AR-Endpoint (AREP).

5.1.2.6.2 AR-endpoints

An AR-Endpoint (AREP) represents a service access point to a DLE within the AE. The AR Send Service conveys APDU and route information to the AREP and implicit to the DLE. The AREP is a system management object presenting the user configurable parameters of the related DLE.

This Type has no special system management services, because the attributes of the AREP objects are declared as Variable Objects by the user application. This makes the attributes network visible. The AR Get Attribute and AR Set Attribute services are used by these Variable Objects to access the attribute values.

When an AREP receives an APDU from its DLE (User-triggered) it conveys the APDU to the REP ASE or an AREP (gateway).

5.1.3 Gateway and routing

5.1.3.1 Overview

This Type supports gateway functionality. To identify the destination of a request, a route description is used. This route description holding a complete list of points to pass is sent with the request. On its way to the destination, the route evolves in such a way that it always contains the route back to the requester and the route to the destination. The elements in the route are endpoint and DL addresses.

5.1.3.2 Route endpoint

Within the REP ASE, Route Endpoint objects (REPs) represent endpoints, holding a Route description. An REP represents the starting point for conveying a message. An REP also represents the destination of the APDU.

5.1.3.3 Identification of endpoints

The endpoints are identified by endpoint addresses. These EP addresses are numerical Addresses, taken within an AE from the same name space of numbers, to achieve a unique identification. These EP addresses are used to identify AREPs and REPs.

5.1.3.4 DL addresses

The DLE is identified from the Link by a DL-address. DL-addresses are defined in IEC 61158-4-4, but as they are part of the route, they are mentioned here. There is a one to one relation between the DL-address of a DLE and the endpoint address of the corresponding AREP.

5.1.3.5 Route

The complete Route contains a Destination-Route and a Source-Route.

The Destination-Route describes how to reach the destination REP. It is a sequence of Endpoint addresses and DL-addresses. On its way to the destination, the first element always indicates the address of the receiving DLE, AREP or REP.

The Source Route in the same way describes how to find the way back to the endpoint that initiated the request. It is a sequence of endpoint addresses and DL-addresses. On its way to the destination, the first element always indicates the address of the transmitting DLE, AREP or REP.

5.1.3.6 Destination /source route conversion

A Destination-Route conveyed by an REP is starting with the endpoint address of the local AREP. The next element is the DL-address of the receiving DLE. If the route goes through gateways a pair of addresses shall be added for each gateway, a DL-address and an AREP address. The final address is the endpoint address of the destination REP. The Source address is the endpoint address of the source REP.

The AR Send service removes the first element of the Destination-Route. The DLE inserts its own DL-address in front of the current Source Route before conveying the request to the link.

When a DLE receives an indication from the link, it removes the first element of the Destination Route (its own DL-address). The corresponding AREP inserts its own AREP address in front of the current Source Route before conveying the request to a REP or an AREP.

At the destination REP the Destination-Route holds the endpoint address of the REP, and the Source Route holds the complete route back to the requesting REP.

If any error appears with a request on its way to the destination, the current Source Route can be used for returning an error response.

5.1.4 Architecture summary

A summary of the FAL architecture is presented in Figure 14. It illustrates the major components of the FAL architecture and how they relate to each other.

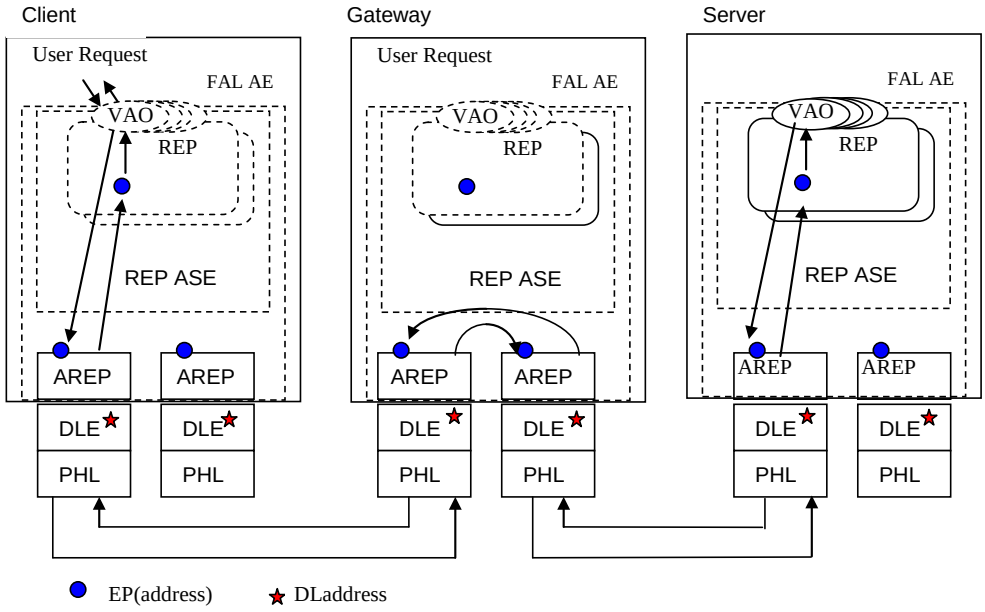


Figure 14 – Summary of the FAL architecture

5.1.5 FAL service procedures and time sequence diagrams

5.1.5.1 Overview

Subclause 5.1.5 describes service procedures for FAL confirmed and unconfirmed services. An overview of the FAL service procedures is given in Figure 15.

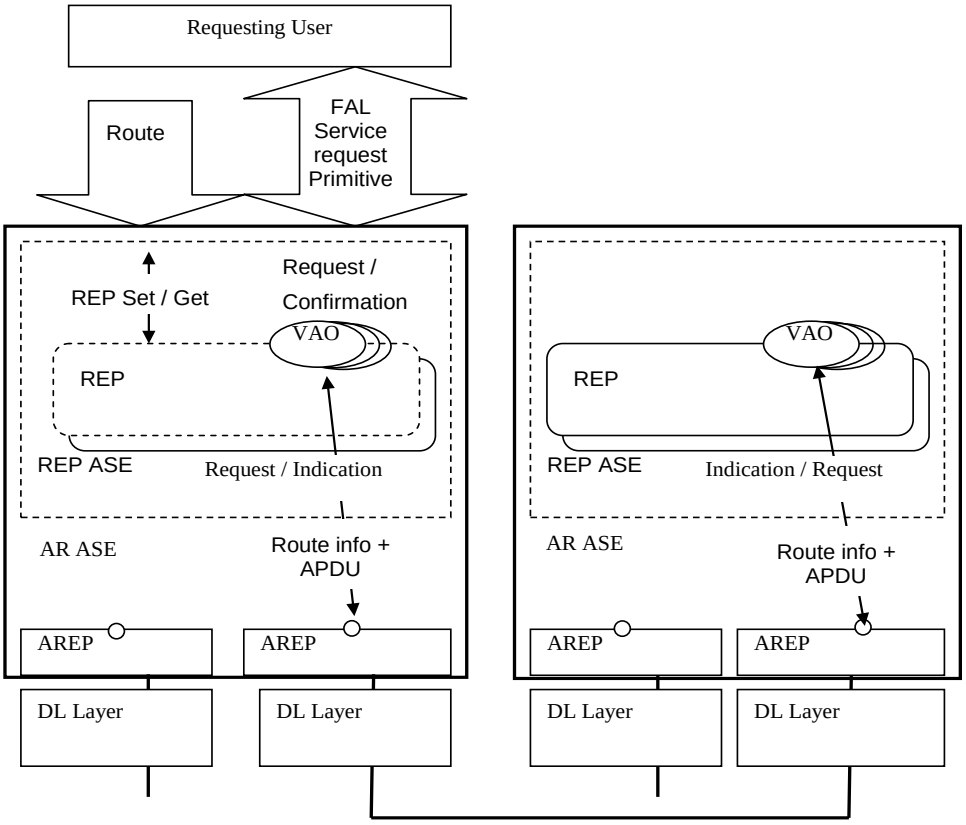


Figure 15 – FAL service procedure overview

5.1.5.2 FAL confirmed service procedures

– At requesting AP:

The user application uses the REP ASE service "Reserve REP" to reserve a REP. The user application then uses the REP ASE service "Set REP Attribute" to set attributes of the REP, including the Destination Route to the remote REP.

The requesting user submits a confirmed request primitive to the REP ASE with the REP address as a parameter. The REP ASE builds the related confirmed request APDU and conveys it on the AREP, using an AR Send request primitive. The AREP address is indicated by the first element of the Destination route. The REP ASE starts an associated timer to monitor the request.

The AREP builds a request DLSDU and submits the route info together with the DLSDU to its DLE. The DLE sends the DLSDU at the first opportunity.

– At responding server AP:

Upon receipt of the DLSDU, the receiving AREP builds an APDU and delivers the APDU and the received Route to the addressed REP by an AR Send indication primitive.

The REP ASE handles the indication by data exchange with the addressed Variable Object, builds the related unconfirmed response APDU and conveys it on the AREP, using an AR Send request primitive. The received Source route is used as Destination route for the response.

The AREP builds a request DLSDU and submits the route info, together with the DLSDU to its DLE. It is the task of the DL layer to decide whether to “queue” the DLSDU or send it immediately.

If the REP ASE is not able to process the indication and respond within "MaxIndicationDelay", the REP ASE shall submit an AR Acknowledge request primitive to the receiving AREP, and in this way free the Token. The AR Acknowledge is submitted to the DLE.

– At requesting AP receiving the response:

Upon receipt of the response DLSDU, the receiving AREP conveys the response to the REP ASE. The destination address conveyed to the REP ASE holds the endpoint address of the requesting REP. The REP object can now deliver the response to the requesting user application. This is accomplished by the user invocation of the REP ASE RESPONSE Service.

If the timer expires before the requesting REP has received the response, the REP cancels the associated transaction state machine and delivers the Error response "No answer" to the user application.

5.1.5.3 Confirmed service time sequence diagram

Figure 16 shows the time sequence diagram for the confirmed services.

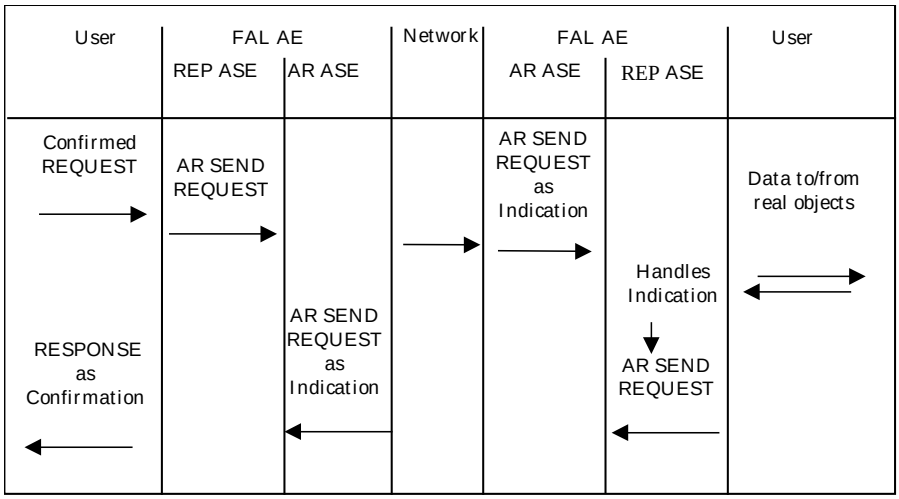


Figure 16 – Time sequence diagram for the confirmed services

5.1.5.4 Unconfirmed request service procedure

– At requesting AP:

The user application uses the REP ASE service "Reserve REP" to reserve a REP. The user application then uses the REP ASE service "Set REP Attribute" to set attributes of the REP, including the Destination Route to the remote REP.

The requesting user submits an unconfirmed request primitive to the REP ASE with the REP address as a parameter. The REP ASE builds the related unconfirmed request APDU and conveys it on the AREP, using an AR Send request primitive. The AREP address is indicated by the first element of the Destination route.

The AREP builds a request DLSDU and submits the route info together with the DLSDU to its DLE. The DLE sends the DLSDU at the first opportunity.

– At responding server AP:

Upon receipt of the DLSDU, the receiving AREP builds an APDU and delivers that and the received Route to the addressed REP by an AR Send indication primitive.

The REP ASE handles the indication by data exchange with the addressed Variable Object.

5.1.5.5 Unconfirmed service time sequence diagram

Figure 17 shows the time sequence diagram for the unconfirmed services.

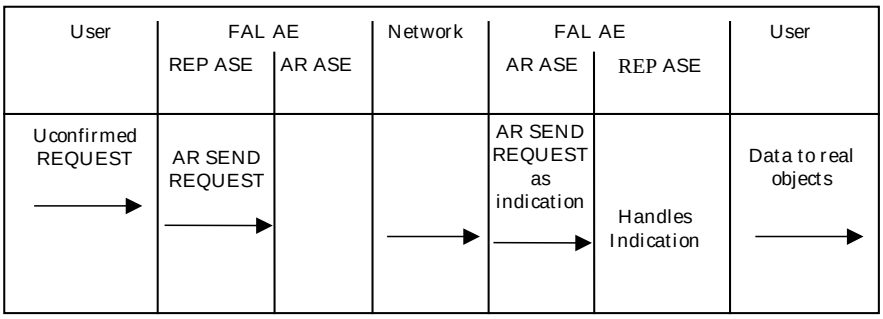


Figure 17 – Time sequence diagram for unconfirmed services

5.1.5.6 Confirmed / unconfirmed service procedure through gateways

Upon receipt of a DLSDU, the receiving AREP delivers an AR Send indication primitive to the local destination. If the endpoint address is a REP address, the APDU + Route is delivered to the REP ASE. If the endpoint address is an AREP address, the AR send service is used to pass it on to the network (Gateway).

For confirmed service Request, an AR Acknowledge is submitted to the “source AREP” unless the response is expected within “MaxIndicationDelay”. “MaxIndicationDelay" is an attribute of the "source AREP".

5.2 Variable ASE

5.2.1 Variable types

5.2.1.1 Overview

The Variable ASE specifies the machine independent syntax for application data conveyed by FAL services. The application layer supports the transfer of both basic and constructed variables. The encoding rules for basic variables and constructed variables as specified in 5.2 are provided in IEC 61158-6-4.

Variable Objects (VAOs) are defined as instances of a variable class. The Variable ASE provides access to VAOs. VAOs are identified by a numeric Variable Identifier, and are instances of a variable class.

Basic types are atomic types that cannot be decomposed into more elemental types. Constructed types are types composed of basic types and other constructed types. Their complexity and depth of nesting is free.

Constructed types are packed according to the encoding rules for constructed types. The structure of a constructed type is not network visible. The structure of a constructed type is generated by a compiler and transferred by other means.

All variable classes have the common attributes Variable Identifier, Data Type Identifier, Octet length, Read enable, Write enable and Write protected. All variable classes have the common services Read, Write, Get Variable Attribute and Set Variable Attribute.

This standard recommends using floating point representation of measurement values scaled in SI units.

5.2.1.2 Basic variable types

Basic variable types have a constant length. The length of basic variable types is an integral number of octets. Defining new basic variable types is accomplished by defining new basic variable type classes.

5.2.1.3 Constructed variable types

5.2.1.3.1 Overview

Constructed variable types are needed to completely convey the variety of information present on the fieldbus. Four kinds of constructed variable types are defined in four classes in this standard: Complex, String, BitString and FIFO. Defining new kinds of constructed variable types is accomplished by defining new constructed variable type classes.

On the fieldbus a complete constructed variable can be transferred, or a part of it, e.g. a single field of a complex variable or an element of a string. When transferring only part of a variable, addressing with offset and maybe Bit-no is used. Offset indicates the distance (in octets) from the first octet of the variable to the part to be transferred, Bit-no indicates the bit number in one octet.

5.2.1.3.2 Complex variable type

A Complex variable type defines Arrays and Structures.

An Array is composed of an ordered set of homogeneously typed elements. This standard places no restriction on the type of array elements, but it does require that each element be of the same type. Once defined, the number of elements in an Array may not be changed.

A Structure is made of an ordered set of heterogeneously typed elements called fields. Like Arrays, this standard does not restrict the type of fields. However, the fields within a Structure can be of different types.

5.2.1.3.3 String variable type

A String is composed of an ordered set of an Actual Number of Elements field and a number of homogeneously typed elements. Once defined, the maximum number of elements in a String may not be changed.

5.2.1.3.4 BitString variable type

A BitString is defined as a series of bits. Once defined, the number of elements in a BitString may not be changed. The octet length of a BitString is the integral part of ((the number of elements + 7) divided by 8).

5.2.1.3.5 FIFO variable type

A FIFO queue is composed of a set of homogeneously typed elements. This standard places no restriction on the type of FIFO elements, but it does require that each element be of the same type. The first written element will be the first element that can be read. On the fieldbus only one, complete element can be transferred as a result of one service invocation.

5.2.2 Variable model class specification

5.2.2.1 Variable model

5.2.2.1.1 Formal model

The Variable formal model defines the characteristics common to all Variable classes. This class is not capable of being instantiated. It is present only for the inheritance of its attributes and services of its subclasses.

FAL ASE:			**VARIABLE ASE**
CLASS:	VARIABLE		
CLASS ID:			1
PARENT CLASS:			TOP
ATTRIBUTES:			
1	(m)	Key Attribute:	Variable Identifier
2	(o)	Attribute:	VariableTypeIdentifier
3	(o)	Attribute:	Octet Length
4	(o)	Attribute:	Read enable
5	(o)	Attribute:	Write enable
6	(o)	Attribute:	Write protected
SERVICES:			
1	(m)	OpsService:	Read
2	(m)	OpsService:	Write
3	(o)	OpsService:	Get Variable Attribute
4	(o)	OpsService:	Set Variable Attribute

5.2.2.1.2 Attributes

1) Variable identifier

   This attribute identifies the Variable Object instantiating the variable type class.

2) VariableTypeIdentifier

   This optional attribute identifies the variable type as a Boolean, Integer8, Integer16, Integer32, Unsigned8, Unsigned16, Float32, Float64, UNICODE Char, Array, String, Structure or FIFO. The VariableTypeIdentifier is a numerical identifier, defined in IEC 61158-6-4. In this standard, a descriptive name corresponding to the identifier is stated.

3) Octet length

   This optional attribute indicates the length in octets of the data of the Variable Object. For all types except FIFO octet length indicates the total number of octets. For FIFO types, octet length indicates the number of octets in one element.

4) Read enable

   The value of this optional attribute indicates, whether the data value of the Variable Object can be read via the fieldbus.

5) Write enable

   The value of this optional attribute indicates, whether the data value of the Variable Object can be updated via the fieldbus.

6) Write protected

   The value of this optional attribute indicates, whether the data value of the Variable Object can only be updated via the fieldbus under certain conditions. The conditions for permitting update of a Write protected VAO are out of scope of this standard.

5.2.2.1.3 Services

1) Read

   This service is used to read the data value of the Variable Object.

2) Write

   This service is used to update the data value of the Variable Object.

3) Get variable attribute

   This service is used to read an attribute of the Variable Object.

4) Set variable attribute

   This service is used to update an attribute of the Variable Object.

5.2.3 Basic variable type specifications

5.2.3.1 Boolean variable type model

5.2.3.1.1 Formal model

This data type expresses a Boolean variable type with the possible values TRUE and FALSE.

FAL ASE:			**VARIABLE ASE**
CLASS:	BOOLEAN VARIABLE TYPE		
CLASS ID:			2
PARENT CLASS:			VARIABLE
ATTRIBUTES:			
SERVICES:			
1	(m)	OpsService:	And
2	(m)	OpsService:	Or
3	(m)	OpsService:	Test-And-Set

5.2.3.1.2 Attributes

Variable Objects of this class will have the following constant values of the following (parent class) attributes:

2	VariableTypeIdentifier	= Boolean
3	Octet Length	= 1

5.2.3.1.3 Services

1) And

   This service is used to update the data value of the Variable Object, by storing the result of a logical AND operation.

2) Or

   This service is used to update the data value of the Variable Object, by storing the result of a logical OR operation.

3) Test-And-Set

   This service is used to read and update the data value of the Variable Object, by returning the result of a logical AND operation, and storing the result of a logical OR operation.

5.2.3.2 Integer8 variable type model

5.2.3.2.1 Formal model

This integer type is a two’s complement binary number with a length of one octet.

FAL ASE: **VARIABLE ASE**
CLASS: INTEGER8 VARIABLE TYPE
CLASS ID: 3
PARENT CLASS: VARIABLE
ATTRIBUTES:
SERVICES:
1 (m) OpsService: And
2 (m) OpsService: Or
3 (m) OpsService: Test-And-Set

5.2.3.2.2 Attributes

Variable Objects of this class will have the following constant values of the following (parent class) attributes:
2 VariableTypeIdentifier = Integer8
3 Octet Length = 1

5.2.3.2.3 Services

1) And

   This service is used to update the data value of the Variable Object, by storing the result of a logical AND operation.

2) Or

   This service is used to update the data value of the Variable Object, by storing the result of a logical OR operation.

3) Test-And-Set

   This service is used to read and update the data value of the Variable Object, by returning the result of a logical AND operation, and storing the result of a logical OR operation.

5.2.3.3 Integer16 variable type model

5.2.3.3.1 Formal model

This integer type is a two's complement binary number with a length of two octets.

FAL ASE: **VARIABLE ASE**
CLASS: INTEGER16 VARIABLE TYPE
CLASS ID: 4
PARENT CLASS: VARIABLE
ATTRIBUTES:
SERVICES:
1 (m) OpsService: And
2 (m) OpsService: Or

5.2.3.3.2 Attributes

Variable Objects of this class will have the following constant values of the following (parent class) attributes:
2 VariableTypeIdentifier = Integer16
3 Octet Length = 2

5.2.3.3.3 Services

1) And

   This service is used to update the data value of the Variable Object, by storing the result of a logical AND operation.

2) Or

   This service is used to update the data value of the Variable Object, by storing the result of a logical OR operation.

5.2.3.4 Integer32 variable type model

5.2.3.4.1 Formal model

This integer type is a two's complement binary number with a length of four octets.

FAL ASE:			**VARIABLE ASE**
CLASS:	INTEGER32 VARIABLE TYPE		
CLASS ID:			5
PARENT CLASS:			VARIABLE
ATTRIBUTES:			
SERVICES:			
1	(m)	OpsService:	And
2	(m)	OpsService:	Or

5.2.3.4.2 Attributes

Variable Objects of this class will have the following constant values of the following (parent class) attributes:

2	VariableTypeIdentifier	= Integer32
3	Octet Length	= 4

5.2.3.4.3 Services

1) And

   This service is used to update the data value of the Variable Object, by storing the result of a logical AND operation.

2) Or

   This service is used to update the data value of the Variable Object, by storing the result of a logical OR operation.

5.2.3.5 Unsigned8 variable type model

5.2.3.5.1 Formal model

This integer type is a binary number with a length of one octet. The most significant bit is always used as the most significant bit of the binary number; no sign bit is included.

FAL ASE:			**VARIABLE ASE**
CLASS:	UNSIGNED8 VARIABLE TYPE		
CLASS ID:			6
PARENT CLASS:			VARIABLE
ATTRIBUTES:			
SERVICES:			
1	(m)	OpsService:	And
2	(m)	OpsService:	Or
3	(m)	OpsService:	Test-And-Set

5.2.3.5.2 Attributes

Variable Objects of this class will have the following constant values of the following (parent class) attributes:

2 VariableTypeIdentifier = Unsigned8

3 Octet Length = 1

5.2.3.5.3 Services

1) And

   This service is used to update the data value of the Variable Object, by storing the result of a logical AND operation.

2) Or

   This service is used to update the data value of the Variable Object, by storing the result of a logical OR operation.

3) Test-And-Set

   This service is used to read and update the data value of the Variable Object, by returning the result of a logical AND operation, and storing the result of a logical OR operation.

5.2.3.6 Unsigned16 variable type model

5.2.3.6.1 Formal model

This integer type is a binary number with a length of two octets. The most significant bit of the most significant octet is always used as the most significant bit of the binary number; no sign bit is included.

FAL ASE:			**VARIABLE ASE**
CLASS:	UNSIGNED16 VAIABLE TYPE		
CLASS ID:			7
PARENT CLASS:			VARIABLE
ATTRIBUTES:			
SERVICES:			
1	(m)	OpsService:	And
2	(m)	OpsService:	Or

5.2.3.6.2 Attributes

Variable Objects of this class will have the following constant values of the following (parent class) attributes:

2 VariableTypeIdentifier = Unsigned16

3 Octet Length = 2

5.2.3.6.3 Services

1) And

   This service is used to update the data value of the Variable Object, by storing the result of a logical AND operation.

2) Or

   This service is used to update the data value of the Variable Object, by storing the result of a logical OR operation.

5.2.3.7 Float32 variable type model

5.2.3.7.1 Formal model

This type has a length of four octets. The format for Float32 is that defined by ISOIEC/IEEE 60559 as single precision.

FAL ASE: **VARIABLE ASE**
CLASS: FLOAT32 VARIABLE TYPE
CLASS ID: 8
PARENT CLASS: VARIABLE
ATTRIBUTES:
SERVICES:

5.2.3.7.2 Attributes

Variable Objects of this class will have the following constant values of the following (parent class) attributes:

2 VariableTypeIdentifier = Float32
3 Octet Length = 4

5.2.3.8 Float64 variable type model

5.2.3.8.1 Formal model

This type has a length of eight octets. The format for Float64 is that defined by ISO/IEC/IEEE 60559 as double precision.

FAL ASE: **VARIABLE ASE**
CLASS: FLOAT64 VARIABLE TYPE
CLASS ID: 9
PARENT CLASS: VARIABLE
ATTRIBUTES:
SERVICES:

5.2.3.8.2 Attributes

Variable Objects of this class will have the following constant values of the following (parent class) attributes:

2 VariableTypeIdentifier = Float64
3 Octet Length = 8

5.2.3.9 UNICODE char variable type model

5.2.3.9.1 Formal model

This type has a length of two octets. It is defined as a single UNICODE character.

FAL ASE: **VARIABLE ASE**
CLASS: UNICODE CHAR VARIABLE TYPE
CLASS ID: 10
PARENT CLASS: VARIABLE
ATTRIBUTES:
SERVICES:

5.2.3.9.2 Attributes

Variable Objects of this class will have the following constant values of the following (parent class) attributes:

2 VariableTypeIdentifier = UNICODE Char
3 Octet Length = 2

5.2.4 Constructed variable type specifications

5.2.4.1 Complex variable type model

5.2.4.1.1 Formal model

A Complex Variable is an Array or a Structure.

An Array is composed of an ordered set of homogeneously typed elements. This standard places no restriction on the type of array elements, but it does require that each element be of the same type. Once defined, the number of elements in an Array may not be changed.

A Structure is made of an ordered set of heterogeneously typed elements called fields. Like Arrays, this standard does not restrict the type of fields. However, the fields within a Structure can be of different types.

FAL ASE:			**VARIABLE ASE**
CLASS:	COMPLEX VARIABLE TYPE		
CLASS ID:			11
PARENT CLASS:			VARIABLE
ATTRIBUTES:			
SERVICES:			
1	(m)	OpsService:	And
2	(m)	OpsService:	Or
3	(m)	OpsService:	Test-And-Set

5.2.4.1.2 Attributes

Variable Objects of this class will have the following constant value of the following (parent class) attribute:

2 VariableTypeIdentifier = Complex

5.2.4.1.3 Services

1) And

   This service is used to update the data value of the Variable Object, by storing the result of a logical AND operation. The operation is legal on subelements of integer or boolean type only.

2) Or

   This service is used to update the data value of the Variable Object, by storing the result of a logical OR operation. The operation is legal on subelements of integer or boolean type only.

3) Test-And-Set

   This service is used to read and update the data value of the Variable Object, by returning the result of a logical AND operation, and storing the result of a logical OR operation. The operation is legal on subelements of Integer8, Unsigned8 or boolean type only.

5.2.4.2 String variable type model

5.2.4.2.1 Formal model

A String is composed of an ordered set of an Actual Number of Elements field and a number of homogeneously typed elements. Once defined, the maximum number of elements in a String may not be changed.

FAL ASE:			**VARIABLE ASE**
CLASS:	STRING VARIABLE TYPE		
CLASS ID:			12
PARENT CLASS:			VARIABLE
ATTRIBUTES:			
SERVICES:			
1	(m)	OpsService:	And
2	(m)	OpsService:	Or
3	(m)	OpsService:	Test-And-Set

5.2.4.2.2 Attributes

Variable Objects of this class will have the following constant value of the following (parent class) attribute:

2 VariableTypeIdentifier = String

5.2.4.2.3 Services

1) And

   This service is used to update the data value of the Variable Object, by storing the result of a logical AND operation. The operation is legal on subelements of integer or boolean type only.

2) Or

   This service is used to update the data value of the Variable Object, by storing the result of a logical OR operation. The operation is legal on subelements of integer or boolean type only.

