

# INTERNATIONAL STANDARD

## NORME INTERNATIONALE

**Industrial communication networks – Fieldbus specifications –  
Part 5-9: Application layer service definition – Type 9 elements**

**Réseaux de communication industriels – Spécifications des bus de terrain –  
Partie 5-9: Définition des services de la couche application – Éléments de type 9**



## THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2014 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](mailto:info@iec.ch)  
[www.iec.ch](http://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](http://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](http://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](http://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](http://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](http://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](http://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](mailto: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](http://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](http://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](http://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](http://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](http://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](http://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](mailto:csc@iec.ch).

# INTERNATIONAL STANDARD

## NORME INTERNATIONALE

**Industrial communication networks – Fieldbus specifications –  
Part 5-9: Application layer service definition – Type 9 elements**

**Réseaux de communication industriels – Spécifications des bus de terrain –  
Partie 5-9: Définition des services de la couche application – Éléments de type 9**

INTERNATIONAL  
ELECTROTECHNICAL  
COMMISSION

COMMISSION  
ELECTROTECHNIQUE  
INTERNATIONALE

PRICE CODE  
CODE PRIX

**XF**

ICS 25.040.40; 35.100.70; 35.110

ISBN 978-2-8322-1735-1

**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.....                                                                           | 5   |
| INTRODUCTION.....                                                                       | 7   |
| 1 Scope.....                                                                            | 8   |
| 1.1 General.....                                                                        | 8   |
| 1.2 Specifications.....                                                                 | 9   |
| 1.3 Conformance.....                                                                    | 9   |
| 2 Normative references .....                                                            | 9   |
| 3 Terms, definitions, symbols, abbreviations and conventions .....                      | 10  |
| 3.1 ISO/IEC 7498-1 terms .....                                                          | 10  |
| 3.2 ISO/IEC 8822 terms .....                                                            | 10  |
| 3.3 ISO/IEC 9545 terms .....                                                            | 10  |
| 3.4 ISO/IEC 8824-1 terms .....                                                          | 10  |
| 3.5 IEC 61158-1 terms .....                                                             | 10  |
| 3.6 Type 9 fieldbus application-layer specific definitions.....                         | 14  |
| 3.7 Abbreviations and symbols.....                                                      | 14  |
| 3.8 Conventions .....                                                                   | 15  |
| 4 Concepts.....                                                                         | 18  |
| 5 Data type ASE.....                                                                    | 18  |
| 5.1 Overview.....                                                                       | 18  |
| 5.2 Formal definition of data type objects .....                                        | 21  |
| 5.3 FAL defined data types.....                                                         | 22  |
| 5.4 Data type ASE service specification .....                                           | 25  |
| 5.5 Summary of data types.....                                                          | 25  |
| 6 Communication model specification .....                                               | 26  |
| 6.1 Concepts.....                                                                       | 26  |
| 6.2 Common parameters .....                                                             | 26  |
| 6.3 ASEs.....                                                                           | 26  |
| 6.4 ARs.....                                                                            | 115 |
| 6.5 Summary of classes .....                                                            | 118 |
| 6.6 Permitted services by AREP role.....                                                | 119 |
| Bibliography.....                                                                       | 121 |
| Figure 1 – Data type class hierarchy.....                                               | 19  |
| Figure 2 – VFD model.....                                                               | 27  |
| Figure 3 – Abstract model of an automation system (VFD).....                            | 27  |
| Figure 4 – Source OD/remote OD .....                                                    | 33  |
| Figure 5 – Put OD state machine .....                                                   | 46  |
| Figure 6 – Transaction object state machine .....                                       | 52  |
| Figure 7 – Context test of two features-supported with different bitstring length ..... | 60  |
| Figure 8 – Overview of event .....                                                      | 80  |
| Figure 9 – Event state machine.....                                                     | 85  |
| Figure 10 – Domain genericdownload/download state machine (server).....                 | 100 |
| Figure 11 – Domain upload state machine (server) .....                                  | 102 |
| Figure 12 – State diagram.....                                                          | 113 |

|                                                                           |    |
|---------------------------------------------------------------------------|----|
| Table 1 – Data type summary .....                                         | 26 |
| Table 2 – Logical status .....                                            | 28 |
| Table 3 – Status .....                                                    | 29 |
| Table 4 – Unsolicited status .....                                        | 30 |
| Table 5 – Identify .....                                                  | 31 |
| Table 6 – Structure of the object dictionary .....                        | 34 |
| Table 7 – Structure of the static list of types .....                     | 34 |
| Table 8 – Structure of the static object dictionary .....                 | 34 |
| Table 9 – Structure of the dynamic list of variable lists .....           | 35 |
| Table 10 – Structure of the dynamic list of program invocations .....     | 35 |
| Table 11 – Empty object dictionary .....                                  | 39 |
| Table 12 – Get OD service parameters .....                                | 42 |
| Table 13 – Initiate put OD service parameters .....                       | 44 |
| Table 14 – Put OD service parameters .....                                | 45 |
| Table 15 – Terminate put OD service parameters .....                      | 46 |
| Table 16 – Put OD state transitions .....                                 | 48 |
| Table 17 – Attribute FMS features supported .....                         | 50 |
| Table 18 – Transaction object state transitions .....                     | 53 |
| Table 19 – Initiate service parameters .....                              | 54 |
| Table 20 – Failure reasons .....                                          | 55 |
| Table 21 – Abort service parameters .....                                 | 56 |
| Table 22 – User abort reasons .....                                       | 57 |
| Table 23 – APO ASE abort reasons .....                                    | 57 |
| Table 24 – Reject service parameters .....                                | 58 |
| Table 25 – Reject APDU reasons .....                                      | 58 |
| Table 26 – Compatibility of the local context to the remote context ..... | 59 |
| Table 27 – Unconfirmed send service parameters .....                      | 62 |
| Table 28 – Confirmed send service parameters .....                        | 63 |
| Table 29 – AR-Abort service parameters .....                              | 64 |
| Table 30 – Compel service parameters .....                                | 64 |
| Table 31 – Get buffered message service parameters .....                  | 65 |
| Table 32 – AR-Status service parameters .....                             | 66 |
| Table 33 – Simple variable access group membership .....                  | 68 |
| Table 34 – Simple variable access rights membership .....                 | 68 |
| Table 35 – Array variable access group membership .....                   | 70 |
| Table 36 – Array variable access rights membership .....                  | 70 |
| Table 37 – Variable list access group membership .....                    | 72 |
| Table 38 – Variable list access rights membership .....                   | 72 |
| Table 39 – Read service parameters .....                                  | 75 |
| Table 40 – Write service parameters .....                                 | 76 |
| Table 41 – Information report service parameters .....                    | 77 |
| Table 42 – Define variable list service parameters .....                  | 78 |

|                                                                         |     |
|-------------------------------------------------------------------------|-----|
| Table 43 – Delete variable list service parameters .....                | 79  |
| Table 44 – Event access group membership .....                          | 81  |
| Table 45 – Event access rights membership .....                         | 81  |
| Table 46 – Event notification service parameters .....                  | 82  |
| Table 47 – Acknowledge event notification service parameters .....      | 83  |
| Table 48 – Alter event condition monitoring service parameters .....    | 84  |
| Table 49 – Event state transitions .....                                | 85  |
| Table 50 – Domain access group membership .....                         | 87  |
| Table 51 – Domain access rights membership .....                        | 87  |
| Table 52 – GenericInitiateDownloadSequence .....                        | 89  |
| Table 53 – GenericDownloadSegment .....                                 | 90  |
| Table 54 – GenericTerminateDownloadSequence .....                       | 91  |
| Table 55 – InitiateDownloadSequence .....                               | 92  |
| Table 56 – DownloadSegment .....                                        | 93  |
| Table 57 – TerminateDownloadSequence .....                              | 94  |
| Table 58 – RequestDomainDownload .....                                  | 95  |
| Table 59 – InitiateUploadSequence .....                                 | 96  |
| Table 60 – UploadSegment .....                                          | 97  |
| Table 61 – TerminateUploadSequence .....                                | 98  |
| Table 62 – RequestDomainUpload .....                                    | 99  |
| Table 63 – Domain genericDownload/download state machine (server) ..... | 100 |
| Table 64 – Domain upload state machine (server) .....                   | 102 |
| Table 65 – Program invocation access group membership .....             | 104 |
| Table 66 – Program invocation access group membership .....             | 104 |
| Table 67 – Create program invocation service parameters .....           | 106 |
| Table 68 – Delete program invocation service parameters .....           | 107 |
| Table 69 – Start service parameters .....                               | 108 |
| Table 70 – Stop service parameters .....                                | 109 |
| Table 71 – Resume service parameters .....                              | 110 |
| Table 72 – Reset service parameters .....                               | 111 |
| Table 73 – Kill service parameters .....                                | 112 |
| Table 74 – Program invocation state machine .....                       | 114 |
| Table 75 – Class summary .....                                          | 119 |
| Table 76 – Services by AREP role .....                                  | 119 |

## INTERNATIONAL ELECTROTECHNICAL COMMISSION

**INDUSTRIAL COMMUNICATION NETWORKS –  
FIELDBUS SPECIFICATIONS –****Part 5-9: Application layer service definition –  
Type 9 elements**

## FOREWORD

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

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

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

International Standard IEC 61158-5-9 has been prepared by subcommittee 65C: Industrial networks, of IEC technical committee 65: Industrial-process measurement, control and automation.

