

INTERNATIONAL STANDARD

NORME INTERNATIONALE

**Industrial communication networks – Fieldbus specifications –
Part 5-7: Application Layer Service definition – Type 7 elements**

**Réseaux de communication industriels – Spécifications des bus de terrain –
Partie 5-7: Définition des services des couches d'application –
Éléments de Type 7**



THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2007 IEC, Geneva, Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either IEC or IEC's member National Committee in the country of the requester. If you have any questions about IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

Droits de reproduction réservés. Sauf indication contraire, aucune partie de cette publication ne peut être reproduite ni utilisée sous quelque forme que ce soit et par aucun procédé, électronique ou mécanique, y compris la photocopie et les microfilms, sans l'accord écrit de l'IEC ou du Comité national de l'IEC du pays du demandeur. Si vous avez des questions sur le copyright de l'IEC ou si vous désirez obtenir des droits supplémentaires sur cette publication, utilisez les coordonnées ci-après ou contactez le Comité national de l'IEC de votre pays de résidence.

IEC Central Office
3, rue de Varembe
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 61158-5-7

Edition 1.0 2007-12

INTERNATIONAL STANDARD

NORME INTERNATIONALE

**Industrial communication networks – Fieldbus specifications –
Part 5-7: Application Layer Service definition – Type 7 elements**

**Réseaux de communication industriels – Spécifications des bus de terrain –
Partie 5-7: Définition des services des couches d'application –
Éléments de Type 7**

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

COMMISSION
ELECTROTECHNIQUE
INTERNATIONALE

PRICE CODE
CODE PRIX

XH

ICS 25.040.40; 35.100.70

ISBN 978-2-8322-2000-9

**Warning! Make sure that you obtained this publication from an authorized distributor.
Attention! Veuillez vous assurer que vous avez obtenu cette publication via un distributeur agréé.**

CONTENTS

FOREWORD.....	7
INTRODUCTION.....	9
1 Scope.....	10
1.1 Overview.....	10
1.2 Specifications.....	11
1.3 Conformance.....	11
2 Normative references	11
3 Terms, definitions, symbols, abbreviations and conventions	12
3.1 ISO/IEC 7498-1 terms	12
3.2 ISO/IEC 8822 terms	12
3.3 ISO/IEC 9545 terms	12
3.4 ISO/IEC 8824 terms	12
3.5 Fieldbus data-link layer terms.....	12
3.6 Fieldbus application layer specific definitions	14
3.7 Abbreviations and symbols.....	21
3.8 Conventions	22
4 Concepts.....	26
5 Data type ASE.....	26
5.1 Overview.....	26
5.2 Formal definition of data type objects	26
5.3 FAL defined data types.....	26
6 Communication model specification.....	28
6.1 Concepts.....	28
6.2 ASEs.....	45
6.3 ARs.....	215
Bibliography.....	236
Figure 1 – Organisation of the ASEs and ARs.....	29
Figure 2 – Object model of the MPS ASE.....	49
Figure 3 – Time-out evaluation net.....	61
Figure 4 – Asynchronous promptness status evaluation net	65
Figure 5 – Synchronous promptness status evaluation net.....	66
Figure 6 – Punctual promptness status evaluation net	68
Figure 7 – Asynchronous refreshment status evaluation net.....	71
Figure 8 – Synchronous refreshment status evaluation net	72
Figure 9 – Punctual refreshment status evaluation net.....	74
Figure 10 – A_Readloc service procedure.....	77
Figure 11 – A_Writeloc service procedure.....	79
Figure 12 – A_Update service procedure	81
Figure 13 – A_Readfar service procedure	83
Figure 14 – A_Writefar service procedure	85
Figure 15 – A_Sent service procedure	86
Figure 16 – A_Received service procedure.....	87
Figure 17 – A_Read service procedure	89

Figure 18 – A_Read service state machine	90
Figure 19 – A_Write service procedure	91
Figure 20 – A_Write service state machine	92
Figure 21 – Model of a resynchronised variable	95
Figure 22 – Principles for resynchronisation of a produced variable	96
Figure 23 – Resynchronisation mechanism state machine for a produced variable.....	98
Figure 24 – Asynchronous refreshment private mechanism evaluation net	99
Figure 25 – Asynchronous refreshment public mechanism evaluation net	100
Figure 26 – Synchronous refreshment private mechanism evaluation net.....	101
Figure 27 – Synchronous refreshment public mechanism evaluation net	102
Figure 28 – Punctual refreshment private mechanism evaluation net	103
Figure 29 – Punctual refreshment public mechanism evaluation net.....	104
Figure 30 – Principles for the resynchronisation of a consumed variable.....	105
Figure 31 – Resynchronisation mechanism state machine for consumed variable	107
Figure 32 – Asynchronous promptness public mechanism evaluation net.....	108
Figure 33 – Asynchronous promptness private mechanism evaluation net	109
Figure 34 – Synchronous promptness public mechanism evaluation net	110
Figure 35 – Synchronous promptness private mechanism evaluation net	111
Figure 36 – Punctual promptness public mechanism evaluation net	113
Figure 37 – Punctual promptness private mechanism evaluation net.....	114
Figure 38 – Spatial consistency list variables interchange mechanism	116
Figure 39 – Spatial consistency – consistency variable interchange mechanism	117
Figure 40 – Spatial consistency – list recovery mechanism	117
Figure 41 – Spatial consistency – validity of the spatial consistency status	118
Figure 42 – Object model of a variable list	118
Figure 43 – A_Readlist service procedure.....	124
Figure 44 – Consistency variable value evaluation net.....	130
Figure 45 – Consistency interchange timing diagram	131
Figure 46 – Recovery mechanism evaluation net	132
Figure 47 – Recovery interchange timing diagram.....	132
Figure 48 – Flowchart of the sub-MMS environment management state	138
Figure 49 – Domain management state chart.....	169
Figure 50 – Domain upload flowchart	171
Figure 51 – Domain download sequence diagram	172
Figure 52 – Domain upload sequence diagram	172
Figure 53 – Program invocation state chart.....	185
Figure 54 – A_Associate service procedure	224
Figure 55 – A_Release service procedure.....	227
Figure 56 – A_Abort service procedure	228
Figure 57 – A_Data service procedure	230
Figure 58 – A_Unidata service procedure	233
Figure 59 – Associated mode service state chart	234
Figure 60 – Non-associated mode service state chart	235

Table 1 – Binary time coding.....	27
Table 2 – Access protection.....	44
Table 3 – Binary time coding.....	60
Table 4 – Asynchronous promptness events and actions	65
Table 5 – Synchronous promptness events and actions	66
Table 6 – Punctual promptness events and actions.....	68
Table 7 – Asynchronous refreshment events and actions.....	71
Table 8 – Synchronous refreshment events and actions.....	72
Table 9 – Punctual refreshment events and actions	75
Table 10 – A_Readloc service parameters	76
Table 11 – A_Writeloc service parameters	78
Table 12 – A_Update service parameters	80
Table 13 – A_Readfar service parameters	82
Table 14 – A_Writefar service parameters	84
Table 15 – A_Sent service parameters	86
Table 16 – A_Received service parameters	87
Table 17 – A_Read service parameters	88
Table 18 – A_Write service parameters	90
Table 19 – Asynchronous refreshment private mechanism events and actions.....	99
Table 20 – Asynchronous refreshment public mechanism events and actions	100
Table 21 – Synchronous refreshment private mechanism events and actions.....	101
Table 22 – Synchronous refreshment public mechanism events and actions	102
Table 23 – Punctual refreshment private mechanism events and actions	104
Table 24 – Punctual refreshment public mechanism events and actions.....	105
Table 25 – Asynchronous promptness public mechanism events and actions.....	108
Table 26 – Asynchronous promptness private mechanism events and actions	109
Table 27 – Synchronous promptness public mechanism events and actions	110
Table 28 – Synchronous promptness privatemechanism events and actions	112
Table 29 – Punctual promptness public mechanism events and actions	113
Table 30 – Punctual promptness privatemechanism events and actions.....	114
Table 31 – A_Readlist service parameters	123
Table 32 – Confirmed initiate service parameters.....	143
Table 33 – Detailed structure of the extension calling parameter	144
Table 34 – Detailed structure of the init request detail parameter.....	145
Table 35 – Detailed structure of the extension called parameter	146
Table 36 – Detailed structure of the init request detail parameter.....	147
Table 37 – Conclude service parameter	148
Table 38 – Unconfirmed abort service parameters	150
Table 39 – Unconfirmed reject service parameters.....	151
Table 40 – Confirmed status service parameters	153
Table 41 – Unconfirmed unsolicited status service parameter	154
Table 42 – Confirmed identify service parameters.....	154