3) Test-And-Set

   This service is used to read and update the data value of the Variable Object, by returning the result of a logical AND operation, and storing the result of a logical OR operation. The operation is legal on subelements of Integer8, Unsigned8 or boolean type only.

5.2.4.3 BitString variable type model

5.2.4.3.1 Formal model

A BitString is defined as a series of bits. Once defined, the number of bits in a BitString may not be changed. The octet length of a BitString is the integral part of ((the number of bits + 7) divided by 8).

FAL ASE:			**VARIABLE ASE**
CLASS:	BITSTRING VARIABLE TYPE		
CLASS ID:			13
PARENT CLASS:			VARIABLE
ATTRIBUTES:			
SERVICES:			
1	(m)	OpsService:	And
2	(m)	OpsService:	Or
3	(m)	OpsService:	Test-And-Set

5.2.4.3.2 Attributes

Variable Objects of this class will have the following constant value of the following (parent class) attribute:

2 VariableTypeIdentifier = BitString

5.2.4.3.3 Services

1) And

   This service is used to update the data value of the Variable Object, by storing the result of a logical AND operation. The operation is legal on a subelement (which is a bit) only.

2) Or

   This service is used to update the data value of the Variable Object, by storing the result of a logical OR operation. The operation is legal on a subelement (which is a bit) only.

3) Test-And-Set

   This service is used to read and update the data value of the Variable Object, by returning the result of a logical AND operation, and storing the result of a logical OR operation. The operation is legal on a subelement (which is a bit) only.

5.2.4.4 FIFO variable type model

5.2.4.4.1 Formal model

A FIFO queue is composed of a set of homogeneously typed elements. This standard places no restriction on the type of FIFO elements, but it does require that each element be of the same type. The first written element will be the first element that can be read. On the fieldbus only one, complete element can be transferred as a result of one service invocation.

FAL ASE:			**VARIABLE ASE**
CLASS:	FIFO VARIABLE TYPE		
CLASS ID:			14
PARENT CLASS:			VARIABLE
ATTRIBUTES:			
1	(m)	Attribute:	Next Element In
2	(m)	Attribute:	Next Element Out
3	(m)	Attribute:	Free elements
4	(m)	Attribute:	Used elements
5	(o)	Attribute:	ReRead
6	(o)	Attribute:	ReWrite

5.2.4.4.2 Attributes

Variable Objects of this class will have the following constant value of the following (parent class) attribute:

2 VariableTypeIdentifier = FIFO

1) Next element in

   Next Element In is an Integer16, which holds the index number for the next element to be written to the FIFO.

2) Next element out

   Next Element Out is an Integer16, which holds the index number for the next element to be read from the FIFO.

3) Free elements

   This attribute is an Integer16 which holds the current number of free elements in the FIFO (= 0 if full).

4) Used elements

This attribute is an Integer16 which holds the current number of used elements in the FIFO (= 0 if empty).

5) Reread

Reread is a Boolean. When the value of Reread is TRUE, a Read will return the previously read element, instead of returning and removing the element indexed by Next Element Out. This facility can be used if a transmission error resulted in a faulty read of an element.

6) Rewrite

Rewrite is a Boolean. When the value of Rewrite is TRUE, a Write will overwrite the previously written element, instead of inserting a new element. This facility can be used if a transmission error resulted in a faulty write of an element.

5.2.4.4.3 Services

5.2.4.4.3.1 Overview

The services defined in the Variable ASE provide access to attributes and data of Variable Objects. The services are invoked by a REP ASE REQUEST Service invocation. The REP ASE REQUEST Service conveys the Variable ASE Service parameters. Refer to 5.3.3 for the documentation of these parameters.

5.2.4.4.3.2 Read

This service is used to read (part of) the specified variable. No data is passed with the request. The value of the specified variable is returned in the response.

5.2.4.4.3.3 Write

This service is used to update (part of) the specified variable. The data passed with the request is written into (part of) the specified variable. No data is returned in the response.

5.2.4.4.3.4 And

This service performs a logical AND of (part of) the specified variable and the data passed with the request. The result is stored in (part of) the variable. The service is used to clear individual bits in a variable without first reading the variable and afterwards writing the result back to the variable. No data is returned in the response.

5.2.4.4.3.5 Or

This service performs a logical OR of (part of) the specified variable and the data passed with the request. The result is stored in (part of) the variable. The service is used to set individual bits in a variable without first reading the variable and afterwards writing the result back to the variable. No data is returned in the response.

5.2.4.4.3.6 Test-And-Set

This service performs a logical AND of (part of) the specified variable and the data passed in the request. The result of the AND operation is returned in the response. Then a logical OR operation of (part of) the specified variable and the data passed in the request is performed. The result of the OR operation is stored in (part of) the variable. The service is used to set individual bits in a variable, and in the same transmission read their state prior to the operation.

5.2.4.4.3.7 Get variable attribute

This service is used to read an attribute of the specified Variable Object. The attribute is specified by the offset parameter.

5.2.4.4.3.8 Set variable attribute

This service is used to update an attribute of the specified Variable Object. The attribute is specified by the offset parameter.

5.2.5 Route endpoint ASE

5.2.5.1 Overview

In the fieldbus environment, application processes contain data that remote applications are able to read and write. The data is accessed by means of the REP ASE. This ASE offers services to user applications in clients. The services REQUEST and RESPONSE give the access to real Variable Objects (VAOs) located in a server. In the server the REP ASE conveys data to / from the real Variable Object and returns responses. The services Get REP Attribute, Set REP Attribute, Reserve REP and Free REP are all for local use.

5.2.5.1.1 Route endpoint Description

The Route Endpoint class is defined to support the on-demand exchange of confirmed and unconfirmed services between two application processes. It uses AR ASE services for conveying APDUs.

An REP is an object container, containing Variable Objects of a variable type. It performs the access to variable objects. The Variable Object ID is unique within an REP. The request service addresses a Variable Object within the REP. The Variable Object ID is one of the request service parameters.

An REP in the proxy container role represents a remote REP in a Server. The remote REP is in the real container role. The Destination Route attribute of the REP in the client holds the Route to the real container REP, which it represents.

The REP ASE handles the segmentation when the data length exceeds the MaxDataSize.

When data is segmented, the order of segments is indicated by the offset part of the APDU.

To initiate access of a proxy Variable Object, the user application uses Reserve REP to get a REP currently not in use.

The attributes of the REP are set, including the Destination Route to the remote REP. This is done by the Set REP Attribute service. The requesting user then issues a REQUEST primitive with the REP ID and a Variable Object ID as parameters. The REP ASE builds a request APDU, and passes that along with Route Info in an AR Send request.

Following a successful transmission, if the REQUEST was confirmed, the AREP receives a response APDU and conveys that to the requesting REP by an AR Send indication. The user application can now read the result by issuing a RESPONSE primitive, with the REP ID as a parameter.

5.2.5.2 Route endpoint model

Formal Model

FAL ASE:			**Route Endpoint ASE**
CLASS:	Route Endpoint		
CLASS ID:			1
PARENT CLASS:			TOP
ATTRIBUTES:			
1.	(m)	Key Attribute:	Endpoint Address
2.	(m)	Attribute:	Role (Proxy object / Real object)
3.	(m)	Attribute:	REP State
4.	(m)	Attribute:	Priority
5.	(m)	Attribute:	Confirmation
6.	(m)	Attribute:	Destination Route
7.	(m)	Attribute:	Source route
8.	(m)	Attribute:	Progress
9.	(m)	Attribute:	Capabilities
10.	(m)	Attribute:	Flat addressing
SERVICES:			
1.	(m)	OpsService:	REQUEST
2.	(m)	OpsService:	RESPONSE
3.	(m)	OpsService:	Reserve REP
4.	(m)	OpsService:	Free REP
5.	(m)	OpsService:	Get REP Attribute
6.	(m)	OpsService:	Set REP Attribute

5.2.5.2.1 Attributes

5.2.5.2.1.1 Endpoint address

This key attribute holds the Endpoint address identifying the REP.

5.2.5.2.1.2 ROLE

This attribute specifies the role of the REP. The valid values are as follows.

Proxy container	Endpoints of this type are used for sending requests to servers and receive responses from them. The objects contained in REPs of this role are proxy objects for the real objects in the server.
Real container	Endpoints of this type are used for receiving confirmed and unconfirmed requests from clients and sending responses to them. The objects contained in REPs of this role are the real Variable Objects.

5.2.5.2.1.3 REP State

This attribute indicates the state of the REP. The values for this attribute are: IDLE RESERVED, WAITING FOR RESPONSE, RESPONSE RECEIVED or NOT IN USE.

5.2.5.2.1.4 Priority

The DLL may provide the possibility to send high priority APDUs before APDUs of lower priority. The priority only refers to the request on the local link, not requests passing gateways and not the response.

5.2.5.2.1.5 Confirmation

This attribute indicates whether the request has to be to confirmed or unconfirmed. The response is always returned unconfirmed. If the Destination Route contains one or more broadcast addresses (126) this attribute shall be set to unconfirmed.

5.2.5.2.1.6 Destination route

The Destination Route describes the Route to the destination REP. It is a sequence of Endpoint addresses and DL-addresses. On its way to the destination, the first part always indicates the address of the next DLE, AREP or REP to receive the APDU. In a request, the Destination Route holds the Route to the REP to respond. In a response, the Destination Route holds the Route to the requesting REP.

5.2.5.2.1.7 Source route

The Source Route describes the Route to the source REP. It is a sequence of endpoint addresses and DL-addresses. On its way to the destination, the first element always indicates the address of the DLE, AREP or REP endpoint from where the APDU was conveyed. In a request, the Source Route holds the Route to the requesting REP. In a response, the Source Route holds the Route to the responding REP.

5.2.5.2.1.8 Progress

This attribute is only relevant in proxy container REPs. It indicates the progress of a request. It indicates number of segments that has been delivered divided with the total number of segments. This means for non-segmented requests the value is zero while waiting for the response, and one when the response is received. For segmented requests, the value will start from zero and gradually increase till it reaches the value of one.

5.2.5.2.1.9 Capabilities

This attribute defines the capabilities of the REP addressed by the Destination Route. It is a local attribute, which shall be set up by the user application, to reflect the capabilities of the REP containing the real Variable Objects. The value of Capabilities is used by the requesting REP to build the APDU. The attribute indicates, whether the responding REP is capable of handling bit addressing or not.

5.2.5.2.1.10 Flat addressing

This attribute indicates, whether the REP should be seen as a container of individual Variable Objects, or as a container of one, flat memory area. If Flat addressing is selected, the Variable Object ID parameter of the REQUEST service indicates the offset in octets from the beginning of the memory area.

5.2.6 Route endpoint ASE service specification

5.2.6.1 REQUEST service

5.2.6.1.1 Overview

This service is used to initiate access of the value of a Variable Object. It may be used for as well reading as updating the value. The result is the operation (if any) is retrieved by the RESPONSE service.

5.2.6.1.2 Service primitives

The service parameters for this service are shown in Table 1. The Indication parameters are indicating the parameters in the APDU received by the remote REP. These parameters are for local use within the remote REP and its variable objects.

Table 1 – REQUEST service parameters

Parameter name	Req	Ind	Cnf
Argument	M	M	
REP	M	M	
Variable object ID	M	M(=)	
Variable Service	M	M(=)	
Data length	M	M(=)	
Offset/Attribute	C	C(=)	
Bit-no	C	C(=)	
Data	C	C(=)	
Result			
Status			M

– Argument

The argument carries the parameters of the REQUEST service invocation.

– REP

This parameter is the EP address of the REP holding the route information.

– Variable ID

This parameter identifies the Variable Object. The Variable ID indicates the Variable Object on which the service is to be performed.

– Variable service

This parameter holds the Variable service to be performed. (Read, Write, And, Or, Test-And-Set, Get attribute and Set Attribute).

– Data length

This parameter indicates the number of octets of data to be transferred.

– Offset/attribute

This parameter is used when accessing only part of a structured variable or an Attribute of the variable. When accessing part of a structured variable, it indicates the offset in octets to the starting octet of the data block to be transferred, relative to the first octet of the variable. When accessing attributes, it identifies the attribute to be accessed.

– Bit-no

This parameter is used to select a bit within a BitString. It indicates the bit number (1-8) within an octet. The Offset/Attribute parameter is used to select the octet.

– Data

This parameter holds the data to be transferred.

– Status

As a result of the request, Status is returned indicating OK or FAILURE.

5.2.6.2 RESPONSE service

5.2.6.2.1 Overview

This service is used to return the result of a confirmed REQUEST to the requesting user application in a requester.

5.2.6.2.2 Service primitives

The service parameters for this service are shown in Table 2. The response parameters are for local use within the responding REP and its variable object.

Table 2 – RESPONSE service parameters

Parameter name	Rsp	Cnf
Argument	M	M
REP	M	M
Result		
Variable Service	M	M
Data length	M	M(=)
Error status	M	M(=)
Data	C	C(=)

– Argument

The argument carries the parameters of the service request.

– REP

This parameter holds the ID of the REP holding the route information.

– Variable service

This parameter holds the Variable service that has been performed. (Read, Write, And, Or, Test and Set, Get attribute and Set Attribute).

– Data length

This parameter indicates the number of octets of data that have been transferred.

– Error status

This parameter holds error information for errors occurred in any layer local or remote. For details, see Table 3.

The error code can be generated anywhere on the route between client and server, or by the responding Variable Object.

Table 3 – Error codes by source

Error description	Client	Gateway	Server
User Application:			
Instruction Error	X		X
Info length error	X		X
FIFO full or empty			X
Write protection			X
Data format error			X
Variable Object ID error			X
Time Out	X		
Actual data error			X
Historical data error			X
FAL:			
Route error	X	X	X
DL-layer:			
No Response	X	X	
WAIT TOO LONG	X	X	
Out of sync	X	X	
CRC ERROR	X	X	
DLE not Client	X	X	
Physical Layer:			
NET SHORTCIRCUIT	X	X	
OVERRUN/FRAMING ERROR	X	X	
RS-232 HANDSHAKE ERROR	X	X	X

– Data

This parameter holds the data that has been transferred.

5.2.6.3 Reserve REP service

5.2.6.3.1 Overview

This service is used by the user application to reserve an REP currently not in use, and to get the ID of the reserved REP.

5.2.6.3.2 Service primitives

The service parameters for this service are shown in Table 4.

Table 4 – Reserve REP service parameters

Parameter name	Req	Cnf
Result		
REP		M

– REP

This parameter holds the ID of the reserved REP. If no REP is available, zero is returned.

5.2.6.4 Free REP Service

5.2.6.4.1 Overview

This service is used by the user application to free an REP by changing the state of the REP to “not in use”.

5.2.6.4.2 Service primitives

The service parameters for this service are shown in Table 5.

Table 5 – Free AREP service parameters

Parameter name	Req
Argument	M
REP	M

– Argument

The argument carries the parameters of the service request.

– REP

This parameter holds the EP address of the reserved REP to free.

5.2.6.5 Get REP attribute service

5.2.6.5.1 Overview

This service is used by the user application to read an attribute of a REP locally.

5.2.6.5.2 Service Primitives

The service parameters for this service are shown in Table 6.

Table 6 – Get REP attribute service parameters

Parameter name	Req	Cnf
Argument	M	
REP	M	
Attribute index	M	
Result (+)		
Value		C
Result (-)		
Status		C

– Argument

The argument carries the parameters of the service request.

– REP

This parameter specifies the REP.

– Attribute index

This parameter identifies the attribute to read.

– Value

This parameter returns the value of the specified attribute.

– Status

This parameter returns error information if the service fails. The possible values are: Illegal attribute index, Illegal REP address.

5.2.6.6 Set REP attribute service

5.2.6.6.1 Overview

This service is used by the user application to write a value into an attribute of a REP locally.

5.2.6.6.2 Service primitives

The service parameters for this service are shown in Table 7.

Table 7 – Set REP attribute service parameters

Parameter name	Req	Cnf
Argument	M	
REP	M	
Attribute index	M	
Result		
Status		M

– Argument

The argument carries the parameters of the service request.

– REP

This parameter specifies the REP.

– Attribute index

This parameter identifies the attribute to update within the specified REP.

– Value

This parameter holds the value to be written to the specified attribute.

– Status

As a result of the request, Status is returned indicating OK or the reason for failure. The possible values are: OK, Illegal attribute index, Illegal value, Illegal REP address.

5.3 Application relationship ASE

5.3.1 Overview

To support access to the remote AP, the Application Relationship ASE is defined. It provides services to the AP for accessing communication-related parameters, and it provides services to the REP ASE for conveying service requests and responses.

The AR ASE provides services at the endpoints of ARs (AREPs).

5.3.2 Application relationship class specification

5.3.2.1 Application relationship formal model

The functionality of the AR ASE is described in 5.1.

The application ASE defines one class, the AREP class.

FAL ASE:			**Application Relationship ASE**
CLASS:	**AREP**		
CLASS ID:			1
PARENT CLASS:			TOP
ATTRIBUTES:			
1.	(m)	Key Attribute:	Endpoint Address
2.	(m)	Attribute:	Role (Client, Server, Peer)
3.	(m)	Attribute:	DLL Reference
4.	(m)	Attribute:	MaxPDUSize
5.	(m)	Attribute:	MaxDataSize
6.	(m)	Attribute:	Acknowledgement
7.	(m)	Attribute:	MaxIndicationDelay
8.	(m)	Attribute:	Local DLE address
9.	(o)	Attribute:	MaxRetryTime
10.	(o)	Attribute:	MaxRetries
11.	(o)	Attribute:	MaxOutstandingRequests
12.	(o)	Attribute:	BaudRate
13.	(o)	Attribute:	NumberOfClientDLEs

5.3.2.1.1.1 SERVICES:

1	(m)	OpsService:	AR-Get Attribute
2	(m)	OpsService:	AR-Set Attribute
3	(m)	OpsService:	AR-Send
4	(m)	OpsService:	AR-Acknowledge

5.3.2.1.1.2 Endpoint address

This attribute specifies the numeric identifier of the AREP. It is used by the FAL to select the AREP, and implicitly the DLE for a request.

5.3.2.1.1.3 Role

This attribute specifies the role of the AREP. The valid values are as follows.

Client Endpoints of this type send confirmed and unconfirmed Request APDUs to servers and receive Response APDUs.

Server Endpoints of this type receive confirmed and unconfirmed Request APDUs from clients and send unconfirmed Response APDUs.

Peer Endpoints of this type act as both Clients and Servers.

5.3.2.1.1.4 MaxPDUSize

This attribute specifies the maximum PDU size that can be sent by the DLL.

5.3.2.1.1.5 MaxDataSize

This attribute specifies the maximum data size that can be sent by the DLL. If the data length exceeds MaxDataSize, the VARIABLE ASE shall segment the request.

5.3.2.1.1.6 DLL reference

This attribute contains the necessary context to convey the DLL relations.

5.3.2.1.1.7 Acknowledgement

This attribute describes the type of Acknowledgement to be used by DLL: Acknowledged or Unacknowledged transfer of Confirmed APDUs, and Acknowledged or Unacknowledged transfer of Unconfirmed APDUs.

5.3.2.1.1.8 MaxIndicationDelay

This attribute indicates to the user application, how long time a Variable Object can use to prepare a response after receiving an indication requiring that. If the Variable Object is unable to prepare a response within MaxIndicationDelay, it shall issue an AR-Acknowledge. The value of MaxIndicationDelay is calculated by the DLE.

5.3.2.1.1.9 Local DLE address

This attribute reflects the DL address of the related DLE. In some situations it may be a read-only attribute.

5.3.2.1.1.10 MaxRetryTime

This attribute indicates the maximum time the DLE should try to re-transmit a request as a result of Acknowledge responses from the remote Variable Object.

5.3.2.1.1.11 MaxRetries

This attribute indicates the maximum number of re-transmissions carried out by the DLE as a result of transmission errors.

5.3.2.1.1.12 MaxOutstandingRequests

This attribute indicates the maximum number of requests that the AREP may initiate without receiving the related responses.

5.3.2.1.1.13 BaudRate

This attribute specifies the baud rate used by the physical layer. It is conveyed to/from the physical layers by the DLEs.

5.3.2.1.1.14 NumberOfClientDLEs

This attribute only relates to a DLE using the P-NET protocol in the client role. It specifies the number of participants in a token round.

5.3.3 Application relationship ASE service specifications

5.3.3.1 Overview

Subclause 5.3.3 contains the definition of the services that are unique to this ASE. The services are the following:

AR Send

AR Acknowledge

AR Get Attribute

AR Set Attribute

5.3.3.2 Common parameter definition

Parameters used in more AR ASE services are defined below:

5.3.3.2.1.1 Route info

The complete DL Route contains the following elements:

5.3.3.2.1.2 Destination route

The Destination-Route describes how to reach the destination REP. It is a sequence of Endpoint addresses and DL-addresses. On its way to the destination, the first part always indicates the address of the next DLE, AREP or REP to receive the APDU.

5.3.3.2.1.3 Source route

The Source Route in the same way describes how to find the way back to the endpoint that initiated the request. It is a sequence of endpoint addresses and DL-addresses. On its way to the destination, the first element always indicates the address of the DLE, AREP or REP endpoint from where the APDU was sent.

5.3.3.2.1.4 Priority

The underlying layer may provide the possibility to send high-priority APDUs before APDUs of lower priority. This priority only refers to the request on the local link, not requests passing gateways and not the response.

5.3.3.2.1.5 Confirmation

This parameter indicates whether the request has to be to confirmed or unconfirmed. A response is always returned unconfirmed. If the destination route contains one or more broadcast addresses (126) this attribute shall be set to unconfirmed.

5.3.3.2.1.6 APDU header

The APDU Header parameter holds a Control/Status subfield indicating the Variable Service, an APDU format subfield, indicating whether the APDU holds an offset, and finally an APDU length subfield, indicating the octet length of the APDU.

5.3.3.2.1.7 APDU body

This parameter holds the APDU Body to send or receive.

5.3.3.3 AR send service

5.3.3.3.1 Overview

The AR Send service is used to send confirmed or unconfirmed APDUs from one AREP to another.

It is the task of the receiving AREP to convey the APDU to the Endpoint specified by the first element in the destination route. In the gateway situation the AR is using this AR Send service to convey the APDU to an AREP.

5.3.3.3.2 Service primitives

The service parameters for this service are shown in Table 8.

Table 8 – AR send service parameters

Parameter name	Req	Ind	Cnf
Argument	M	M	
Route info	M	M	
APDU Header	M	M(=)	
APDU Body	C	C(=)	
Result			
Status			M

– Argument

The argument carries the parameters of the service request.

– Status

As a result of the request, Status is returned indicating OK or the locally detected reason for failure. Status only refers to local conditions. The possible values are: OK, Route Error.

5.3.3.4 AR acknowledge service

5.3.3.4.1 Overview

When the Variable ASE in a confirmed request is not able to process the indication and respond within the time limit, “MaxIndicationDelay”, it shall submit an AR Send Acknowledge request primitive to the local AREP, and in this way free the Token. “MaxIndicationDelay” is an attribute of the AREP.

5.3.3.4.2 Service Primitives

The service parameters for this service are shown in Table 9.

Table 9 – AR acknowledge service parameters

Parameter name	Req
Argument	M
Route info	M

– Argument

The argument carries the parameters of the service request.

AR get attribute service

5.3.3.4.3 Overview

This confirmed service is used to read the value of attributes of an AREP locally.

5.3.3.4.4 Service parameters

The service parameters for the AR Get Attribute Service are shown in Table 10.

Table 10 – AR get attributes service parameters

Parameter name	Req	Cnf
Argument	M	
AREP	M	
Attribute index	M	
Result (+)		S
Value		C
Result (-)		S
Status		C

– Argument

The argument carries the parameters of the service request.

– AREP

This parameter identifies the AREP by its endpoint address.

– Attribute index

This parameter identifies the attribute.

– Result (+)

This selection type parameter indicates that the request succeeded.

– Value

This parameter is returned with the requested attribute value if the request succeeded.

– Result (-)

This selection type parameter indicates that the request failed.

– Status

If the request failed, Status is returned indicating the reason for failure. The possible values are: Illegal attribute index, Illegal AREP address.

5.3.3.5 AR set attributes service

This confirmed service is used to set the current value of attributes of an AREP locally.

5.3.3.5.1 Service parameters

The service parameters for the AR Set Attribute Service are shown in Table 11.

Table 11 – AR set attributes service parameters

Parameter name	Req	Cnf
Argument	M	
AREP	M	
Attribute index	M	
Value	M	
Result		
Status		M

– Argument

  The argument carries the parameters of the service request.

– AREP

  This parameter identifies the AREP by its endpoint address. The legal values are 1 to 16 inclusive.

– Attribute index

  This parameter identifies the attribute to be updated.

– Value

  This parameter holds the new attribute value.

– Status

  As a result of the request, Status is returned indicating OK or the reason for failure. The possible values are: OK, Illegal attribute index, Illegal value, Illegal AREP address.

Bibliography

IEC 61158-1:2014, *Industrial communication networks – Fieldbus specifications – Part 1: Overview and guidance for the IEC 61158 and IEC 61784 series*

IEC 61784-1:2014, *Industrial communication networks – Profiles – Part 1: Fieldbus profiles*

IEC 61784-2:2014, *Industrial communication networks – Profiles – Part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC 8802-3*

_____

SOMMAIRE

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

_____

RÉSEAUX DE COMMUNICATION INDUSTRIELS – SPÉCIFICATIONS DES BUS DE TERRAIN –

Partie 5-4: Définition des services de la couche application – Eléments de type 4

AVANT-PROPOS

1) La Commission Electrotechnique Internationale (CEI) est une organisation mondiale de normalisation composée de l'ensemble des comités électrotechniques nationaux (Comités nationaux de la CEI). La CEI a pour objet de favoriser la coopération internationale pour toutes les questions de normalisation dans les domaines de l'électricité et de l'électronique. A cet effet, la CEI – entre autres activités – publie des Normes internationales, des Spécifications techniques, des Rapports techniques, des Spécifications accessibles au public (PAS) et des Guides (ci-après dénommés "Publication(s) de la CEI"). Leur élaboration est confiée à des comités d'études, aux travaux desquels tout Comité national intéressé par le sujet traité peut participer. Les organisations internationales, gouvernementales et non gouvernementales, en liaison avec la CEI, participent également aux travaux. La CEI collabore étroitement avec l'Organisation Internationale de Normalisation (ISO), selon des conditions fixées par accord entre les deux organisations.
2) Les décisions ou accords officiels de la CEI concernant les questions techniques représentent, dans la mesure du possible, un accord international sur les sujets étudiés, étant donné que les Comités nationaux de la CEI intéressés sont représentés dans chaque comité d'études.
3) Les Publications de la CEI se présentent sous la forme de recommandations internationales et sont agréées comme telles par les Comités nationaux de la CEI. Tous les efforts raisonnables sont entrepris afin que la CEI s'assure de l'exactitude du contenu technique de ses publications; la CEI ne peut pas être tenue responsable de l'éventuelle mauvaise utilisation ou interprétation qui en est faite par un quelconque utilisateur final.
4) Dans le but d'encourager l'uniformité internationale, les Comités nationaux de la CEI s'engagent, dans toute la mesure possible, à appliquer de façon transparente les Publications de la CEI dans leurs publications nationales et régionales. Toutes divergences entre toutes Publications de la CEI et toutes publications nationales ou régionales correspondantes doivent être indiquées en termes clairs dans ces dernières.
5) La CEI elle-même ne fournit aucune attestation de conformité. Des organismes de certification indépendants fournissent des services d'évaluation de conformité et, dans certains secteurs, accèdent aux marques de conformité de la CEI. La CEI n'est responsable d'aucun des services effectués par les organismes de certification indépendants.
6) Tous les utilisateurs doivent s'assurer qu'ils sont en possession de la dernière édition de cette publication.
7) Aucune responsabilité ne doit être imputée à la CEI, à ses administrateurs, employés, auxiliaires ou mandataires, y compris ses experts particuliers et les membres de ses comités d'études et des Comités nationaux de la CEI, pour tout préjudice causé en cas de dommages corporels et matériels, ou de tout autre dommage de quelque nature que ce soit, directe ou indirecte, ou pour supporter les coûts (y compris les frais de justice) et les dépenses découlant de la publication ou de l'utilisation de cette Publication de la CEI ou de toute autre Publication de la CEI, ou au crédit qui lui est accordé.
8) L'attention est attirée sur les références normatives citées dans cette publication. L'utilisation de publications référencées est obligatoire pour une application correcte de la présente publication.
9) L'attention est attirée sur le fait que certains des éléments de la présente Publication de la CEI peuvent faire l'objet de droits de brevet. La CEI ne saurait être tenue pour responsable de ne pas avoir identifié de tels droits de brevets et de ne pas avoir signalé leur existence.

L'attention est attirée sur le fait que l'utilisation du type de protocole associé est restreinte par les détenteurs des droits de propriété intellectuelle. En tout état de cause, l'engagement de renonciation partielle aux droits de propriété intellectuelle pris par les détenteurs de ces droits autorise l'utilisation d'un type de protocole de couche avec les autres protocoles de couche du même type, ou dans des combinaisons avec d'autres types autorisées explicitement par les détenteurs des droits de propriété intellectuelle pour ce type.