This second edition cancels and replaces the first edition published in 2007. This edition constitutes a technical revision. The main change with respect to the previous edition is listed below:

- Correct download state machine event service

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

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

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

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

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

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

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

## INTRODUCTION

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

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

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

## **INDUSTRIAL COMMUNICATION NETWORKS – FIELDBUS SPECIFICATIONS –**

### **Part 5-9: Application layer service definition – Type 9 elements**

#### **1 Scope**

##### **1.1 General**

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

This standard provides common elements for basic time-critical and non-time-critical messaging communications between application programs in an automation environment and material specific to Type 9 fieldbus. The term “time-critical” is used to represent the presence of a time-window, within which one or more specified actions are required to be completed with some defined level of certainty. Failure to complete specified actions within the time window risks failure of the applications requesting the actions, with attendant risk to equipment, plant and possibly human life.

This standard defines in an abstract way the externally visible service provided by the different Types of the 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 7498-1) and the OSI Application Layer Structure (ISO/IEC 9545).

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

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

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

## 1.2 Specifications

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

A secondary objective is to provide migration paths from previously-existing industrial communications protocols. It is this latter objective which gives rise to the diversity of services standardized as the various types of IEC 61158.

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 9 application layer services as defined in this standard.

## 2 Normative references

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

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

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

ISO/IEC 646, *Information technology – ISO 7-bit coded character set for information interchange*

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

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

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

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

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

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

### **3 Terms, definitions, symbols, abbreviations and conventions**

For the purposes of this document, the following terms, definitions, symbols, abbreviations and conventions as defined in these publications apply.

#### **3.1 ISO/IEC 7498-1 terms**

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

#### **3.2 ISO/IEC 8822 terms**

- 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 8824-1 terms**

- a) object identifier
- b) type

#### **3.5 IEC 61158-1 terms**

For the purposes of the present document, the following IEC 61158-1 terms apply.

**3.5.1****application**

function or data structure for which data is consumed or produced

**3.5.2****application layer interoperability**

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

**3.5.3****application object**

object class that manages and provides the run time exchange of messages across the network and within the network device

Note 1 to entry: Multiple types of application object classes may be defined.

**3.5.4****application process**

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

**3.5.5****application process identifier**

component that distinguishes multiple application processes used in a device

**3.5.6****application process object**

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

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

**3.5.7****application process object class**

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

**3.5.8****application relationship**

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

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

**3.5.9****application relationship application service element**

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

**3.5.10****application relationship endpoint**

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

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

### **3.5.11**

#### **attribute**

description of an externally visible characteristic or feature of an object

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

### **3.5.12**

#### **behaviour**

indication of how the object responds to particular events

Note 1 to entry: Its description includes the relationship between attribute values and services.

### **3.5.13**

#### **class**

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

Note 1 to entry: A class is a generalisation of the 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.

### **3.5.14**

#### **class attributes**

attribute that is shared by all objects within the same class

### **3.5.15**

#### **class code**

unique identifier assigned to each object class

### **3.5.16**

#### **class specific service**

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

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

### **3.5.17**

#### **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, such as the role of an AR endpoint in which it issues confirmed service request APDUs to a single AR endpoint acting as a server

### **3.5.18**

#### **conveyance path**

unidirectional flow of APDUs across an application relationship

### **3.5.19**

#### **cyclic**

events which repeat in a regular and repetitive manner

### **3.5.20**

#### **dedicated AR**

AR used directly by the FAL User

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

### **3.5.21**

#### **device**

physical hardware connection to the link

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

**3.5.22****device profile**

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

**3.5.23****dynamic AR**

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

**3.5.24****endpoint**

one of the communicating entities involved in a connection

**3.5.25****error**

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

**3.5.26****error class**

general grouping for error definitions

Note 1 to entry: Error codes for specific errors are defined within an error class.

**3.5.27****error code**

identification of a specific type of error within an error class

**3.5.28****FAL subnet**

networks composed of one or more data link segments

Note 1 to entry: They are permitted to contain bridges, but not routers. FAL subnets are identified by a subset of the network address.

**3.5.29****logical device**

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

**3.5.30****management information**

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

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

**3.5.31****network**

series of nodes connected by some type of communication medium

Note 1 to entry: The connection paths between any pair of nodes can include repeaters, routers and gateways.

**3.5.32****peer**

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

### 3.5.33

#### **pre-defined AR endpoint**

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

Note 1 to entry: Pre-defined ARs that are not pre-established are established before being used.

### 3.5.34

#### **pre-established AR endpoint**

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

### 3.5.35

#### **publisher**

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

Note 1 to entry: The publisher may not be aware of the identity or the number of subscribers and it may publish its APDUs using a dedicated AR. Two types of publishers are defined by this standard, Pull Publishers and Push Publishers, each of which is defined separately.

### 3.5.36

#### **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

### 3.5.37

#### **service**

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

Note 1 to entry: A set of common services is defined and provisions for the definition of object-specific services are provided. Object-specific services are those which are defined by a particular object class to perform a required function which is not performed by a common service.

### 3.5.38

#### **subscriber**

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

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

## 3.6 Type 9 fieldbus application-layer specific definitions

There are no additional terms defined.

## 3.7 Abbreviations and symbols

|      |                                     |
|------|-------------------------------------|
| AE   | Application Entity                  |
| AL   | Application Layer                   |
| ALME | Application Layer Management Entity |
| ALP  | Application Layer Protocol          |
| APO  | Application Object                  |
| AP   | Application Process                 |
| APDU | Application Protocol Data Unit      |
| API  | Application Process Identifier      |
| AR   | Application Relationship            |

|       |                                                    |
|-------|----------------------------------------------------|
| AREP  | Application Relationship End Point                 |
| ASCII | American Standard Code for Information Interchange |
| ASE   | Application Service Element                        |
| Cnf   | Confirmation                                       |
| 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                               |
| FAL   | Fieldbus Application Layer                         |
| ID    | Identifier                                         |
| IEC   | International Electrotechnical Commission          |
| Ind   | Indication                                         |
| LME   | Layer Management Entity                            |
| OSI   | Open Systems Interconnect                          |
| QoS   | Quality of Service                                 |
| Req   | Request                                            |
| Rsp   | Response                                           |
| SAP   | Service Access Point                               |
| SDU   | Service Data Unit                                  |
| SMIB  | System Management Information Base                 |
| SMK   | System Management Kernel                           |
| VFD   | Virtual Field Device                               |

### 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 Dictionary ASE services specified in 6.3.2 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:

|                 |                                       |
|-----------------|---------------------------------------|
| <b>FAL ASE:</b> | <b>ASE Name</b>                       |
| 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 2 048 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 a 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 subseries 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 subseries. 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 subseries protocol specification standard but not in an IEC 61158-5 subseries 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 subseries abstract service definitions can be found in IEC 61158-1, 9.6.

## 4 Concepts

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

## 5 Data type ASE

### 5.1 Overview

Fieldbus data types specify the machine independent syntax for application data conveyed by FAL services. The Fieldbus application layer supports the definition and transfer of both basic and constructed data types. Encoding rules for the data types specified in Clause 5 are provided in IEC 61158-6-9.

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

Data types are defined as instances of the data type class, as shown in Figure 1.

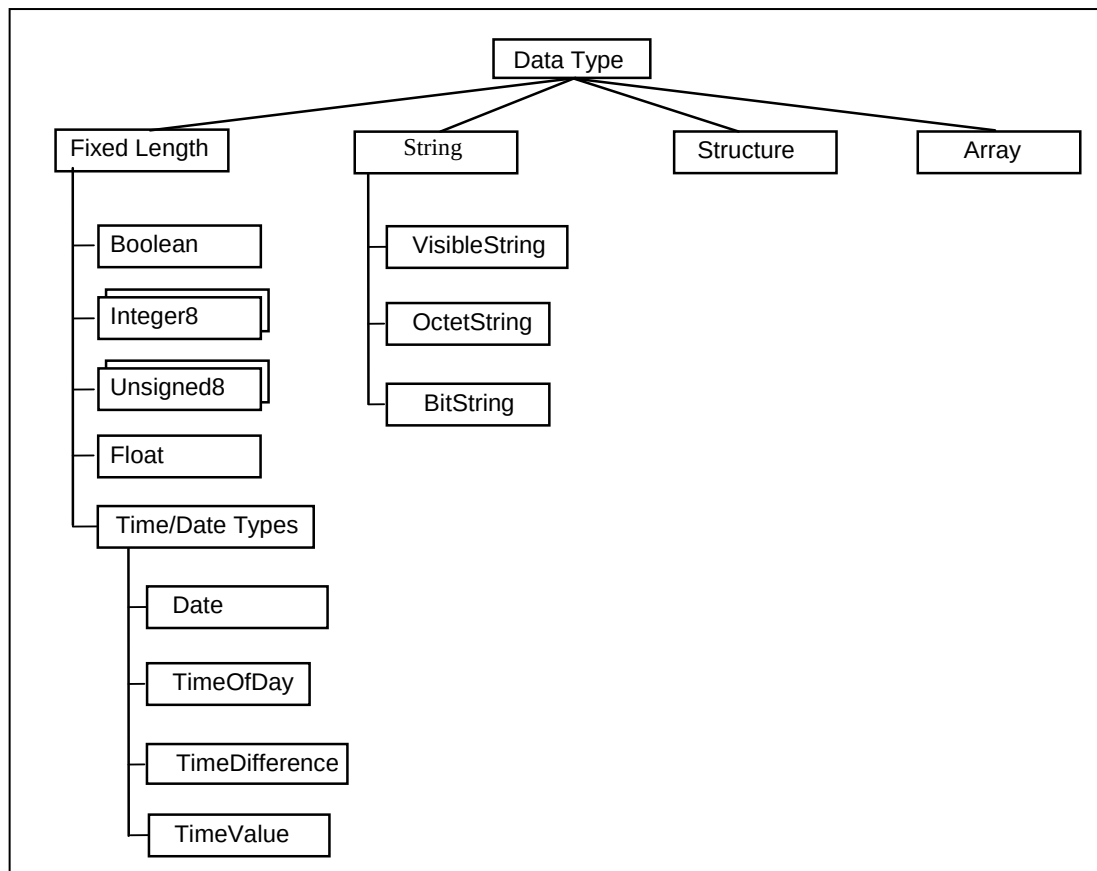


**Figure 1 – Data type class hierarchy**

The data type definitions are represented as a class/format/instance structure beginning with data type class entitled "Data type". The formats for data types are defined by the data type class.