Table 43 – Confirmed get name list service parameters	155
Table 44 – Access group attribute description for domain object	158
Table 45 – Access rights attribute description for domain object	158
Table 46 – Confirmed delete domain service parameters	159
Table 47 – Confirmed initiate download sequence service parameters	160
Table 48 – Confirmed download segment service parameters	161
Table 49 – Confirmed terminate download sequence service parameters	162
Table 50 – Confirmed initiate upload sequence service parameters	164
Table 51 – Confirmed upload segment service parameters	165
Table 52 – Confirmed terminate upload sequence service parameters	166
Table 53 – Confirmed get domain attributes service parameters	167
Table 54 – Access group attribute details for program invocation object	174
Table 55 – Access rights attribute details for program invocation object	175
Table 56 – Confirmed create program invocation service parameters	176
Table 57 – Confirmed delete program invocation service parameters	178
Table 58 – Confirmed start service parameters	179
Table 59 – Confirmed stop service parameters	180
Table 60 – Confirmed resume service parameters	181
Table 61 – Confirmed reset service parameters	182
Table 62 – Confirmed kill service parameters	183
Table 63 – Access group attribute details for variable object	187
Table 64 – Access rights attribute details for variable object	188
Table 65 – Access group attribute details for variable list object	189
Table 66 – Access right attribute details for variable list objects	189
Table 67 – Confirmed read service parameters	190
Table 68 – Confirmed write service parameters	192
Table 69 – Unconfirmed information report service parameters	193
Table 70 – Confirmed define variable-list service parameters	194
Table 71 – Confirmed delete variable-list service parameters	196
Table 72 – Confirmed get variable access attributes service parameters	197
Table 73 – Confirmed get variable-list attributes service parameters	198
Table 74 – Data type specification	200
Table 75 – Variable access specification	201
Table 76 – Variable access description attribute details	201
Table 77 – Path selection parameters	202
Table 78 – Access group attribute detail for event object	205
Table 79 – Access rights attribute details for event object	206
Table 80 – Unconfirmed event notification service parameters	207
Table 81 – Event type parameter details	207
Table 82 – Confirmed acknowledged event notification service parameter	209
Table 83 – Confirmed alter event condition monitoring service parameters	210
Table 84 – Confirmed get alarm summary service parameters	212
Table 85 – Confirmed get event condition attributes service parameters	214

Table 86 – Classification of service quality parameters	217
Table 87 – Identification parameters	221
Table 88 – List of MCS AR ASE services	222
Table 89 – A_Associate service parameters.....	222
Table 90 – A_Release service parameters	227
Table 91 – A_Abort service parameters	228
Table 92 – A_Data service parameters	229
Table 93 – A_Unidata service parameters.....	230

INTERNATIONAL ELECTROTECHNICAL COMMISSION

**INDUSTRIAL COMMUNICATION NETWORKS –
FIELDBUS SPECIFICATIONS –****Part 5-7: Application Layer Service definition – Type 7 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 provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with an IEC Publication.
- 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.

NOTE Use of some of the associated protocol types is restricted by their 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 particular data-link layer protocol type to be used with physical layer and application layer protocols in Type combinations as specified explicitly in the IEC 61784 series. Use of the various protocol types in other combinations may require permission from their respective intellectual-property-right holders.

International Standard IEC 61158-5-7 has been prepared by subcommittee 65C: Industrial networks, of IEC technical committee 65: Industrial-process measurement, control and automation.

This first edition and its companion parts of the IEC 61158-5 subseries cancel and replace IEC 61158-5:2003. This edition of this part constitutes an editorial revision.

This edition of IEC 61158-5 includes the following significant changes from the previous edition:

- a) deletion of the former Type 6 fieldbus for lack of market relevance;
- b) addition of new types of fieldbuses;
- c) partition of part 5 of the third edition into multiple parts numbered -5-2, -5-3, ...

This bilingual version (2014-12) corresponds to the monolingual English version, published in 2007-12.

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

FDIS	Report on voting
65C/475/FDIS	65C/486/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.

The French version of this standard has not been voted upon.

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

The committee has decided that the contents of this publication will remain unchanged until the maintenance result 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.

NOTE The revision of this standard will be synchronized with the other parts of the IEC 61158 series.

The 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.

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/TR 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 5-7: Application Layer Service definition – Type 7 elements

1 Scope

1.1 Overview

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. 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 7 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 IEC fieldbus Application Layer, in conformance with the OSI Basic Reference Model (ISO/IEC 7498) 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.

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 7 application layer services as defined in this standard.

2 Normative references

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

IEC 60559, *Binary Floating-point Arithmetic for Microprocessor Systems*

IEC/TR 61158-1 (Ed.2.0), *Industrial communication networks – Fieldbus specifications – Part 1: Overview and guidance for the IEC 61158 and IEC 61784 series*

IEC 61158-3-7, *Industrial communication networks – Fieldbus specifications – Part 3-7: Data-link layer service definition – Type 7 elements*

IEC 61158-4-7, *Industrial communication networks – Fieldbus specifications – Part 4-7: Data-link layer protocol specification – Type 7 elements*

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, *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*

3 Terms, definitions, symbols, abbreviations and conventions

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

3.1 ISO/IEC 7498-1 terms

For the purposes of this document, the following terms as defined in ISO/IEC 7498-1 apply:

- 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

- a) abstract syntax
- b) presentation context

3.3 ISO/IEC 9545 terms

- 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 terms

- a) object identifier
- b) type

3.5 Fieldbus data-link layer terms

The following terms as defined in IEC 61158-3-7 and IEC 61158-4-7 apply.

- a) acknowledgement response DLPDU
- b) basic cycle
- c) basic transaction

- d) bus-arbitrator (BA)
- e) control field
- f) destination address
- e) DL-segment, local link
- g) DLCEP-identifier
- h) DLCEP-identifier DLPDU
- i) end of message transaction indication DLPDU
- j) identified variable (or simply « variable »)
- k) invalid DLCEP-identifier
- l) macrocycle
- m) message DLPDU identifier
- n) message response DLPDU
- o) periodic scanning of variables
- p) published identified variable
- q) request response DLPDU
- r) source address
- s) subscribed identified variable
- t) triggered message scanning
- u) triggered periodic scanning of messages
- w) triggered periodic scanning of variables
- x) triggered scanning of variables
- y) turnaround time
- z) variable response DLPDU

The following symbols and abbreviations as defined in IEC 61158-3-7 and IEC 61158-4-7 apply.

- a) BA
- b) B_Dat_cons
- c) B_Dat_Prod
- d) B_Req1/2
- e) ID_DAT
- f) ID_MSG
- g) ID_RQ1/2
- h) PRT
- i) Q_IDMSG
- j) Q_IDRQ1/2
- k) Q_Msg_Aper
- l) Q_Req1/2
- m) Q_RPRQ
- n) RQ_Inhibit
- o) RP_ACK
- p) RP_DAT
- q) RP_DAT_MSG

- r) RP_DAT_RQ1/2
- s) RP_DAT_RQ1/2_MSG
- t) RP_MSG_ACK
- u) RP_MSG_NOACK
- v) RP_NAK
- w) RP_END
- x) STT
- y) TI

3.6 Fieldbus application layer specific definitions

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

3.6.1

access protection

limitation of the usage of an application object to one client

3.6.2

active connection control object

instance of a certain FAL class that abstracts the interconnection facility (as Consumer and Provider) of an automation device

3.6.3

address assignment table

mapping of the client's internal I/O-Data object storage to the decentralised input and output data objects

3.6.4

allocate

take a resource from a common area and assign that resource for the exclusive use of a specific entity

3.6.5

application

function or data structure for which data is consumed or produced

3.6.6

application layer interoperability

capability of application entities to perform coordinated and cooperative operations using the services of the FAL

3.6.7

application objects

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

3.6.8

application process

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

3.6.9

application process identifier

distinguishes multiple application processes used in a device

3.6.10**application process object**

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

NOTE 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.