NOTE Les combinaisons de Types de protocoles sont spécifiées dans les normes CEI 61784-1 et CEI 61784-2.

La Norme internationale CEI 61158-5-4 a été établie par le sous-comité 65C: Réseaux industriels, du comité d'études 65 de la CEI: Mesure, commande et automation dans les processus industriels.

Cette deuxième édition annule et remplace la première édition parue en 2007. Cette édition constitue une révision éditoriale avec seulement des révisions éditoriales mineures.

Cette édition inclut les modifications majeures suivantes par rapport à l'édition précédente:

a) améliorations éditoriales;

b) corrections éditoriales.

Le texte de cette norme est issu des documents suivants:

FDIS	Rapport de vote
65C/763/FDIS	65C/773/RVD

Le rapport de vote indiqué dans le tableau ci-dessus donne toute information sur le vote ayant abouti à l'approbation de cette norme.

Cette publication a été rédigée selon les Directives ISO/CEI, Partie 2.

Une liste de toutes les parties de la série CEI 61158, publiées sous le titre général *Réseaux de communication industriels – Spécifications des bus de terrain*, est disponible sur le site web de la CEI.

Le comité a décidé que le contenu de cette publication ne sera pas modifié avant la date de stabilité indiquée sur le site web de la CEI sous http://webstore.iec.ch dans les données relatives à la publication recherchée. A cette date, la publication sera

- reconduite,
- supprimée,
- remplacée par une édition révisée, ou
- amendée.

IMPORTANT – Le logo *"colour inside"* qui se trouve sur la page de couverture de cette publication indique qu'elle contient des couleurs qui sont considérées comme utiles à une bonne compréhension de son contenu. Les utilisateurs devraient, par conséquent, imprimer cette publication en utilisant une imprimante couleur.

INTRODUCTION

La présente partie de la CEI 61158 s'inscrit dans une série créée pour faciliter l'interconnexion des composants de systèmes d'automation. Elle renvoie aux autres normes de l'ensemble défini par le modèle de référence de bus de terrain "à trois couches" décrit dans la CEI 61158-1.

Le protocole d'application fournit le service d'application au moyen des services disponibles au niveau de la couche Liaison de données ou de la couche immédiatement inférieure. La présente norme définit les caractéristiques du service d'application que les applications de bus de terrain et/ou la gestion de système peuvent exploiter.

Dans l'ensemble de normes relatives aux bus de terrain, le terme "service" désigne une capacité abstraite fournie par une couche du modèle de référence de base de l'interconnexion des systèmes ouverts (Open Systems Interconnection, OSI) à la couche immédiatement supérieure. Ainsi, le service de couche Application défini dans la présente norme est un service architectural conceptuel, indépendant des divisions administratives et de mise en œuvre.

RÉSEAUX DE COMMUNICATION INDUSTRIELS – SPÉCIFICATIONS DES BUS DE TERRAIN –

Partie 5-4: Définition des services de la couche application – Eléments de type 4

1 Domaine d'application

1.1 Généralités

La couche application de bus de terrain (Fieldbus Application Layer, FAL) procure aux programmes de l'utilisateur un moyen d'accès à l'environnement de communication des bus de terrain. A cet égard, la FAL peut être considérée comme une "fenêtre entre programmes d'application correspondants".

La présente norme fournit des éléments communs pour les communications à temps critique ou non entre des programmes d'application dans un environnement et avec un matériel d'automation spécifiques aux bus de terrain de Type 4. Le terme "en temps critique" signale l'existence d'une fenêtre temporelle dans laquelle des actions spécifiées doivent être exécutées, avec un niveau de certitude défini. La non-réalisation des actions spécifiées dans la fenêtre temporelle induit un risque de défaillance des applications qui demandent ces actions, avec les risques afférents pour l'équipement, les installations et éventuellement la vie humaine.

La présente norme définit de manière abstraite le service visible de l'extérieur fourni par la couche application de bus de terrain de Type 4 en termes

a) de modèle abstrait visant à la définition des ressources d'application (objets) qui peuvent être manipulées par des utilisateurs utilisant un service FAL;

b) d'événements et d'actions liées aux primitives du service;

c) de paramètres associés à chaque événement et action de primitive, ainsi que de forme prise par ces paramètres; et

d) d'interaction entre ces événements et ces actions, ainsi que de séquences valides desdits événements et actions.

La présente norme vise à définir les services mis en place pour

1) l'utilisateur de FAL, à la frontière entre l'utilisateur et la couche application du modèle de référence de bus de terrain; et

2) la Gestion des systèmes, à la frontière entre la couche application et la Gestion des systèmes selon le modèle de référence de bus de terrain.

La présente norme spécifie la structure et les services de la couche application de bus de terrain de Type 4, en conformité avec le modèle de référence de base de l'OSI (ISO/CEI 7498-1) et la structure de la couche application de l'OSI (ISO/CEI 9545).

Les services et protocoles de couche FAL sont fournis par des entités AE de couche FAL contenues dans les processus d'application. L'AE de la FAL se compose d'un jeu d'éléments de service application (Application Service Element, ASE) orientés objet et d'une entité de gestion de couche (Layer Management Entity, LME) qui gère l'AE. Les éléments ASE délivrent des services de communication agissant sur un ensemble de classes d'objets de processus d'application (Application Process Object, APO) associées. L'un des éléments ASE de couche FAL est un élément ASE de gestion qui fournit un ensemble commun de services destinés à la gestion des instances des classes de couche FAL.

Quoique ces services spécifient, du point de vue des applications, les modalités d'émission et de remise des demandes et des réponses, ils ne comprennent pas de spécification du traitement que doivent en faire les applications demandeuse et répondeuse. En d'autres termes, les aspects comportementaux des applications ne sont pas définis; seule une définition des demandes et réponses que ces applications peuvent envoyer/recevoir est établie. Cela laisse une plus grande marge de manœuvre aux utilisateurs de la couche FAL dans la normalisation du comportement de ces objets. Outre ces services, la présente norme définit également certains services de soutien donnant accès à la couche FAL dans un but de commande de certains aspects de son fonctionnement.

1.2 Spécifications

La présente norme a pour principal objet de préciser les caractéristiques des services conceptuels de couche application adaptés aux communications à temps critique; elle vise ainsi à compléter le modèle de référence de base OSI en guidant le développement de protocoles de couche application destinés aux communications à temps critique.

Un objectif secondaire consiste à fournir des voies d'évolution à partir des protocoles de communication industriels antérieurs. Ce dernier objectif explique la diversité des services normalisés sous la forme des différents Types CEI 61158, ainsi que celle des protocoles correspondants, normalisés dans la série CEI 61158-6.

La présente spécification peut être utilisée comme la base pour les interfaces de programmation d'applications (Application Programming-Interfaces) formelles. Cependant, elle ne constitue pas une interface de programmation formelle, et toute interface de ce type devra faire face à des problèmes de mise en œuvre non couverts par la présente spécification, notamment

a) les dimensions et l'ordre des octets de plusieurs paramètres de service multi-octet, et

b) la corrélation des primitives associées (demande et confirmation, ou indication et réponse).

1.3 Conformité

La présente norme ne définit pas de mises en œuvre ni de produits particuliers, pas plus qu'elle ne limite les mises en œuvre des entités de couche application dans les systèmes d'automation industriels.

Il n'existe pas de conformité de l'équipement à la présente norme de définition de service de couche application. Au contraire, la conformité est obtenue par une mise en œuvre de protocoles conformes de couche application qui satisfont aux de services de couche application de Type 2 définis dans la présente norme.

2 Références normatives

Les documents suivants sont cités en référence de manière normative, en intégralité ou en partie, dans le présent document et sont indispensables pour son application. Pour les références datées, seule l'édition citée s'applique. Pour les références non datées, la dernière édition du document de référence s'applique (y compris les éventuels amendements).

NOTE Toutes les parties de la série CEI 61158, ainsi que la CEI 61784-1 et la CEI 61784-2 font l'objet d'une maintenance simultanée. Les références croisées à ces documents dans le texte se rapportent par conséquent aux éditions datées dans la présente liste de références normatives.

CEI 61158-3-4:2014, *Réseaux de communication industriels – Spécifications des bus de terrain – Partie 3-4: Définition du service de la couche liaison de données – Eléments de type 4*

CEI 61158-4-4:2014, *Réseaux de communication industriels – Spécifications des bus de terrain – Partie 4-4: Spécification du protocole de la couche liaison de données – Eléments de type 4*

CEI 61158-6-4:2014, *Réseaux de communication industriels – Spécifications des bus de terrain – Partie 6-4: Spécification de protocole de la couche application – Eléments de type 4*

CEI 61158-6 (toutes les sous-parties), *Réseaux de communication industriels – Spécifications des bus de terrain – Partie 6: Spécification de protocole de la couche application*

ISO/CEI 7498-1, *Technologies de l'information – Interconnexion de systèmes ouverts (OSI) – Modèle de référence de base – Partie 1: Le modèle de base*

ISO/CEI 7498-3, *Technologies de l'information – Interconnexion de systèmes ouverts (OSI) – Modèle de référence de base – Partie 3: Dénomination et adressage*

ISO/CEI 8822, *Technologies de l'information – Interconnexion de systèmes ouverts – Définition du service de présentation*

ISO/IEC 8824-1, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation* (disponible en anglais seulement)

ISO/CEI 9545, *Technologies de l'information – Interconnexion de systèmes ouverts (OSI) – Structure de la couche Application*

ISO/CEI 10731, *Technologies de l'information – Interconnexion de systèmes ouverts – Modèle de référence de base – Conventions pour la définition des services OSI*

ISO/IEC/IEEE 60559, *Information technology – Microprocessor Systems – Floating-Point arithmetic* (disponible en anglais seulement)

3 Termes et définitions

Pour les besoins du présent document, les termes suivants, définis dans ces publications, s'appliquent:

3.1 Termes de l'ISO/CEI 7498-1

a) entité d'application
b) processus d'application
c) unité de données de protocole d'application
d) élément de service d'application
e) invocation d'entité d'application
f) invocation de processus d'application
g) transaction d'application
h) système ouvert réel
i) syntaxe de transfert

3.2 Termes de l'ISO/CEI 8822

Pour les besoins du présent document, les termes suivants, définis dans l'ISO/CEI 8822, s'appliquent:

a) syntaxe abstraite
b) contexte de présentation

3.3 Termes de l'ISO/CEI 9545

Pour les besoins du présent document, les termes suivants, définis dans l' ISO/CEI 9545, s'appliquent:

a) application-association (association d'applications)
b) application-context (contexte d'application)
c) nom de contexte d'application
d) application-entity-invocation (invocation d'entité d'application)
e) application-entity-type (type d'entité d'application)
f) application-process-invocation (invocation de processus d'application)
g) application-process-type (type de processus d'application)
h) application-service-element (élément de service d'application)
i) élément de service de contrôle d'application

3.4 Termes de l'ISO/CEI 8824-1

Pour les besoins du présent document, les termes suivants, définis dans l'ISO/CEI 8824-1, s'appliquent:

a) identificateur d'objet
b) type

3.5 Termes relatifs à la couche Liaison de données de bus de terrain

Pour les besoins du présent document, les termes suivants s'appliquent.

a) délai DL
b) politique de planification DL
c) DLCEP
d) DLC
e) DLPDU
f) DLSDU
g) DLSAP
h) balise fixe
i) balise générique
j) liaison
k) adresse réseau
l) adresse de nœud
m) node
n) balise
o) planifié
p) non planifié

3.6 Définitions relatives à la couche application de bus de terrain

Pour les besoins de la présente norme, les termes et définitions suivants s'appliquent.

3.6.1
application
fonction ou structure de données pour laquelle des données sont consommées ou produites

3.6.2
objets d'application
classes d'objets multiples qui gèrent et assurent un échange de messages pendant le mode exécution à travers le réseau et à l'intérieur de l'appareil de réseau

3.6.3
processus d'application
partie d'une application distribuée sur un réseau, qui est située sur un appareil et adressée sans ambiguïté

3.6.4
identificateur de processus d'application
identificateur qui permet de distinguer différents processus d'application utilisés dans un appareil

3.6.5
objet de processus d'application
composant d'un processus d'application, identifiable et accessible via une relation d'applications de la FAL

Note 1 à l'article: Les définitions d'objets de processus d'application se composent d'un ensemble de valeurs destinées aux attributs de leur classe (se reporter à la définition "classe d'objets de processus d'application"). On peut accéder aux définitions d'objets de processus d'application à distance au moyen des services de l'élément ASE de gestion des objets FAL. Les services de gestion des objets FAL peuvent être utilisés pour charger ou mettre à jour des définitions d'objets, pour lire des définitions d'objets, ainsi que pour créer et supprimer de manière dynamique des objets d'application et les définitions correspondantes.

3.6.6
classe d'objets de processus d'application
classe d'objets de processus d'application définis par un ensemble de services et d'attributs accessibles via le réseau

3.6.7
relation d'applications
association de type coopératif entre deux invocations d'entités d'application (application-entity-invocation) ou plus, dans un but d'échange d'informations et de coordination de leur action conjointe

Note 1 à l'article: Cette relation est activée soit par l'échange d'unités de données de protocole d'application (application-protocol-data-unit), soit à la suite d'activités de préconfiguration.

3.6.8
élément de service d'application de relation d'applications
élément de service d'application (application-service-element) qui fournit le seul moyen d'établir et de rompre toute relation d'applications

3.6.9
point d'extrémité de relation d'applications
contexte et comportement d'une relation d'applications, vus et maintenus par un des processus d'application impliqués dans la relation d'applications

Note 1 à l'article: Chaque processus d'application impliqué dans la relation d'applications maintient son propre point d'extrémité de relation d'applications.

3.6.10
attribut
description d'une caractéristique ou fonction, visible par un observateur externe, d'un objet

Note 1 à l'article: Les attributs d'un objet contiennent des informations sur les portions variables d'un objet. En règle générale, ils fournissent des informations de statut ou régissent l'action d'un objet. Les attributs peuvent également influer sur le comportement d'un objet. Les attributs se répartissent en deux catégories: les attributs de classe et les attributs d'instance.

3.6.11
comportement
indication de la manière dont un objet réagit à des événements particuliers

3.6.12
no-bit
Désigne le numéro d'un bit dans une chaîne de bits ou un octet

3.6.13
canal
liaison physique ou logique unique d'un objet d'application d'entrée ou de sortie d'un serveur avec le processus

3.6.14
classe
ensemble d'objets représentant le même type de composant du système

Note 1 à l'article: Une classe est une généralisation d'un objet, un modèle qui permet de définir des variables et des méthodes. Tous les objets d'une classe possèdent une forme et un comportement identiques, mais leurs attributs contiennent généralement des données différentes.

3.6.15
attributs de classe
attribut commun à tous les objets d'une classe

3.6.16
code de classe
identificateur unique attribué à chaque classe d'objets

3.6.17
service spécifique à une classe
service défini par une classe d'objets particulière permettant d'exécuter une fonction qui doit être exécutée et qu'un service classique ne peut pas assurer

Note 1 à l'article: Tout objet spécifique à une classe est réservé à l'usage exclusif de la classe d'objets qui le définit.

3.6.18
client
a) objet qui utilise les services d'un autre objet (serveur) pour réaliser une tâche

b) initiateur d'un message auquel un serveur réagit

3.6.19
objets de communication
composants qui gèrent et assurent un échange de messages pendant le mode exécution à travers le réseau

EXEMPLES: objet Connection Manager (gestionnaire de connexion), objet Unconnected Message Manager (UCMM, gestionnaire de messages non connectés) et objet Message Router (routeur de message).

3.6.20
connexion
liaison logique entre deux objets d'application qui peuvent se trouver au sein du même appareil ou dans des appareils différents

Note 1 à l'article: Les connexions peuvent être point à point ou multipoint.

3.6.21
chemin de transport
flux unidirectionnel d'unités APDU, d'un bout à l'autre d'une relation entre applications

3.6.22
AR dédiée
AR directement utilisée par l'utilisateur FAL

Note 1 à l'article: Dans les AR dédiées, seuls l'en-tête FAL et les données d'utilisateur sont transférés.

3.6.23
adresse DL par défaut
la valeur 126 en tant que valeur initiale de l'adresse DL, devant être changée (par exemple par l'affectation d'une adresse DL via le bus de terrain) avant utilisation avec un DP-maître (classe 1)

3.6.24
appareil
matériel physique connecté à la liaison

Note 1 à l'article: Un appareil peut contenir plusieurs nœuds.

3.6.25
AR dynamique
AR qui doit être mise dans un état établi par le biais des procédures d'établissement d'AR

3.6.26
point d'extrémité
une des entités en communication impliquées dans une connexion

3.6.27
erreur
divergence entre une valeur ou condition calculée, observée ou mesurée et la valeur ou condition spécifiée ou théoriquement correcte

3.6.28
classe d'erreurs
groupement général de définitions d'erreurs proches et des codes d'erreur associés

3.6.29
code d'erreur
identification d'un type d'erreur particulier dans une classe d'erreurs

3.6.30
événement
instance d'un changement de conditions

3.6.31
sous-réseau FAL
sous-réseaux composés d'un ou plusieurs segments de liaison de données, identifiés par un sous-ensemble de l'adresse de réseau

Note 1 à l'article: les sous-réseaux FAL peuvent contenir des ponts, mais pas de routeurs.

3.6.32
variable FIFO
classe d'objets de variable composée d'un ensemble d'éléments saisis de manière homogène dans laquelle le premier élément écrit constitue le premier élément qui peut être lu

Note 1 à l'article: Dans un bus de terrain, seul un élément complet peut être transféré à la suite d'une invocation de services.

3.6.33
trame
synonyme discrédité de DLPDU

3.6.34
interface
a) frontière commune entre deux unités fonctionnelles, définie par des caractéristiques fonctionnelles, des caractéristiques de signal ou d'autres caractéristiques adaptées

b) ensemble d'attributs et de services de classe FAL représentant une vue spécifique de la classe FAL

3.6.35
invocation
acte d'utiliser un service ou une autre ressource d'un processus d'application

Note 1 à l'article: Chaque invocation représente un fil de commande distinct qui peut être décrit par son contexte. Une fois le service terminé ou la ressource libérée, l'invocation cesse d'exister. Dans le cas des invocations de services, un service lancé, mais pas encore terminé est considéré comme une invocation de services en cours. Dans le cas des invocations de services, un ID d'invocation (Invoke ID) peut être utilisé pour identifier une invocation de services de manière non ambiguë et la différencier des autres invocations de services en cours.

3.6.36
index
adresse d'un objet au sein d'un processus d'application

3.6.37
instance
occurrence physique permettant d'identifier un objet parmi d'autres au sein d'une même classe d'objets

EXEMPLE "California" (La Californie) est une instance de la classe d'objets "state" (état).

Note 1 à l'article: Les termes objet, instance et instance d'objet font référence à une instance particulière.

3.6.38
attributs d'instance
attribut qui est réservé à l'usage exclusif d'une instance d'objet et n'est pas commun à la classe d'objets

3.6.39
instancié
se dit d'un objet créé dans un appareil

3.6.40
appareil logique
classe FAL particulière qui extrait un composant logiciel ou de microprogramme sous la forme d'un dispositif intégré et autonome au sein d'un appareil d'automatisation

3.6.41
ID fabricant
identification de chaque fabricant de produits par un numéro unique

3.6.42
informations de gestion
informations accessibles via le réseau qui facilitent la gestion de l'exploitation du système de bus de terrain, y compris la couche application

Note 1 à l'article: La gestion inclut des fonctions telles que la commande, la surveillance et le diagnostic.

3.6.43
membre
élément d'un attribut qui est structuré comme une partie d'une matrice

3.6.44
méthode
<objet> synonyme d'un service de fonctionnement délivré par l'élément ASE serveur et appelé par un client

3.6.45
module
composant matériel ou logique d'un appareil physique

3.6.46
réseau
ensemble de nœuds reliés par un support de communication d'un type ou d'un autre, avec d'éventuels répétiteurs intermédiaires, ponts, routeurs et passerelles de couche inférieure

3.6.47
objet
représentation abstraite d'un composant particulier dans un appareil; il s'agit généralement d'un ensemble de données (sous la forme de variables) et de méthodes (procédures) associées, destiné à l'exploitation des données dont l'interface et le comportement sont clairement définis

3.6.48
service spécifique à un objet
service réservé à l'usage exclusif de la classe d'objets qui le définit

3.6.49
homologue
rôle d'un point d'extrémité d'AR dans lequel il est capable d'agir à la fois comme client et comme serveur

3.6.50
appareil physique
appareil d'automation ou autre appareil de réseau

3.6.51
propriété
terme général désignant toute information descriptive concernant un objet

3.6.52
fournisseur
source d'une connexion de données

3.6.53
éditeur
rôle d'un point de fin d'AR qui transmet les unités APDU sur le bus de terrain afin qu'elles soient consommées par un ou plusieurs abonnés

Note 1 à l'article: Un éditeur peut ne pas connaître l'identité des abonnés ou leur nombre; il peut publier ses unités APDU au moyen d'une AR dédiée.

3.6.54
gestionnaire d'édition
rôle d'un point de fin d'AR dans lequel il envoie une ou plusieurs unités APDU de demande de service confirmé à un éditeur pour demander l'édition d'un objet spécifié

Note 1 à l'article: La présente norme définit deux types de gestionnaires d'édition: les gestionnaires d'édition par extraction et les gestionnaires d'édition par émission; chacun d'eux fait l'objet d'une définition distincte

3.6.55
abonné par extraction
type d'abonné qui identifie les unités APDU de réponse de service confirmé reçues comme étant des données d'objet éditées

3.6.56
ressource
capacité de traitement ou d'information d'un sous-système

3.6.57
point d'extrémité de chemin (route endpoint)
conteneur d'objet contenant les objets de variable d'une classe de variables

3.6.58
serveur
a) rôle d'un AREP dans lequel il renvoie une APDU de réponse de service confirmé au client à l'origine de la demande

b) objet qui offre des services à un autre objet (client)

3.6.59
service
opération ou fonction qu'un objet et/ou une classe d'objets exécute à la demande d'un autre objet et/ou une autre classe d'objets

3.6.60
abonné
rôle d'un AREP dans lequel il reçoit les unités APDU produites par un éditeur

3.7 Abréviations et symboles

AE	Entité d'application
AL	Couche application
ALME	Entité de gestion de couche application
ALP	Protocole de couche application
APO	Objet d'application
AP	Processus d'application
APDU	Unité de données de protocole d'application

API	Identificateur de processus d'application
AR	Relation entre applications
AREP	Point de fin de relation entre applications
ASCII	American Standard Code for Information Interchange
ASE	Elément de service d'application
Cnf	Confirmation
CR	Relation de communication
CREP	Point de fin de relation de communication
DL-	(comme préfixe) Liaison de données
DLC	Connexion de liaison de données
DLCEP	Point d'extrémité de connexion de liaison de données
DLL	Couche de liaison de données
DLM	Gestion de liaison de données
DLSAP	Point d'accès au service liaison de données
DLSDU	Unité de données de service de liaison de données
DNS	Service de nom de domaine
DP	Périphériques décentralisés
FAL	Couche application de bus de terrain
FIFO	Premier entré, premier sorti
IHM	Interface Homme/Machine
ID	Identificateur
IDL	Langage de définition d'interface
CEI	Commission Electrotechnique Internationale
Ind	Indication
IP	Internet Protocol
ISO	Organisation internationale de normalisation
LDev	Appareil logique
LME	Entité de gestion de couche
OSI	Interconnexion de systèmes ouverts
PDev	Appareil physique
PDU	Unité de données de protocole
PL	Couche physique
QoS	Qualité de service
REP	Point d'extrémité de chemin
Req	Demande
Rsp	Réponse
RT	Exécution
SAP	Point d'accès de service
SCL	Niveau de sécurité
SDU	Unité de données de service
SEM	Matrice d'événement d'état

SMIB Base d'informations de gestion du système

SMK Noyau de gestion du système

STD Diagramme de transition d'états, utilisé pour décrire le comportement de l'objet

VAO Objet Variable

3.8 Conventions

3.8.1 Présentation

La couche FAL se compose d'un ensemble d'ASE orientés objet. Chaque ASE est spécifié dans un paragraphe distinct. Chaque spécification d'ASE est constituée de deux parties, à savoir sa spécification de classe et sa spécification de service.

La spécification de classe définit les attributs de la classe. Les attributs sont accessibles à partir d'instances de la classe en utilisant les services d'ASE de gestion d'objets spécifiés à l'Article 5 de la présente norme. La spécification de service définit les services fournis par l'élément ASE.

3.8.2 Conventions générales

La présente norme emploie les conventions de description énoncées dans l'ISO/CEI 10731.

3.8.3 Conventions pour les définitions de classe

Les définitions de classe sont décrites à l'aide de modèles. Chaque modèle se compose d'une liste d'attributs associés à la classe. La forme générale du modèle est montrée ci-dessous:

FAL ASE:			**Nom de l'ASE**
CLASSE:			**Nom de la classe**
ID CLASSE:			#
CLASSE PARENT:			nom de la classe parent
ATTRIBUTS:			
1	(o)	Attribut clé:	identificateur numérique
2	(o)	Attribut clé:	nom
3	(m)	Attribut:	nom d'attribut(valeurs)
4	(m)	Attribut:	nom d'attribut(valeurs)
4.1	(s)	Attribut:	nom d'attribut(valeurs)
4.2	(s)	Attribut:	nom d'attribut(valeurs)
4.3	(s)	Attribut:	nom d'attribut(valeurs)
5.	(c)	Contrainte:	expression de la contrainte
5.1	(m)	Attribut:	nom d'attribut(valeurs)
5.2	(o)	Attribut:	nom d'attribut(valeurs)
6	(m)	Attribut:	nom d'attribut(valeurs)
6.1	(s)	Attribut:	nom d'attribut(valeurs)
6.2	(s)	Attribut:	nom d'attribut(valeurs)
SERVICES:			
1	(o)	OpsService:	nom de service
2.	(c)	Contrainte:	expression de la contrainte
2.1	(o)	OpsService:	nom de service
3	(m)	MgtService:	nom de service

(1) La rubrique "FAL ASE:" est le nom de l'élément ASE de la couche FAL (FAL ASE) qui fournit les services pour la classe spécifiée.

(2) La rubrique "CLASS:" est le nom de la classe spécifiée. Tous les objets définis à l'aide de ce modèle seront une instance de cette classe. La classe peut être spécifiée par la présente norme ou par un utilisateur de la présente norme.

(3) La rubrique "CLASS ID:" est un numéro qui identifie la classe spécifiée. Ce numéro est unique au sein du FAL ASE qui fournira les services pour cette classe. Lorsqu'il est qualifié par l'identité de son FAL ASE, il identifie sans ambiguïté la classe relevant du domaine d'application de la FAL. La valeur "NULL" indique que la classe ne peut pas être instanciée. Les Class ID (identificateurs de classe) entre 1 et 255 sont réservés par la présente norme pour identifier des classes normalisées. Ils ont été attribués pour maintenir la compatibilité avec des normes nationales existantes. Les CLASS ID entre 256 et 2048 sont alloués pour identifier les classes définies par l'utilisateur.

(4) La rubrique "PARENT CLASS:" est le nom de la classe parent pour la classe spécifiée. Tous les attributs définis pour la classe parent et hérités par celle-ci sont hérités pour la classe définie, et ils n'ont donc pas à être redéfinis dans le modèle pour cette classe.

NOTE La classe parent "TOP" indique que la classe définie est une définition de classe initiale. La classe parent "TOP" est utilisée comme point de départ à partir duquel toutes les autres classes sont définies. L'usage de "TOP" est réservé pour les classes définies par la présente norme.

(5) L'étiquette "ATTRIBUTES" (ATTRIBUTS) indique que les entrées suivantes sont des attributs définis pour la classe.

a) Chacune des entrées d'attribut contient un numéro de ligne dans la colonne 1, un indicateur obligatoire (m) / facultatif (o) / conditionnel (c) / sélecteur (s) dans la colonne 2, une étiquette de type d'attribut dans la colonne 3, un nom ou une expression conditionnelle dans la colonne 4, et, facultativement, une liste de valeurs énumérées dans la colonne 5. Dans la colonne suivant la liste de valeurs, la valeur par défaut pour l'attribut peut être spécifiée.

b) Les objets sont normalement identifiés par un identificateur numérique et/ou par un nom d'objet. Dans les modèles de classe, ces attributs clés sont définis sous l'attribut clé.

c) Le numéro de ligne définit la séquence et le niveau d'imbrication de la ligne. Chaque niveau d'imbrication est identifié par un point (period). L'imbrication est utilisée pour spécifier

i) des champs d'un attribut structuré (4.1, 4.2, 4.3),

ii) des attributs conditionnés à un énoncé de contrainte (5). Les attributs peuvent être obligatoires (5.1) ou facultatifs (5.2) si la contrainte est vraie. Tous les attributs facultatifs n'exigent pas des énoncés de contraintes comme le fait l'attribut défini en 5.2,

iii) les champs sélection d'un attribut de type choix (6.1 et 6.2).