The basic data classes are used to define fixed length and bitstring data types. Standard types taken from ISO/IEC 8824-1 are referred to as *simple* data types. Other standard basic data types are defined specifically for Fieldbus applications and are referred to as *specific types*.

The constructed types specified in this standard are strings, arrays and structures. There are no standard types defined for arrays and structures.

### 5.1.1 Overview of basic types

Most basic types are defined from a set of ISO/IEC 8824 types (simple types). Some ISO/IEC 8824 types have been extended for Fieldbus specific use (specific types).

Simple types are ISO/IEC 8824 universal types. They are defined in this standard to provide them with Fieldbus class identifiers.

Specific types are basic types defined specifically for use in the Fieldbus environment. They are defined as simple class subtypes.

Basic types have a constant length. Two variations are defined, one for defining data types whose length is an integral number of octets, and one for defining data types whose length is bits.

NOTE Boolean, Integer, OctetString, VisibleString, and UniversalTime are defined in this standard for the purpose of assigning Fieldbus class identifiers to them. This standard does not change their definitions as specified in ISO/IEC 8824.

#### **5.1.1.1 Fixed length types**

The length of Fixed length types is an integral number of octets.

### **5.1.2 Overview of constructed types**

Constructed data types are needed to completely convey the variety of information present on the Fieldbus. There are two kinds of constructed types defined for this standard, arrays and structures.

#### **5.1.2.1 Strings**

A string is composed of an ordered set, variable in number, of homogeneously typed fixed-length elements.

#### **5.1.2.2 Arrays**

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

#### **5.1.2.3 Structures**

A structure is made of an ordered set of heterogeneously-typed elements called fields. Like arrays, this standard does not restrict the data type of fields. However, the fields within a structure do not have to be of the same type.

#### **5.1.2.4 Nesting level**

This standard permits arrays and structures to contain arrays and structures. It places a restriction on the number of nesting levels allowed to one.

### **5.1.3 Specification of user defined data types**

Users may find it necessary to define custom data types for their own applications. User defined types are supported by this standard as instances of data type 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.

#### **5.1.4 Transfer of user data**

User data is transferred between applications by the FAL protocol. All encoding and decoding are performed by the FAL user.

The rules for encoding user data in FAL protocol data units is data type dependent. These rules are defined in IEC 61158-6-9. User-defined data types for which there are no encoding rules are transferred as a variable-length sequence of octets. The format of the data within the octet string is defined by the user.

## 5.2 Formal definition of data type objects

### 5.2.1 Data type class

The data type class specifies the root of the data type class tree. Its parent class "top" indicates the top of the FAL class tree.

| FAL ASE:             |                    | DATA TYPE ASE                                        |
|----------------------|--------------------|------------------------------------------------------|
| <b>CLASS:</b>        | <b>DATA TYPE</b>   |                                                      |
| <b>CLASS ID:</b>     |                    | 5 (FIXED LENGTH & STRING), 6 (STRUCTURE), 12 (ARRAY) |
| <b>PARENT CLASS:</b> |                    | TOP                                                  |
| <b>ATTRIBUTES:</b>   |                    |                                                      |
| 1                    | (m) Key Attribute: | Data type Numeric Identifier                         |
| 2                    | (o) Key Attribute: | Data type Name                                       |
| 3                    | (m) Attribute:     | Format (FIXED LENGTH, STRING, STRUCTURE, ARRAY)      |
| 4                    | (c) Constraint:    | Format = FIXED LENGTH   STRING                       |
| 4.1                  | (m) Attribute:     | Octet Length                                         |
| 5                    | (c) Constraint:    | Format = STRUCTURE                                   |
| 5.1                  | (m) Attribute:     | Number of Fields                                     |
| 5.2                  | (m) Attribute:     | List of Fields                                       |
| 5.2.1                | (o) Attribute:     | Field Name                                           |
| 5.2.2                | (m) Attribute:     | Field Data type                                      |
| 6                    | (c) Constraint:    | Format = ARRAY                                       |
| 6.1                  | (m) Attribute:     | Number of Array Elements                             |
| 6.2                  | (m) Attribute :    | Array Element Data type                              |

#### 5.2.1.1 Attributes

##### Format

This attribute identifies the data type as a fixed-length, string, array, or data structure.

##### Octet Length

This conditional attribute defines the representation of the dimensions of the associated type object. It is present when the value of the format attribute is "FIXED LENGTH" or "STRING". For FIXED LENGTH data types, it represents the length in octets. For STRING data types, it represents the length in octets for a single element of a string.

##### Number of Fields

This conditional attribute defines the number of fields in a structure. It is present when the value of the format attribute is "STRUCTURE".

##### List of Fields

This conditional attribute is an ordered list of fields contained in the structure. Each field is specified by its number and its type. Fields are numbered sequentially from 0 (zero) in the order in which they occur. Partial access to fields within a structure is supported by identifying the field by number. This attribute is present when the value of the format attribute is "STRUCTURE".

##### Field Name

This conditional, optional attribute specifies the name of the field. It may be present when the value of the format attribute is "STRUCTURE".

##### Field Data type

This conditional attribute specifies the data type of the field. It is present when the value of the format attribute is "STRUCTURE". This attribute may itself specify a constructed data type either by referencing a constructed data type definition by its numeric id, or by

embedding a constructed data type definition here. When embedding a description, the Embedded Data type description shown below is used.

### Number of Array Elements

This conditional attribute defines the number of elements for the array type. Array elements are indexed starting at "0" through "n-1" where the size of the array is "n" elements. This attribute is present when the value of the format attribute is "ARRAY".

### Array Element Data type

This conditional attribute specifies the data type for the elements of an array. All elements of the array have the same data type. It is present when the value of the format attribute is "ARRAY". This attribute may itself specify a constructed data type either by referencing a constructed data type definition by its numeric id, or by embedding a constructed data type definition here. When embedding a description, the Embedded Data type description shown below is used.

### Embedded Data type Description

This attribute is used to recursively define embedded data types within a structure or array. The template below defines its contents. The attributes shown in the template are defined above in the data type class, except for the Embedded Data type attribute, which is a recursive reference to this attribute. It is used to define nested elements.

|       |     |             |                                                |
|-------|-----|-------------|------------------------------------------------|
| 1     | (m) | Attribute:  | Format(FIXED LENGTH, STRING, STRUCTURE, ARRAY) |
| 2     | (c) | Constraint: | Format = FIXED LENGTH   STRING                 |
| 2.1   | (m) | Attribute:  | Data type Numeric ID value                     |
| 2.2   | (m) | Attribute:  | Octet Length                                   |
| 3     | (c) | Constraint: | Format = STRUCTURE                             |
| 3.1   | (m) | Attribute:  | Number of Fields                               |
| 3.2   | (m) | Attribute:  | List of Fields                                 |
| 3.2.1 | (m) | Attribute:  | Embedded Data type Description                 |
| 4     | (c) | Constraint: | Format = ARRAY                                 |
| 4.1   | (m) | Attribute:  | Number of Array Elements                       |
| 4.2   | (m) | Attribute:  | Embedded Data type Description                 |

## 5.3 FAL defined data types

### 5.3.1 Fixed length types

#### 5.3.1.1 Boolean

| CLASS:             | Data type                        |
|--------------------|----------------------------------|
| <b>ATTRIBUTES:</b> |                                  |
| 1                  | Data type Numeric Identifier = 1 |
| 2                  | Data type Name = Boolean         |
| 3                  | Format = FIXED LENGTH            |
| 4.1                | Octet Length = 1                 |

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

#### 5.3.1.2 Integer8

| CLASS:             | Data type                        |
|--------------------|----------------------------------|
| <b>ATTRIBUTES:</b> |                                  |
| 1                  | Data type Numeric Identifier = 2 |
| 2                  | Data type Name = Integer8        |
| 3                  | Format = FIXED LENGTH            |
| 4.1                | Octet Length = 1                 |

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

### 5.3.1.3 Integer16

**CLASS:** Data type

**ATTRIBUTES:**

- |     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data type Numeric Identifier | = | 3            |
| 2   | Data type Name               | = | Integer16    |
| 3   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 2            |

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

### 5.3.1.4 Integer32

**CLASS:** Data type

**ATTRIBUTES:**

- |     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data type Numeric Identifier | = | 4            |
| 2   | Data type Name               | = | Integer32    |
| 3   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 4            |

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

### 5.3.1.5 Unsigned8

**CLASS:** Data type

**ATTRIBUTES:**

- |     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data type Numeric Identifier | = | 5            |
| 2   | Data type Name               | = | Unsigned8    |
| 3   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 1            |

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

### 5.3.1.6 Unsigned16

**CLASS:** Data type

**ATTRIBUTES:**

- |     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data type Numeric Identifier | = | 6            |
| 2   | Data type Name               | = | Unsigned16   |
| 3   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 2            |

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

### 5.3.1.7 Unsigned32

**CLASS:** Data type

**ATTRIBUTES:**

- |     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data type Numeric Identifier | = | 7            |
| 2   | Data type Name               | = | Unsigned32   |
| 3   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 4            |

This type is a binary number. The most significant bit of the most significant octet is always used as the most significant bit of the binary number; no sign bit is included. This unsigned type has a length of four octets.