3.6.11**application process object class**

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

3.6.12**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 This relationship is activated either by the exchange of application-protocol-data-units or as a result of preconfiguration activities.

3.6.13**application relationship application service element**

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

3.6.14**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 Each application process involved in the application relationship maintains its own application relationship endpoint.

3.6.15**attribute**

description of an externally visible characteristic or feature of an object

NOTE 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.

3.6.16**behaviour**

indication of how an object responds to particular eventss

3.6.17**bit-no**

designates the number of a bit in a bitstring or an octet

3.6.18**channel**

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

3.6.19**class**

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

NOTE 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.

3.6.20

class attributes

attribute that is shared by all objects within the same class

3.6.21

class code

unique identifier assigned to each object class

3.6.22

class specific service

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

NOTE A class specific object is unique to the object class which defines it.

3.6.23

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

3.6.24

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

3.6.25

configuration identifier

representation of a portion of I/O Data of a single input- and/or output-module of a server

3.6.26

connection

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

NOTE 1 Connections may be either point-to-point or multipoint.

NOTE 2 The logical link between sink and source of attributes and services at different custom interfaces of RT-Auto ASES is referred to as interconnection. There is a distinction between data and event interconnections. The logical link and the data flow between sink and source of automation data items is referred to as data interconnection. The logical link and the data flow between sink (method) and source (event) of operational services is referred to as event interconnection.

3.6.27

connection channel

description of a connection between a sink and a source of data items

3.6.28

connection ID (CID)

identifier assigned to a transmission that is associated with a particular connection between producers and consumers, providing a name for a specific piece of application information

3.6.29

connection path

octet stream that defines the application object to which a connection instance applies

3.6.30

connection point

buffer which is represented as a subinstance of an Assembly object

3.6.31**consume**

act of receiving data from a producer

3.6.32**consumer**

node or sink that is receiving data from a producer

3.6.33**consuming application**

application that consumes data

3.6.34**consumerID**

unambiguous identifier within the scope of the ACCO assigned by the consumer to recognize the internal data of a configured interconnection sink

3.6.35**control commands**

action invocations transferred from client to server to clear outputs, freeze inputs and/or synchronise outputs

3.6.36**conveyance path**

unidirectional flow of APDUs across an application relationship

3.6.37**cyclic**

repetitive in a regular manner

3.6.38**data consistency**

means for coherent transmission and access of the input- or output-data object between and within client and server

3.6.39**dedicated AR**

AR used directly by the FAL User

NOTE On Dedicated ARs, only the FAL Header and the user data are transferred.

3.6.40**device**

physical hardware connected to the link

NOTE A device may contain more than one node.

3.6.41**device profile**

collection of device dependent information and functionality providing consistency between similar devices of the same device type

3.6.42**end node**

producing or consuming node

3.6.43**endpoint**

one of the communicating entities involved in a connection

3.6.44

error

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

3.6.45

error class

general grouping for related error definitions and corresponding error codes

3.6.46

error code

identification of a specific type of error within an error class

3.6.47

event

instance of a change of conditions

3.6.48

FIFO variable

a 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 On the fieldbus only one, complete element can be transferred as a result of one service invocation.

3.6.49

frame

denigrated synonym for DLPDU

3.6.50

group

- a) <general> a general term for a collection of objects.
- b) <addressing> when describing an address, an address that identifies more than one entity

3.6.51

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

3.6.52

interface definition language

syntax and semantics of describing service parameters in a formal way

NOTE This description is the input for the ORPC model, especially for the ORPC wire protocol.

3.6.53

interface pointer

key attribute that unambiguously addresses an object interface instance

3.6.54

invocation

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

NOTE 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.

3.6.55**index**

address of an object within an application process

3.6.56**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 The terms object, instance, and object instance are used to refer to a specific instance.

3.6.57**instance attributes**

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

3.6.58**instantiated**

object that has been created in a device

3.6.59**logical device**

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

3.6.60**manufacturer ID**

identification of each product manufacturer by a unique number

3.6.61**management information**

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

NOTE Managing includes functions such as controlling, monitoring, and diagnosing.

3.6.62**member**

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

3.6.63**method**

synonym for an operational service which is provided by the server ASE and invoked by a client

3.6.64**module**

hardware or logical component of a physical device

3.6.65**multipoint connection**

connection from one node to many

NOTE Multipoint connections allow messages from a single producer to be received by many consumer nodes.

3.6.66**network**

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

3.6.67

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

3.6.68

object remote procedure call

model for object oriented or component based remote method invocation

3.6.69

object specific service

service unique to the object class which defines it

3.6.70

originator

client responsible for establishing a connection path to the target

3.6.71

peer

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

3.6.72

physical device

an automation or other network device

3.6.73

point-to-point connection

connection that exists between exactly two application objects

3.6.74

pre-defined AR endpoint

AR endpoint that is defined locally within a device without use of the create service

NOTE Pre-defined ARs that are not pre-established are established before being used.

3.6.75

pre-established AR endpoint

AR endpoint that is placed in an established state during configuration of the AEs that control its endpoints

3.6.76

process data

object(s) which are already pre-processed and transferred acyclically for the purpose of information or further processing

3.6.77

produce

act of sending data to be received by a consumer

3.6.78

producer

node that is responsible for sending data

3.6.79

provider

source of a data connection

3.6.80**publisher**

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

NOTE A publisher may not be aware of the identity or the number of subscribers and it may publish its APDUs using a dedicated AR.

3.6.81**resource**

processing or information capability of a subsystem

3.6.82**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

3.6.83**service**

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

3.6.84**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
CID	Connection ID
CIM	Computer Integrated Manufacturing
CIP	Control and Information Protocol
Cnf	Confirmation
COR	Connection originator
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
ORPC	Object Remote Procedure Call
OSI	Open Systems Interconnect
PDev	Physical Device
PDU	Protocol Data Unit
PL	Physical Layer
QoS	Quality of Service
Req	Request
Rsp	Response
RT	Runtime
SAP	Service Access Point
SCL	Security Level
SDU	Service Data Unit
SMIB	System Management Information Base
SMK	System Management Kernel
STD	State transition diagram, used to describe object behaviour
S-VFD	Simple Virtual Field Device
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 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.

3.8.3 Conventions for service definitions

3.8.3.1 General

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

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

3.8.3.2 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.

NOTE 1 See the note under 3.8.3.3 relative to the non-inclusion of service parameters that are appropriate to a protocol specification or programming interface specification or implementation specification, but not to an abstract service definition.

This standard uses a tabular format to describe the component parameters of the service primitives. The parameters that apply to each group of service primitives are set out in tables throughout the remainder of this standard. Each table consists of up to six columns: a column for the name of the service parameter, and a column each for those primitives and parameter-transfer directions used by the service. The possible six columns are:

- 1) the parameter name;
- 2) the request primitive's input parameters;

3) the request primitive's output parameters;

NOTE 2 This is a seldom-used capability. Unless otherwise specified, request primitive parameters are input parameters.

4) the indication primitive's output parameters;

5) the response primitive's input parameters; and

6) the confirm primitive's output parameters.

NOTE 3 The request, indication, response and confirm primitives are also known as requestor.submit, acceptor.deliver, acceptor.submit, and requestor.deliver primitives, respectively (see ISO/IEC 10731).

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.

3.8.3.3 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.

NOTE The IEC 61158-5 series of standards define sets of abstract services. They are neither protocol specifications nor implementation specifications nor concrete programming interface specifications. Therefore there are restrictions on the extent to which service procedures can be mandated in the parts of IEC 61158-5. Protocol aspects that can vary among different protocol specifications or different implementations that instantiate the same abstract services are unsuitable for inclusion in these service definitions, except at the level of abstraction that is necessarily common to all such expressions.

For example, the means by which service providers pair request and reply PDUs is appropriate for specification in an IEC 61158-6 protocol specification standard but not in an IEC 61158-5 abstract service definition standard. Similarly, local implementation methods by which a service provider or service user pairs request and confirm(ation) primitives, or indication and response primitives, is appropriate for an implementation specification or for a programming interface specification, but not for an abstract service standard or for a protocol standard, except at a level of abstraction that is necessarily common to all embodiments of the specifying standard. In all cases, the abstract definition is not permitted to over-specify the more concrete instantiating realization.