(6) L'étiquette "SERVICES" indique que les entrées suivantes sont des services définis pour la classe.

a) Un (m) dans la colonne 2 indique que le service est obligatoire pour la classe, alors qu'un (o) indique qu'il est facultatif. Un (c) dans cette colonne indique que le service est conditionnel. Lorsque tous les services définis pour une classe le sont comme étant facultatifs, l'un au moins doit être sélectionné quand une instance de la classe est définie.

b) L'étiquette "OpsService" désigne un service opérationnel (1).

c) L'étiquette "MgtService" désigne un service de gestion (2).

d) Le numéro de ligne définit la séquence et le niveau d'imbrication de la ligne. Chaque niveau d'imbrication est identifié par un point (period). L'imbrication dans la liste de services sert à spécifier des services conditionnés à un énoncé de contrainte.

3.8.4 Conventions pour les définitions de service

3.8.4.1 Généralités

Le modèle de service, les primitives de service et les diagrammes séquentiels de temps utilisés sont des descriptions entièrement abstraites; ils ne représentent pas une spécification de mise en œuvre.

3.8.4.2 Paramètres du service

Les primitives de service sont utilisées pour représenter les interactions entre utilisateur de service et fournisseur de service (ISO/CEI 10731). Elles acheminent des paramètres qui indiquent des informations disponibles dans l'interaction entre utilisateur et fournisseur. Pour toute interface particulière, il n'est pas nécessaire d'indiquer explicitement tous les paramètres.

Les spécifications de service selon la présente norme utilisent un format de tableau pour décrire les paramètres de composants des primitives du service d'ASE. Les paramètres qui s'appliquent à chaque groupe de primitives de service sont consignés en tableaux. Chaque tableau comporte jusqu'à cinq colonnes pour le/la:

1) nom de paramètre,
2) primitive "request",
3) primitive "indication",
4) primitive "response", et
5) primitive "confirm".

Un paramètre (ou un composant de celui-ci) est énuméré dans chaque ligne de chaque tableau. Dans les colonnes appropriées de la primitive de service, un code est utilisé pour spécifier le type d'usage du paramètre sur la primitive spécifiée dans la colonne:

M le paramètre est obligatoire pour la primitive.

U le paramètre est une option de l'utilisateur et peut ou peut ne pas être fourni, en fonction de l'usage dynamique de l'utilisateur du service. Lorsque ce paramètre n'est pas fourni, une valeur par défaut est supposée.

C le paramètre est conditionné à d'autres paramètres ou à l'environnement de l'utilisateur du service.

— (blanc/vide) le paramètre n'est jamais présent.

S le paramètre est un élément sélectionné.

Certaines entrées sont par ailleurs qualifiées par des éléments entre parenthèses. Ces entrées peuvent être:

a) une contrainte spécifique au paramètre:
"(=)" indique que le paramètre équivaut du point de vue de la sémantique au paramètre dans la primitive de service située immédiatement à sa gauche dans le tableau.

b) une indication qu'une certaine note s'applique à l'article:
"(n)" indique que la note "n" suivante contient des informations complémentaires relatives au paramètre et à son utilisation.

3.8.4.3 Procédures de service

Les procédures sont définies en termes

- d'interactions entre entités d'application par l'échange d'unités de données de protocole d'application de bus de terrain, et
- d'interactions entre un fournisseur de service de couche application et un utilisateur de service de couche application dans le même système par l'invocation de primitives de service de couche application.

Ces procédures sont applicables à des instances de communication entre systèmes qui prennent en charge des services de communication à contrainte temporelle au sein de la couche Application de bus de terrain.

4 Concepts

4.1 Présentation

Le bus de terrain est destiné à être utilisé dans les usines et les installations de traitement pour interconnecter les appareils primaires d'automatisation (par exemple, capteurs, organes de commande, dispositifs d'affichage locaux, organes de signalisation, automates programmables, petits régulateurs en boucle unique et contrôles de champ autonomes) avec des équipements de commande et de surveillance situés dans des centres de contrôle.

Les appareils primaires d'automatisation sont associés aux niveaux les plus faibles de la hiérarchie d'automatisation industrielle et effectuent un ensemble limité de fonctions dans une fenêtre temporelle définie. Certaines de ces fonctions incluent les diagnostics, la validation des données et la gestion de plusieurs entrées et sorties.

Ces appareils primaires d'automatisation, également appelés dispositifs de terrain, sont situés à proximité des fluides de traitement, de la pièce fabriquée, de la machine, de l'opérateur et de l'environnement. Cette utilisation positionne le bus de terrain aux niveaux les plus bas de l'architecture CIM (Computer Integrated Manufacturing).

Certains des avantages attendus de l'utilisation de bus de terrain sont la réduction du câblage, l'augmentation de la quantité des données échangées, la distribution plus large du contrôle entre les appareils primaires d'automatisation et l'équipement du centre de contrôle, et le respect des contraintes temporelles critiques.

L'Article 4 décrit les principes de la couche FAL. Des informations descriptives détaillées relatives aux éléments ASE de couche FAL peuvent être trouvées dans le paragraphe "Présentation" de chacune des spécifications de modèle de communication.

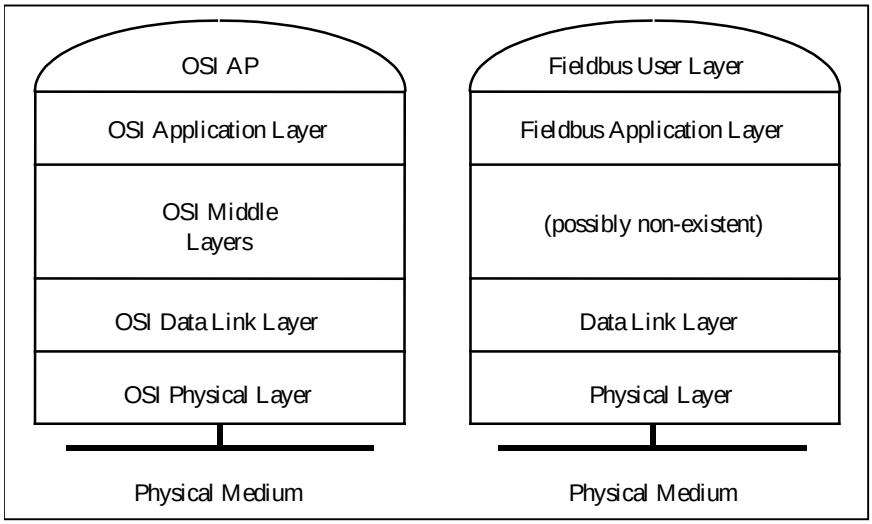
4.2 Relations de l'architecture

4.2.1 Relation à la couche Application du modèle de référence de base OSI

Les fonctions de la couche FAL ont été décrites conformément aux principes des couches OSI. Toutefois, sa relation d'architecture aux couches inférieures est différente, comme le montre la Figure 1.

– La couche FAL inclut les fonctions OSI avec des extensions pour faire face aux contraintes temporelles critiques. La norme relative à la structure de la couche Application OSI (ISO/CEI 9545) a servi de base pour la spécification de la couche FAL.

– La couche FAL utilise directement les services de la couche sous-jacente. La couche sous jacente peut être la couche DLL ou n'importe quelle autre couche intermédiaire. Lorsqu'elle utilise la couche sous-jacente, la couche FAL peut fournir des fonctions normalement associées aux couches OSI intermédiaires pour le mapping correct sur la couche sous-jacente.



Légende

Anglais	Français
Fieldbus User Layer	Couche utilisateur de bus de terrain
OSI Application Layer	Couche d'application OSI
Fieldbus Application Layer	Couche d'application de bus de terrain
OSI Middle Layers	Couches intermédiaire OSI
(possibly non-existent)	(peut-être inexistant)
OSI Data Link Layer	Couche de liaison OSI
Data Link Layer	Couche de liaison de données
OSI Physical Layer	Couche physique OSI
Physical Layer	Couche physique
Physical Medium	Support physique

Figure 1 – Relation au modèle de référence de base OSI

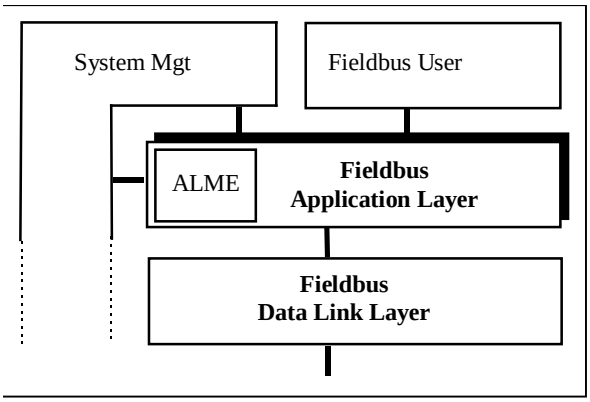
4.2.2 Relations aux autres entités du bus de terrain

4.2.2.1 Généralités

Les relations d'architecture de la couche Application de bus de terrain (FAL), illustrées à la Figure 2, ont été conçues pour faire face aux besoins d'interopérabilité des systèmes à temps critique distribués au sein de l'environnement du bus de terrain.

Au sein de cet environnement, la couche FAL fournit des services de communications aux applications en temps critique ou non situées dans les appareils de bus de terrain.

En outre, la couche FAL utilise directement la couche DLL pour transférer ses unités de données de protocole de couche application. Elle le fait en utilisant un ensemble de services de transfert de données et un ensemble de services de soutien qui permettent de contrôler les aspects opérationnels de la couche DLL.



Légende

Anglais	Français
System Mgt	Gestion système
Fieldbus User	Utilisateur de bus de terrain
Fieldbus Application Layer	Couche d'application de bus de terrain
Fieldbus Data Link Layer	Couche de liaison de données de bus de terrain

Figure 2 – Positionnement architectural de la couche Application de bus de terrain

4.2.2.2 Utilisation de la couche Liaison de données de bus de terrain

La couche Application de bus de terrain (FAL) assure l'accès réseau aux AP de bus de terrain. Elle s'interface directement à la couche Liaison de données de bus de terrain pour le transfert de ses APDU.

La couche DLL offre différents types de services à la couche FAL pour le transfert des données entre les points de fin de liaison de données (par exemple DLSAP, DLCEP).

4.2.2.3 Support pour les applications de bus de terrain

Les applications de bus de terrain sont représentées sur le réseau en tant que procédés d'application (AP). Les AP sont les composants d'un système distribué qui peuvent être individuellement identifiés et adressés.

Chaque AP contient une entité d'application (AE) FAL qui assure l'accès réseau à l'AP. C'est-à-dire que chaque AP communique avec les autres par l'intermédiaire de son AE. Dans ce sens, l'AE fournit une fenêtre de visibilité dans l'AP.

Les AP contiennent des composants identifiables qui sont également visibles sur le réseau. Ces composants sont représentés sur le réseau en tant qu'objets de processus d'application (APO). Ils peuvent être identifiés ou plusieurs attributs clés. Ils se trouvent à l'adresse du processus d'application qui les contient.

Les services utilisés pour y accéder sont fournis par les éléments de service application (ASE) spécifiques de l'APO contenus au sein de la couche FAL. Ces ASE sont conçus pour prendre en charge les applications utilisateur, de bloc de fonction et de gestion.

4.2.2.4 Prise en charge de la gestion du système

Les services de la couche FAL peuvent être utilisés pour prendre en charge diverses opérations de gestion, y compris la gestion des systèmes de bus de terrain, des applications et du réseau de bus de terrain.

4.2.2.5 Accès aux entités de gestion de couche de la couche FAL

Une entité de gestion de couche (LME) peut être présente dans chaque entité FAL du réseau. Les FALME donnent accès à la couche FAL pour les besoins de gestion du système.

L'ensemble des données accessibles par le gestionnaire du système s'appelle Base d'informations de gestion du système (SMIB). Chaque entité de gestion de couche Application de bus de terrain (FALME) fournit la partie FAL de la SMIB. Le mode d'implémentation de la SMIB ne relève pas du domaine d'application de la présente norme.

4.3 Structure de la couche Application de bus de terrain

4.3.1 Présentation

La structure de la couche FAL est un affinement de la structure de la couche Application OSI (ISO/CEI 9545). Par conséquent, l'organisation de 4.3 est semblable à celle de l'ISO/CEI 9545. Certains concepts présentés ici ont été précisés par rapport à l'ISO/CEI 9545 pour l'environnement du bus de terrain.

La couche FAL diffère des autres couches OSI par deux aspects principaux.

- OSI définit un type unique de voie de communication de la couche application, l'association, pour connecter les APPAREILS PORTABLES les uns aux autres. La couche FAL définit la relation entre applications (AR), dont il existe plusieurs types, pour permettre aux procédés d'application (AP) de communiquer.
- La couche FAL utilise la DLL pour transférer ses PDU mais pas la couche de présentation. Par conséquent, il n'existe pas de contexte de présentation explicite à la disposition de la couche FAL. Entre la même paire (ou ensemble) de points d'accès de service de liaison de données, le protocole de la couche FAL ne peut pas être utilisé simultanément avec d'autres protocoles de couche application.

4.3.2 Concepts fondamentaux

Le fonctionnement des systèmes ouverts réels à temps critiques est modélisé en termes d'interactions entre les AP à temps critiques. La couche FAL permet à ces AP de transmettre des commandes et des données entre elles.

La coopération entre AP exige qu'elles partagent des informations suffisantes pour interagir et réaliser les activités de traitement de façon coordonnée. Leurs activités peuvent être restreintes à un segment unique de bus de terrain ou s'étendre sur plusieurs segments. La couche FAL a été conçue en utilisant une architecture modulaire pour prendre en charge les besoins de messagerie de ces applications.

En outre, la coopération entre AP exige parfois qu'elles partagent le même sens de la durée. La couche FAL ou la couche DLL (CEI 61158-3-4 et CEI 61158-4-4) peut assurer la distribution du temps sur tous les appareils. Elles peuvent également définir des services d'appareil locaux qui peuvent être utilisés par les AP pour accéder au temps distribué.

Le reste de 4.3 décrit chacun des composants modulaires de l'architecture ainsi que leurs relations les uns aux autres. Les composants de la couche FAL sont modélisés en tant qu'objets, chacun d'entre eux fournissant un ensemble de services de communications FAL utilisables par les applications. Les objets FAL et leurs relations sont décrits ci-dessous. Les spécifications détaillées des objets FAL et leurs services sont fournis dans les articles suivants de la présente norme. La CEI 61158-6-4 spécifie les protocoles nécessaires pour transmettre ces services d'objet entre applications.

4.3.3 Procédés d'application de bus de terrain

4.3.3.1 Définition de l'AP de bus de terrain

Dans l'environnement du bus de terrain, une application peut être fractionnée en un ensemble de composants et distribuée sur plusieurs appareils du réseau. Chacun de ces composants est appelé procédé d'application (AP) de bus de terrain. Un AP de bus de terrain est une variante du procédé d'application tel qu'il est défini dans le modèle de référence OSI ISO (ISO/CEI 7498). Les AP de bus de terrain peuvent être adressés sans ambiguïté par au moins une adresse individuelle de point d'accès au service de couche de liaison de données. Adressé sans ambiguïté, dans ce contexte, signifie qu'aucun autre AP ne peut être localisé simultanément par la même adresse. Cette définition n'interdit pas qu'un AP soit localisé par une ou plusieurs adresses individuelles (ou groupe) de point d'accès au service de couche liaison de données.

4.3.3.2 Services de communication

Les AP de bus de terrain communiquent entre eux en utilisant des services confirmés ou non confirmés (ISO/CEI 10731). Les services définis dans cette norme pour la couche FAL spécifient la sémantique des services vus par les AP demandeuse et répondeuse. La syntaxe des messages utilisée pour acheminer les demandes et réponses de services est définie dans la CEI 61158-6-4. Le comportement AP associé aux services est spécifié par l'AP.

Les services confirmés sont utilisés pour définir les échanges de demande/réponse entre les AP.

En revanche, les services non confirmés sont utilisés pour définir le transfert unidirectionnel des messages d'une AP vers un ou plusieurs AP distants. Du point de vue des communications, il n'existe pas de relations entre des invocations séparées des services non confirmés comme il en existe entre la demande et la réponse d'un service confirmé.

4.3.3.3 Interactions entre AP

4.3.3.3.1 Généralités

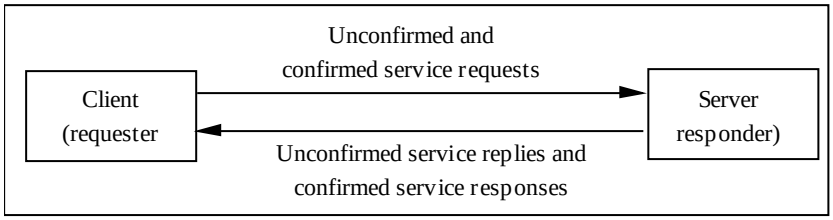
Au sein de l'environnement du bus de terrain, les AP peuvent interagir avec d'autres AP selon les besoins pour atteindre leurs objectifs fonctionnels. Aucune contrainte n'est imposée par la présente norme quant à l'organisation de ces interactions ou quant aux possibles relations qui peuvent exister entre elles.

Par exemple, dans l'environnement du bus de terrain, les interactions peuvent être basées sur des messages de demande/réponse envoyés directement entre les AP, ou sur des données/événements envoyés par un AP pour être utilisés par d'autres. Ces deux modèles d'interactions entre AP sont nommés interactions client/serveur et éditeur/abonné.

Les services pris en charge par un modèle d'interaction sont transmis les points de fin de relation entre applications (AREP) associés aux AP qui communiquent. Le rôle joué par l'AREP dans l'interaction (par exemple client, serveur, homologue, éditeur, abonné) est défini en tant qu'attribut de l'AREP.

4.3.3.3.2 Interactions client/serveur

Les interactions client/serveur sont caractérisées par un flot de données bidirectionnel entre un AP client et un ou plusieurs AP serveur. La Figure 3 illustre l'interaction entre un client unique et un serveur unique. Dans ce type d'interaction, le client peut émettre une demande confirmée ou non confirmée au serveur pour réaliser une tâche. Si le service est confirmé, le serveur retourne toujours une réponse. Si le service n'est pas confirmé, le serveur peut retourner une réponse en utilisant un service non confirmé défini à cette fin.



Légende

Anglais	Français
Unconfirmed and confirmed service requests	Demandes de service non confirmées et confirmées
Client (requester	Client (demandeur)
Server responder)	Serveur (répondeur)
Unconfirmed service replies and confirmed service responses	Réponses des services non confirmés et confirmés

Figure 3 – Interactions client/serveur

4.3.3.3.3 Interactions éditeur/abonné

4.3.3.3.3.1 Généralités

Les interactions éditeur/abonné, quant à elles, impliquent un seul AP éditeur, et un groupe d'un ou plusieurs AP abonnés. Ce type d'interaction a été défini pour prendre en charge les variantes de deux modèles d'interaction entre les AP, le modèle "par extraction" et le modèle "par émission". Dans les deux modèles, la configuration de l'AP d'édition est réalisée par la gestion et ne relève pas du domaine d'application de la présente norme.

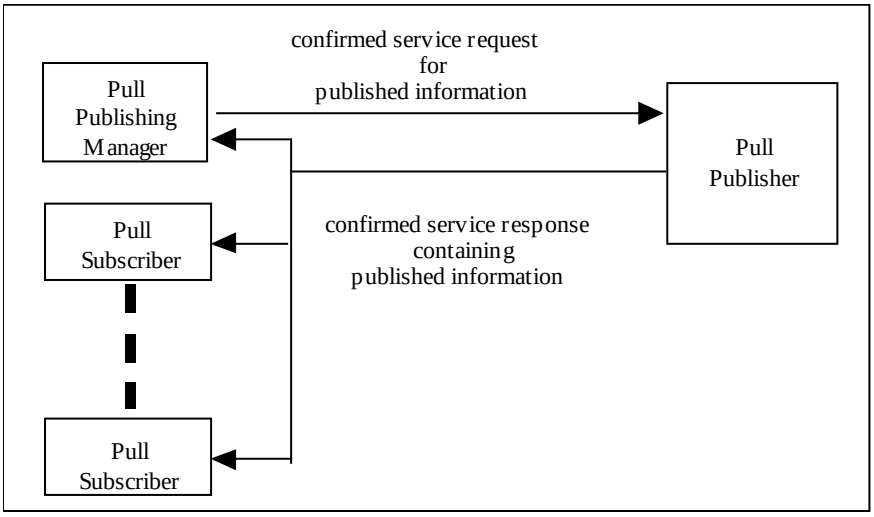
4.3.3.3.3.2 Interactions selon le modèle par extraction

Dans le modèle "par extraction", l'éditeur reçoit une demande d'édition à partir d'un *gestionnaire d'édition distant*, puis diffuse (ou multidiffuse) sa réponse sur le réseau. Le gestionnaire d'édition n'est responsable que d'initier l'édition en envoyant une demande à l'éditeur.

Les abonnés souhaitant recevoir les données publiées écoutent les réponses transmises par l'éditeur. De cette manière, les données sont "extraites" de l'éditeur par les demandes du gestionnaire d'édition.

Les services FAL confirmés sont utilisés pour prendre en charge ce type d'interaction. Deux caractéristiques de ce type d'interaction le différencient des autres types d'interaction. En premier lieu, un échange type de demande/réponse confirmé est effectué entre le gestionnaire d'édition et l'éditeur. Toutefois, le mécanisme sous-jacent d'acheminement fourni par la couche FAL retourne la réponse non seulement au gestionnaire d'édition, mais également à l'ensemble des abonnés qui souhaitent recevoir les informations publiées. Cela s'effectue via la couche DLL qui transmet la réponse à une adresse de groupe, plutôt qu'à l'adresse individuelle du gestionnaire d'édition. Par conséquent, la réponse envoyée par l'éditeur contient les données publiées et est multidiffusée au gestionnaire d'édition et à l'ensemble des abonnés.

La seconde différence concerne le comportement des abonnés. Les abonnés du modèle par émission, appelés abonnés par émission, sont capables d'accepter les données publiées dans les réponses de service confirmé sans avoir émis la demande correspondante. La Figure 4 illustre ces concepts.



Légende

Anglais	Français
confirmed service request for published information	demande de service confirmée pour les informations publiées
Pull Publishing Manager	Gestionnaire d'abonnement par extraction
Pull Publisher	Abonné par extraction
Pull Susbcriber	Abonné par extraction
confirmed service response containing published information	réponse de service confirmée contenant des informations publiées

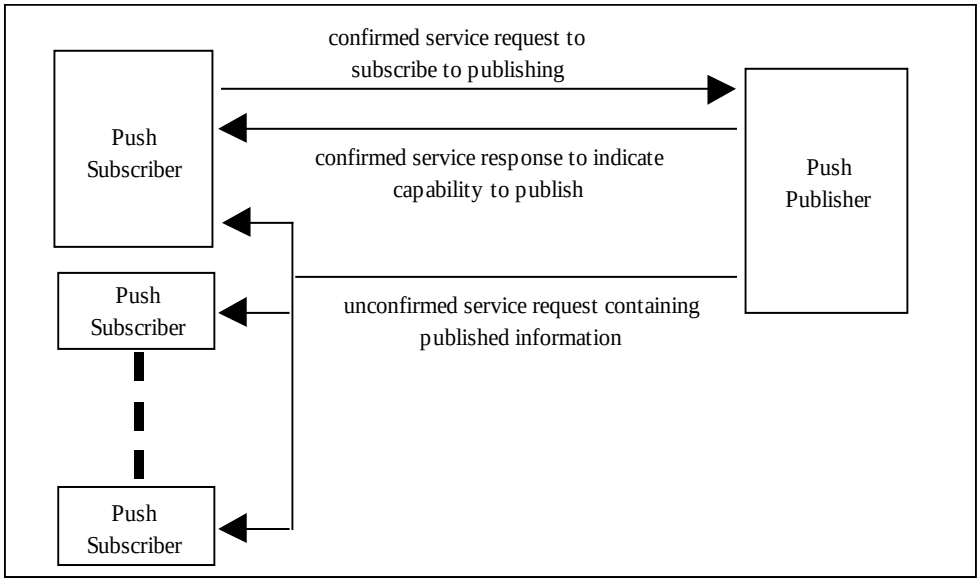
Figure 4 – Interactions selon le modèle par extraction

4.3.3.3.3.3 Interactions selon le modèle par émission

Dans le modèle "par émission", deux services peuvent être utilisés, un confirmé et un non confirmé. Le service confirmé est utilisé par l'abonné pour demander à rejoindre l'édition. La réponse à cette demande est retournée à l'abonné, en respectant le modèle client/serveur d'interaction. Cet échange n'est nécessaire que lorsque l'abonné et l'éditeur sont situés dans des AP différents.

Le service non confirmé utilisé dans le modèle par émission est utilisé par l'éditeur pour distribuer ses informations aux abonnés. Dans ce cas, l'éditeur est responsable d'appeler le service non confirmé correct au moment approprié et de fournir les informations appropriées. De cette manière, il est configuré pour "émettre" ses données sur le réseau.

Les abonnés du modèle par émission reçoivent les services non confirmés publiés distribués par les éditeurs. La Figure 5 illustre le concept du modèle par émission.



Légende

Anglais	Français
confirmed service request to subscribe to publishing	demande de service confirmée pour s'abonner à la publication
confirmed service response to indicate capability to publish	demande de service confirmée pour indiquer la capacité à publier
unconfirmed service request containing published information	demande de service non confirmée contenant des informations publiées
Push Susbcriber	Abonné par émission
Push Publisher	Editeur par émission

Figure 5 – Interactions selon le modèle par émission

4.3.3.3.3.4 Ponctualité des informations publiées

Pour prendre en charge la nature périssable des informations publiées, la couche FAL peut prendre en charge quatre types de ponctualités définies pour les interactions éditeur/abonné. Chaque type permet aux abonnés des données publiées de déterminer si les données qu'ils reçoivent sont à jour ou "dépassées". Ces types sont réalisés par l'intermédiaire de mécanismes au sein de la couche DLL. Chacun d'eux est brièvement décrit ci-dessous. Pour une description plus détaillée, se reporter à la CEI 61158-3-4 et à la CEI 61158-4-4.

Type	Description
Transparent	Ce type de ponctualité permet au processus application utilisateur de déterminer la qualité des données de ponctualité qu'il génère et de faire que la qualité de ponctualité accompagne les informations lors de leur transfert sur le réseau. Dans ce type de ponctualité, le réseau n'effectue ni calcul ni mesure de ponctualité. Il se contente de transmettre la qualité de ponctualité fournie avec les données par le processus application utilisateur.
Résidence	Lorsque la couche FAL présente des données de l'AP d'édition à la couche DLL pour les transmettre, la couche DLL démarre un temporisateur. Si le temporisateur expire avant que les données n'aient été transmises, la couche DLL marque le tampon comme "non opportun" et transmet ces informations de ponctualité avec les données.
Synchronisée	Ce type de ponctualité nécessite la coordination de deux informations publiées. Les unes sont les données à publier les autres sont une "marque de synchronisation" spéciale. Lorsque la marque de synchronisation est reçue du réseau un temporisateur démarre dans chacune des stations participantes. Par la suite, lorsque les données sont reçues pour être transmises par la couche DLL à la station d'édition, ou lorsque les données transmises sont reçues du réseau à la station de l'abonné, l'attribut de ponctualité de la couche DLL pour les données est défini sur TRUE. Il reste défini sur TRUE jusqu'à la réception de la marque de synchronisation

suivante ou jusqu'à l'expiration du temporisateur. Les données reçues après l'expiration du temporisateur mais avant la marque de synchronisation suivante ne provoquent pas la réinitialisation de l'attribut de ponctualité sur TRUE. Il n'est réinitialisé sur TRUE que si les données sont reçues dans la fenêtre temporelle après réception de la marque de synchronisation. Les données transmises par la station de l'éditeur avec l'attribut de ponctualité défini sur FALSE conservent le paramètre FALSE à chacun des abonnés, quel que soit le fonctionnement de leur temporisateur.

Mettre à jour — Ce type de ponctualité nécessite la coordination de deux mêmes informations publiées définies pour la ponctualité *synchronisée*. Dans ce type, la marque de synchronisation démarre également un temporisateur dans chacune des stations participantes. De même que la ponctualité *synchronisée*, l'expiration du temporisateur provoque toujours la définition de l'attribut de ponctualité sur FALSE. Contrairement à la ponctualité *synchronisée*, la réception de nouvelles données à tout moment (pas seulement dans la fenêtre temporelle démarrée après réception de la marque de synchronisation) provoque toujours la définition de l'attribut de ponctualité sur TRUE.

4.3.3.4 Structure des AP

La structure des AP peut être représentée par un ou plusieurs objets de processus d'application (APO) et accessible par une ou plusieurs entités d'application (AE). Les AE fournissent les capacités de communication de l'AP. Pour chaque AP de bus de terrain, il n'existe qu'une seule entité AE de couche FAL. Les APO sont la représentation en réseau des capacités propres à l'application (objets de processus d'application utilisateur) d'un AP qui sont accessibles par l'intermédiaire de son entité AE de couche FAL.