### 5.3.1.8 Float

**CLASS:** Data type

**ATTRIBUTES:**

|     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data type Numeric Identifier | = | 8            |
| 2   | Data type Name               | = | Float        |
| 4   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 4            |

This type has a length of four octets. The format for float32 is that defined by ISO/IEC/IEEE 60559 as single precision.

### 5.3.1.9 Date

**CLASS:** Data type

**ATTRIBUTES:**

|     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data type Numeric Identifier | = | 11           |
| 2   | Data type Name               | = | Date         |
| 4   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 7            |

This data type is consists of years, months, days or month, days of week, hours, standard/daylight savings time, minutes, and milliseconds.

### 5.3.1.10 TimeOfDay

**CLASS:** Data type

**ATTRIBUTES:**

|     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data type Numeric Identifier | = | 12           |
| 2   | Data type Name               | = | TimeOfDay    |
| 4   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 6            |

This data type is composed of two elements of unsigned values and expresses the time of day and the date. The first element is an Unsigned32 data type and gives the time after the midnight in milliseconds. The second element is an Unsigned16 data type and gives the date.

### 5.3.1.11 TimeDifference

**CLASS:** Data type

**ATTRIBUTES:**

|     |                              |   |                |
|-----|------------------------------|---|----------------|
| 1   | Data type Numeric Identifier | = | 13             |
| 2   | Data type Name               | = | TimeDifference |
| 3   | Format                       | = | FIXED LENGTH   |
| 4.1 | Octet Length                 | = | 4 or 6         |

This data type is composed of two elements of unsigned values that express the difference in time. The first element is an Unsigned32 data type that provides the fractional portion of one day in milliseconds. The optional second element is an Unsigned16 data type that provides the difference in days.

**CLASS:** Data type

**ATTRIBUTES:**

|     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data type Numeric Identifier | = | 21           |
| 2   | Data type Name               | = | Time Value   |
| 3   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 8            |

This simple type expresses the time or time difference in a two's complement binary number with a length of eight octets. The unit of time is 1/32 millisecond.

### 5.3.1.12 TimeValue

|                                |                  |
|--------------------------------|------------------|
| <b>CLASS:</b>                  | <b>Data type</b> |
| <b>ATTRIBUTES:</b>             |                  |
| 1 Data type Numeric Identifier | = 21             |
| 2 Data type Name               | = Time Value     |
| 3 Format                       | = FIXED LENGTH   |
| 4.1 Octet Length               | = 8              |

This simple type expresses the time or time difference in a two's complement binary number with a length of eight octets. The unit of time is 1/32 millisecond.

## 5.3.2 String types

### 5.3.2.1 VisibleString

|                                |                  |
|--------------------------------|------------------|
| <b>CLASS:</b>                  | <b>Data type</b> |
| <b>ATTRIBUTES:</b>             |                  |
| 1 Data type Numeric Identifier | = 9              |
| 2 Data type Name               | = VisibleString  |
| 3 Format                       | = STRING         |
| 4.1 Octet Length               | = 1 to n         |

This type is defined as the ISO/IEC 646 string type.

### 5.3.2.2 OctetString

|                                |                  |
|--------------------------------|------------------|
| <b>CLASS:</b>                  | <b>Data type</b> |
| <b>ATTRIBUTES:</b>             |                  |
| 1 Data type Numeric Identifier | = 10             |
| 2 Data type Name               | = OctetString    |
| 3 Format                       | = STRING         |
| 4.1 Octet Length               | = 1 to n         |

An OctetString is an ordered sequence of octets, numbered from 1 to n. For the purposes of discussion, octet 1 of the sequence is referred to as the first octet. IEC 61158-6-9 defines the order of transmission.

### 5.3.2.3 BitString

|                                |                  |
|--------------------------------|------------------|
| <b>CLASS:</b>                  | <b>Data type</b> |
| <b>ATTRIBUTES:</b>             |                  |
| 1 Data type Numeric Identifier | = 14             |
| 2 Data type Name               | = Bitstring      |
| 3 Format                       | = STRING         |
| 5.1 Octet Length               | = 1 to n         |

This string type is defined as a series of eight bits, numbered from 0 to 7.

## 5.4 Data type ASE service specification

There are no operational services defined for the type object.

## 5.5 Summary of data types

This clause contains a summary of the defined data types. The Class ID values have been assigned to be compatible with existing standards. Table 1 is sorted by class id.

**Table 1 – Data type summary**

| Class Id | Data type  | Class Id | Data type     |
|----------|------------|----------|---------------|
| 1        | Boolean    | 8        | Float         |
| 2        | Integer8   | 9        | VisibleString |
| 3        | Integer16  | 10       | OctetString   |
| 4        | Integer32  | 11       | Date          |
| 5        | Unsigned8  | 12       | TimeOfDay     |
| 6        | Unsigned16 | 14       | BitString     |
| 7        | Unsigned32 | 21       | TimeValue     |

## 6 Communication model specification

### 6.1 Concepts

The concepts are those of the basic communications model described in IEC 61158-1.

### 6.2 Common parameters

Many services have the following parameters. Instead of defining them with each service, the following common definitions are provided.

#### AREP ID

This parameter specifies sufficient information to locally identify the AR used to convey the service.

#### VCR

This parameter specifies sufficient information to locally identify the VCR and AR used to convey the service.

#### Error Type

This parameter specifies information about the reason for the unsuccessful execution of the service.

### 6.3 ASEs

#### 6.3.1 Virtual field device ASE

##### 6.3.1.1 Virtual field device class specification

##### 6.3.1.1.1 Class overview

The Virtual Field Device (VFD) is an abstract model for the description of the data and the behavior of an Application Process. VFDs contain APOs. The attributes of an APO are described by object descriptions. The collection of a VFDs object descriptions is defined as the Object Dictionary (OD). For each VFD there is exactly one Object Dictionary. Services are defined for accessing a VFD's OD and its APOs, as shown in Figure 2.

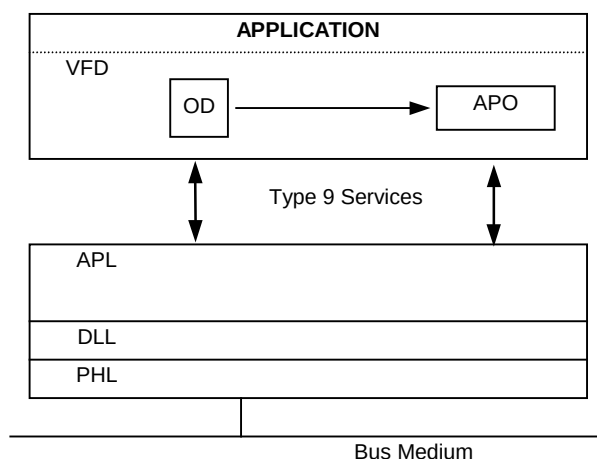


**Figure 2 – VFD model**

The services define no concrete interface for an implementation. They describe in an abstract form which functions may be used.

The application is not a subject of this standard. It should only be indicated, how the abstractly described services may be made available to the application.

Access to VFD objects is defined by Virtual Communication Relationships (VCR). Several VFD Objects may be present in a device, as shown in Figure 3.

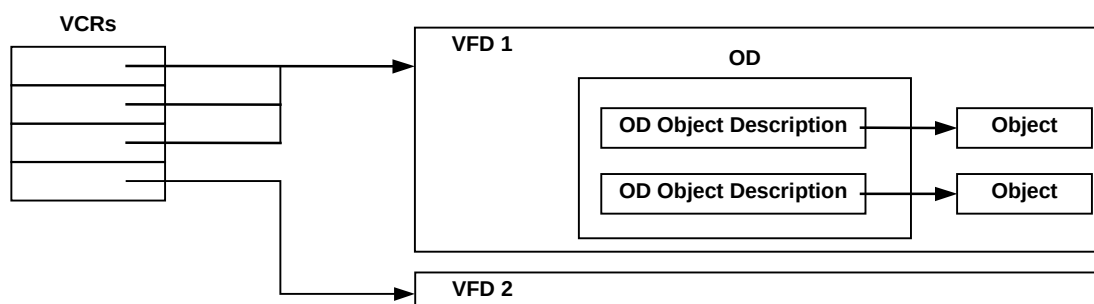


**Figure 3 – Abstract model of an automation system (VFD)**

#### 6.3.1.1.2 Formal model

The VFD class specifies the attributes and services defined for application processes. Its parent class "top" indicates the top of the class tree.

|                       |                              |
|-----------------------|------------------------------|
| <b>ASE:</b>           | <b>VFD ASE</b>               |
| <b>CLASS:</b>         | <b>VFD</b>                   |
| <b>CLASS ID:</b>      | —                            |
| <b>PARENT CLASS:</b>  | <b>TOP</b>                   |
| 1. (m) Key Attribute: | implicit through the VCR     |
| 2. (m) Attribute:     | VendorName                   |
| 3. (m) Attribute:     | ModelName                    |
| 4. (m) Attribute:     | Revision                     |
| 5. (m) Attribute:     | ProfileNumber                |
| 6. (m) Attribute:     | LogicalStatus                |
| 7. (m) Attribute:     | PhysicalStatus               |
| 8. (m) Attribute:     | List of VFD Specific Objects |

#### SERVICES:

1. (m) Ops Service: Status
2. (m) Ops Service: Unsolicited Status
3. (m) Ops Service: Identify

#### 6.3.1.1.3 Attributes

##### Implicit through the VCR

The VCR which is entered in the VCR List points to the assigned VFD Object.