Further information on the conceptual service procedures of an implementation of a protocol that realizes the services of one of the IEC 61158-5 abstract service definitions can be found in IEC/TR 61158-1, 9.6.

4 Concepts

The common concepts and templates used to describe the application layer service in this fieldbus are detailed in IEC/TR 61158-1, Clause 9.

5 Data type ASE

5.1 Overview

An overview of the data type ASE and the relationships between data types is provided in IEC/TR 61158-1, 10.1

5.2 Formal definition of data type objects

The template used to describe the data type class is detailed in IEC/TR 61158-1, 10.2. This includes the specific ASE structure and the definition of its attributes.

5.3 FAL defined data types

5.3.1 Fixed length types

Primitive types that are selected and their characteristics are mostly specified in the ISO 8824 standard.

The Used Data types and the discrepancies between the 8824 and this standard are listed hereafter.

5.3.1.1 BOOLEAN

Conform to the 8824 Standard.

5.3.1.2 FLOATING POINT

CLASS:		Data type
ATTRIBUTES:		
1	Data type Numeric Identifier	= Not used
2	Data type Name	= FloatingPoint
3	Format	= FIXED LENGTH
5.1	Octet Length	= Boolean

This primitive type defines value representation for positive and negative real numbers, including, zero, negative infinite and positive infinite.

The values at the limits (0, +∞ -∞), as well as the various representation formats are defined in IEC 60559.

For this type, the attribut Octet length, of boolean type, indicates whether the representation format is simple precision "4 octets" (FALSE) or double precision "8 octets" (TRUE).

5.3.1.3 BCD

Conform to the ISO/IEC 8824.

5.3.1.4 INTEGER

CLASS:		Data type
ATTRIBUTES:		

- 1 Data type Numeric Identifier = Not used
- 2 Data type Name = Integer
- 3 Format = FIXED LENGTH
- 5.1 Octet Length = n

The definition of this primitive type is the same as the integer type definition in ISO 8824 standard. For this type, the Octet Length attribute defines the number of bits that are necessary to have a twos complement representation of all possible values.

Ex : Octet Length = 1 if $-128 \leq n \leq 127$; 2 if $-32768 \leq n \leq 32767$;

5.3.1.5 UNSIGNED

CLASS: Data type

ATTRIBUTES:

- 1 Data type Numeric Identifier = Not used
- 2 Data type Name = Unsigned
- 3 Format = FIXED LENGTH
- 5.1 Octet Length = n

The definition of this primitive type is the same as the integer type definition in ISO 8824 with the exclusion of negative numbers. For this type, the Octet Length attribute defines the number of bits that are necessary to have a twos complement representation of all possible values.

Ex : 1 : for unsigned8 ; 2 : for unsigned16 ; 4 : for unsigned32 ; 8 for unsigned64

5.3.1.6 GENERALISED TIME

Conform to the ISO/IEC 8824.

5.3.1.7 BINARY TIME

CLASS: Data type

ATTRIBUTES:

- 1 Data type Numeric Identifier = Not used
- 2 Data type Name = BinaryTime
- 3 Format = FIXED LENGTH
- 5.1 Octet Length = n

This primitive type is defined as the unsigned type of the present standard, the value of which corresponds to a number of elementary times.

For this type, each value of the *Octet Length* attribute defines the value of an elementary time as well as a number of bits used to represent any possible binary time value, as show in the Table 1.

Table 1 – Binary time coding

Size	10 ms	0,1 ms	1 ms	10 ms
16 Bits	0	1	2	3
32 Bits	4	5	6	7
48 Bits	8	9	—	—

5.3.2 String types

5.3.2.1 BIT STRING

Conform to the ISO/IEC 8824.

5.3.2.2 OCTET STRING

Conform to the ISO/IEC 8824.

5.3.2.3 VISIBLE STRING

Conform to the ISO/IEC 8824.

5.3.3 Array types

There are defined as user defined data types for the specific application. User data types are supported by this standard as instances of the data types classes.

User defined types are specified in the same manner that all FAL objects are specified. They are defined by providing values for the attributes specified for their class.

5.3.4 Structures types

There are defined as user defined data types for the specific application. User data types are supported by this standard as instances of the data types classes.

User defined types are specified in the same manner that all FAL objects are specified. They are defined by providing values for the attributes specified for their class.

6 Communication model specification

6.1 Concepts

6.1.1 AL environment

This general model presents the relationship between the AL environment and the application layer services which is the purpose of this standard.

The AL environment is both process control and discrete parts manufacturing. In those systems, manufacturing devices are controlled by a system that performs the automation functions.

6.1.2 Application services

Application services can be divided into two sets. See Figure 1.

The standard specifies the MPS (Manufacturing Periodic/aperiodic Services) application service set. The other service set corresponds to a subset (subMMS) of that specified in the MMS standard.

MPS services are particularly well suited to the real time requirements of a distributed application in the fact that they support:

- The periodic broadcast update of a distributed database in the various control system devices.

- The elaboration of temporal qualifiers associated with the data base.
- The elaboration of consistency qualifiers associated with sets of data within this database.

SubMMS and MCS are essentially considered to fulfil the requirements for configuration and operating mode management within the distributed application.

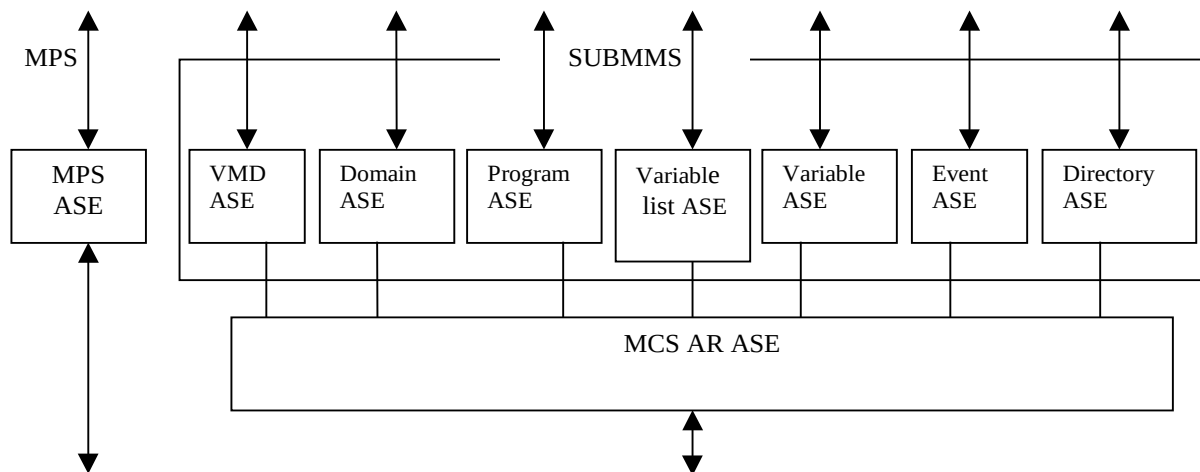


Figure 1 – Organisation of the ASEs and ARs

A distributed application is characterized by various information processing tasks on various devices of a communication system for the requirements of check a production system control application.

Cooperation between the tasks belonging to different devices is modelled in the form of interactions between application processes (APs).

A distributed application is composed of two or more APs, each of the devices of the communication system which can have zero, one or several of them.

As provided for by this standard, the communication between two APs passes via the network.

NOTE The breaking down of a distributed application in several AP's at the level of a device is decided by the user according to his own criteria (Methodology, Functional structuring of the application, Setting into service facilities, etc.).

6.1.3 Application process

6.1.3.1 Application process overview

An AP is modelled by one or several information processing tasks in a device for the requirements of a particular function of the distributed application.

The aspects of an AP, taken into account by the AL, only concern its communication capacities which are modelled by Application Entities (AEs). Each AP can have at the most one AE of a given type.

On the other hand, the cooperation between APs in the framework of a distributed application are performed via the setting up of the relations between AP invocations (APIs) which represent its activities. An AP can, at a given instant, zero, one or several APIs. Besides, each API has one particular utilization of the AE (AEI) which satisfies the communication requirements.

6.1.3.2 AP model

6.1.3.2.1 AP model description

The purpose of the AP model is to show all the aspects of an AP which are taken into account by the AL communication.

NOTE The other aspects concerning the type of information, processing and execution of the latter are not modeled, since they are not involved at the communication level.