4.3.3.5 Classe AP

Une classe AP est une définition des attributs et des services d'un AP. La définition de classe standard des AP est définie dans la présente norme. Des classes définies par l'utilisateur peuvent également être spécifiées. Les identificateurs de classe (décrits en 3.8.2) sont attribués à partir d'un ensemble réservé à cette fin.

4.3.3.6 Type AP

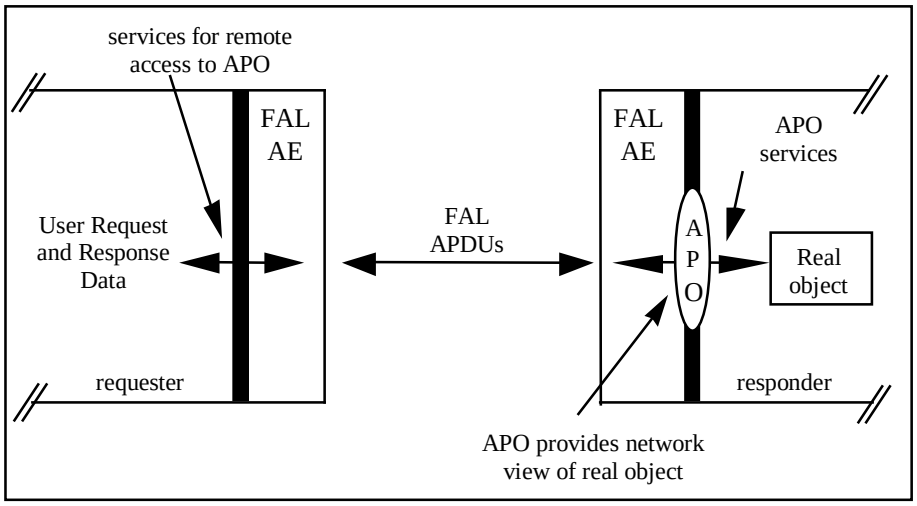
Comme décrit ci-dessus dans les paragraphes précédents, les AP sont définis en instanciant une classe AP. Chaque définition d'AP est composée des attributs et des services sélectionnés pour l'AP à partir de ceux définis par sa classe AP. De surcroît, une définition d'AP contient des valeurs pour un ou plusieurs des attributs sélectionnés pour celle-ci. Lorsque deux AP partagent la même définition, celle-ci est appelée type AP. Ainsi, un type AP est une spécification générique d'un AP qui peut être utilisé pour définir un ou plusieurs AP.

4.3.4 Objets de processus d'application (APO)

4.3.4.1 Définition d'un APO

Un objet de processus d'application (APO) est une représentation en réseau d'un aspect particulier d'un AP. Chaque APO représente un ensemble particulier d'information et de capacités de traitement d'un AP qui sont accessibles par l'intermédiaire des services de la couche FAL. Les APO sont utilisés pour représenter ces capacités vers d'autres AP dans un système de bus de terrain.

Du point de vue de la couche FAL, un APO est modélisé en tant qu'objet accessible à partir du réseau contenu au sein d'un AP ou d'un autre APO (les APO peuvent contenir d'autres APO). Les APO fournissent la définition du réseau pour les objets contenus au sein d'un AP qui sont accessibles à distance. La définition d'un APO inclut une identification des services FAL qui peuvent être utilisés par les AP distants pour l'accès à distance. Les services FAL, illustrés à la Figure 6, sont fournis par l'entité de communications de la couche FAL de l'AP, connue sous le nom d'entité d'application FAL (AE FAL).



Légende

Anglais	Français
services for remote access to APO	services pour accéder à distance à APO
User Request and Response Data	Données de demande et de réponse utilisateur
requester	demandeur
APO services	Services APO
Real object	Objet réel
responder	répondeur
APO provides network view of real object	APO fournit une vue réseau d'un objet réel

Figure 6 – Services APO transmis par la couche FAL

Dans la Figure 6, les AP distants agissant comme des clients peuvent accéder à l'objet réel en envoyant des demandes par l'intermédiaire de l'APO qui représente l'objet réel. Les aspects locaux de l'AP convertissent entre la vue réseau (l'APO) de l'objet réel et la vue AP interne de l'objet réel.

Pour prendre en charge le modèle d'interaction éditeur/abonné, les informations concernant l'objet réel peuvent être publiées par l'intermédiaire de son APO. Les AP distants agissant en tant qu'abonnés voient la vue APO des informations publiées au lieu de devoir connaître les détails spécifiques de l'objet réel.

4.3.4.2 Classes APO

Une classe APO est spécification générique pour un ensemble d'APO, chacun étant décrit par le même ensemble d'attributs et accessible en utilisant le même ensemble de services.

Les classes APO fournissent le mécanisme permettant la normalisation des aspects réseau visibles des AP. Chaque définition de classe APO standard spécifie un ensemble particulier d'attributs et de services AP accessible sur le réseau. La CEI 61158-6-4 spécifie la syntaxe et les procédures utilisées par le protocole de la couche FAL pour fournir l'accès à distance aux attributs et aux services d'une classe APO.

Les classes standard APO sont spécifiées par la présente norme dans le but de normaliser l'accès distant aux AP. Des classes définies par l'utilisateur peuvent également être spécifiées.

Les classes définies par l'utilisateur sont définies en tant que sous-classes des classes APO normalisées ou d'autres classes définies par l'utilisateur. Elles peuvent être définies en identifiant de nouveaux attributs ou en indiquant que les attributs facultatifs pour la classe parent sont obligatoires pour la sous-classe. Les conventions de définition des classes définies en 3.8.2 peuvent être utilisées à cette fin. La méthode permettant d'enregistrer ou de mettre ces nouvelles définitions de classe à la disposition du public ne relève pas du domaine d'application de la présente norme.

4.3.4.3 APO en tant qu'instances des classes APO

Les classes APO sont définies dans la présente norme à l'aide de modèles. Ces modèles sont utilisés non seulement pour définir les classes APO, mais également pour spécifier les instances d'une classe.

Chaque APO défini pour un AP est une instance d'une classe APO. Chaque APO fournit la vue réseau d'un objet réel contenu dans un AP. Un APO est défini par

(1) la sélection des attributs à partir de son modèle de classe APO qui doivent être accessibles à partir de l'objet réel;

(2) l'affectation de valeurs à un ou plusieurs attributs indiqués comme attributs clés dans le modèle. les attributs clés sont utilisés pour identifier l'APO;

(3) l'affectation de valeurs à zéro, un ou plusieurs attributs non-clés pour l'APO. les attributs non-clés sont utilisés pour caractériser l'APO;

(4) la sélection des services à partir du modèle qui peuvent être utilisés par des AP distants pour accéder à l'objet réel.

Le Paragraphe 3.8.2 de la présente norme spécifie les conventions des modèles de classe. Ces conventions fournissent la définition des attributs et des services obligatoires, facultatifs et conditionnels.

Les attributs et les services obligatoires doivent être présents dans tous les APO de la classe. Les attributs et les services facultatifs peuvent être sélectionnés, sur un APO par base APO, pour inclusion dans un APO. Les attributs et les services conditionnels sont définis avec un énoncé de contrainte qui les accompagne. Les énoncés de contrainte spécifient les conditions qui indiquent si l'attribut doit être présent ou non dans un APO.

4.3.4.4 Types APO

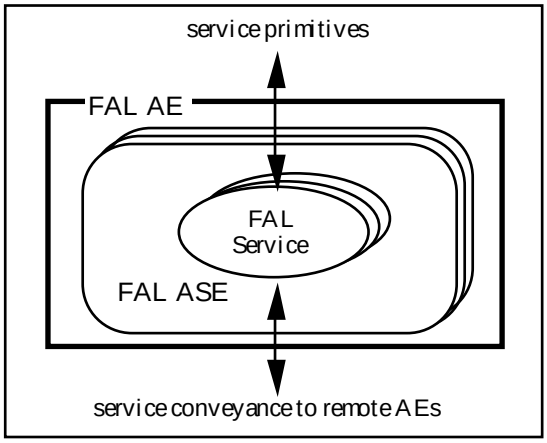
Les types APO fournissent le mécanisme permettant la définition des APO standard.

Comme décrit ci-dessus dans les paragraphes précédents, les APO sont définis en instanciant une classe APO. Chaque définition d'APO est composée des attributs et des services sélectionnés pour l'APO à partir de ceux définis par sa classe APO. De surcroît, une définition d'APO contient des valeurs pour un ou plusieurs des attributs sélectionnés pour celle-ci. Lorsque deux APO partagent la même définition, excepté pour la configuration des attributs clés, cette définition est appelée type APO. Ainsi, un type APO est une spécification générique d'un APO qui peut être utilisé pour définir un ou plusieurs APO.

4.3.5 Entités d'application

4.3.5.1 Définition de FAL AE

Une entité d'application fournit les capacités de communication d'un AP unique. Un FAL AE fournit un ensemble de services ainsi que les protocoles de prise en charge qui permettent les communications entre AP dans un environnement du bus de terrain. Les services fournis par les FAL AE sont regroupés dans des éléments de service d'application (ASE), tels que les services FAL fournis à un AP, sont définis par les ASE que contient son FAL AE. La Figure 7 illustre ce concept.



Légende

Anglais	Français
service primitives	primitives de service
FAL Service	Service FAL
service conveyance to remote AEs	transport de service vers AE à distance

Figure 7 – Structure d'entité d'application

4.3.5.2 Type AE

Les entités d'application qui fournissent le même ensemble d'ASE sont du même type AE. Deux AE partageant un ensemble commun d'ASEs sont capables de communiquer ensemble.

4.3.6 Eléments de service d'application de bus de terrain

4.3.6.1 Généralités

Un élément de service d'application (ASE), tel que défini dans l'ISO/CEI 9545, est un ensemble de fonctions d'application qui offrent une capacité d'interfonctionnement des invocations d'entités d'application à des fins particulières. Les ASE fournissent un ensemble de services pour transmettre des demandes et des réponses vers et à partir des processus d'application et de leurs objets. Les AE, tels que définis ci-dessus, sont représentés par un ensemble d'invocations ASE au sein de l'AE.

4.3.6.2 Services de la FAL

Les services de la FAL transmettent les demandes/réponses fonctionnelles entre les AP. Chaque service de la FAL est défini pour transmettre les demandes et les réponses pour accéder à un objet réel modélisé en tant qu'objet FAL accessible.

La couche FAL définit les services confirmés et non confirmés. Les demandes de service confirmé sont envoyées à l'AP contenant l'objet réel. Une invocation d'une demande de service confirmé peut être identifiée par un ID d'invocation fourni par l'utilisateur. Cet ID d'invocation est retourné dans la réponse par l'AP contenant l'objet réel. Lorsqu'il est présent, il est utilisé par l'AP demandeur et son FAL AE pour associer la réponse avec la demande appropriée.

Les services non confirmés peuvent être envoyés à partir de l'AP contenant l'objet réel pour envoyer des informations concernant l'objet. Ils peuvent également être envoyés à l'AP contenant l'objet réel pour accéder à l'objet réel. Les deux types de services non confirmés peuvent être définis pour la couche FAL.

4.3.6.3 Définition des FAL ASE

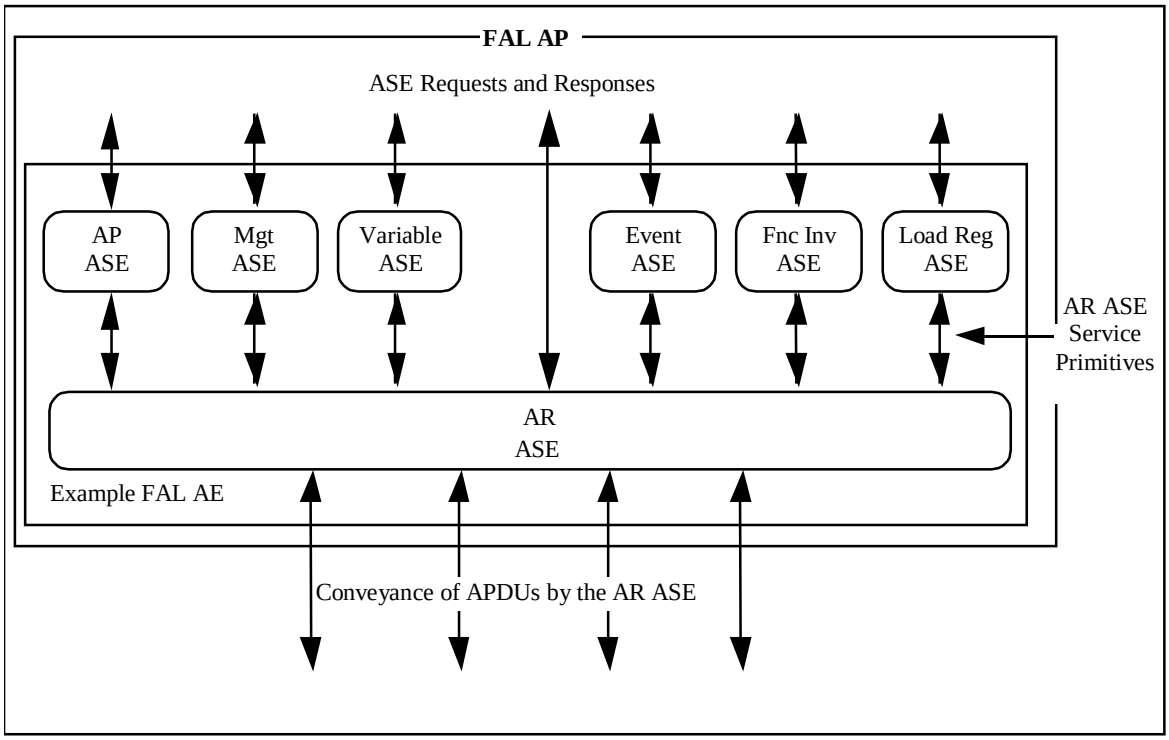
4.3.6.3.1 Généralités

La définition des FAL ASE suit une approche modulaire. Les ASE définis pour la couche FAL sont également orientés objet. En général, les ASE fournissent un ensemble de services conçus pour une classe d'objets spécifique ou pour un ensemble associé de classes. Les ASE de gestion d'objets communs, lorsqu'ils sont présents, fournissent un ensemble commun de services de gestion applicable à toutes les classes d'objets.

Pour prendre en charge l'accès distant à l'AP, l'ASE de relations entre applications est défini. Il fournit des services à l'AP pour définir et établir des relations entre applications avec d'autres AP, et il fournit des services aux autres ASE pour transmettre leurs demandes et leurs réponses de service.

Chaque FAL ASE définit un ensemble de services, d'APDU et de procédures qui opèrent sur les classes qu'il représente. Seul un sous-ensemble des services ASE peut être fourni pour satisfaire aux besoins d'une application. Des profils peuvent être utilisés pour définir de tels sous-ensembles. La définition des profils ne relève pas du domaine d'application de la présente norme.

Les APDU sont envoyés et reçus entre les FAL ASE qui prennent en charge les mêmes services. Chaque FAL AE contient, au minimum, l'AR ASE et au moins un autre ASE. La Figure 8 illustre un exemple d'ensemble de FAL ASE ainsi que leurs relations d'architecture. Tous les APO ASE suivent cet exemple.



Légende

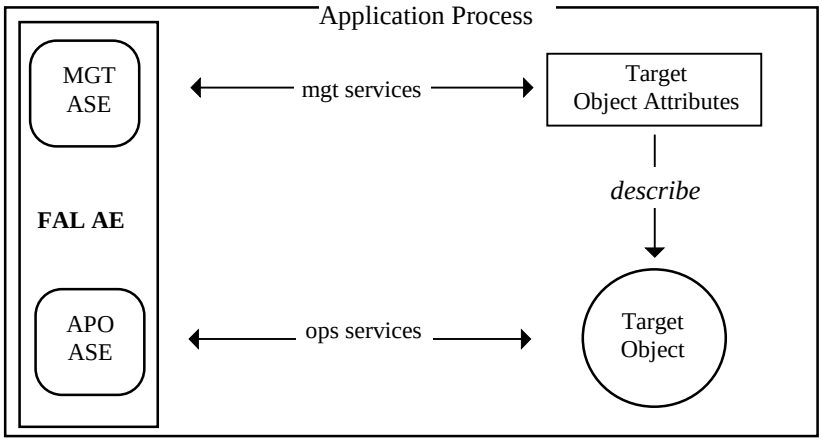
Anglais	Français
ASE Requests and Responses	Demandes et réponses ASE
Mgt ASE	Gérer ASE
Variable ASE	Elément ASE de variable
Event ASE	Elément ASE d'événement

Anglais	Français
Fnc Inv ASE	Fnc Inv ASE
Load Reg ASE	Charger ASE enregistré
AR ASE Service Primitives	Primitives de service AR ASE
Example FAL AE	Exemple de FAL AE
Conveyance of APDUs by the AR ASE	Acheminement d'unités de données de protocole application par l'AR ASE

Figure 8 – Exemple de FAL ASE

4.3.6.3.2 Gestion d'objets ASE

Un ASE de gestion d'objets spécial peut être spécifié pour la couche FAL pour fournir des services pour la gestion des objets. Ses services sont utilisés pour accéder aux attributs d'objet, et créer et supprimer des instances d'objets. Ces services sont utilisés pour gérer les objets AP visibles du réseau accessibles par l'intermédiaire de la couche FAL. Les services opérationnels spécifiques qui s'appliquent à chaque type d'objet sont spécifiés dans la définition de l'ASE pour le type d'objet. La Figure 9 illustre l'intégration de la gestion et des services opérationnels pour un objet au sein d'un AP.



Légende

Anglais	Français
Application Process	Processus d'application
mgt services	services mgt
ops services	services ops
Target Object Attributes	Attributs d'objet cible
describe	décrire
Target Object	Objet cible

Figure 9 – Gestion FAL des objets

4.3.6.3.3 AP ASE

Un AP ASE peut être spécifié pour l'identification et le contrôle des FAL AP. Les attributs définis par l'AP ASE spécifient les caractéristiques de l'AP concernant son fabricant et énumèrent son contenu et ses capacités.

4.3.6.3.4 APO ASE

La couche FAL spécifie un ensemble d'ASE avec les services définis pour accéder aux APO d'un AP. Les APO ASE définis pour la couche FAL sont définis par chaque modèle de communication.

4.3.6.3.5 AR ASE

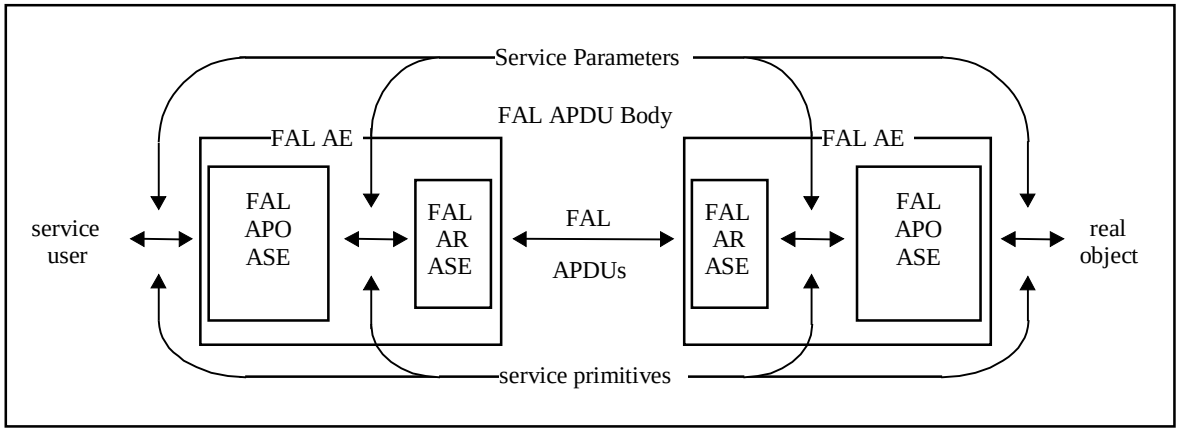
Un AR ASE est spécifié pour établir et gérer les relations entre applications (AR) qui sont utilisées pour transmettre les FAL APDU entre/parmi les AP. Les AR représentent les canaux de communication de la couche application entre les AP. Les AR ASE sont responsables de fournir les services aux points de fin des AR. Les services AR ASE peuvent être définis pour établir, rompre et abandonner les AR, pour transmettre les APDU pour l'AE, et pour indiquer l'état local de l'AR à l'utilisateur. De plus, des services locaux peuvent être définis pour accéder à certains aspects des points de fin des AR.

4.3.6.4 Transport des services FAL

Les FAL APO ASE fournissent des services pour transmettre les demandes et les réponses entre les utilisateurs du service et les objets réels.

Pour accomplir la tâche de transmettre des demandes et des réponses de service, trois types d'activités pour l'utilisateur expéditeur et trois types correspondants pour l'utilisateur récepteur sont définis. Du côté de l'utilisateur expéditeur, ils acceptent les demandes et les réponses de service à transmettre. Ensuite, ils sélectionnent le type de FAL APDU qui sera utilisé pour transmettre la demande ou la réponse et coder les paramètres de service sans sa portion du corps. Puis, ils soumettent de corps APDU codé à l'AR ASE pour le transport.

Du côté de l'utilisateur récepteur, ils reçoivent les corps APDU codés à partir de l'AR ASE. Ils décodent les corps APDU et extraient les paramètres de service qu'ils ont transmis. Pour conclure le transport, ils fournissent la demande ou la réponse de service à l'utilisateur. La Figure 10 illustre ces concepts.



Légende

Anglais	Français
Service Parameters	Paramètres de service
service primitives	primitives de service
service user	utilisateur service
real object	objet réel
FAL ADPU Body	Corps FAL ADPU

Figure 10 – Transport des services ASE

4.3.6.5 Contexte de présentation de la couche FAL

Le contexte de présentation de l'environnement OSI permet de distinguer les APDU d'un ASE et d'un autre, et d'identifier les règles de syntaxe de transfert utilisées pour coder chaque APDU. Toutefois, l'architecture de communication des bus de terrain n'inclut pas la couche de présentation. Par conséquent, un autre mécanisme est fourni par la couche FAL par chacun des types particuliers de modèles de communication.

4.3.7 Relations entre applications

4.3.7.1 Définition de AR

Les AR représentent les canaux de communication entre les AP. Ils définissent la manière dont les informations sont communiquées entre les AP. Chaque AR est caractérisé par la manière dont il transmet les demandes et les réponses de service ASE d'un AP à un autre. Ces caractéristiques sont décrites ci-dessous.

4.3.7.2 Points d'extrémité AR

Les AR sont définis en tant qu'ensemble d'AP coopérant. L'AR ASE de chaque AP gère un point d'extrémité de l'AR, et gère son contexte local. Le contexte local d'un point d'extrémité de l'AR est utilisé par l'AR ASE pour contrôler le transport des APDU sur l'AR.

4.3.7.3 Classes de points d'extrémité AR

Les AR sont composés d'un ensemble de points d'extrémité de classes compatibles. Les classes de points d'extrémité AR sont utilisées pour représenter des points d'extrémité AR qui acheminent des APDU de la même manière. Par l'intermédiaire de la normalisation des classes de points d'extrémité, des AR pour différents modèles d'interaction peuvent être définis.

4.3.7.4 Cardinalité d'AR

Les AR caractérisent les canaux communications entre les AP. L'une des caractéristiques d'un AR est le nombre de points d'extrémité de l'AR. Les AR qui transmettent des services entre deux AP ont une cardinalité de 1-à-1. Ceux qui transmettent des services d'un AP à plusieurs AP ont une cardinalité de 1-à-plusieurs. Ceux qui transmettent des services de ou vers plusieurs AP ont une cardinalité de plusieurs-à-plusieurs.

4.3.7.5 Accès aux objets par l'intermédiaire des AR

Les AR donnent accès aux AP et aux objets qu'ils contiennent par l'intermédiaire des services d'un ou plusieurs ASE. Par conséquent, une caractéristique est l'ensemble des services ASE qui peuvent être transmis vers et à partir de ces objets par l'AR. Les services qui peuvent être transmis par l'AR sont sélectionnés à partir de ceux définis pour l'AE.

4.3.7.6 Chemins de transport entre AR

Les AR sont modélisés en tant qu'un ou deux chemins de transport entre points de fin des AR. Chaque chemin de transport transmet des APDU dans une direction entre un ou plusieurs points de fin des AR. Chaque point de fin de l'AR récepteur pour un chemin de transport reçoit tous les APDU émettant sur l'AR par le point de fin de l'AR expéditeur.

4.3.7.7 Rôles AREP

Du fait que les AP interagissent les uns avec les autres par l'intermédiaire des points de fin, un déterminant de base de leur compatibilité est le rôle qu'ils jouent dans l'AR. Le rôle définit comment un AREP interagit avec les autres AREP dans l'AR.

Par exemple, un AREP peut fonctionner en tant que client, serveur, éditeur ou abonné. Lorsqu'un AREP interagit avec un autre AREP sur un AR unique à la fois en tant que client et serveur, il est défini comme ayant le rôle d'un "homologue".

Certains rôles peuvent être capables de démarrer des demandes de service, tandis que d'autres peuvent limiter leur capacité à la simple réponse aux demandes de service. Cette partie de la définition d'un rôle identifie les exigences d'un AR pour qu'il soit capable de transmettre les demandes dans les deux directions, ou seulement dans une seule.

4.3.7.8 Tampons et files d'attente des AREP

Les AREP peuvent être modélisés en tant que file d'attente ou de tampon. Les APDU transférés sur un AREP mis en file d'attente sont fournis dans l'ordre reçu pour le transport. Le transfert des APDU sur un AREP mis en mémoire tampon est différent. Dans ce cas, un APDU à transmettre par l'AR ASE est placé dans une mémoire tampon pour le transfert. Lorsque la couche DLL a accès au réseau, elle transmet le contenu de la mémoire tampon.

Lorsque l'AR ASE reçoit une autre demande de transport, il remplace le contenu précédent de la mémoire tampon qu'il ait été transmis ou non. Lorsqu'un APDU est écrit dans une mémoire tampon pour être transféré, il est conservé dans la mémoire tampon jusqu'à ce que l'APDU suivant à transmettre le remplace. Lorsqu'il se trouve en mémoire tampon, un APDU peut être lu plusieurs fois sans le supprimer de la mémoire tampon ou sans en changer le contenu.

A l'extrémité de réception, l'opération est similaire. Le point de fin de réception place un APDU reçu dans une mémoire tampon pour que l'AR ASE puisse y accéder. Lorsqu'un APDU suivant est reçu, il remplace l'APDU précédent dans la mémoire tampon qu'il ait été lu ou non. La lecture de l'APDU à partir de la mémoire tampon n'est pas destructive — elle ne détruit pas ou ne change pas le contenu de la mémoire tampon, ce qui permet au contenu d'être lu à partir de la mémoire tampon une ou plusieurs fois.

4.3.7.9 Transport déclenché par l'utilisateur et planifié

Une autre caractéristique d'un AREP apparaît lorsqu'ils transmettent les demandes et les réponses de service. Les AREP qui les transmettent lors de la soumission par l'utilisateur sont nommés déclenchés par l'utilisateur. Leur transport est asynchrone par rapport au fonctionnement du réseau.

Les AREP qui transmettent les demandes et les réponses selon des intervalles prédéfinis, quel que soit le moment où ils sont reçus pour être transférés sont nommés planifiés. Les AREP planifiés peuvent être capables d'indiquer quand des données transférées ont été soumises en retard pour la transmission, ou lorsqu'elles ont été soumises dans les délais, mais transmises en retard.

4.3.7.10 Ponctualité des AREP

Les AREP transmettent les APDU entre applications à l'aide des services de la couche DLL. Lorsque les capacités de ponctualité sont définies pour un AREP et prises en charge par la couche DLL, l'AREP fait suivre les indicateurs de ponctualité fournis par la couche DLL. Ces indicateurs de ponctualité permettent aux abonnés des données publiées de déterminer si les données qu'ils reçoivent sont à jour ou "dépassées".

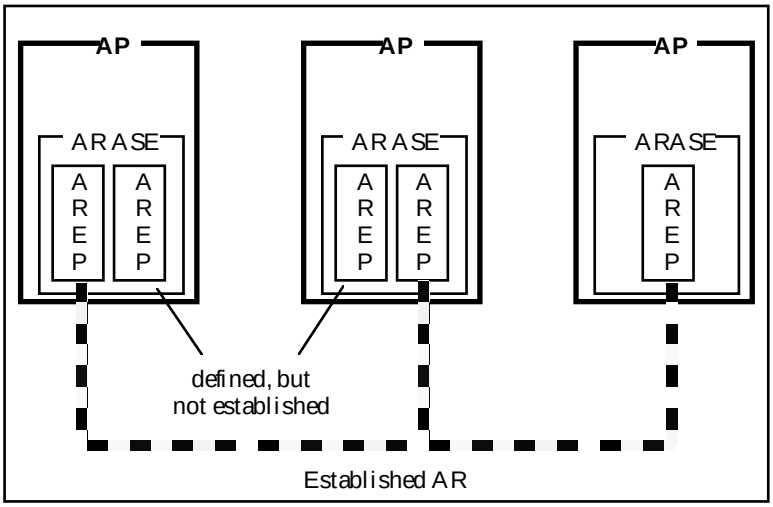
Pour prendre en charge ces types de ponctualité, l'AREP d'édition établit une connexion de la liaison de données éditeur reflétant le type de ponctualité configuré pour lui par la gestion. Après l'établissement de la connexion, l'AREP reçoit les données de l'utilisateur et les envoie à la couche DLL pour transmission, où des procédures de ponctualité sont effectuées. Lorsque la couche DLL a la possibilité de transmettre les données, elle transmet l'état actuel de la ponctualité avec les données.