##### VendorName

This attribute identifies the vendor of the device.

##### ModelName

This attribute specifies the model name of the device. The value of this attribute is defined by the vendor.

##### Revision

This attribute describes the revision level of the device. The value of this attribute is defined by the vendor.

##### ProfileNumber

This attribute identifies the profile of the VFD. The profile value is defined by an external administrative authority. If the device does not correspond to a profile, then this attribute has a null value.

##### Logical Status

This attribute specifies information about the state of the communication capabilities of the device. Its values are shown in Table 2.

**Table 2 – Logical status**

| Logical status             | Meaning                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |          |           |       |       |        |               |        |       |          |                |          |  |
|----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------|-----------|-------|-------|--------|---------------|--------|-------|----------|----------------|----------|--|
| Ready for communication    | All services may be used normally.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |          |           |       |       |        |               |        |       |          |                |          |  |
| Limited number of services | Limited number of services may be supported for some cases.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |          |           |       |       |        |               |        |       |          |                |          |  |
| OD-LOADING-NON-INTERACTING | If the Object Dictionary is in the state OD-LOADING-NON-INTERACTING, it is not allowed to execute the service InitiatePutOD.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |          |           |       |       |        |               |        |       |          |                |          |  |
| OD-LOADING-INTERACTING     | <p>If the Object Dictionary is in the state OD-LOADING-INTERACTING, then all connections, except for the connection over which the InitiatePutOD Service was received, are locked and the establishment of a further connection should be rejected. On this connection only the following services are allowed:</p> <table border="0"> <tr> <td>Initiate</td><td>PhysWrite</td></tr> <tr> <td>Abort</td><td>GetOD</td></tr> <tr> <td>Reject</td><td>InitiatePutOD</td></tr> <tr> <td>Status</td><td>PutOD</td></tr> <tr> <td>Identify</td><td>TerminatePutOD</td></tr> <tr> <td>PhysRead</td><td></td></tr> </table> | Initiate | PhysWrite | Abort | GetOD | Reject | InitiatePutOD | Status | PutOD | Identify | TerminatePutOD | PhysRead |  |
| Initiate                   | PhysWrite                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |          |           |       |       |        |               |        |       |          |                |          |  |
| Abort                      | GetOD                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |          |           |       |       |        |               |        |       |          |                |          |  |
| Reject                     | InitiatePutOD                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |          |           |       |       |        |               |        |       |          |                |          |  |
| Status                     | PutOD                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |          |           |       |       |        |               |        |       |          |                |          |  |
| Identify                   | TerminatePutOD                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |          |           |       |       |        |               |        |       |          |                |          |  |
| PhysRead                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |          |           |       |       |        |               |        |       |          |                |          |  |

##### Physical Status

This attribute gives a coarse summary of the state of the real device. Its values are:

- operational
- partially operational
- not operational
- needs maintenance

##### List of VFD Specific Objects

Contains all VFD Specific Objects.

#### 6.3.1.1.4 Services

##### Status

This service is used to request the network visible state from the AP.

##### Unsolicited Status

This service is used by the AP to report its network visible state.

##### Identify

This service is used to request manufacturer information from the AP.

#### 6.3.1.2 Virtual field device ASE service specification

##### 6.3.1.2.1 Supported services

This subclause contains the definition of services that are unique to this ASE. The services defined for this ASE are:

- Status
- Unsolicited Status
- Identify

##### 6.3.1.2.2 Status service

###### 6.3.1.2.2.1 Service overview

The device/user state is read with the Status service.

###### 6.3.1.2.2.2 Service primitives

The service parameters for this service are shown in Table 3.

**Table 3 – Status**

| Parameter name                                                                                                                                                                                                                                                            | Req | Ind | Rsp | Cnf   |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-----|-----|-------|
| Argument                                                                                                                                                                                                                                                                  |     |     |     |       |
| VCR                                                                                                                                                                                                                                                                       | M   | M   |     |       |
| Result(+)                                                                                                                                                                                                                                                                 |     |     | S   | S (=) |
| Logical Status                                                                                                                                                                                                                                                            |     |     | M   | M (=) |
| Physical Status                                                                                                                                                                                                                                                           |     |     | M   | M (=) |
| Local Detail                                                                                                                                                                                                                                                              |     |     | U   | U (=) |
| Result(-)                                                                                                                                                                                                                                                                 |     |     | S   | S (=) |
| Error Type                                                                                                                                                                                                                                                                |     |     | M   | M (=) |
| NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. The method by which a response primitive is correlated with its corresponding preceding indication primitive is a local matter. See 1.2. |     |     |     |       |

##### Argument

The argument contains the parameters of the service request.

##### Result(+)

This selection type parameter indicates that the service request succeeded.

### Logical Status

This parameter specifies the value of the Logical Status attribute of the VFD.

### Physical Status

This parameter specifies the value of the Physical Status attribute of the VFD.

### Local Detail

This parameter specifies the user defined state of the application.

### Result(-)

This selection type parameter indicates that the service request failed.

#### 6.3.1.2.2.3 Service procedure

The Confirmed Service Procedure specified in Clause 4 applies to this service.

#### 6.3.1.2.3 Unsolicited status service

##### 6.3.1.2.3.1 Service overview

The Status Notification service is used by the AP to report its state spontaneously.

##### 6.3.1.2.3.2 Service primitives

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

**Table 4 – Unsolicited status**

| Parameter name         | Req | Ind   |
|------------------------|-----|-------|
| Argument               |     |       |
| VCR                    | M   | M     |
| Destination DL-Address | C   |       |
| Source DL-Address      |     | C     |
| LogicalStatus          | M   | M (=) |
| Physical Status        | M   | M (=) |
| Local Detail           | U   | U (=) |

### Argument

The argument contains the parameters of the service request.

### Destination DL-Address

This parameter exists only when the corresponding AREP supports it and the Remote Address Configuration Type is FREE. It gives the remote address to which the Status Notification should be sent.

### Source DL-Address

This parameter exists only when the corresponding AREP supports it and the Remote Address Configuration Type is FREE. This parameter indicates the remote address from which the indicated Status Notification is to be sent.

### Logical Status

This parameter specifies the value of the Logical Status attribute of the VFD.

### Physical Status

This parameter specifies the value of the Physical Status attribute of the VFD.

## Local Detail

This parameter specifies the user defined state of the application.

### 6.3.1.2.3.3 Service procedure

The Unconfirmed Service Procedure specified in Clause 4 applies to this service.

### 6.3.1.2.4 Identify service

#### 6.3.1.2.4.1 Service overview

Information to identify an AP is read with this service.

NOTE The attribute ProfileNumber of the AP is transmitted with the Initiate service of the Context Management ASE.

#### 6.3.1.2.4.2 Service primitives

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

**Table 5 – Identify**

| Parameter name                                                                                                                                                                                                                                                            | Req | Ind | Rsp | Cnf   |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-----|-----|-------|
| Argument                                                                                                                                                                                                                                                                  |     |     |     |       |
| VCR                                                                                                                                                                                                                                                                       | M   | M   |     |       |
| Result(+)                                                                                                                                                                                                                                                                 |     |     | S   | S (=) |
| Vendor Name                                                                                                                                                                                                                                                               |     |     | M   | M (=) |
| Model Identifier                                                                                                                                                                                                                                                          |     |     | M   | M (=) |
| Vendor Revision                                                                                                                                                                                                                                                           |     |     | M   | M (=) |
| Result(-)                                                                                                                                                                                                                                                                 |     |     | S   | S (=) |
| Error Type                                                                                                                                                                                                                                                                |     |     | M   | M (=) |
| NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. The method by which a response primitive is correlated with its corresponding preceding indication primitive is a local matter. See 1.2. |     |     |     |       |

#### Argument

The argument contains the parameters of the service request.

#### Result(+)

This selection type parameter indicates that the service request succeeded.

#### Vendor Name

This parameter specifies the value of the Vendor Name attribute of the VFD.

#### Model Identifier

This parameter specifies the value of the Model Identifier attribute of the VFD.

#### Vendor Revision

This parameter specifies the value of the Vendor Revision attribute of the VFD.

#### Result(-)

This selection type parameter indicates that the service request failed.

#### **6.3.1.2.4.3 Service procedure**

The Confirmed Service Procedure specified in Clause 4 applies to this service.

### **6.3.2 Object dictionary (OD) ASE**

#### **6.3.2.1 Overview**

The ASE Models each specify one or more related APO classes as a collection of attributes and services.

The collection of attributes for the APOs of an AP is referred to as the Object Dictionary (OD). The attributes for a single APO is referred to as its object description. To access the object descriptions of APOs, this ASE defines common OD services. These services operate using the client/server model.