Class: APPLICATION PROCESS (AP)

Key Attribute: AP title

Attribute: AP type

Attribute: List of Inherit from Application Entity

Attribute: List of Reference API

6.1.3.2.2 Attributes:

AP title:

A title is attributed to an AP to identify it, among all the other APs, in a non-ambiguous manner in a multisegment AL network. For the requirements of the universal identification of an AP in the AL environment, this AP tile is involved as a constituent of the ISO name proper to AL user site.

This identification can be structured from the name of the device to which it belongs.

NOTE To structure this identification, this AP should remain permanently in the equipment.

AP type:

The type of an AP helps to designate a set of APs to which it belongs.

NOTE This type is not covered by a title which can be handled in the AL environment.

List of Inherit from Application Entity

The Application Process class inherits an Application Entity class list which gives it the capacity to communicate in the AL environment.

Thus an AP has one or several application entities, which are necessarily of different types.

List of Reference API:

Designates the lists of APIs which model the AP activities. This list can vary dynamically but at a given instant zero, can designate one or several APIs.

6.1.3.3 API model

6.1.3.3.1 API model description

An API model describes the activities of an AP.

Class: APPLICATION PROCESS INVOCATION (API)

Key Attribute: API identification

Attribute: Reference Application Process

Attribute: List of Reference AEI

6.1.3.3.2 Attributes:

API identification:

Identifies an Application Process Invocation in a non-ambiguous manner in the given AP field. It is underlined that the scope of this identification is global for the AP and independent of the rules for naming the AP title.

Reference Application process

The purpose of this attribute is to show that this API belongs to a given AP.

The lifetime of an API is necessarily less than or equal to that of the AP to which it belongs.

List of Reference AEI:

This list designates AE uses which are reserved for the API requirements.

An API can use zero, one or several AEIs for each AE of the AP to which it belongs.

The number of referenced AEIs can vary dynamically in time. Consequently, an API, at a given moment, can have one or several AEIs (Complete definition of an AEI in the following paragraph).

6.1.4 Application entity

An AE represents a set of communication capacities for the requirements of an AP. These communication capacities are defined by a set of Application Service Elements (ASEs). The particular uses of the communication capacities of an AE are represented by AE invocations (AEIs).

An AEI consequently models the communication functions and their state for the particular activities of an API. Communications between APIs take place via AEIs which, as required, may or may not be related by Application Associations (AAs) or Application Relationship (AR) in the AL context.

An AEI in the AL environment gives an API the exclusive possibility of communicating with or without association. An AEI can only participate in one and in only one association application.

The communications without association allow an AEI to exchange information with one or several other AEIs.

An association application allows an AEI to communicate with one or several other AEIs in the framework of a common Context Application (CA).

This context can be negotiated or predefined by configuration for each AEI, for a point to point association which makes use of a pair of AEIs. The context is necessarily prenegotiated for each AEI.

The state of an AEI is thus directly linked to the fact that the latter participates or not in an association and to the state of the association if it participates in a negotiated association.

Generally it can be said that the lifetime of an AEI is equivalent to that of the service exchanged for the communications without association and equivalent to that of the association for communications with associations. However, for a negotiated association, the lifetime of the pair of AEIs is lengthened by their opening and closing transients.

The lifetime of an AEI is controlled by the API which it represents in the AL environment. An API, on the other hand, can have a much longer lifetime than that of its AEIs, equal to that of the AP to which it belongs.

Particular AR are predefined by configuration, these AR are only used by the MPS ASE to exchange plain data (variable). All the parameters relative to the variable are pre-established at the configuration, these parameters never change. Only a new configuration can cancel or modify these particular AR. The identification of this peculiar AR is made by a number unique on the network, this number refer to the whole set of parameters of the AR and the DLL resources implied for the communication. This number similar to a DLCEP Id is called Identifier.

6.1.5 AE model

6.1.5.1 AE model description

Description of the communication capacities offered by an AE.

Class: APPLICATION ENTITY (AE)

Attribute: AE type

Key Attribute: Qualifier

Attribute: List of Inherit from ASE

Attribute: List of Reference AEI

6.1.5.2 Attributes:

Qualifier:

An AE qualifier identifies an AE non-ambiguously in the field of an application process. The combination of this AE qualifier with the AP title forms an AE title which identifies this AE non-ambiguously in the AL environment.

This AE title allows recovery from the directory of the addressing information, the link addresses.

An AE title can have "Synonyms of the AE title" to be distinguished from one or several other AEs by different titles.

An AE can be globally identified with other AEs with the help of an "AE group designation".

AE type:

The AE type designates a set of AEs sharing the same communication characteristics. Various AEs of the same type inherit the same list of application service elements.

List of Inherit from ASE

The Application Entity (AE) class inherits an ASE class list which gives it the specificity of its communication functions. An AE can support one or several ASEs, the latter being necessarily of different types.

List of Reference AEI

Designates the list of AEIs which model the AE uses. This list, which can vary dynamically, can designate at a given moment, zero, one or several AEIs.

6.1.6 AEI model

6.1.6.1 AEI model description

An AEI model is used to describe a particular use of the AE.

Class: APPLICATION ENTITY INVOCATION (AEI)*Key attribute:* AEI identification*Attribute:* Reference AE*Attribute:* Reference API*Attribute:* Local AE address*Attribute:* Remote AE address*Attribute:* Communication mode

[ASSOCIATED; NON-ASSOCIATED]

Constraint: Communication mode = ASSOCIATED*Attribute:* Association state

[OPENING;

ESTABLISHED;

CLOSED]

Attribute: Association type

[NEGOTIATED; PRENEGOTIATED]

Attribute: Association type

[NEGOTIATED; PRENEGOTIATED]

Attribute: Local AEI access point*Attribute:* Remote AEI access point*Attribute:* Inherit from Application context**6.1.6.2 Attributes:****AEI identification:**

Allows non-ambiguous identification of an application entity invocation in the sub-field of an AE reserved for an API of a given AP.

The Non-ambiguous designation of an AEI in the environment requires indicating: AP title / API identification / AE qualifier /AEI identification.

Reference AE:

This reference is used to designate the AE object to which this AEI belongs,

The lifetime of an AEI is necessarily less than or equal to that of the AE to which it belongs.

Reference API:

This reference helps to designate the API object which is represented in the AL environment by this AEI. An AEI can represent one and only one API in the AL environment.

Local AE address:

Corresponds to the link address which allows locating the AE to which this AEI belongs.

This address is completed with the help of directory functions during the creation of this AEI and remains static for its lifetime.

Remote AE address:

Corresponds, either to a link address which allows locating the AE with which this AEI is corresponding, for point to point exchanges, or with a link address which helps to locate an AE group with which this AEI is corresponding for multipoint exchanges.

This address is filled at the AEI level during the opening of an association with one other AEI or when exchanging data in a non-associated mode.

Communication mode [ASSOCIATED; NON-ASSOCIATED]

(The communication mode in the ASSOCIATED state indicated that this AEI allows an API to communicate in a application context which was previously defined or negotiated. The type of exchanges between AEIs, possible in such a communication mode, is point to point or multipoint.

When the communication mode is NON-ASSOCIATED, this AEI allows an API to communicate outside all pre-established context. The nature of the possible exchanges in a non-associated communication mode is point to point or multipoint.

Association state

[OPENING;
ESTABLISHED;
CLOSING];

In the associated mode, the possibilities of communication of an AEI with another AEI depends on the state of the association. The various states correspond to the phases in the life of an association application:

OPENING:

Corresponds to an AEI which, at the request of the API which it represents, is negotiating an application context with another AEI, for the requirements of an association. In this state, no exchanges can take place outside those reserved for negotiation of the context.

ESTABLISHED:

Corresponds to an AEI which offers the API which it represents, the possibility of communicating with another API via one of its determined AEIs, within the framework of an application context. This application context has been obtained either by negotiation between the AEI pair or by local predefinition at the AEIs.

CLOSING:

Corresponds to an AEI which, at the request of the API which it represents, or with which it had been associated, is being freed from its application context.

Association type

[NEGOTIATED;
PRENEGOTIATED]

This type indicates if the association has been negotiated or not. This attribute, if it takes the PRENEGOTIATED value, imposes on the association to remain exclusively in the ESTABLISHED state.