A l'AREP abonné, une connexion de liaison de données est ouverte pour recevoir les données publiées qui reflètent le type de ponctualité configuré pour lui par la gestion. La couche DLL calcule la ponctualité des données reçues, puis les fournit à l'AREP. Les données sont ensuite fournies à l'AP de l'utilisateur par l'intermédiaire de l'ASE approprié.

4.3.7.11 Définition et création des AREP

Les définitions d'AREP spécifient les instances des classes AREP. Les AREP peuvent être prédéfinis ou ils peuvent être définis en utilisant un service "create" si leur AE prend en charge cette capacité.

Les AREP peuvent être prédéfinis et préétablis ou être prédéfinis et établis de façon dynamique. La Figure 11 décrit ces deux cas. Les AREP peuvent également nécessiter à la fois la définition et l'établissement dynamique ou être définis de façon dynamique de telle sorte qu'ils puissent être utilisés sans aucun établissement (ils sont définis dans un état établi).



Légende

Anglais	Français
defined, but not established	défini, mais pas établi
Established AR	AR établi

Figure 11 – AREP définis et établis

4.3.7.12 Etablissement et arrêt des AR

Les AR peuvent être établis soit avant la phase opérationnelle de l'AP ou pendant son fonctionnement. Lorsqu'il est établi lors du fonctionnement d'un AP, l'AR est établi par l'intermédiaire de l'échange des AR APDU.

Lorsqu'un AR a été établi, il peut être arrêté normalement ou être abandonné, selon les capacités de l'AR.

4.4 Désignation et adressage de la couche Application de bus de terrain

4.4.1 Généralités

Les principes définis dans l'ISO 7498-3 qui implique l'identification (désignation) et l'emplacement (adressage) APO référencés par l'intermédiaire de la couche Application de bus de terrain sont détaillés en 4.4.

La manière dont les noms et les identificateurs numériques sont utilisés pour identifier les APO accessibles par l'intermédiaire de la couche FAL est détaillée en 4.4. Le Paragraphe 4.4 indique également comment les adresses des couches sous-jacentes sont utilisées pour localiser les AP dans l'environnement du bus de terrain.

4.4.2 Identification des objets accessibles par l'intermédiaire de la couche FAL

4.4.2.1 Généralités

Les APO accessibles par l'intermédiaire de la couche FAL sont identifiés indépendamment de leur emplacement. C'est-à-dire que si l'emplacement de l'AP qui contient l'APO change, l'APO peut toujours être référencé en utilisant le même ensemble d'identificateurs.

Les identificateurs pour les AP et les APO au sein de la couche FAL sont définis comme les attributs clés dans les définitions de classe pour les APO. Au sein de ces définitions d'APO, deux types d'attributs clés sont couramment utilisés, les noms et les identificateurs numériques.

4.4.2.2 Noms

Les noms sont des identificateurs orientés chaîne. Ils sont définis pour permettre de nommer les AP et les APO au sein du système où ils sont utilisés. Ainsi, bien que l'étendue du nom d'un APO soit spécifique de l'AP dans lequel il réside, l'attribution du nom est administrée au sein du système où il est configuré.

Les noms peuvent être descriptifs, bien que ce ne soit pas une nécessité. Les noms descriptifs permettent de fournir des informations significatives, par exemple l'utilisation, concernant l'objet qu'ils nomment.

Les noms peuvent également être codés. Les noms codés permettent d'identifier un objet en utilisant une forme courte, compressée d'un nom. Ils sont généralement plus simples à transférer et à traiter, mais sont moins simples à comprendre que les noms descriptifs.

4.4.2.3 Identificateurs numériques

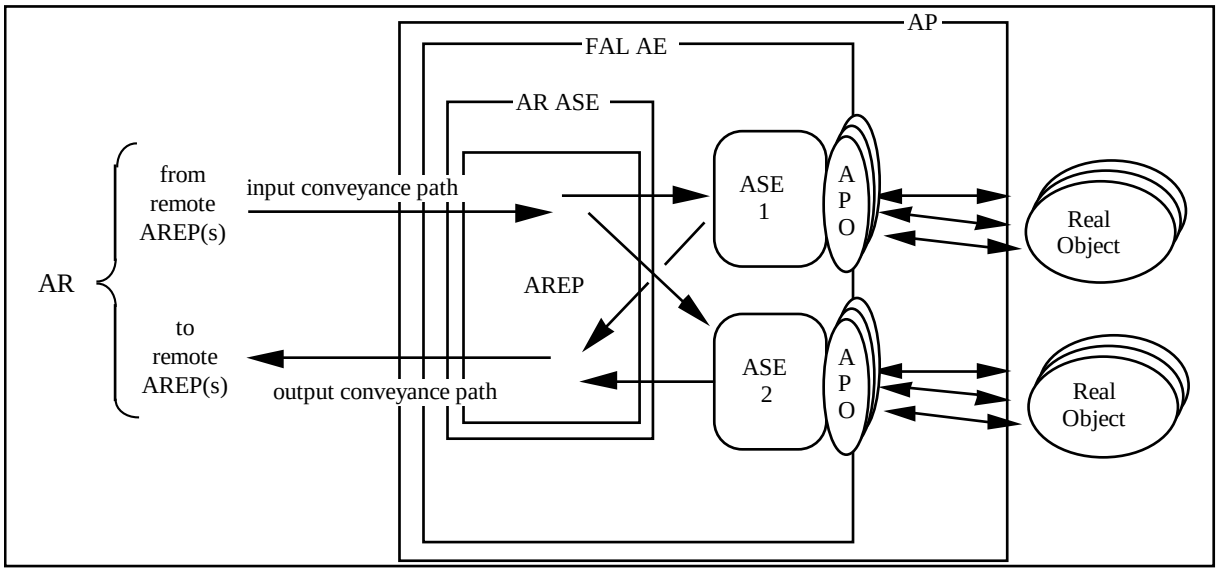
Les identificateurs numériques sont ceux dans lesquels les valeurs sont des nombres. Ils sont conçus pour une utilisation efficace au sein du système de bus de terrain et peuvent être attribués pour l'accès efficace aux APO par leur AP.

4.4.3 Adressage des AP accessibles par l'intermédiaire de la couche FAL

Les adresses de bus de terrain représentent les emplacements réseau des AP. Les adresses associées à la couche FAL sont celles des couches sous-jacentes qui sont utilisées pour localiser les AREP d'un AP.

4.5 Résumé de l'architecture

Le résumé de l'architecture de la couche FAL est présenté en 4.5. La Figure 12 illustre les principaux composants de l'architecture de la couche FAL ainsi que la manière dont ils sont reliés.



Légende

Anglais	Français
input conveyance path	chemin de transport d'entrée
output conveyance path	chemin de transport de sortie
from remote AREP(s)	depuis AREP à distance
to remote AREP(s)	vers AREP à distance

Figure 12 – Composants d'architecture de la couche FAL

La Figure 12 décrit un AP qui communique par l'intermédiaire du FAL AE. L'AP représente ses objets internes réels en tant qu'APO pour l'accès distant à ceux-ci. Deux ASE sont illustrés qui fournissent les services d'accès à distance à leurs APO associés. L'AR ASE contient un AREP unique qui transmet les demandes et les réponses de service pour les ASE à un ou plusieurs AREP distants situés dans des AP distants.

4.6 Procédure de service FAL

4.6.1 Procédures de service confirmé FAL

L'utilisateur demandeur soumet une primitive de demande de service confirmé à sa couche FAL. L'élément ASE de la couche FAL génère le corps APDU de la demande de service confirmé associée et la transmet sur l'AR spécifié.

A la réception du corps APDU de la demande de service confirmé, l'ASE qui répond le décode. Si aucune erreur de protocole ne s'est produite, l'ASE envoie une primitive indication de service confirmé à son utilisateur.

Si l'utilisateur qui répond est capable de traiter la demande avec succès, il retourne une primitive (+) de réponse de service confirmé.

Si l'utilisateur qui répond n'est pas capable de traiter la demande avec succès, le service échoue et l'utilisateur émet une primitive (-) de réponse de service confirmé qui en indique le motif.

L'ASE qui répond génère un corps APDU de réponse de service confirmé pour une primitive (+) de réponse de service confirmé ou un corps APDU d'erreur de service confirmé pour une primitive (-) de réponse de service confirmé et le transmet sur l'AR spécifié.

A la réception du corps APDU retourné, l'ASE initiateur fournir une primitive de confirmation de service confirmé à son utilisateur qui indique le succès ou l'échec, ainsi que le motif de l'échec le cas échéant.

4.6.2 Procédures de service non confirmé FAL

L'utilisateur demandeur soumet une primitive de demande de service non confirmé à l'entité AE de sa couche FAL. L'élément ASE de la couche FAL génère le corps APDU de la demande de service non confirmé associée et la transmet sur l'AR spécifié.

A la réception d'un corps APDU de demande de service non confirmé, les ASE récepteur qui participent à l'AR fournissent la primitive d'indication de service non confirmé appropriée à son utilisateur. Les paramètres de ponctualité sont inclus dans la primitive d'indication si l'AR qui a transmis le corps APDU prend en charge la ponctualité.

4.7 Attributs FAL courants

Dans les spécifications des classes FAL qui suivent, un grand nombre de classes utilisent les attributs suivants. Par conséquent, ces attributs sont définis ici au lieu de l'être avec les autres attributs pour chacune des classes, excepté pour la classe du type de données.

ATTRIBUTS:

1	(o)	Attribut clé:	Identificateur numérique
2	(o)	Attribut clé:	Nom
3	(o)	Attribut:	Description d'utilisateur
4	(o)	Attribut:	Révision de l'objet

1) Identificateur numérique

Cet attribut clé facultatif spécifie l'identificateur numérique de l'objet. Il est utilisé comme une référence abrégée par le protocole de la couche FAL pour identifier l'objet. Il existe trois possibilités pour l'identification: un identificateur numérique ou un nom ou les deux. Cet attribut doit être présent pour le modèle type de données.

2) Nom

Cet attribut clé facultatif spécifie le nom de l'objet. Il existe trois possibilités pour l'identification: un identificateur numérique ou un nom ou les deux.

3) Description d'utilisateur

Cet attribut facultatif contient des informations descriptives fournies par l'utilisateur concernant l'objet.

4) Révision de l'objet

Cet attribut facultatif spécifie le niveau de révision de l'objet. C'est un attribut structuré composé des numéros des révisions majeures et mineures. Si la révision de l'objet est prise en charge, il contient à la fois une révision majeure et une révision mineure avec une plage de valeurs de 0 à 15 pour chacune. L'utilisation des champs majeur/mineur est destinée à fournir les fonctions suivantes:

– Révision majeure

Le champ révision majeure contient la valeur de révision majeure de l'objet. Un changement de révision majeure indique que l'interopérabilité est affectée par le changement.

– Révision mineure

Le champ révision mineure contient la valeur de révision mineure de l'objet. Un changement de révision mineure indique que l'interopérabilité n'a pas été affectée par le changement -- c'est-à-dire que les utilisateurs de l'objet seront toujours en mesure d'interopérer avec l'objet lorsque sa révision mineure aura changé, dans la mesure où la révision majeure reste la même.

4.8 Paramètres communs aux services de la FAL

Dans les spécifications des services FAL qui suivent, un grand nombre de services utilisent les paramètres suivants. Par conséquent, ils sont définis ici au lieu de l'être avec les autres paramètres pour chacun des services.

– AREP

Ce paramètre contient des informations suffisantes pour identifier localement l'AREP devant être utilisé pour acheminer le service: Ce paramètre peut utiliser un attribut clé de l'AREP pour identifier la relation entre applications. Lorsqu'un AREP prend en charge plusieurs contextes (établis en utilisant le service de lancement) en même temps, le paramètre AREP est étendu pour identifier le contexte ainsi que l'AREP.

– ID d'invocation

Ce paramètre identifie cette invocation du service. Il est utilisé pour associer des demandes à des réponses. Par conséquent, deux quelconques invocations de services en cours ne peuvent pas être identifiées par la même valeur d'ID d'invocation.

– Classe FAL ASE/FAL

Ce paramètre spécifie l'élément ASE de la couche FAL (par exemple AP, AR, Variable, Type de données, Evénement, Invocation de fonctions, Région de charge) et la classe FAL au sein de l'ASE (par exemple AREP, Liste de variables, Notificateur, Action).

– ID numérique

Ce paramètre est l'identificateur numérique de l'objet.

– Informations d'erreur

Ce paramètre fournit des informations d'erreur pour les erreurs de service. Il est renvoyé dans les primitives (-) de réponse de service confirm. Il se compose des éléments suivants:

- Classe d'erreurs

  Ce paramètre indique la classe générale d'erreur. Les valeurs valides sont spécifiées dans la définition du paramètre Code d'erreur, ci-dessous.

- Code d'erreur

  Ce paramètre identifie l'erreur de service spécifique.

- Code supplémentaire

  Ce paramètre facultatif identifie l'erreur qui s'est produite lors du traitement de la demande spécifique à l'objet auquel l'accès est effectué. Lorsqu'il est utilisé, la valeur présentée dans la primitive "response" est délivrée inchangée dans la primitive "confirmation".

- Informations supplémentaires

  Ce paramètre facultatif contient les données d'utilisateur qui accompagnent la réponse négative. Lorsqu'il est utilisé, la valeur présentée dans la primitive "response" est délivrée inchangée dans la primitive "confirmation".

4.9 Taille APDU

La taille APDU dépend du modèle de communication.

5 Spécification du modèle de communication de type 4

5.1 Concepts

5.1.1 Présentation

Les principes de ce modèle reposent essentiellement sur les concepts communs avec quelques exceptions. Le principe de base décrit du Paragraphe 4.2.2.1 au Paragraphe 4.2.2.3 est applicable à l'Article 5. Ce type prend en charge la fonctionnalité de passerelle au sein de la couche application. Une partie peut également être reconnue à partir des concepts communs. Toutefois, pour encourager la lisibilité, certains éléments de l'Article 4 sont répétés dans l'Article 5.

5.1.2 Entités d'application

5.1.2.1 Définition de FAL AE

Un FAL AE fournit un ensemble de services qui permettent les communications entre AP dans un environnement du bus de terrain. Ces services permettent à une application utilisateur d'un client d'accéder aux objets utilisateur d'un serveur.

La fonctionnalité du FAL AE peut être répartie en deux groupes.

Relation d'application (AR). La fonctionnalité AR est responsable de l'organisation du transport des APDU via le réseau.

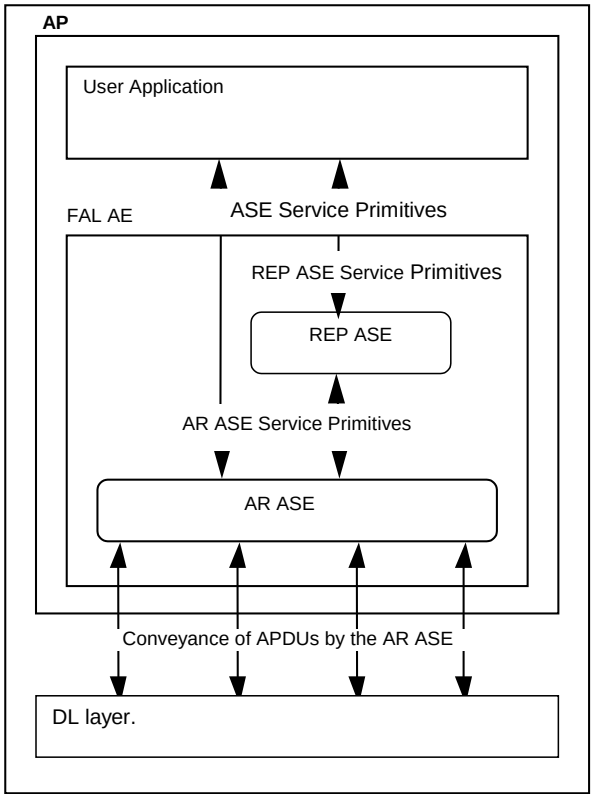
Accès variable (VA). La fonctionnalité d'accès variable est responsable du codage et de l'encodage des APDU. Dans un serveur le VA transporte également les données directement vers / à partir des objets utilisateur. Cette opération est réalisée en local et est en dehors du domaine d'application de la présente norme.

5.1.2.2 Services de la FAL

Les services offerts par le FAL AE à l'application utilisateur sont appelés Services de la FAL. Les services pour l'AR sont appelés services AR ASE et les services pour l'accès variable sont nommés services de point d'extrémité de chemin ASE (services REP ASE).

Les services de la FAL donnent accès aux objets AR locaux et assurent le transport des demandes et des réponses pour accéder aux objets réels modélisés en tant qu'objets FAL accessibles. La couche FAL fournit à la fois des services confirmés et non confirmés. Les demandes de service sont transmises à l'AP qui contient l'objet réel.

La Figure 13 illustre le FAL AE et les relations d'architecture.



Légende

Anglais	Français
User Application	Application utilisateur
ASE Service Primitives	Primitives de service ASE
REP ASE Service Primitives	Primitives de service REP ASE
AR ASE Service Primitives	Primitives de service AR ASE
Conveyance of APDUs by the AR ASE	Acheminement d'unités de données de protocole application par l'AR ASE
DL layer.	Couche DL.

Figure 13 – FAL AE

5.1.2.3 AR ASE

Pour prendre en charge l'accès à l'AP distant, l'ASE de relations entre applications est défini. Il fournit des services à l'AP pour accéder aux paramètres associés à la communication, ainsi que des services au REP ASE pour transmettre des demandes et des réponses de service.

L'AR ASE fournit les services aux points de fin des AR (AREP).

5.1.2.4 REP ASE

Le REP ASE fournit un ensemble de services utilisés pour lire et écrire les attributs visibles du réseau des objets de variable et leurs données d'application. Dans la mesure où un seul type ASE est utilisé à distance, le type est implicite et non transféré.

5.1.2.5 Transport et passerelle des services FAL

Les FAL AE fournissent des services pour transmettre les demandes et les réponses entre les applications utilisateur et les objets de variable incluant la fonctionnalité de passerelle.

Dans un système distribué, les procédés d'application communiquent les uns avec les autres en échangeant des messages de couche application. L'AR est défini pour prendre en charge l'échange à la demande des services confirmés et non confirmés entre deux procédés d'application. Les services de liaison de données sans connexion sont utilisés pour les échanges. Les transferts de la couche DLL peuvent être acquittés ou non acquittés.

5.1.2.6 Relations entre applications

5.1.2.6.1 Définition de AR

L'AR est responsable de transmettre des messages entre applications. Les messages qui sont acheminés sont des demandes et des réponses de services de la couche FAL. Chaque demande et réponse de service est soumise à l'AR ASE pour le transfert. Au sein de l'AR il n'existe qu'un type d'objet, le point d'extrémité AR (AREP).

5.1.2.6.2 Points d'extrémité AR

Un point d'extrémité AR (AREP) représente un point d'accès de service à un DLE au sein de l'AE. Le service d'envoi AR transmet les APDU et les informations relatives au chemin vers l'AREP et implicitement au DLE. L'AREP est un objet de gestion de système présentant les paramètres configurables de l'utilisateur au DLE associé.

Ce type n'a pas de services de gestion de système spéciaux, car les attributs des objets AREP sont déclarés en tant qu'objets de variable par l'application utilisateur. Cela rend le réseau d'attributs visible. Les services AR Get Attribute et AR Set Attribute sont utilisés par ces objets de variable pour accéder aux valeurs des attributs.

Lorsqu'un AREP reçoit un APDU à partir de son DLE (déclenché par l'utilisateur) il transmet l'APDU au REP ASE ou à un AREP (passerelle).

5.1.3 Passerelle et routage

5.1.3.1 Présentation

Ce type prend en charge la fonctionnalité de passerelle. Pour identifier la destination d'une demande, une description de chemin est utilisée. Cette description de chemin contenant une liste complète de points à transmettre est envoyée avec la demande. Sur son itinéraire de destination, le chemin évolue de manière à toujours contenir le chemin de retour vers le demandeur et le chemin vers la destination. Les éléments du chemin sont les adresses de point d'extrémité et de DL.

5.1.3.2 Point d'extrémité de chemin

Au sein du REP ASE, les objets point d'extrémité de chemin (REP) représentent des points d'extrémité, contenant une description de chemin. Un REP représente le point de départ pour transmettre un message. Un REP représente également la destination de l'APDU.

5.1.3.3 Identification des points d'extrémité

Les points d'extrémité sont identifiés par des adresses de point d'extrémité. Ces adresses EP sont des adresses numériques, prises au sein d'un AE à partir du même espace de nom de nombres, pour obtenir une identification unique. Ces adresses EP sont utilisées pour identifier les AREP et les REP.

5.1.3.4 Adresses DL

Le DLE est identifié à partir de la liaison par une adresse DL. Les adresses DL sont définies dans la CEI 61158-4-4, mais comme elles font partie du chemin, elles sont mentionnées ici. Il existe une relation bi-univoque entre l'adresse DL d'un DLE et l'adresse de point d'extrémité de l'AREP correspondant.

5.1.3.5 Route

Le chemin complet contient un chemin de destination et un chemin de départ

Le chemin de destination décrit comment atteindre le REP de destination. C'est une séquence d'adresses de point d'extrémité et d'adresses DL. Sur son itinéraire de destination, le premier élément indique toujours l'adresse du DLE, de l'AREP ou du REP de réception.

De la même manière, le chemin de départ décrit comment retrouver le chemin vers le point d'extrémité qui est à l'origine de la demande. C'est une séquence d'adresses de point d'extrémité et d'adresses DL. Sur son itinéraire de destination, le premier élément indique toujours l'adresse du DLE, de l'AREP ou du REP d'émission.

5.1.3.6 Conversion de chemin Destination / départ

Un chemin de destination transmis par un REP démarre par l'adresse de point d'extrémité de l'AREP local. L'élément suivant est l'adresse DL du DLE de réception. Si le chemin traverse des passerelles, une paire d'adresses doit être ajoutée pour chaque passerelle: une adresse DL et une adresse AREP. L'adresse finale est l'adresse de point d'extrémité du REP de destination. L'adresse de départ est l'adresse de point d'extrémité du REP de départ.

Le service d'envoi AR supprime le premier élément du chemin de destination. Le DLE insère sa propre adresse DL avant le chemin de départ actuel avant de transmettre la demande à la liaison.

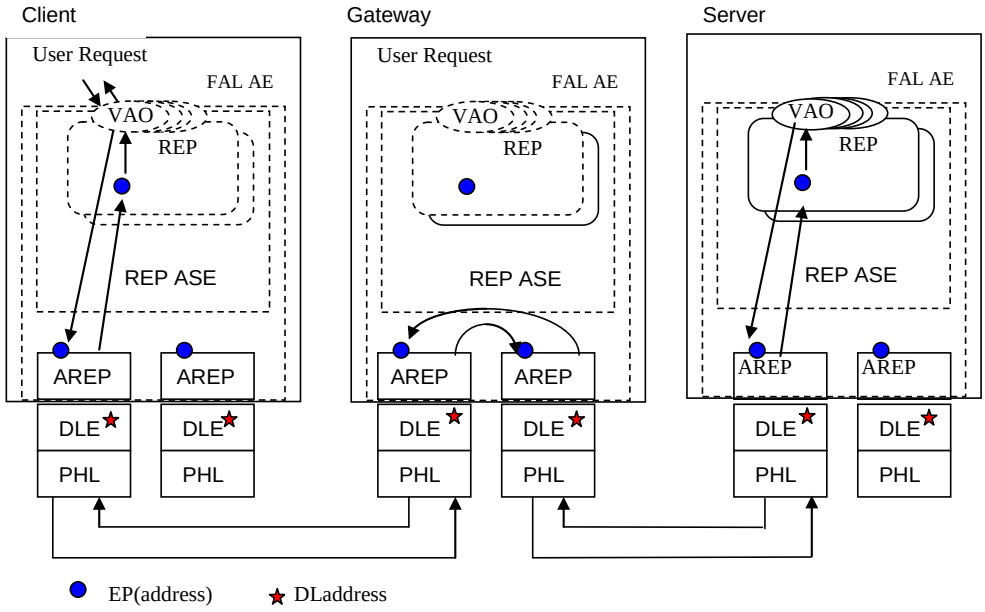
Lorsqu'un DLE reçoit une indication à partir de la liaison, il supprime le premier élément du chemin de destination (sa propre adresse DL). L'AREP correspondant insère sa propre adresse AREP avant le chemin de départ actuel avant de transmettre la demande à un REP ou un AREP.

Au REP de destination, le chemin de destination contient l'adresse de point d'extrémité du REP, et le chemin de départ contient le chemin complet de retour vers le REP demandeur.

Si des erreurs apparaissent lorsqu'une demande est en route vers la destination, le chemin de départ actuel peut être utilisé pour retourner une réponse d'erreur.

5.1.4 Résumé de l'architecture

Un résumé de l'architecture FAL est présenté à la Figure 14. Elle illustre les principaux composants de l'architecture FAL et la manière dont ils se relient les uns aux autres.



Légende

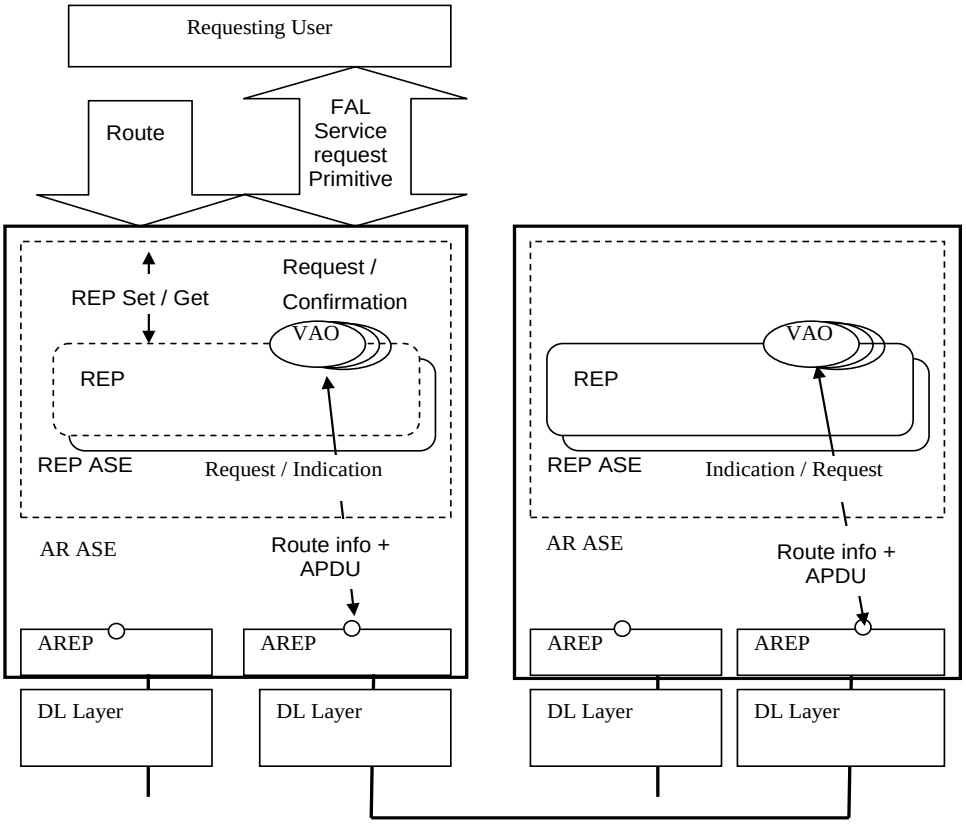
Anglais	Français
Gateway	Passerelle
Server	Serveur
User Request	Demande utilisateur
EP (address)	EP (adresse)
DL address	Adresse DL
Gateway	Passerelle

Figure 14 – Résumé de l'architecture FAL

5.1.5 Procédures de service FAL et diagrammes de séquence temporelle

5.1.5.1 Présentation

Le Paragraphe 5.1.5 décrit les procédures de service pour les services FAL confirmés et non confirmés. Une présentation des procédures de service FAL est donnée à la Figure 15.