The Object Dictionary (OD) contains the Object Descriptions of the following Communication Objects:

- DataType
- Domain
- SimpleVariable
- Record
- Event
- DataTypeStructureDescription
- ProgramInvocation
- Array
- VariableList

An object description "Object Dictionary" (OD object description) is uniquely assigned to the Object Dictionary. The OD object description contains information about the structure of the Object Dictionary.

The object descriptions are marked with a unique index of type Unsigned16.

A name of type visible string may also be assigned to the following objects:

- Domain
- SimpleVariable
- Record
- Event
- ProgramInvocation
- Array
- VariableList

The index or the name serves as a key to the object and the object description.

The services address on principle the object of the server, because the real object exists there.

Before a remote communication partner can access an object, it needs to know its description. The logical address (Index) or the name may be used, when objects are accessed remotely.

The object descriptions may be specified during system configuration, but they may also be transferred between stations at any later time.

The definition of the object description takes place at that station, in which the object really exists (Source OD). The length of an object description may be different for the individual objects. The communication partners maintain a complete or a partial copy of the Remote Object Description (Remote OD). This is illustrated in Figure 4.

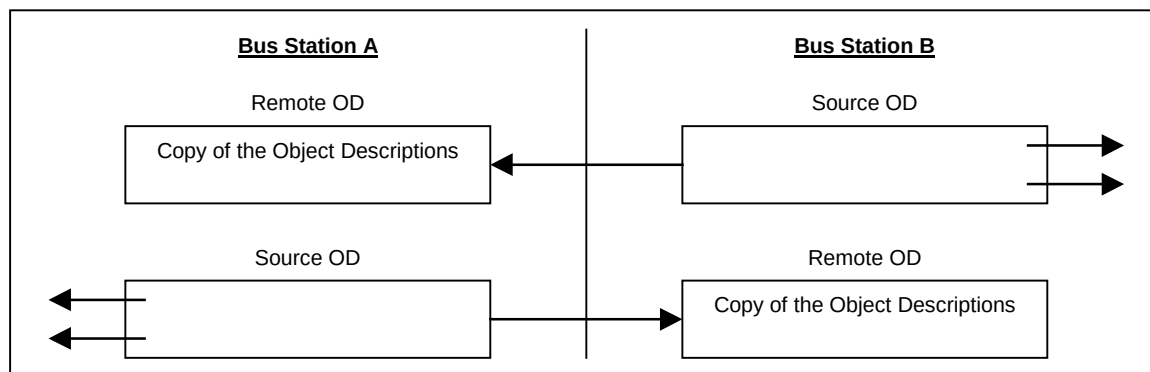


**Figure 4 – Source OD/remote OD**

Thus every station may possess a Source OD for locally existing communication objects and one or more Remote ODs. The Remote ODs are connection specific, i.e. for each connection another Remote OD may be valid.

The OD model provides corresponding services for reading and writing of Source ODs. If a Source OD is modified by a remote communication partner, then this should be notified to all other communication partners which use the same data base (Source OD). For this purpose, the user releases the connections to all other communication partners. After the OD has been modified, the connections may be established again. During the connection establishment, the new version number (Version OD) is given to each communication partner to notify that the corresponding Source OD has been modified. The addition of object descriptions is permissible without connection release.

The structure of the OD described as follows does not prescribe a concrete implementation of the OD. It only defines the structure of the object descriptions which are transmitted via the bus. Even the existence of an OD is not necessary, if the needed information can be obtained in another way (e.g. by an algorithm). Detailed attributes and transfer syntaxes are specified in the appropriate models.

The Object Dictionary (OD) is divided into several parts, as shown in Table 6:

- The Object description of the "Object Dictionary" (OD object description) contains structural information about the OD.
- Static Type Dictionary (ST-OD) contains Data types and Data type Structure Description.
- Static Object Dictionary (S-OD) contains object descriptions of Simple Variables, Arrays, Records, Domains and Events.
- Dynamic List of Variable Lists (DV-OD) contains object descriptions of variable lists.
- Dynamic List of Program Invocations (DP-OD) contains object descriptions of Program Invocations.

**Table 6 – Structure of the object dictionary**

| Index         | Object Dictionary (OD)                      |
|---------------|---------------------------------------------|
| 0             | OD object description (OD-ODES)             |
| 1<br>...<br>i | Static List of Types (ST-OD)                |
| k<br>...<br>n | Static Object Dictionary (S-OD)             |
| p<br>...<br>t | Dynamic List of Variable Lists (DV-OD)      |
| v<br>...<br>z | Dynamic List of Program Invocations (DP-OD) |

#### 6.3.2.1.1 Static list of types (ST-OD)

The Static List of Types (ST-OD) contains the object descriptions of Data types and Data type Structure Descriptions for the Variable Access objects of the S-OD, as shown in Table 7.

**Table 7 – Structure of the static list of types**

| Index         | Static List of Types (ST-OD)                                            |
|---------------|-------------------------------------------------------------------------|
| 1<br>...<br>c | contains objects: Data types                                            |
| d<br>...<br>i | contains objects: Data types<br>and<br>Data type Structure Descriptions |

Exactly one object description is assigned to every index. This assignment is static and may not be deleted. The ST-OD may be loaded by the remote partner.

The Static List of Types always begins with the index 1 in the Object Dictionary. The remaining object descriptions should follow with increasing index. The number of assignments "index <=> object description" is entered as length of the ST-OD in the Object Dictionary.

The ST-OD is divided in two segments. The first segment contains the object descriptions of standardized Data types (Standard Data types) and begins with index 1. The second segment follows immediately and contains object descriptions of Data types and Data type Structure Descriptions, which can be defined individually. This segment may be further specified by user groups with the help of profiles.

#### 6.3.2.1.2 Static object dictionary (S-OD)

The Static Object Dictionary (S-OD) contains the object descriptions of Simple Variables, Arrays, Records, Domains and Events, as shown in Table 8.

**Table 8 – Structure of the static object dictionary**

| Index         | Static Object Dictionary (S-OD)                                            |
|---------------|----------------------------------------------------------------------------|
| k<br>...<br>n | contains objects: Simple Variables,<br>Arrays, Records, Domains and Events |

An object description is assigned to each index. The first index, which is associated to an object description, is entered in the object description of the Object Dictionary. The remaining object descriptions should follow with increasing index. The number of assignments "index <=> object description" is entered as length of the S-OD in the object description of Object Dictionary.

Besides the assignment "index <=> object description", an additional assignment "name <=> object description" may exist. The length of the name may be from 0 to 32 octets. It is entered in the Name Length Field of the OD object description. If this field contains a "0", no names are present. The objects which are defined in the S-OD may be provided with access rights. Whether Access Rights are possible is entered in the field AccessProtectionSupported of the OD object description.

An object description in the S-OD contains an extension, which may contain further object specific definitions.

#### 6.3.2.1.3 Dynamic list of variable lists (DV-OD)

The Dynamic List of Variable Lists (DV-OD) contains the object description of the Variable Lists, as shown in Table 9.

**Table 9 – Structure of the dynamic list of variable lists**

| Index         | Dynamic List of Variable Lists (DV-OD) |
|---------------|----------------------------------------|
| p<br>...<br>t | contains objects: Variable List        |

An object "Variable List" and its object description in the DV-OD is dynamically created with the service DefineVariableList and may be deleted with the service DeleteVariableList. Upon creation of the object access rights may be assigned to it. Moreover it is possible, to create these objects by loading the whole OD.

The first index, which has the object description of a Variable List assigned to it, and the number of available indices are entered in the OD object description.

Fundamental information is entered in the object description about the Variable List, for example access rights, the number of Variable Access objects and the logical addresses (indices) of Variable Access objects in the S-OD.

Besides the assignment "index <=> object description", an additional assignment "name <=> object description" may exist.

#### 6.3.2.1.4 Dynamic list of program invocations (DP-OD)

The Dynamic List of Program Invocations (DP-OD) contains the object descriptions of the objects Program Invocation, as shown in Table 10.

**Table 10 – Structure of the dynamic list of program invocations**

| Index         | Dynamic List of Program Invocations (PD-OD) |
|---------------|---------------------------------------------|
| v<br>...<br>w | Predefined Program Invocations              |
| x<br>...<br>z | Dynamic Program Invocations                 |

An object Program Invocation and its object description in the DP-OD is dynamically created with the service Create Program Invocation and deleted with the service Delete Program Invocation. Upon creation of the object access rights may be assigned to it. Moreover it is possible to create these objects by loading the whole OD.

The first index, which has the object description of a Program Invocation assigned to it, and the number of available indices are entered in the OD object description.

The object description of a Program Invocation contains information such as access rights and the number of Domain objects and their logical addresses (indices).

Besides the assignment "index <=> object description", an additional assignment "name <=> object description" may exist.

The DP-OD may contain a segment with pre-configured Program Invocations.

### 6.3.2.2 OD class specifications

#### 6.3.2.2.1 Class overview

Every object description contains an index, an object code, object attributes, system specific references to the real object and, if need be, a name and an extension.

The object is addressed by the index and identified by the object description. The object code is the identifier of the object and indicates which class this object belongs to. The other object attributes are object specific. The content of the extension field is application-specific.

The local address serves the internal addressing of the object. Besides the index, optionally a name for the addressing of the object may also be used. The extension contains a length information for the extension and optionally further object attributes. The value "0" for the length means: no further object attributes.