Local AEI access point:

Each AEI, in the ASSOCIATED mode, can be directly localized by its AEI access point. This addressing information is filled with the help of the directory functions during the creation of an AEI instance and remains static for its full lifetime.

AEI remote access point:

Each AEI, which is in the ASSOCIATED mode, has this addressing information which localizes the AEI with which it is in correspondence for point to point exchanges. This information is filled in for an AEI at the opening of an association.

This information locates the AEIs with which it is in correspondence for multipoint exchanges.

Inherit from Context Application

This inheritance allows designating the object of the application context which conditions communications with this AEI.

The attributes of this application context are valued during installation of an AEI on the basis of the values proposed by the user in the negotiated associated mode and the non associated mode, whereas they are filled by configuration in the prenegotiated associated mode.

6.1.7 Application context

6.1.7.1 Application context overview

A pair of AEIs require common knowledge of the rules which govern their communications. This set of rules is called Context Application. The application context which can be supported by an AEI is necessarily derived from the possibilities offered by the various ASEs possessed by the AE to which it belongs.

6.1.7.2 Application service elements

6.1.7.2.1 ASE description

The possibilities offered by the various ASEs of an AE are defined by the specification (in ASN1) of one or more set of Application Protocol Data Units (APDU) which form abstract syntaxes. Besides, the use of an abstract syntax to dialogue between two AEs requires the implementation of a transfer syntax which defines the rules for encoding of the APDUs on the bus.

NOTE Among the abstract syntaxes associated with the ASE MMS, the general abstract syntax ISO-9506-MMS-1 can be noted as well as other abstract syntaxes such as those meant for digital controls ISO-9506-NCCS-1 and process control ISO-9506-PROCESS-1.

Class: ELEMENT SERVICE APPLICATION

Attribute: ASE type

Attribute: List of Abstract syntax

Attribute: List of Transfer syntax

6.1.7.2.2 Attributes:

ASE type:

Several ASE types can be met with in the AL environment. This standard is covered by the SubMMSE definition derived from the ASE MMS defined by the ISO.

The MCSE (Messaging Common Service Element) service corresponds to the control ASE which should necessarily be used to project another ASE application on the link layer messaging. This service element has unique encoding rules which are universally known in the AL messaging environment.

NOTE The MCS encoding rules are not given in ASN1. They are directly given in the representation used for the transfer.

List of Abstract Syntax:

An application service element can have one or several abstract syntaxes. Each of them correspond to a set of APDUs described in an ASN1 module universally identified in the AL environment.

NOTE For example, an abstract syntax defined in ASN1 has the following appearance:

```
<module name> <module universal identifier>
Definitions ::= BEGIN{
IMPORTS <..>, <..>, ....FROM <external module>
EXPORTS <..>, <..>, ...
```

```
PDU module ::= CHOICE {
  {<production name> [N°] <production>} +
}
```

END

This standard keeps AL SUBMMS-1, derived from the reference core, as one of its abstract syntaxes for the MMS service element.

List of Transfer syntax:

A transfer syntax defines the abstract syntax encoding rules used. An ASE could support several transfer syntaxes within the framework of an AE, which could be used according to the requirements depending on the possibilities of their correspondents.

One of the transfer syntaxes taken into account by this AL environment standard for the implementation of the AL abstract syntax, AL SUBMMS-1, is proposed by the ISO basic encoding rules (BER).

Like the abstract syntaxes, the transfer syntaxes are universally identified in the AL environment.

6.1.7.3 Type of application context

6.1.7.3.1 Application context description

The application context belonging to an AEI conditions the possible communications with it. In the information contained in the application context, some is specific to the AE architecture, i.e. with the type of ASE resting on MCS as well as with the abstract transfer syntax selected. Other information is specific to the choice performed on the API level and corresponding to the restrictions to the negotiated abstract syntax and to the quantity of communication resources used.

Class: APPLICATION CONTEXT

Attribute: Context name

Key Attribute: Context identification

Attribute: Reference ASE

Attribute: Abstract syntax

Attribute: Transfer syntax

Constraint: Com. mode = ASSOCIATED

Attribute: Application reductions

Attribute: Service quality

6.1.7.3.2 Attributes:

Identification of the context:

Allows identifying an application context in the field of an AE. It thus has an exclusive local role.

Context name:

Used to identify, in the AL environment, the bases of a context consisting of the selected ASE, an abstract syntax and a transfer syntax. This name is used during the association negotiation or during a transfer of data in the non-associated mode.

Reference ASE:

One of the AE application service elements which is selected by this application context.

Abstract syntax:

An abstract syntax among those of the ASE within the framework of the AE, which is selected by this context.

Transfer syntax:

The transfer syntax selected for encoding the ASE PDUs, implemented within the framework of this AE, which is selected by this context.

Application reductions:

The abstract syntax selected at the level of an AEI application context for a given service element can be used in a reduced manner. This possibility is used only for transmission in the associated mode.

Service quality:

The selected service quality.

6.1.7.4 Directory**6.1.7.4.1 Directory overview**

In order to communicate mutually, the AEIs make use of directory functions which given them the addressing information which they require. These directory functions are necessary exclusively during the opening of association or during a transmission in the mode without association.

6.1.7.4.2 Addressing model

The link layer offers data transfer services in the non-connected mode and an addressing space allowing location of the application entities. This addressing space allows locating the entities individually and/or by groups, and on the other hand to locate them with the help of physical or logic addresses. These two types of addresses respectively allow taking into account or not of the network topology in the location of application entities.

The application layer has a double location requirement, that of the AEs during negotiation of the associations or transmission of data without association, and that of the AEIs during transmission of data within the framework of an association.

Consequently, the AE addresses and the AEI access points which need to be located are addressed with the link addresses from the single link addressing space.

In the case of point to point exchanges, the AE addresses and the AEI access points are designated with the help of individual link addresses. In the case of multipoint exchanges, the AE address and the AEI access point locating the transmitter are designated by individual link addresses whereas the AE address and the AEI access point locating the receivers are designated by group link addresses.

The rules allocating the liaison addresses used for the AE addresses and the AEI access points are not defined in this standard.

The abbreviations used to speak about these two uses of link addresses are: ADAE for those which are associated with AE addresses and ADAEI for those which are associated with AEI access points.

NOTE The ADAEIs are allocated by the directory functions of a device to the AEIs, in the sub-set which was attributed to an AE.

The address values contained in this sub-set can be, for example selected by dividing up the addressing space according to the network architecture or well chosen according to other criteria better satisfying the communication requirements of the user application.

Addressing rules:

Rule_1: Each AE is identified by one and only one "AE title" which is in a bi-univocal relation with an ADAE, allowing it to be located in the AL environment.

Rule_2: Each AE can be designated by one or several synonyms, each being in relation with one and only one address.

Rule_3: Each AE can be designated in the framework of one or several "AE group designations", each being in relation with one and only one address.

NOTE It is important to note that biunivocity is not a property of rules 2 and 3.

Rule_3: Each AEI which is identified by an AEI identification in the field of an AE, an API, of a given AP, is in biunivocal relation with an ADAEI, allowing it to be located in the AL environment.

6.1.7.4.3 Directory functions

The projection between the names allowing identification of the various objects (AEs and AEIs) in the application, and the link addresses (respectively ADAE and ADAEI) used to locate them in the AL environment, is managed by the directory service.

For any communication initiative at the level of an AEI, the latter calls the directory functions which give it the addresses necessary for the exchanges.

— Initiator Directory Function (IDF):

This function is implemented for an AEI initiating an exchange concerning an association opening or a transmission of data in the non-associated mode.

IDF is used to:

- obtain an ADAE pair allowing location of the local and remote AEs according to their AE titles,
- allocate an ADAEI for the local AEI access point and optionally of another for the remote AEI access point. This allocation only takes place when setting up an association.

<Input parameters> ::=

<Local AE title>
 <Remote AE title
 | Synonym of the AE title
 | Designation of the AE group>
 <Transmission mode>

with <Transmission mode>::=<ASSOCIATED | NON-ASSOCIATED>;

<Output parameter> ::=

<Local ADAE>

<Remote ADAE>

[<Local ADAEI>] (*)

[<Remove ADAE>] (**)

(*) This ADAEI is only returned in the case of an ASSOCIATED" transmission mode.

(**) This ADAEI is returned or not according to the remote AE title.