Légende

Anglais	Français
Requesting User	Utilisateur qui demande
Route	Route
FAL Service request Primitive	Primitive de demande de service FAL
Request /Confirmation	Demande / Confirmation
Request / Indication	Demande / Indication
Indication / Request	Indication / Demande
Route info + APDU	Info route + APDU
DL Layer	Couche DL

Figure 15 – Présentation de la procédure de service FAL

5.1.5.2 Procédures de service confirmé FAL

– Sur l'AP demandeur:

L'application utilisateur utilise le service REP ASE "Reserve REP" pour réserver un REP. L'application utilisateur utilise ensuite le service REP ASE "Set REP Attribute" pour définir les attributs du REP, y compris le chemin de destination vers le REP distant.

L'utilisateur demandeur soumet une primitive de demande confirmée au REP ASE avec l'adresse REP comme paramètre. Le REP ASE génère l'APDU de la demande confirmée associée et la transmet via l'AREP, à l'aide d'une primitive de demande d'envoi AR. L'adresse AREP est indiquée par le premier élément du chemin de destination. Le REP ASE démarre un temporisateur associé pour surveiller la demande.

L'AREP génère une DLSDU de demande et soumet les informations de chemin avec la DLSDU à son DLE. Le DLE envoie la DLSDU dès que la première opportunité se présente.

– Sur l'AP serveur répondant:

A la réception de la DLSDU, l'AREP de réception génère un APDU et fournit l'APDU et le chemin reçu au REP adressé par une primitive d'indication d'envoi AR.

Le REP ASE gère l'indication par échange de données avec l'objet de variable adressé, génère l'APDU de la demande non confirmée associée et la transmet sur l'AREP, en utilisant une primitive de demande d'envoi AR. Le chemin de départ reçu est utilisé comme chemin de destination pour la réponse.

L'AREP génère une DLSDU de demande et soumet les informations de chemin avec la DLSDU à son DLE. Il revient à la couche DL de décider s'il faut "mettre en file d'attente" la DLSDU ou l'envoyer immédiatement.

Si le REP ASE n'est pas en mesure de traiter l'indication et de répondre au sein du "MaxIndicationDelay", il doit soumettre une primitive de demande AR Acknowledge à l'AREP de réception, et de cette manière, il libère le jeton. L'AR Acknowledge est soumis au DLE.

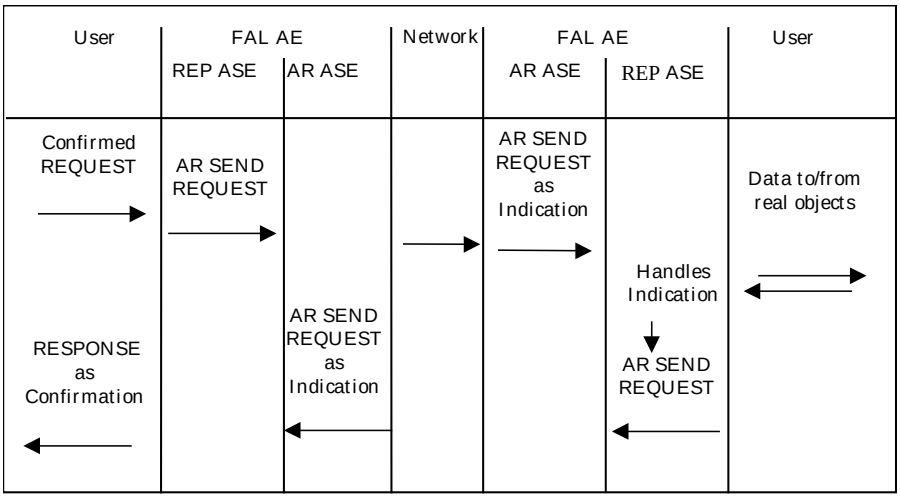
– Sur l'AP demandeur recevant la réponse:

A la réception de la DLSDU de réponse, l'AREP de réception transmet la réponse au REP ASE. L'adresse de destination transmise au REP ASE contient l'adresse de point d'extrémité du REP demandeur. L'objet REP peut maintenant fournir la réponse à l'application utilisateur demandeuse. Cela s'effectue par l'invocation de l'utilisateur du service REP ASE RESPONSE.

Si le temporisateur expire avant que le REP demandeur ait reçu la réponse, le REP annule le diagramme d'états de transactions associé et fournit la réponse d'erreur "Pas de réponse" à l'application utilisateur.

5.1.5.3 Diagramme de séquence temporelle de service confirmé

La Figure 16 affiche le diagramme de séquence temporelle des services confirmés.



Légende

Anglais	Français
User	Utilisateur
Network	Réseau
Confirmed REQUEST	DEMANDE confirmée
AR SEND REQUEST as Indication	DEMANDE D'ENVOI AR sous forme d'indication
Data to/from real objects	Données vers/des objets réels

Anglais	Français
RESPONSE as Confirmation	REPONSE sous forme de confirmation
AR SEND REQUEST as Indication	DEMANDE D'ENVOI AR sous forme d'indication
Handles Indication	Gère l'indication

Figure 16 – Diagramme de séquence temporelle des services confirmés

5.1.5.4 Procédure de service de demande non confirmé

– Sur l'AP demandeur:

L'application utilisateur utilise le service REP ASE "Reserve REP" pour réserver un REP. L'application utilisateur utilise ensuite le service REP ASE "Set REP Attribute" pour définir les attributs du REP, y compris le chemin de destination vers le REP distant.

L'utilisateur demandeur soumet une primitive de demande non confirmée au REP ASE avec l'adresse REP comme paramètre. Le REP ASE génère l'APDU de la demande non confirmée associée et la transmet sur l'AREP, en utilisant une primitive de demande d'envoi AR. L'adresse AREP est indiquée par le premier élément du chemin de destination.

L'AREP génère une DLSDU de demande et soumet les informations de chemin avec la DLSDU à son DLE. Le DLE envoie la DLSDU dès que la première opportunité se présente.

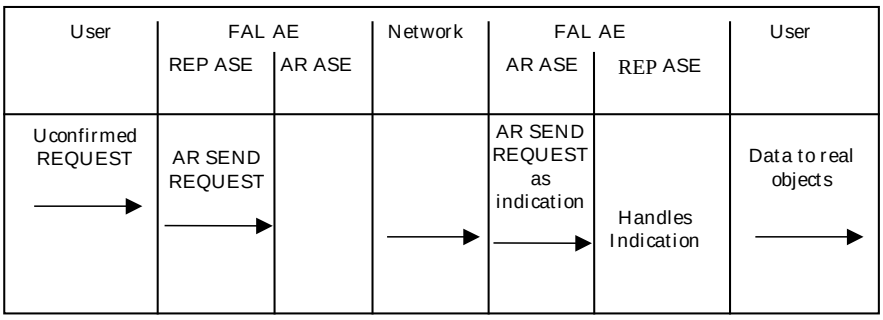
– Sur l'AP serveur répondant:

A la réception de la DLSDU, l'AREP de réception génère un APDU et fournit cela et le chemin reçu au REP adressé par une primitive d'indication d'envoi AR.

Le REP ASE gère l'indication par échange de données avec l'objet de variable adressé.

5.1.5.5 Diagramme de séquence temporelle de service non confirmé

La Figure 17 affiche le diagramme de séquence temporelle des services non confirmés.



Légende

Anglais	Français
User	Utilisateur
Network	Réseau
Unconfirmed REQUEST	DEMANDE non confirmée
AR SEND REQUEST as Indication	DEMANDE D'ENVOI AR sous forme d'indication
Data to/from real objects	Données vers/des objets réels
Handles Indication	Gère l'indication

Figure 17 – Diagramme de séquence temporelle des services non confirmés

5.1.5.6 Procédure de service confirmé / non confirmé par l'intermédiaire de passerelles

A la réception de la DLSDU, l'AREP de réception fournit une primitive d'indication d'envoi AR à la destination locale. Si l'adresse de point d'extrémité est une adresse REP, l'APDU + chemin est fourni au REP ASE. Si l'adresse de point d'extrémité est une adresse AREP, le service d'envoi AR est utilisé pour la transmettre sur le réseau (passerelle).

Pour la demande de service confirmé, un AR Acknowledge est envoyé à l'"AREP source" sauf si la réponse est attendue au sein du "MaxIndicationDelay". "MaxIndicationDelay" est un attribut de l'"AREP source".

5.2 Elément ASE de variable

5.2.1 Types de variable

5.2.1.1 Présentation

L'ASE de variable spécifie la syntaxe indépendante de la machine pour les données d'application transmises par les services FAL. La couche d'application prend en charge le transfert des variables de base et construites. Les règles de codage pour les variables de base et les variables construites sont spécifiées en 5.2 tel qu'indiqué dans la CEI 61158-6-4.

Les objets de variable (VAO) sont définis comme des instances d'une classe de variables. L'ASE de variable donne accès aux VAO. Les VAO sont identifiés par un identificateur de variable numérique, et sont des instances d'une classe de variables.

Les types de base sont des types atomiques qui ne peuvent pas être décomposés en des types plus élémentaires. Les types construits sont des types composés de types de base et d'autres types construits. Leur complexité et la profondeur de l'imbrication sont libres.

Les types construits sont condensés conformément aux règles de codage des types construits. La structure d'un type construit n'est pas visible du réseau. La structure d'un type construit est générée par un assembleur et transférée par d'autres moyens.

Toutes les classes de variables ont les attributs communs identificateur de variable, identificateur de type de données, longueur en octets, activation en lecture, activation en écriture et protégé en écriture. Toutes les classes de variable ont les services communs Read, Write, Get Variable Attribute et Set Variable Attribute.

La présente norme recommande d'utiliser une représentation à virgule flottante des valeurs de mesure échelonnées en unités SI.

5.2.1.2 Types de variable de base

Les types de variable de base ont une longueur constante. La longueur des types de variable de base est un nombre d'octets intégral. La définition des nouveaux types de variable de base s'effectue en définissant des nouvelles classes de types de variable de base.

5.2.1.3 Types de variable construite

5.2.1.3.1 Présentation

Les types de variable construite sont nécessaires pour transmettre complètement la variété des informations présentes sur le bus de terrain. Quatre sortes de types de variable construite sont définies dans quatre classes dans la présente norme: Complex, String, BitString et FIFO. La définition des nouvelles sortes de types de variable construite s'effectue en définissant de nouvelles classes de types de variable construite.

Sur le bus de terrain une variable construite complète peut être transférée, ou une partie de celle-ci, par exemple un champ unique d'une variable complexe ou un élément d'une chaîne. Lors du transfert d'une partie seulement d'une variable, l'adressage avec décalage et peut-être no-bit est utilisé. Le décalage indique la distance (en octets) du premier octet de la variable à la partie à transférer, no-bit indique le numéro de bit dans un octet.

5.2.1.3.2 Type de variable complexe

Un type de variable complexe définit des matrices et des structures.

Une matrice est composée d'un ensemble ordonné d'éléments saisis de manière homogène. La présente norme n'impose aucune restriction quant au type d'élément de la matrice, mais elle exige que chaque élément soit du même type. Une fois défini, le nombre d'éléments d'une matrice ne peut pas être changé.

Une structure est composée d'un ensemble ordonné d'éléments de type hétérogène nommés champs. De même que pour les matrices, la présente norme n'impose aucune restriction quant au type de champs. Toutefois, les champs d'une structure peuvent être de types différents.

5.2.1.3.3 Type de variable chaîne

Une chaîne est composée d'un champ Nombre réel d'éléments et d'un nombre d'éléments saisis de manière homogène. Une fois défini, le nombre maximum d'éléments d'une chaîne ne peut pas être changé.

5.2.1.3.4 Type de variable BitString

Un BitString est défini comme une série de bits. Une fois le BitString défini, le nombre d'éléments qu'il contient ne peut pas être changé. La longueur en octets d'un BitString est la partie intégrale de ((le nombre d'éléments + 7) divisé par 8).

5.2.1.3.5 Type de variable FIFO

Une file d'attente FIFO est composée d'un ensemble d'éléments saisis de manière homogène. La présente norme n'impose aucune restriction quant au type d'élément FIFO, mais elle exige que chaque élément soit du même type. Le premier élément écrit sera le premier élément qui peut être lu. Dans un bus de terrain, seul un élément complet peut être transféré à la suite d'une invocation de services.

5.2.2 Spécification de la classe de modèle de variable

5.2.2.1 Modèle de variable

5.2.2.1.1 Modèle formel

Le modèle formel de variable définit les caractéristiques communes à toutes les classes de variables. Cette classe ne peut pas être instanciée. Elle n'est présente que pour l'héritage de ses attributs et des services de ses sous-classes.

FAL ASE:			**VARIABLE ASE**
CLASSE:			VARIABLE
ID CLASSE:			1
CLASSE PARENT:			HAUT
ATTRIBUTS:			
1	(m)	Attribut clé:	Identificateur de variable
2	(o)	Attribut:	VariableTypeIdentifier
3	(o)	Attribut:	Longueur en octets
4	(o)	Attribut:	Activation en lecture

5	(o)	Attribut:	Activation en écriture
6	(o)	Attribut:	Protégé en écriture

SERVICES:

1	(m)	OpsService:	Lire
2	(m)	OpsService:	Ecrire
3	(o)	OpsService:	Obtenir l'attribut de variable
4	(o)	OpsService:	Définir l'attribut de variable

5.2.2.1.2 Attributs

1) Identificateur de variable

   Cet attribut identifie l'objet de variable qui instancie la classe de type de variable.

2) VariableTypeIdentifier

   Cet attribut facultatif identifie le type de variable en tant que Boolean, Integer8, Integer16, Integer32, Unsigned8, Unsigned16, Float32, Float64, UNICODE Char, Array, String, Structure ou FIFO. L'identificateur VariableTypeIdentifier est un identificateur numérique, défini dans la CEI 61158-6-4. Dans la présente norme, un nom descriptif correspondant à l'identificateur est spécifié.

3) Longueur en octets

   Cet attribut facultatif indique la longueur en octets des données dans l'objet de variable. Pour tous les types excepté FIFO, longueur en octets indique le nombre total d'octets. Pour les types FIFO, longueur en octets indique le nombre d'octets d'un élément.

4) Activation en lecture

   La valeur de cet attribut facultatif indique si la valeur des données de l'objet de variable peut être lue via le bus de terrain.

5) Activation en écriture

   La valeur de cet attribut facultatif indique si la valeur des données de l'objet de variable peut être mise à jour via le bus de terrain.

6) Protégé en écriture

   La valeur de cet attribut facultatif indique si la valeur des données de l'objet de variable peut être mise à jour via le bus de terrain seulement dans certaines conditions. Les conditions permettant la mise à jour un VAO protégé en écriture sont en dehors du domaine d'application de la présente norme.

5.2.2.1.3 Services

1) Lire

   Ce service est utilisé pour lire la valeur des données de l'objet de variable.

2) Ecrire

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable.

3) Obtenir l'attribut de variable

   Ce service est utilisé pour lire un attribut de l'objet de variable.

4) Définir l'attribut de variable

   Ce service est utilisé pour mettre à jour un attribut de l'objet de variable.

5.2.3 Spécifications du type de variable de base

5.2.3.1 Modèle de type de variable Boolean

5.2.3.1.1 Modèle formel

Ce type de données exprime un type de variable Boolean avec les valeurs possibles TRUE et FALSE.

FAL ASE:			**VARIABLE ASE**
CLASSE:			TYPE DE VARIABLE BOOLEAN
CLASS ID:			2
CLASSE PARENT:			VARIABLE
ATTRIBUTS:			
SERVICES:			
1	(m)	OpsService:	Et
2	(m)	OpsService:	Ou
3	(m)	OpsService:	Tester et définir

5.2.3.1.2 Attributs

Les objets de variable de cette classe auront les valeurs constantes suivantes des attributs suivants (classe parent):

2	VariableTypeIdentifier	= Boolean
3	Longueur en octets	= 1

5.2.3.1.3 Services

1) Et

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique AND.

2) Ou

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique OR.

3) Tester et définir

   Ce service est utilisé pour lire et mettre à jour la valeur des données de l'objet de variable, en retournant le résultat d'une opération logique AND, et en stockant le résultat d'une opération logique OR.

5.2.3.2 Modèle de type de variable Integer8

5.2.3.2.1 Modèle formel

Ce type entier est un nombre binaire en complément à deux d'une longueur de un octet.

FAL ASE:			**VARIABLE ASE**
CLASSE:			TYPE DE VARIABLE INTEGER8
CLASS ID:			3
CLASSE PARENT:			VARIABLE
ATTRIBUTS:			
SERVICES:			
1	(m)	OpsService:	Et
2	(m)	OpsService:	Ou
3	(m)	OpsService:	Tester et définir

5.2.3.2.2 Attributs

Les objets de variable de cette classe auront les valeurs constantes suivantes des attributs suivants (classe parent):

2	VariableTypeIdentifier	= Integer8
3	Longueur en octets	= 1

5.2.3.2.3 Services

1) Et

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique AND.

2) Ou

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique OR.

3) Tester et définir

   Ce service est utilisé pour lire et mettre à jour la valeur des données de l'objet de variable, en retournant le résultat d'une opération logique AND, et en stockant le résultat d'une opération logique OR.

5.2.3.3 Modèle de type de variable Integer16

5.2.3.3.1 Modèle formel

Ce type entier est un nombre binaire en complément à deux d'une longueur de deux octets.

FAL ASE:			**VARIABLE ASE**
CLASSE:			TYPE DE VARIABLE INTEGER16
CLASS ID:			4
CLASSE PARENT:			VARIABLE
ATTRIBUTS:			
SERVICES:			
1	(m)	OpsService:	Et
2	(m)	OpsService:	Ou

5.2.3.3.2 Attributs

Les objets de variable de cette classe auront les valeurs constantes suivantes des attributs suivants (classe parent):

2	VariableTypeIdentifier	= Integer16
3	Longueur en octets	= 2

5.2.3.3.3 Services

1) Et

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique AND.

2) Ou

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique OR.

5.2.3.4 Modèle de type de variable Integer32

5.2.3.4.1 Modèle formel

Ce type entier est un nombre binaire en complément à deux d'une longueur de quatre octets.

FAL ASE:	**VARIABLE ASE**
CLASSE:	TYPE DE VARIABLE INTEGER32
CLASS ID:	5
CLASSE PARENT:	VARIABLE
ATTRIBUTS:	

SERVICES:

1	(m)	OpsService:	Et
2	(m)	OpsService:	Ou

5.2.3.4.2 Attributs

Les objets de variable de cette classe auront les valeurs constantes suivantes des attributs suivants (classe parent):

2	VariableTypeIdentifier	= Integer32
3	Longueur en octets	= 4

5.2.3.4.3 Services

1) Et

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique AND.

2) Ou

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique OR.

5.2.3.5 Modèle de type de variable Unsigned8

5.2.3.5.1 Modèle formel

Ce type entier est un nombre binaire d'une longueur d'un octet. Le bit de poids le plus fort est toujours utilisé comme bit de poids le plus fort du nombre binaire; aucun bit de signe n'est inclus.

FAL ASE:	**VARIABLE ASE**
CLASSE:	TYPE DE VARIABLE UNSIGNED8
CLASS ID:	6
CLASSE PARENT:	VARIABLE
ATTRIBUTS:	

SERVICES:

1	(m)	OpsService:	Et
2	(m)	OpsService:	Ou
3	(m)	OpsService:	Tester et définir

5.2.3.5.2 Attributs

Les objets de variable de cette classe auront les valeurs constantes suivantes des attributs suivants (classe parent):

2	VariableTypeIdentifier	= Unsigned8
3	Longueur en octets	= 1

5.2.3.5.3 Services

1) Et

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique AND.

2) Ou

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique OR.

3) Tester et définir

Ce service est utilisé pour lire et mettre à jour la valeur des données de l'objet de variable, en retournant le résultat d'une opération logique AND, et en stockant le résultat d'une opération logique OR.

5.2.3.6 Modèle de type de variable Unsigned16

5.2.3.6.1 Modèle formel

Ce type entier est un nombre binaire d'une longueur de deux octets. Le bit de poids le plus fort de l'octet de poids le plus fort est toujours utilisé comme bit de poids le plus fort du nombre binaire; aucun bit de signe n'est inclus.

FAL ASE:			**VARIABLE ASE**
CLASSE:			TYPE DE VARIABLE UNSIGNED16
CLASS ID:			7
CLASSE PARENT:			VARIABLE
ATTRIBUTS:			
SERVICES:			
1	(m)	OpsService:	Et
2	(m)	OpsService:	Ou

5.2.3.6.2 Attributs

Les objets de variable de cette classe auront les valeurs constantes suivantes des attributs suivants (classe parent):

2	VariableTypeIdentifier	= Unsigned16
3	Longueur en octets	= 2

5.2.3.6.3 Services

1) Et

Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique AND.

2) Ou

Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique OR.

5.2.3.7 Modèle de type de variable Float32

5.2.3.7.1 Modèle formel

Ce type a une longueur de quatre octets. Le format de Float32 est celui défini par la ISO/CEI/IEEE 60559 simple précision.

FAL ASE:	**VARIABLE ASE**
CLASSE:	TYPE DE VARIABLE FLOAT32
CLASS ID:	8
CLASSE PARENT:	VARIABLE
ATTRIBUTS:	
SERVICES:	

5.2.3.7.2 Attributs

Les objets de variable de cette classe auront les valeurs constantes suivantes des attributs suivants (classe parent):

2 VariableTypeIdentifier = Float32

3 Longueur en octets = 4

5.2.3.8 Modèle de type de variable Float64

5.2.3.8.1 Modèle formel

Ce type a une longueur de huit octets. Le format de Float64 est celui défini par la ISO/CEI/IEEE 60559 double précision.

FAL ASE:	**VARIABLE ASE**
CLASSE:	TYPE DE VARIABLE FLOAT64
CLASS ID:	9
CLASSE PARENT:	VARIABLE
ATTRIBUTS:	
SERVICES:	

5.2.3.8.2 Attributs

Les objets de variable de cette classe auront les valeurs constantes suivantes des attributs suivants (classe parent):

2 VariableTypeIdentifier = Float64

3 Longueur en octets = 8

5.2.3.9 Modèle de type de variable UNICODE char

5.2.3.9.1 Modèle formel

Ce type a une longueur de deux octets. Il est défini comme un seul caractère UNICODE.

FAL ASE:	**VARIABLE ASE**
CLASSE:	TYPE DE VARIABLE UNICODE CHAR
CLASS ID:	10
CLASSE PARENT:	VARIABLE
ATTRIBUTS:	
SERVICES:	

5.2.3.9.2 Attributs

Les objets de variable de cette classe auront les valeurs constantes suivantes des attributs suivants (classe parent):

2 VariableTypeIdentifier = UNICODE Char

3 Longueur en octets = 2

5.2.4 Spécifications du type de variable construite

5.2.4.1 Modèle de type de variable complexe

5.2.4.1.1 Modèle formel

Une variable complexe est une matrice ou une structure.

Une matrice est composée d'un ensemble ordonné d'éléments saisis de manière homogène. La présente norme n'impose aucune restriction quant au type d'élément de la matrice, mais elle exige que chaque élément soit du même type. Une fois défini, le nombre d'éléments d'une matrice ne peut pas être changé.

Une structure est composée d'un ensemble ordonné d'éléments de type hétérogène nommés champs. De même que pour les matrices, la présente norme n'impose aucune restriction quant au type de champs. Toutefois, les champs d'une structure peuvent être de types différents.

FAL ASE:	**VARIABLE ASE**
CLASSE:	TYPE DE VARIABLE COMPLEXE
CLASS ID:	11
CLASSE PARENT:	VARIABLE
ATTRIBUTS:	
SERVICES:	

1	(m)	OpsService:	Et
2	(m)	OpsService:	Ou
3	(m)	OpsService:	Tester et définir

5.2.4.1.2 Attributs

Les objets de variable de cette classe auront la valeur constante suivante de l'attribut suivant (classe parent):

2 VariableTypeIdentifier = Complex

5.2.4.1.3 Services

1) Et

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique AND. Cette opération est légale sur les sous-éléments de type entier ou booléen seulement.

2) Ou

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique OR. Cette opération est légale sur les sous-éléments de type entier ou booléen seulement.

3) Tester et définir

   Ce service est utilisé pour lire et mettre à jour la valeur des données de l'objet de variable, en retournant le résultat d'une opération logique AND, et en stockant le résultat d'une opération logique OR. Cette opération est légale sur les sous-éléments de type Integer8, Unsigned8 ou booléen seulement.

5.2.4.2 Modèle de type de variable chaîne

5.2.4.2.1 Modèle formel

Une chaîne est composée d'un champ Nombre réel d'éléments et d'un nombre d'éléments saisis de manière homogène. Une fois défini, le nombre maximum d'éléments d'une chaîne ne peut pas être changé.

FAL ASE:	**VARIABLE ASE**
CLASSE:	TYPE DE VARIABLE CHAINE
CLASS ID:	12
CLASSE PARENT:	VARIABLE
ATTRIBUTS:	

SERVICES:

1	(m)	OpsService:	Et
2	(m)	OpsService:	Ou
3	(m)	OpsService:	Tester et définir

5.2.4.2.2 Attributs

Les objets de variable de cette classe auront la valeur constante suivante de l'attribut suivant (classe parent):

2 VariableTypeIdentifier = String

5.2.4.2.3 Services

1) Et

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique AND. Cette opération est légale sur les sous-éléments de type entier ou booléen seulement.

2) Ou

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique OR. Cette opération est légale sur les sous-éléments de type entier ou booléen seulement.

3) Tester et définir

   Ce service est utilisé pour lire et mettre à jour la valeur des données de l'objet de variable, en retournant le résultat d'une opération logique AND, et en stockant le résultat d'une opération logique OR. Cette opération est légale sur les sous-éléments de type Integer8, Unsigned8 ou booléen seulement.

5.2.4.3 Modèle de type de variable BitString

5.2.4.3.1 Modèle formel

Un BitString est défini comme une série de bits. Une fois le BitString défini, le nombre de bits qu'il contient ne peut pas être changé. La longueur en octets d'un BitString est la partie intégrale de ((le nombre de bits + 7) divisé par 8).

FAL ASE:	**VARIABLE ASE**
CLASSE:	TYPE DE VARIABLE BITSTRING
CLASS ID:	13
CLASSE PARENT:	VARIABLE
ATTRIBUTS:	

SERVICES:

1	(m)	OpsService:	Et
2	(m)	OpsService:	Ou
3	(m)	OpsService:	Tester et définir

5.2.4.3.2 Attributs

Les objets de variable de cette classe auront la valeur constante suivante de l'attribut suivant (classe parent):

2 VariableTypeIdentifier = BitString

5.2.4.3.3 Services

1) Et

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique AND. Cette opération est légale sur un sous-élément (qui est un bit) seulement.

2) Ou

   Ce service est utilisé pour mettre à jour la valeur des données de l'objet de variable, en stockant le résultat d'une opération logique OR. Cette opération est légale sur un sous-élément (qui est un bit) seulement.

3) Tester et définir

   Ce service est utilisé pour lire et mettre à jour la valeur des données de l'objet de variable, en retournant le résultat d'une opération logique AND, et en stockant le résultat d'une opération logique OR. Cette opération est légale sur un sous-élément (qui est un bit) seulement.

5.2.4.4 Modèle de type de variable FIFO

5.2.4.4.1 Modèle formel

Une file d'attente FIFO est composée d'un ensemble d'éléments saisis de manière homogène. La présente norme n'impose aucune restriction quant au type d'élément FIFO, mais elle exige que chaque élément soit du même type. Le premier élément écrit sera le premier élément qui peut être lu. Dans un bus de terrain, seul un élément complet peut être transféré à la suite d'une invocation de services.

FAL ASE:			**VARIABLE ASE**
CLASSE:			TYPE DE VARIABLE FIFO
CLASS ID:			14
CLASSE PARENT:			VARIABLE
ATTRIBUTS:			
1	(m)	Attribut:	Elément d'entrée suivant
2	(m)	Attribut:	Elément de sortie suivant
3	(m)	Attribut:	Eléments libres
4	(m)	Attribut:	Eléments utilisés
5	(o)	Attribut:	Relire
6	(o)	Attribut:	Réécrire

5.2.4.4.2 Attributs

Les objets de variable de cette classe auront la valeur constante suivante de l'attribut suivant (classe parent):

2 VariableTypeIdentifier = FIFO

1) Elément d'entrée suivant

   Elément d'entrée suivant est un Integer16, qui contient le numéro d'index de l'élément suivant à écrite dans le FIFO.

2) Elément de sortie suivant

   Elément de sortie suivant est un Integer16, qui contient le numéro d'index de l'élément suivant à lu à partir du FIFO.

3) Eléments libres

   Cet attribut est un Integer16 qui contient le numéro actuel des éléments libres du FIFO (= 0 si complet).

4) Eléments utilisés

Cet attribut est un Integer16 qui contient le numéro actuel des éléments utilisés du FIFO (= 0 si vide).

5) Relire

Relire est un booléen. Lorsque la valeur de Réécrire est TRUE, un Read retourne l'élément lu précédemment, au lieu de retourner et de supprimer l'élément indexé par Elément de sortie suivant. Ce dispositif peut être utilisé si une erreur de transmission s'est traduite par l'erreur de lecture d'un élément.