#### 6.3.2.2.2 OD description class specification

##### 6.3.2.2.2.1 Formal model

The structure of the Object Dictionary is described by the first entry in the OD. This has the index "0" in the Object Dictionary. By reading the OD object description the structure description and the version number of the OD are made available.

|                       |                              |
|-----------------------|------------------------------|
| <b>ASE:</b>           | <b>Object Dictionary ASE</b> |
| <b>CLASS:</b>         | <b>OD Description</b>        |
| <b>CLASS ID:</b>      | —                            |
| <b>PARENT CLASS:</b>  | <b>TOP</b>                   |
| <b>ATTRIBUTES:</b>    |                              |
| 1. (m) Key Attribute: | OD Index                     |
| 2. (m) Attribute:     | ROM/RAM Flag                 |
| 3. (m) Attribute:     | Name Length                  |
| 4. (m) Attribute:     | Access Protection Supported  |
| 5. (m) Attribute:     | Version OD                   |
| 6. (m) Attribute:     | Local Address OD-ODES        |
| 7. (m) Attribute:     | ST-OD Length                 |
| 8. (m) Attribute:     | LocalAddress ST-OD           |
| 9. (m) Attribute:     | First Index S-OD             |
| 10. (m) Attribute:    | S-OD Length                  |
| 11. (m) Attribute:    | Local Address S-OD           |
| 12. (m) Attribute:    | First Index DV-OD            |

- 13. (m) Attribute: DV-OD Length
- 14. (m) Attribute: LocalAddress DV-OD
- 15. (m) Attribute: First Index DP-OD
- 16. (m) Attribute: DP-OD Length
- 17. (m) Attribute: Local Address DP-OD

**SERVICES:**

- 1. (m) OpsService: Get OD
- 2. (m) OpsService: Initiate Put OD
- 3. (m) OpsService: Put OD
- 4. (m) OpsService: Terminate Put OD

**6.3.2.2.2 Attributes****OD Index**

OD Index of the OD Object Description, here always = 0.

**ROM/RAM Flag**

This attribute of type Boolean describes whether modifications in the OD are allowed.

false <=> no modifications in the OD are permitted

true <=> modifications in the OD are permitted

**Name Length**

This field gives the length of the name and may take only the values "0..32":

0 <=> no names are used

**Access Protection Supported**

This attribute states, whether access rights for pass-word, access groups and all communication partners are supported (true), or whether the access to all objects is permitted for every communication partner (false).

**Version OD**

Gives the version of the OD.

**Local Address OD-ODES**

This attribute is a system specific reference to the real OD object description and serves the internal addressing. If no representation of this kind is used, this attribute is set to the value FFFFFFFF hex.

**ST-OD Length**

This attribute gives the maximal number of available entries in the Static List of Types.

NOTE The first available index of the Static List of Types is always = 1, therefore the highest available index in the Static List of Types is equal to its length.

**Local Address ST-OD**

This attribute is a system specific reference to the real ST-OD and serves the internal addressing. If no representation of this kind takes place, this attribute is set to the value FFFFFFFF hex.

**First Index S-OD**

The first available index in the Static Object Dictionary.

**S-OD Length**

This attribute gives the maximum number of available entries in the Static Object Dictionary.

NOTE The highest available index is  $S\text{-OD-Length} = \text{FirstIndex-S-OD} + S\text{-OD-Length} - 1$  (done)

#### **Local Address S-OD**

This attribute is a system specific reference to the real S-OD and serves the internal addressing. If no representation of this kind takes place, this attribute is set to the value FFFFFFFF hex.

#### **First Index DV-OD**

First available index in the Dynamic List of Variable Lists.

0 <=> no Dynamic List of Variable Lists.

#### **DV-OD Length**

This attribute gives the maximum number of available entries in the Dynamic List of Variable Lists.

NOTE The highest available index is  $DV\text{-OD-Length} = \text{FirstIndex-DV-OD} + OD\text{-OD-Length} - 1$ .

#### **Local Address DV-OD**

This attribute is a system specific reference to the real DV-OD and serves the internal addressing. If no representation of this kind takes place, this attribute is set to the value FFFFFFFF hex.

#### **First Index DP-OD**

First available index in the Dynamic List of Program Invocations.

0 <=> no Dynamic List of Program Invocations.

#### **DP-OD Length**

This attribute gives the maximum number of available entries in the Dynamic List of Program Invocations.

NOTE The highest available index is  $DP\text{-OD-Length} = \text{FirstIndex-DP-OD} + DP\text{-OD-Length} - 1$ .

#### **Local Address DP-OD**

This attribute is a system specific reference to the real DP-OD and serves the internal addressing. If no representation of this kind takes place, this attribute is set to the value FFFFFFFF hex.

### **6.3.2.2.2.3 Services**

All services defined for this class are mandatory.

#### **Get OD**

This confirmed service is used to read entries of the OD.

#### **Initiate Put OD**

This confirmed service is used to begin the process of writing entries of the OD.

#### **Put OD**

This confirmed service is used to write entries of the OD.

#### **Terminate Put OD**

This confirmed service is used to terminate the process of writing entries of the OD.

#### 6.3.2.2.2.4 Empty object dictionary

The Object Dictionary is loadable. Therefore if an Object Dictionary is not yet loaded, at least an empty Object Dictionary should exist.

An empty Object Dictionary is defined by pre-setting the attributes of the OD object description shown in Table 11:

**Table 11 – Empty object dictionary**

| Attribute                 | Value |
|---------------------------|-------|
| ROM/RAM-Flag              | True  |
| NameLength                | 0     |
| AccessProtectionSupported | False |
| VersionOD                 | 0     |
| ST-OD-Length              | 0     |
| FirstIndex-S-OD           | 0     |
| S-OD-Length               | 0     |
| FirstIndex-DV-OD          | 0     |
| DV-OD-Length              | 0     |
| FirstIndex-DP-OD          | 0     |
| DP-OD-Length              | 0     |

#### 6.3.2.2.3 OD object formal model

##### 6.3.2.2.3.1 Formal model

The structure of the Object Dictionary is described by the first entry in the OD. This has the index "0" in the Object Dictionary. By reading the OD object description the structure description and the version number of the OD are made available.

**ASE:** Object Dictionary ASE

**CLASS:** OD Object

**CLASS ID:** —

**PARENT CLASS:** TOP

##### ATTRIBUTES:

1. (m) Key Attribute: implicit by VFD
2. (m) Attribute: OD object description
3. (m) Attribute: List of object description
4. SERVICES:
  1. (m) OpsService: Get OD
  2. (m) OpsService: Initiate Put OD
  3. (m) OpsService: PutOD
  4. (m) OpsService: Terminate Put OD

##### 6.3.2.2.3.2 Attributes

###### implicit by VFD

The VCR, which is entered in the VCR List points to the assigned Object Dictionary.

###### OD object description

Contains the object description of the Object Dictionary.

###### List of object description

Contains object descriptions of the following objects:

Domain

ProgramInvocation

SimpleVariable  
 Array  
 Record  
 VariableList  
 DataType  
 DataTypeStructureDescription  
 Event

#### **6.3.2.2.3.3 Services**

All services defined for this class are mandatory.

##### **Get OD**

This confirmed service is used to read entries of the OD.

##### **Initiate Put OD**

This confirmed service is used to begin the process of writing entries of the OD.

##### **Put OD**

This confirmed service is used to write entries of the OD.

##### **Terminate Put OD**

This confirmed service is used to terminate the process of writing entries of the OD.

### **6.3.2.3 Object dictionary model services**

#### **6.3.2.3.1 Supported services**

OD services are specified to read or write the object descriptions of Domain, Program Invocation, Simple Variable, Array, Record, Variable List, Event, Object Dictionary, Data type and Data type Structure Description objects. The attributes of an APO which can be written are specified as part of the attribute definition.

Because unexpected changes to the object definitions within an AP can affect operation of the AP, the Initiate Put OD and the Terminate Put OD services are defined. The Initiate Put OD service is used to request the remote AP for permission to change the OD. The Terminate Put OD service is used to indicate to the remote AP that changes to its OD are complete and that it can resume normal operation.

With the GetOD service one or several object descriptions are read. With the PutOD service one or several object descriptions are written. The number of object descriptions, which are transferable with one PutOD or GetOD Service, depends on their length and the maximum available PDU size.

With the PutOD services one or several object descriptions are written. Before the beginning of a modification of the OD an user may take appropriate measures (e.g. to put a process into a safe state). The communication does not test the plausibility of the new Object Dictionary.

Writing in the OD is performed with the sequence InitiatePutOD service, one or several PutOD services and a TerminatePutOD service. For this it is distinguished between a loading which is free of interactions and a loading which is not free of interactions (OD-LOADING-NON-INTERACTING and OD-LOADING-INTERACTING). The loading which is not free of interactions differentiates between a new loading (complete) and a post loading (partial). If need be, new communication objects and their state machines are created with the TerminatePutOD.

**Loading free of interactions ( OD-LOADING-NON-INTERACTING ):**

The client communicates to the server with the parameter Consequence = 0, whether the intended modifications or extensions are free of interactions for the other communication partners. In this case the VCRs are continued. For example, modifications free of interactions are

- addition of new object descriptions in the static segment,
- deletion of private object descriptions for stations which are no longer in the communication system.

The communication does not check whether the write to the OD is really free of interactions.

The loading free of interaction is initiated with the service InitiatePutOD with the parameter Consequence = 0. After that object descriptions may be loaded with PutOD. As long as the loading process is not terminated with TerminatePutOD, every further InitiatePutOD is rejected.