Responder Directory Function (RDF):

This function is implemented at the level of a called AE, for the opening of an association.

RDF:

Used to allocate an ADAE at the level of the called entity during the response phase for an association opening request.

<Input parameters> ::=

<Local ADAE>

<Remote ADAE>

<Output parameters> ::=

<Local ADAEI>

NOTE This function is not used on the level of a responder if the initiator of the communication had already performed the allocation by use of the function.

6.1.8 Coordination of the AEs

6.1.8.1 AEI communication description

The initiative and the type of communications belong to the APIs. The four types of communication are:

- Information exchanges in the non-associated mode.
- Association opening requests.
- Information exchanges in the associated mode.
- Association closing requests

NOTE For each of these types of communication, the AL environment objects are implemented in a specific manner which is covered by the following paragraphs.

6.1.8.2 Information exchanges in the non-associated mode

When an API wishes to initialize a data transfer in the non-associated mode, it proceeds in the following manner:

Exchange scenario:

- It addresses, as applicable:

- either the source AE identified by its <AE title>, indicating the addressee (either by its <AE title>, or by an <AE synonym title>, or by an <AE group designation> and the <Application context>,
 - or an AEI whose mode is NON-ASSOCIATED, if the latter has been created to allow a series of exchanges.
- The local procedure which takes place at the AE after this request by the API is as follows:

If the API addresses the AE, the following local procedure is executed:

 - Creation of an AEI object in the NON-ASSOCIATED mode,
 - Entering of the attributes of this object:
 - <Application context> takes the value indicated by the request,
 - <Remote AE address> takes the value returned by IDF for the non-associated communication mode,
 - <Local AE address>: takes the value returned by ID for the non-associated communication mode.
 - Then whatever the addressing which took place, a data transfer service in the MCSE non-associated mode takes place within the framework of this AEI. The user data encapsulated in this data transfer is that of the MCS user application service which was requested.
 - The AEI thus used could be maintained or revoked after this exchange. In case of revocation, this will be effective immediately if the service is not confirmed or after reception of the response and passage to API of the data contained in it, if the service is confirmed.
 - The following procedure is executed at the addressee AE:
 - Depending on whether an AEI does not yet exist or exists already, the following operations are executed or not:
 - Creation of an AEI object in the NON-ASSOCIATED mode.
 - Filling of the attributes of this object:
 - <Application context>: takes the value indicated in the received PDU, if it is supported by the AE.
 - <Remote AE address> takes the value contained in the received PDU.
 - <Local AE address> takes the value contained in the PDU.
 - Identification of an API object designated implicitly or explicitly in the PDU.
 - Passage of the <Application service> contained in the PDU received at the API identified for the execution.
 - The AEI thus used can be revoked or preserved at the end of this exchange. In case of revocation, this will be effective immediately if the service is not confirmed or after transmission of the response if it is confirmed.

6.1.8.3 Association opening request

If the API wishes to initialize an association opening request, it proceeds in the following manner:

Opening scenario:

- The API locally addresses the calling AE identified by its <AE title>, indicating:
 - the called <AE title>,
 - the <opening service> (Initiate Rq in MMS) which contains the <Application reductions> of the association, which will be negotiated,
 - the <Application context> description other than the application reductions which will be negotiated at the MCS level.
- The local procedure which will be executed at the AE level after this API request, is the following:
 - Creation of an AEI object in the ASSOCIATED mode in the OPENING state.
 - Entering of the attributes of this object:
 - <Remote AE address; local AEI access point>: take the value returned by IDF.
 - Consideration of the application context requested by the user, and possible reduction of this, if its local AE cannot offer it.
 - A_Associate.Rq service call by MCSE in the framework of this AEI with the application context given by the user as well as the opening request user PDU.
 - On reception of the response, the other AEI attributes can be filled:
 - <Remote AEI access point>: takes the value contained in the PDU.
 - <Application context> takes the value contained in the response which can correspond to a reduction of that requested.
 - Passage of the AEI state to ESTABLISHED,
- The procedure which takes place at the responder AE level on reception of an opening request transmitted by the source procedure is the following:
 - Creation of an AEI object in the ASSOCIATED mode in the OPENING state.
 - Filling of the attributes of this object:
 - <Application Context>: takes the value indicated in the PDU, or a reduction, if the AE does not support it.
 - <Local AE address> takes the value received in the PDU.
 - <Local AEI access point> can take the optional value contained in the PDU or that returned by the RDF (Responder Directory Function).
 - <Remote AE address; Remote access point> take the values received in the PDU.
 - Addressing of the API from its implicit or explicit identification in the PDU.
 - Passing to the user of the opening request PDU.
 - When the API gives its response, a call of the A_ASSOCIATE.Rp service of MCSE in the framework of this AEI, with the PDU user, as a response to the opening request and possible updating of the context if this has been reduced by the API.
 - Passage of the AEI state to ESTABLISHED.

6.1.8.4 Exchange of information in the associated mode

When an API wishes to initialize an associated mode exchange, it proceeds as follows.

Exchange scenario:

- It addresses the AEI locally in the ESTABLISHED state, identified by its <AEI identifier> relative to its area by indicating:
 - the <application service> which should be executed with the called entity.
- The local procedure which should be performed at the AEI level is as follows:
 - During reception of the response, for a confirmed service, sending of this to the user.
 - Call of the data transfer service in the MCSE associated mode in the AEI framework, with as user data the PDU associated with the service.
- The procedure executed at the responder AEI on reception of the service request is as follows:
 - Passing of the <Application service> contained in the API PDU.
 - Service call and information transfer in the MCSE associated mode, in the framework of the AEI, with as user data the response PDU associated with this service.

6.1.8.5 Request for termination of an association

When an API wishes to initialize a request for termination of an association, it proceeds as follows:

Closing scenario:

- It locally addresses the AEI identified by its <AEI identification> in its area, indicating:
 - the <Closing application service>.
- The local procedure which is executed at the AEI initiating this closing is as follows:
 - Setting of the AEI to the CLOSING state (Blocking of the data transfers beyond what has been engaged).
 - MCSE A-RELEASE service call to transmit the closing request service in the framework of this AEI.
 - On reception of the positive response, revocation of the AEI object, return to the ESTABLISHED state in the other cases;
- The remote procedure which is performed at the responder AEI level is as follows:
 - Setting of the AEI to the CLOSING state.
 - (Blocking of the data transfers outside those engaged).
 - Passing of the closing request service to the user.
 - At a positive response from the user, revocation of the AEI object and sending of the response to the initiator using the MCSE.

6.1.9 Concepts for sub_MMS

6.1.9.1 General

Sub-MMS communications can be established through negotiated or predefined association as well as without association.

The assignment of access rights to association as well as the protections of objects, allow and the protection of objects authorizes selective access to the objects.

6.1.9.2 Sub-MMS communication channels

- 1- A Sub-MMS communication can be achieved without association:

These communication can take place in point to point (confirmed/unconfirmed services), in multipoint or in distribution (unconfirmed services).

In this case a service request (other than Initiate, Conclude and Abort) should be refused if the service is not supported, if the resources are insufficient, if it is not in compliance with the protocol, etc.

The Initiate, Conclude and Abort services have no significance.

This type of communication does not support the access protection mechanism on the objects.

2- A Sub-MMS communication can be performed in a predefined association:

These communication can take place in point to point (confirmed/unconfirmed services), in multipoint or in distribution (unconfirmed services).

In this case a service request (other than Initiate, Conclude and Abort) should be refused if the service is not supported, if it is not in compliance with the protocol or if the resources reserved for the association are insufficient, etc.

The Initiate, Conclude and Abort services have no significance.

This type of communication does/does not support the access protection mechanism on the objects.

3- A Sub-MMS communication can be performed in a negotiated association:

These communication can take place in point to point (services confirmed/unconfirmed),

In this case a service request (other than Initiate, Conclude and Abort) should be refused if the service is not supported, if it is not in compliance with the protocol, or if the resources reserved for the association are insufficient, etc.

This type of communication does/does not support the access protection mechanism on the objects.

6.1.9.3 Access protection mechanism

6.1.9.3.1 Protection level introduction

The access to Sub-MMS object can be refused to some clients. For this we define the protection levels assigned to an object and access right assigned to the association.

6.1.9.3.2 Protection level assigned to an object

The definition of objects other than the VMD object includes an OPTIONAL attribute called "ACCESS PROTECTION" which determines if an object is protected or not.