6) Réécrire

Réécrire est un booléen. Lorsque la valeur de Réécrire est TRUE, un Write remplace l'élément écrit précédemment, au lieu d'insérer un nouvel élément. Ce dispositif peut être utilisé si une erreur de transmission s'est traduite par l'erreur d'écriture d'un élément.

5.2.4.4.3 Services

5.2.4.4.3.1 Présentation

Les services définis dans la Variable ASE donnent accès aux attributs et aux données des objets de variable. Les services sont appelés par une invocation du service REP ASE REQUEST. Le service REP ASE REQUEST transmet les paramètres du service variable ASE. Se référer à 5.3.3 pour obtenir la documentation de ces paramètres.

5.2.4.4.3.2 Lire

Ce service est utilisé pour lire (une partie de) la variable spécifiée. Aucune donnée n'est transmise avec la demande. La valeur de la variable spécifiée est retournée dans la réponse.

5.2.4.4.3.3 Ecrire

Ce service est utilisé pour mettre à jour (une partie de) la variable spécifiée. Les données transmises avec la demande sont écrites dans (une partie de) la variable spécifiée. Aucune donnée n'est retournée dans la réponse.

5.2.4.4.3.4 Et

Ce service effectue un AND logique de (une partie de) la variable spécifiée et des données transmises avec la demande. Le résultat est stocké dans (une partie de) la variable. Ce service permet d'effacer les différents bits dans une variable sans lire d'abord la variable puis réécrire le résultat dans la variable. Aucune donnée n'est retournée dans la réponse.

5.2.4.4.3.5 Ou

Ce service effectue un OR logique de (une partie de) la variable spécifiée et des données transmises avec la demande. Le résultat est stocké dans (une partie de) la variable. Ce service permet de définir les différents bits dans une variable sans lire d'abord la variable puis réécrire le résultat dans la variable. Aucune donnée n'est retournée dans la réponse.

5.2.4.4.3.6 Tester et définir

Ce service effectue un AND logique de (une partie de) la variable spécifiée et des données transmises dans la demande. Le résultat de l'opération AND est retourné dans la réponse. Ensuite, une opération logique OR de (une partie de) la variable spécifiée et des données transmises dans la demande est effectuée. Le résultat de l'opération OR est stocké dans (une partie de) la variable. Ce service permet de définir les différents bits dans une variable, et dans la même transmission de lire leur état avant l'opération.

5.2.4.4.3.7 Obtenir l'attribut de variable

Ce service est utilisé pour lire un attribut de l'objet de variable spécifié. L'attribut est spécifié par le paramètre de décalage.

5.2.4.4.3.8 Définir l'attribut de variable

Ce service est utilisé pour mettre à jour un attribut de l'objet de variable spécifié. L'attribut est spécifié par le paramètre de décalage.

5.2.5 Point d'extrémité de chemin ASE

5.2.5.1 Présentation

Dans l'environnement du bus de terrain, les procédés d'application contiennent des données que les applications distantes sont capables de lire et d'écrire. Les données sont accessibles au moyen du REP ASE. Cet ASE offre des services aux applications utilisateur dans les clients. Les services REQUEST et RESPONSE donnent accès aux objets de variable (VAO) réels situés dans un serveur. Dans le serveur le REP ASE transmet les données vers / à partir de l'objet de variable réel et retourne des réponses. Les services Get REP Attribute, Set REP Attribute, Reserve REP et Free REP sont tous destinés à une utilisation locale.

5.2.5.1.1 Description du point d'extrémité de chemin

La classe Route Endpoint est définie pour prendre en charge l'échange à la demande des services confirmés et non confirmés entre deux procédés d'application. Elle utilise les services AR ASE pour transmettre les APDU.

Un REP est un conteneur d'objet contenant les objets de variable d'un type de variable. Il donne accès aux objets de variable. L'ID de l'objet de variable est unique au sein d'un REP. Le service de demande adresse un objet de variable au sein du REP. L'ID de l'objet de variable est l'un des paramètres du service de demande.

Un REP dans le rôle de conteneur de proxy représente un REP distant dans un serveur. Le REP distant est dans le rôle de conteneur réel. L'attribut Destination Route du REP dans le client contient le chemin vers le REP de conteneur réel, qu'il représente.

Le REP ASE gère la segmentation lorsque la longueur des données dépasse la MaxDataSize.

Lorsque les données sont segmentées, l'ordre des segments est indiqué par la partie décalée de l'APDU.

Pour donner accès d'un objet de variable proxy, l'application utilisateur utilise Reserve REP pour obtenir un REP qui n'est pas utilisé actuellement.

Les attributs du REP sont définis, y compris le chemin de destination (Destination Route) vers le REP distant. Cela s'effectue par le service Set REP Attribute. L'utilisateur demandeur émet ensuite une primitive REQUEST avec le REP ID et un ID de l'objet de variable comme paramètres. Le REP ASE génère un APDU de demande, et le transmet avec les informations de chemin dans une demande d'envoi AR.

A la suite d'une transmission réussie, si la REQUEST a été confirmée, l'AREP reçoit un APDU de réponse et le transmet au REP demandeur par une indication d'envoi AR. L'application utilisateur peut maintenant lire le résultat en émettant une primitive RESPONSE, avec le REP ID comme paramètre.

5.2.5.2 Modèle Point d'extrémité de chemin (Route End Point)

Modèle formel

FAL ASE:			**Point d'extrémité de chemin ASE**
CLASSE:			Point d'extrémité de chemin
ID CLASSE:			1
CLASSE PARENT:			HAUT
ATTRIBUTS:			
1.	(m)	Attribut clé:	Adresse de point de fin
2.	(m)	Attribut:	Rôle (objet Proxy / objet Réel)
3.	(m)	Attribut:	Etat REP
4.	(m)	Attribut:	Priorité
5.	(m)	Attribut:	Confirmation
6.	(m)	Attribut:	Route de destination
7.	(m)	Attribut:	Route source
8.	(m)	Attribut:	Avancement
9.	(m)	Attribut:	Fonctionnalités
10.	(m)	Attribut:	Adressage plat
SERVICES:			
1.	(m)	OpsService:	DEMANDE
2.	(m)	OpsService:	REPONSE
3.	(m)	OpsService:	Réserver REP
4.	(m)	OpsService:	Libérer REP
5.	(m)	OpsService:	Obtenir attribut REP
6.	(m)	OpsService:	Définir attribut REP

5.2.5.2.1 Attributs

5.2.5.2.1.1 Adresse de point de fin

Cet attribut clé contient l'adresse de point d'extrémité identifiant le REP.

5.2.5.2.1.2 ROLE

Cet attribut spécifie le rôle du REP. Les valeurs valides sont les suivantes:

Proxy container (Conteneur de proxy)	Les points d'extrémité de ce type sont utilisés pour l'envoi des demandes aux serveurs et la réception des réponses qui en proviennent. Les objets contenus dans les REP de ce rôle sont les objets proxy pour les objets réels du serveur.
Real container (Conteneur réel)	Les points d'extrémité de ce type sont utilisés pour la réception des réponses confirmées et non confirmées à partir des clients et l'envoi des réponses vers ceux-ci. Les objets contenus dans les REP de ce rôle sont les objets de variable réels.

5.2.5.2.1.3 Etat REP

Cet attribut spécifie l'état du REP. Les valeurs de cet attribut sont les suivantes: IDLE RESERVED, WAITING FOR RESPONSE, RESPONSE RECEIVED ou NOT IN USE.

5.2.5.2.1.4 Priorité

La DLL peut permettre d'envoyer des APDU à haute priorité avant les APDU à faible priorité. La priorité ne fait référence qu'à la demande sur le lien local, pas aux demandes franchissant des passerelles et pas à la réponse.

5.2.5.2.1.5 Confirmation

Cet attribut indique si la demande doit être confirmée ou non confirmée. La réponse est toujours retournée non confirmée. Si le chemin de destination contient une ou plusieurs adresses de diffusion (126), cet attribut doit être défini sur non confirmé.

5.2.5.2.1.6 Route de destination

L'attribut Destination Route (chemin de destination) décrit le chemin vers le REP de destination. C'est une séquence d'adresses de point d'extrémité et d'adresses DL. Sur son itinéraire de destination, la première partie indique toujours l'adresse du DLE, de l'AREP ou du REP suivant pour recevoir l'APDU. Dans une demande, l'attribut Destination Route contient le chemin vers le REP auquel répondre. Dans une réponse, l'attribut Destination Route contient le chemin vers le REP demandeur.

5.2.5.2.1.7 Route source

L'attribut Source Route (chemin de destination) décrit le chemin vers le REP de départ. C'est une séquence d'adresses de point d'extrémité et d'adresses DL. Sur son itinéraire de destination, le premier élément indique toujours l'adresse du DLE, de l'AREP ou du point d'extrémité du REP où l'APDU a été transmis. Dans une demande, l'attribut Source Route contient le chemin vers le REP demandeur. Dans une réponse, l'attribut Source Route contient le chemin vers le REP qui répond.

5.2.5.2.1.8 Avancement

Cet attribut n'est pertinent que dans les REP de conteneur de proxy. Il indique la progression d'une demande. Il indique le nombre de segments ayant été livrés divisé par le nombre total de segments. Cela signifie que pour les demandes non segmentées la valeur est zéro dans l'attente de la réponse, et un lorsque la réponse est reçue. Pour les demandes segmentées, la valeur commence par zéro et croît graduellement jusqu'à ce qu'elle atteigne la valeur un.

5.2.5.2.1.9 Fonctionnalités

Cet attribut définit les capacités du REP adressé par l'attribut Destination Route. Il s'agit d'un attribut local, qui doit être configuré par l'application utilisateur afin de refléter les capacités du REP contenant les objets de variable réels. La valeur de l'attribut Capabilities est utilisée par le REP demandeur pour générer l'APDU. Cet attribut indique si le REP qui répond est capable de gérer ou non l'adressage de bit.

5.2.5.2.1.10 Adressage plat

Cet attribut indique s'il convient de considérer le REP comme un conteneur d'objets de variable individuels ou comme un conteneur d'une zone mémoire linéaire. Si l'attribut Flat addressing est sélectionné, le paramètre Variable Object ID du service REQUEST indique le décalage en octets par rapport au début de la zone mémoire.

5.2.6 Spécification du service Route endpoint ASE

5.2.6.1 Service REQUEST

5.2.6.1.1 Présentation

Ce service est utilisé pour donner accès à la valeur d'un objet de variable. Il peut être utilisé pour la lecture ou la mise à jour de la valeur. Le résultat est que l'opération (le cas échéant) est récupérée par le service RESPONSE.

5.2.6.1.2 Primitives de service

Les paramètres de service pour ce service sont montrés dans le Tableau 1. Les paramètres Indication indiquent les paramètres de l'APDU reçus par le REP distant. Ces paramètres sont destinés à une utilisation locale au sein du REP distant et de ses objets de variable.

Tableau 1 – Paramètres du service REQUEST

Nom du paramètre	Req	Ind	Cnf
Argument	M	M	
REP	M	M	
ID de l'objet de variable	M	M(=)	
Service variable	M	M(=)	
Longueur des données	M	M(=)	
Décalage/attribut	C	C(=)	
N° de bit	C	C(=)	
Données	C	C(=)	
Résultat			
Statut			M

– Argument

L'argument contient les paramètres de l'invocation du service REQUEST.

– REP

Ce paramètre est l'adresse EP du REP contenant les informations relatives au chemin.

– ID de variable

Ce paramètre identifie l'objet de variable. L'ID de variable indique l'objet de variable sur lequel le service doit être effectué.

– Service variable

Ce paramètre contient le service variable à effectuer. (Read, Write, And, Or, Test-And-Set, Get attribute et Set Attribute).

– Longueur des données

Ce paramètre indique le nombre d'octets des données à transférer.

– Décalage/attribut

Ce paramètre est utilisé pour n'accéder qu'à une partie d'une variable structurée ou d'un attribut de la variable. Pour accéder à une partie d'une variable structurée, il indique le décalage en octets à l'octet de départ du bloc de données à transférer, par rapport au premier octet de la variable. Lors de l'accès aux attributs, il identifie l'attribut auquel accéder.

– N° de bit

Ce paramètre est utilisé pour sélectionner un bit au sein d'un BitString. Il indique le numéro de bit (1-8) au sein d'un octet. Le paramètre Offset/Attribute est utilisé pour sélectionner l'octet.

– Données

Ce paramètre contient les données à transférer.

– Statut

Résultat de la demande, Statut est retourné en indiquant OK ou FAILURE.

5.2.6.2 Service REPONSE

5.2.6.2.1 Présentation

Ce service est utilisé pour retourner le résultat d'une REQUEST confirmée à l'application utilisateur demandeuse dans un demandeur.

5.2.6.2.2 Primitives de service

Les paramètres de service pour ce service sont montrés dans le Tableau 2. Les paramètres de réponse sont destinés à une utilisation locale au sein du REP qui répond et de son objet de variable.

Tableau 2 – Paramètres du service REPONSE

Nom du paramètre	Rsp	Cnf
Argument	M	M
REP	M	M
Résultat		
Service variable	M	M
Longueur des données	M	M(=)
Etat d'erreur	M	M(=)
Données	C	C(=)

– Argument

L'argument contient les paramètres de la demande de service.

– REP

Ce paramètre contient l'ID du REP contenant les informations relatives au chemin.

– Service variable

Ce paramètre contient le service variable ayant été effectué. (Read, Write, And, Or, Test and Set, Get attribute et Set Attribute).

– Longueur des données

Ce paramètre indique le nombre d'octets des données ayant été transférées.

– Etat d'erreur

Ce paramètre contient les informations relatives aux erreurs qui se sont produites dans une couche locale ou distante. Pour plus de détails, se reporter au Tableau 3.

Le code d'erreur peut être généré n'importe ou sur le chemin entre le client et le serveur, ou par l'objet de variable qui répond.

Tableau 3 – Codes d'erreur par source

Description de l'erreur	Client	Passerelle	Serveur
Application utilisateur			
Erreur d'instruction	X		X
Erreur de longueur d'info	X		X
FIFO plein ou vide			X
Protection en écriture			X
Erreur de format de données			X
Erreur d'ID d'objet variable			X
Temporisation	X		
Erreur des données réelles			X
Erreur des données historiques			X
FAL			
Erreur de route	X	X	X
Couche DL			
Pas de réponse	X	X	
DELAI D'ATTENTE TROP LONG	X	X	
Désynchronisé	X	X	
ERREUR CRC	X	X	
DLE pas client	X	X	
Couche physique			
COURT-CIRCUIT NET	X	X	
ERREUR DE DEPASSEMENT/D'ENCADREMENT	X	X	
ERREUR D'ETABLISSEMENT DE LIAISON RS-232	X	X	X

– Données

Ce paramètre contient les données qui ont été transférées.

5.2.6.3 Service Reserve REP

5.2.6.3.1 Présentation

Ce service est utilisé par l'application utilisateur pour réserver un REP qui n'est pas utilisé actuellement, et pour obtenir l'ID du REP réservé.

5.2.6.3.2 Primitives de service

Les paramètres de service pour ce service sont montrés dans le Tableau 4.

Tableau 4 – Paramètre du service Reserve REP

Nom du paramètre	Req	Cnf
Résultat		
REP		M

– REP

Ce paramètre contient l'ID du REP réservé. Si aucun REP n'est disponible, zéro est retourné.

5.2.6.4 Service Free REP

5.2.6.4.1 Présentation

Ce service est utilisé par l'application utilisateur pour libérer un REP en changeant son état pour "not in use".

5.2.6.4.2 Primitives de service

Les paramètres de service pour ce service sont montrés dans le Tableau 5.

Tableau 5 – Paramètres du service Free AREP

Nom du paramètre	Req
Argument	M
REP	M

– Argument

L'argument contient les paramètres de la demande de service.

– REP

Ce paramètre contient l'adresse EP du REP réservé à libérer.

5.2.6.5 Service Get REP attribute

5.2.6.5.1 Présentation

Ce service est utilisé par l'application utilisateur pour lire un attribut d'un REP localement.

5.2.6.5.2 Primitives de service

Les paramètres de service pour ce service sont montrés dans le Tableau 6.

Tableau 6 – Paramètres du service de l'attribut Get REP

Nom du paramètre	Req	Cnf
Argument	M	
REP	M	
Index d'attribut	M	
Résultat (+)		
Valeur		C
Résultat (-)		
Statut		C

– Argument

L'argument contient les paramètres de la demande de service.

– REP

Ce paramètre spécifie le REP.

– Index d'attribut

Ce paramètre identifie l'attribut à lire.

– Valeur

Ce paramètre renvoie la valeur de l'attribut spécifié

– Statut

Ce paramètre renvoie des informations d'erreur si le service échoue. Les valeurs possibles sont: Illegal attribute index, Illegal REP address.

5.2.6.6 Service Set REP attribute

5.2.6.6.1 Présentation

Ce service est utilisé par l'application utilisateur pour écrire une valeur dans un attribut d'un REP localement.

5.2.6.6.2 Primitives de service

Les paramètres de service pour ce service sont montrés dans le Tableau 7.

Tableau 7 – Paramètres du service de l'attribut Set REP

Nom du paramètre	Req	Cnf
Argument	M	
REP	M	
Index d'attribut	M	
Résultat		
Statut		M

– Argument

L'argument contient les paramètres de la demande de service.

– REP

Ce paramètre spécifie le REP.

– Index d'attribut

Ce paramètre identifie l'attribut à mettre à jour au sein du REP spécifié.

– Valeur

Ce paramètre contient la valeur à écrire dans l'attribut spécifié.

– Statut

Résultat de la demande, Statut est retourné en indiquant OK ou la raison de la défaillance. Les valeurs possibles sont: OK, Illegal attribute index, Illegal value, Illegal REP address.

5.3 ASE de relations entre applications

5.3.1 Présentation

Pour prendre en charge l'accès à l'AP distant, l'ASE de relations entre applications est défini. Il fournit des services à l'AP pour accéder aux paramètres associés à la communication, ainsi que des services au REP ASE pour transmettre des demandes et des réponses de service.

L'AR ASE fournit les services aux points de fin des AR (AREP).

5.3.2 Spécification de la classe de relation d'applications

5.3.2.1 Modèle formel de relation d'applications

La fonctionnalité de l'AR ASE est décrite au en 5.1.

L'ASE application définit une classe, la classe AREP.

FAL ASE:			**ASE de relations entre applications**
CLASSE:			**AREP**
ID CLASSE:			1
CLASSE PARENT:			HAUT
ATTRIBUTS:			
1.	(m)	Attribut clé:	Adresse de point de fin
2.	(m)	Attribut:	Rôle (Client, Serveur, Pair)
3.	(m)	Attribut:	Référence DLL
4.	(m)	Attribut:	MaxPDUSize
5.	(m)	Attribut:	MaxDataSize
6.	(m)	Attribut:	Acquittement
7.	(m)	Attribut:	MaxIndicationDelay
8.	(m)	Attribut:	Adresse DLE locale
9.	(o)	Attribut:	MaxRetryTime
10.	(o)	Attribut:	MaxRetries
11.	(o)	Attribut:	MaxOutstandingRequests
12.	(o)	Attribut:	BaudRate
13.	(o)	Attribut:	NumberOfClientDLEs

5.3.2.1.1.1 SERVICES:

1	(m)	OpsService:	AR-Get Attribute
2	(m)	OpsService:	AR-Set Attribute
3	(m)	OpsService:	AR-Send
4	(m)	OpsService:	AR-Acknowledge

5.3.2.1.1.2 Adresse de point de fin

Cet attribut spécifie l'identificateur numérique de l'AREP. Il est utilisé par le FAL pour sélectionner l'AREP et implicitement le DLE d'une demande.

5.3.2.1.1.3 Rôle

Cet attribut spécifie le rôle de l'AREP. Les valeurs valides sont les suivantes:

Client Les points d'extrémité de ce type envoient des APDU de demande confirmée et non confirmée aux serveurs et reçoivent des APDU de réponse.

Serveur Les points d'extrémité de ce type reçoivent des APDU de demande confirmée et non confirmée à partir des clients et envoient des APDU de réponse non confirmée.

Pair Les points d'extrémité de ce type agissent à la fois comme clients et comme serveurs.

5.3.2.1.1.4 MaxPDUSize

Cet attribut spécifie la taille maximum d'une unité PDU qui peut être envoyée par la couche DLL.

5.3.2.1.1.5 MaxDataSize

Cet attribut spécifie la taille maximum des données qui peuvent être envoyées par la couche DLL. Si la longueur des données dépasse MaxDataSize, le service VARIABLE ASE doit segmenter la demande.

5.3.2.1.1.6 Référence DLL

Cet attribut contient le contexte nécessaire pour transmettre les relations DLL.

5.3.2.1.1.7 Acquittement

Cet attribut décrit le type d'acquittement à utiliser par la couche DLL: Le transfert acquitté ou non acquitté des APDU confirmés, et le transfert acquitté ou non acquitté des APDU non confirmés.

5.3.2.1.1.8 MaxIndicationDelay

Cet attribut indique à l'application combien de temps un objet de variable peut utiliser pour préparer une réponse après avoir reçu une indication qui le nécessite. Si l'objet de variable n'est pas en mesure de préparer une réponse au sein du MaxIndicationDelay, il doit émettre un AR-Acknowledge. La valeur de MaxIndicationDelay est calculée par le DLE.

5.3.2.1.1.9 Adresse DLE locale

Cet attribut reflète l'adresse DL du DLE associé. Dans certaines situations il peut s'agir d'un attribut en lecture seule.

5.3.2.1.1.10 MaxRetryTime

Cet attribut indique le délai maximal pendant lequel il convient de tenter pour le DLE de retransmettre une demande comme résultat des réponses d'acquittement à partir de l'objet de variable distant.

5.3.2.1.1.11 MaxRetries

Cet attribut indique le nombre maximal de retransmissions effectuées par le DLE à la suite des erreurs de transmission.

5.3.2.1.1.12 MaxOutstandingRequests

Cet attribut indique le nombre maximal de demandes que l'AREP peut effectuer sans recevoir les réponses associées.

5.3.2.1.1.13 BaudRate

Cet attribut spécifie le débit utilisé par la couche physique. Il est transmis vers/à partir des couches physiques par les DLE.

5.3.2.1.1.14 NumberOfClientDLEs

Cet attribut ne s'associe qu'à un DLE avec le protocole P-NET dans le rôle du client. Il spécifie le nombre de participants d'un cycle de jeton.

5.3.3 Spécifications de service de l'ASE de relations entre applications

5.3.3.1 Présentation

Le Paragraphe 5.3.3 contient la définition des services qui sont propres à cet ASE. Ces services sont les suivants:

AR Send

AR Acknowledge

AR Get Attribute

AR Set Attribute

5.3.3.2 Définition des paramètres communs

Les paramètres utilisés dans plus de services AR ASE sont définis ci-dessous:

5.3.3.2.1.1 Info de route

Le chemin DL complet contient les éléments suivants:

5.3.3.2.1.2 Route de destination

Le chemin de destination décrit comment atteindre le REP de destination. C'est une séquence d'adresses de point d'extrémité et d'adresses DL. Sur son itinéraire de destination, la première partie indique toujours l'adresse du DLE, de l'AREP ou du REP suivant pour recevoir l'APDU.

5.3.3.2.1.3 Route source

De la même manière, le chemin de départ décrit comment retrouver le chemin vers le point d'extrémité qui est à l'origine de la demande. C'est une séquence d'adresses de point d'extrémité et d'adresses DL. Sur son itinéraire de destination, le premier élément indique toujours l'adresse du DLE, de l'AREP ou du point d'extrémité du REP où l'APDU a été envoyé.

5.3.3.2.1.4 Priorité

La couche sous-jacente peut permettre d'envoyer des APDU à haute priorité avant les APDU à faible priorité. Cette priorité ne fait référence qu'à la demande sur le lien local, pas aux demandes franchissant des passerelles et pas à la réponse.

5.3.3.2.1.5 Confirmation

Ce paramètre indique si la demande doit être confirmée ou non confirmée. Une réponse est toujours retournée non confirmée. Si le chemin de destination contient une ou plusieurs adresses de diffusion (126), cet attribut doit être défini sur non confirmé.

5.3.3.2.1.6 En-tête APDU

Le paramètre En-tête APDU contient une sous-trame Control/Statut indiquant le service de variable, une sous-trame de format d'APDU, indiquant si l'APDU contient un décalage, et enfin une sous-trame de longueur d'APDU, indiquant la longueur d'octet de l'APDU.

5.3.3.2.1.7 Corps APDU

Ce paramètre contient le corps d'APDU à envoyer ou recevoir.

5.3.3.3 Service AR send

5.3.3.3.1 Présentation

Le service AR Send permet d'envoyer des APDU confirmés et non confirmés d'un AREP vers un autre.

C'est l'AREP de réception qui doit transmettre l'APDU au point d'extrémité spécifié par le premier élément du chemin de destination. Dans la situation de passerelle, l'AR utilise ce service AR Send pour transmettre l'APDU à un AREP.

5.3.3.3.2 Primitives de service

Les paramètres de service pour ce service sont montrés dans le Tableau 8.

Tableau 8 – Paramètres du service AR send

Nom du paramètre	Req	Ind	Cnf
Argument	M	M	
Info de route	M	M	
En-tête APDU	M	M(=)	
Corps APDU	C	C(=)	
Résultat			
Statut			M

– Argument

  L'argument contient les paramètres de la demande de service.

– Statut

  Résultat de la demande, Statut est retourné en indiquant OK ou la raison de la défaillance détectée localement. Statut ne fait référence qu'aux conditions locales. Les valeurs possibles sont: OK, Route Error.

5.3.3.4 Service AR acknowledge

5.3.3.4.1 Présentation

Si l'ASE de variable d'une demande confirmée n'est pas en mesure de traiter l'indication et de répondre au sein du délai, "MaxIndicationDelay", il doit soumettre une primitive de demande AR Send Acknowledge à l'AREP local, et de cette manière, il libère le jeton. "MaxIndicationDelay" est un attribut de l'AREP.

5.3.3.4.2 Primitives de service

Les paramètres de service pour ce service sont montrés dans le Tableau 9.

Tableau 9 – Paramètres du service AR acknowledge

Nom du paramètre	Req
Argument	M
Info de route	M

– Argument

L'argument contient les paramètres de la demande de service.

Service AR get attribute

5.3.3.4.3 Présentation

Ce service confirmé est utilisé pour lire la valeur des attributs d'un AREP localement.

5.3.3.4.4 Paramètres du service

Les paramètres de service du service AR Get Attribute sont illustrés dans le Tableau 10.

Tableau 10 – Paramètres du service AR get attributes

Nom du paramètre	Req	Cnf
Argument	M	
AREP	M	
Index d'attribut	M	
Résultat (+)		S
Valeur		C
Résultat (-)		S
Statut		C

– Argument

L'argument contient les paramètres de la demande de service.

– AREP

Ce paramètre identifie l'AREP par son adresse de point d'extrémité.

– Index d'attribut

Ce paramètre identifie l'attribut.

– Résultat (+)

Ce paramètre de type sélection indique que la demande a réussi.

– Valeur

Ce paramètre est envoyé avec la valeur de l'attribut demandé si la demande a réussi.

– Résultat (-)

Ce paramètre de type sélection indique que la demande a échoué.

– Statut

Si la demande a échoué, Statut est retourné en indiquant la raison de la défaillance. Les valeurs possibles sont: Illegal attribute index, Illegal AREP address.

5.3.3.5 Service AR set attributes

Ce service confirmé est utilisé pour définir la valeur actuelle des attributs d'un AREP localement.

5.3.3.5.1 Paramètres du service

Les paramètres de service du service AR Set Attribute sont illustrés dans le Tableau 11.

Tableau 11 – Paramètres du service AR set attributes

Nom du paramètre	Req	Cnf
Argument	M	
AREP	M	
Index d'attribut	M	
Valeur	M	
Résultat		
Statut		M

– Argument

L'argument contient les paramètres de la demande de service.

– AREP

Ce paramètre identifie l'AREP par son adresse de point d'extrémité. Les valeurs légales vont de 1 à 16 inclus.

– Index d'attribut

Ce paramètre identifie l'attribut à mettre à jour.

– Valeur

Ce paramètre contient la nouvelle valeur de l'attribut.

– Statut

Résultat de la demande, Statut est retourné en indiquant OK ou la raison de la défaillance. Les valeurs possibles sont: OK, Illegal attribute index, Illegal value, Illegal AREP address.

Bibliographie

CEI 61158-1:2014, *Réseaux de communication industriels – Spécifications des bus de terrain – Partie 1: Présentation et lignes directrices des séries CEI 61158 et CEI 61784*

CEI 61784-1:2014, *Réseaux de communication industriels – Profils – Partie 1: Profils de bus de terrain*

CEI 61784-2:2014, *Réseaux de communication industriels – Profils – Partie 2: Profils de bus de terrain supplémentaires pour les réseaux en temps réel basés sur l'ISO/CEI 8802-3*

_____

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

3, rue de Varembé
PO Box 131
CH-1211 Geneva 20
Switzerland

Tel: + 41 22 919 02 11
Fax: + 41 22 919 03 00
info@iec.ch
www.iec.ch