An object which is not protected has its "ACCESS PROTECTION" attribute forced to the value "FALSE".

An object which is protected has its "ACCESS PROTECTION" attribute forced to the value "TRUE".

- 1 – Password,
- 2 – Access Groups,
- 3 – Access Rights.

Password:

The "Password" attribute allows restricting access only to clients holding the password.

Access Groups:

"Access Groups" attribute allows to restrict the access only to clients who belong to one of the groups defined by the attribute.

Access Rights:

The "Access Rights" attribute indicates the various possibilities to obtain access to the object.

It also indicates for each of these access possibilities, the services which can be executed as well as the conditions to be fulfilled, as shown in Table 2.

Table 2 – Access protection

Object class	Executable operations
Domain	Load, Upload, Delete, Connect to a program invocation
Program invocation	Execute, Initialize, Stop, Delete
Variable	Read, Write
Variable-List	Read, Write, Delete
Event	Read, Alter, Acknowledge
Access granted because of password	Conditions to be fulfilled
Yes	a) The association password is identical to that of the object.
No	b) The value of the password of the association is ZERO; condition (a) is not fulfilled
Access granted because of group membership	Conditions to be fulfilled
Yes	a) The group word for the association and the object have at least one identical group No
No	b) The value of the association group word is ZERO; condition (c) is not fulfilled
Access granted to all	Condition to be fulfilled
Yes	e) The "Access Rights" attribute authorizes access to all clients.

6.1.9.3.3 Level of the access rights assigned to an association

It is possible to assign access rights to predefined or negotiated associations. In this case these access rights are characterized by the parameters "Password" and "Access Groups".

The "Password" and "Access Group" parameters are granted implicitly to the predefined association.

The "Password" and "Access Group" parameters are granted to the negotiated association by the initiate service.

The "Password" and "Access Group" parameters thus characterize the Client's access rights.

6.1.9.3.4 Access control mechanisms for protected objects

Access to the objects can be granted:

- either when the "Password" attribute of the object is identical to the association "Password" parameter
- when the "Access Groups" attribute of the object and the "Access Groups" parameter of the association have at least one group No. in common.
- or to all the clients when the "Access Rights" attribute permits.

NOTE No access restriction is imposed for the use of the following services:
- Get Name List,

- Get Domain Attributes,
- Get Program Invocation Attributes,
- Get Variable Access Attributes,
- Get Variable List Attributes,
- Get Event Condition Attributes,
- Get OD Header/Data type Attributes.

6.1.9.4 Directory management

6.1.9.4.1 General

All the sub-MMS object describers supported by an application form a directory.

The arrangement of the describers in a directory can be either standardized or free.

The standardization of the directory is performed by an object called "OD" (Object Dictionary).

6.1.9.4.2 Use of the "OD" object

It is not mandatory for a device to support the "OD" object.

6.2 ASEs

6.2.1 MPS ASE (periodic/aPeriodic manufacturing services)

6.2.1.1 Overview

In a MPS environment, a certain number of application objects are implemented. These objects come directly from the instantiation of the different classes previously described.

All these application objects representing the network configuration should be described partially or globally in order to allow:

- the user to have access to the application configuration (e.g.: Access to the variable type),
- the checking of the configuration consistency of the application between several subscribers (e.g.: configuration consistency between the producer and the consumers of a variable).

Moreover, this description is specified in this standard in order to ensure the interoperability in the configuration interchanges between different implementations.

6.2.1.2 Object concepts

6.2.1.2.1 Application objects

The description of periodic/apperiodic MPS services leads to describe all the resources involved in the application layer. This specification defines the static part of the specification.

The specification reactive aspects describe the various service elements actions on the application layer resources:

MPS environment application objects are divided into two sets:

- objects related to variables,
- objects related to variable lists.

Variables are abstract elements that associate a transfer syntax and a semantics to a data.

Variables, and related information, are provided to the user by the communication system, either individually, or by lists.

Other variables are provided with an event characteristic to which the user may associate actions relative to the execution of the distributed application.

6.2.1.2.2 Application objects addressing

The key-attributes (A_Name) used to address the MPS application layer objects all belong to the same addressing space.

Each of these names in the addressing space is defined within a maximum length of 16 characters.

The characters belong to the set defined for the "visible string" type of the abstract syntax notation ASN1 (ISO 8824).

Within that set, the following characters are to be used:

Upper case letters (A..Z)

Lower case letters (a..z)

Digits (0..9)

Underscore (_)

Currency symbol (\$)

The "space" character is not to be used.

Upper and lower case letters are considered as different characters.

An A_Name should not begin with a digit.

The "underscore" character as a second character defines a standard-reserved name.

6.2.1.3 Variable access sub-ASE

6.2.1.3.1 Concepts

6.2.1.3.1.1 MPS environment variables

An MPS environment variable is an abstraction in the MPS environment, in which it is referenced by a unique name or description. The data associated with this variable corresponds to the representation handled by the data link services, which is exchanged on the network.

6.2.1.3.1.2 Network memory

Control system devices provide an AP with a value of each variable concerned an image of which is stored in local network memory.

6.2.1.3.1.3 Producer/consumer/third party concepts

Application entities that recognise a variable are provided with a local image of it.

Within a distributed application, one unique application entity is declared as the producer of the variable value whereas one or several application entities are declared as consumers of that value.

Moreover, some application entities, while being neither producer, nor consumer of the variable value, may have a knowledge of it in order to invoke the updating of its value. Such application entities are called third party entities.

6.2.1.3.1.4 Periodicity and aperiodicity concepts

The exchange of variable value between the producing entity and the consuming entities can be performed periodically or aperiodically.

A periodic updating is initiated by the network and is defined during the configuration step.

The updating period is determined by

- the user requirements to fulfil the distributed application,
- constraints related to the network.

Aperiodic updating is initiated by a user, such as the producing entity, a consuming entity, or a third party entity.

6.2.1.3.1.5 Synchronous and asynchronous properties

Application entities handling variables elaborated according to production validity, for the produced variables, and transmission validity for the consumed variables.

Definitions:

Production or transmission validity associated with each variable have an asynchronous character as they are elaborated independently from the network activity.

Production or transmission validity associated with each variable have a synchronous character when they involve synchronization orders issued from the network.

A synchronization order corresponds to the event associated with the emission or the reception of a variable of class SYNCHRONIZATION.

6.2.1.3.2 Variable class specification

6.2.1.3.2.1 Variable class modelisation

A variable in the MPS environment is modelled using the following objects (see Figure 2):

- one *Produced Variable Local Image* object that includes all the attributes necessary to a producer user.
- one or several *Consumed Variable Local Image* object that includes all the attributes necessary to one or several Consumer users.

Those two object types are issued from inheritance from several classes:

The *Produced Variable Local Image* objects are instantiated from the *Produced Variable* class.

The *Produced Variable Class* includes all the attributes proper to a producing entity, and inherits from the *Identified Variable* class. That latter class includes all the attributes that are common to every local image associated with that variable.

Moreover, the *Produced Variable* class optionally inherits from two classes *Refreshment* and *Punctual Refreshment* that include all the attributes necessary to the elaboration of informations relative to the production validity associated with that variable, within the producing application entity.

The Consumed Variable Local Image objects are instantiated from the Consumed Variable class.

The *Consumed Variable* class includes all the variable attributes proper to a consuming entity, and inherits from the *Identified Variable* class. That latter class includes all the attributes common to every local image associated with that variable.

Moreover, the *Consumed Variable* class optionally inherits from two classes *Promptness* and *Punctual Promptness* that includes the attributes necessary to the elaboration of the informations relative to the transmission validity associated with the variable within the consuming application entities.

Each of the classes *Produced Variable* and *Consumed Variable* optionally inherits respectively from a class *Production Resynchronization* and Consumption Resynchronization that include the attributes relative to the double memorisation of a variable in order to allow the providing of the variable:

- to the network by a producing entity,
- to the consuming entities by the network, upon a synchronization order issued from the network.

All the common attributes of a variable are described in a *Variable* generic class.

The variable type is described in a *Type Constructor* class which is referred by the variable and that allow the elaboration of all the type that may exist in the MPS environment.

Concerning the access to a variable, the *Variable Access* class refers to a variable and indicates the access mode that is used.