**Loading not free of interactions ( OD-LOADING-INTERACTING ):**

The loading not free of interactions is initiated with the service InitiatePutOD with the Parameter Consequence = 1 (post-loading) or Consequence = 2 (new-loading). In this case the user may abort all active VCRs, except the one over which the InitiatePutOD is received to this VFD. After that the object description may be loaded with PutOD. It is also the task of the client user to enter the new version number in the OD. The loading process is terminated with the service TerminatePutOD. After that all VCRs may be re-established. With the re-establishment of the VCRs the new version number of the OD is transferred. The communication partners may then recognize that the corresponding OD has been modified. Whereas for post-loading solely parts of the OD are loaded, the existing OD is deleted and subsequently loaded completely anew upon receipt of an InitiatePutOD with Consequence = 2.

**6.3.2.3.2 Get OD service****6.3.2.3.2.1 Service overview**

The service distinguishes between a short and a long form. The long form of the GetOD service is optional. To read a single object description, the index or the name should be given.

To read several or all object descriptions, the index of the first required object description (StartIndex) should be given. To read the whole OD, the GetOD service should be used repeatedly.

**6.3.2.3.2.2 Service parameters**

The service parameters for this service are shown in Table 12.

**Table 12 – Get OD service parameters**

| Parameter name                                                                                                                                                                                                                                                            | Req | Ind   | Rsp | Cnf   |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-------|-----|-------|
| Argument                                                                                                                                                                                                                                                                  |     |       |     |       |
| VCR                                                                                                                                                                                                                                                                       | M   | M     |     |       |
| All Attributes                                                                                                                                                                                                                                                            | M   | M (=) |     |       |
| Access Specification                                                                                                                                                                                                                                                      | M   | M (=) |     |       |
| Index                                                                                                                                                                                                                                                                     | S   | S (=) |     |       |
| Variable Name                                                                                                                                                                                                                                                             | S   | S (=) |     |       |
| Variable List Name                                                                                                                                                                                                                                                        | S   | S (=) |     |       |
| Domain Name                                                                                                                                                                                                                                                               | S   | S (=) |     |       |
| PI Name                                                                                                                                                                                                                                                                   | S   | S (=) |     |       |
| Event Name                                                                                                                                                                                                                                                                | S   | S (=) |     |       |
| Start Index                                                                                                                                                                                                                                                               | S   | S (=) |     |       |
| Result(+)                                                                                                                                                                                                                                                                 |     |       | S   | S (=) |
| List of Object Description                                                                                                                                                                                                                                                |     |       | M   | M (=) |
| More Follows                                                                                                                                                                                                                                                              |     |       | M   | M (=) |
| Result(-)                                                                                                                                                                                                                                                                 |     |       | S   | S (=) |
| Error Type                                                                                                                                                                                                                                                                |     |       | M   | M (=) |
| NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. The method by which a response primitive is correlated with its corresponding preceding indication primitive is a local matter. See 1.2. |     |       |     |       |

### Argument

The argument contains the parameters of the service request.

### All Attributes

This parameter states whether the object description should be transferred in the short form (false) or in the long form (true).

The following object attributes are not contained in the short form:

- Password
- AccessGroups
- AccessRights
- LocalAddress
- Name
- LocalAddress-OD-ODES
- LocalAddress-ST-OD
- LocalAddress-S-OD
- LocalAddress-DV-OD
- LocalAddress-DP-OD
- Extension

### Access Specification

This parameter states which object description is accessed.

**Index**

Index of the object description.

**Variable Name**

Name of a variable.

**Variable List Name**

Name of a Variable List.

**PI Name**

Name of a Program Invocation.

**Domain Name**

Name of a Domain.

**Event Name**

Name of an Event.

**StartIndex**

Index, starting with which the object descriptions should be read. When this option is selected, the responding VFD sends object descriptions of indexes greater than StartIndex in the order of increasing Index. When all of requested object descriptions are not sent in the response, the MoreFollows flag should be set.

**Result(+)**

This selection type parameter indicates that the service request succeeded.

**List of object description**

For logical access and for access by name one, otherwise several complete object descriptions. The number depends on the maximum possible PDU size and the length of the single object descriptions. If access rights are not supported (Access Protection = false), the attributes Password, Access Groups and Access Rights are meaningless.

**More Follows**

States whether further object descriptions exist after reading several object descriptions selecting StartIndex for Access Specification. This parameter is set true when the responding VFD has at least one object description of Index greater than that of the last object description in the response. It is set false when the responding VFD has no more object descriptions. For the access to single object descriptions selecting other options for Access Specification, descriptions this parameter is always set to false.

**Result(-)**

This selection type parameter indicates that the service request failed.

**6.3.2.3.2.3 Service procedure**

The Confirmed Service Procedure specified in Clause 4 applies to this service.

If the user is able to access the specified object descriptions, and if they can be conveyed in a single APDU, they are returned in the response (+) primitive.

If they cannot be conveyed in a single APDU, the user returns those that can and sets the more flag in the response (+) primitive. The requester is then responsible for submitting additional requests identifying the remaining attributes to be retrieved.

If the user is unable to get the description for the first object in the list, the service fails and the user issues a response (-) primitive indicating the reason.

### 6.3.2.3.3 InitiatePutOD

#### 6.3.2.3.3.1 Service overview

The InitiatePutOD opens the process of loading the OD free of interaction or not free of interaction.

#### 6.3.2.3.3.2 Service parameters

The service parameters for this service are shown in Table 13.

**Table 13 – Initiate put OD service parameters**

| Parameter name                                                                                                                                                                                                                                                            | Req | Ind   | Rsp | Cnf   |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-------|-----|-------|
| Argument                                                                                                                                                                                                                                                                  |     |       |     |       |
| VCR                                                                                                                                                                                                                                                                       | M   | M     |     |       |
| Consequence                                                                                                                                                                                                                                                               | M   | M (=) |     |       |
| Result (+)                                                                                                                                                                                                                                                                |     |       | S   | S (=) |
| Result (-)                                                                                                                                                                                                                                                                |     |       | S   | S (=) |
| Error Type                                                                                                                                                                                                                                                                |     |       | M   | M (=) |
| NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. The method by which a response primitive is correlated with its corresponding preceding indication primitive is a local matter. See 1.2. |     |       |     |       |

#### Argument

The argument contains the parameters of the service request.

#### Consequence

This parameter states whether the intended modifications are free of interaction for the other communication partners, not free of interaction with partial reload of the OD, or not free of interaction with completely new load of the OD.

#### Result(+)

This selection type parameter indicates that the service request succeeded.

#### Result(-)

This selection type parameter indicates that the service request failed.

#### 6.3.2.3.3.3 Service procedure

The Confirmed Service Procedure specified in Clause 4 applies to this service.

#### 6.3.2.3.4 Put OD service

##### 6.3.2.3.4.1 Service overview

With the PutOD service one or several object descriptions are written into the OD.

##### 6.3.2.3.4.2 Service parameters

The service parameters for this service are shown in Table 14.

**Table 14 – Put OD service parameters**

| Parameter name                                                                                                                                                                                                                                                            | Req | Ind   | Rsp | Cnf   |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-------|-----|-------|
| Argument                                                                                                                                                                                                                                                                  |     |       |     |       |
| VCR                                                                                                                                                                                                                                                                       | M   | M     |     |       |
| List of Object Description                                                                                                                                                                                                                                                | M   | M (=) |     |       |
| Result (+)                                                                                                                                                                                                                                                                |     |       | S   | S (=) |
| Result (-)                                                                                                                                                                                                                                                                |     |       | S   | S (=) |
| Error Type                                                                                                                                                                                                                                                                |     |       | M   | M (=) |
| NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. The method by which a response primitive is correlated with its corresponding preceding indication primitive is a local matter. See 1.2. |     |       |     |       |

**Argument**

The argument contains the parameters of the service request.

**List of Object Description**

One or several object descriptions which should be entered into the OD. If access rights are not supported (Access Protection = false), the attributes Password, Access Groups and Access Rights are meaningless.

An object description may also be deleted with the PutOD service. In this case, an empty object description with the following structure is transferred:

- Index of the object description to be deleted
- Object Code = Null Object.

**Result(+)**

This selection type parameter indicates that the service request succeeded.

**Result(-)**

This selection type parameter indicates that the service request failed.

**6.3.2.3.4.3 Service procedure**

The Confirmed Service Procedure specified in Clause 4 applies to this service.

**6.3.2.3.5 Terminate put OD service****6.3.2.3.5.1 Service overview**

The Terminate Put OD service terminates the process of loading of the OD. The objects with their state machines are generated.

**6.3.2.3.5.2 Service parameters**

The service parameters for this service are shown in Table 15.

### Table 15 – Terminate put OD service parameters

| Parameter name | Req | Ind | Rsp | Cnf   |
|----------------|-----|-----|-----|-------|
| Argument       |     |     |     |       |
| VCR            | M   | M   |     |       |
| Result (+)     |     |     | S   | S (=) |
| Result (-)     |     |     | S   | S (=) |
| Error Type     |     |     | M   | M (=) |
| Index          |     |     | C   | C (=) |

NOTE The method by which a confirm primitive is correlated with its corresponding preceding request primitive is a local matter. The method by which a response primitive is correlated with its corresponding preceding indication primitive is a local matter. See 1.2.

### Argument

The argument contains the parameters of the service request.

**Result(+)**

This selection type parameter indicates that the service request succeeded.

**Result(-)**

This selection type parameter indicates that the service request failed.

## Index

This parameter gives the Index of the object, for which the generation was not successful.

#### 6.3.2.3.5.3 Service procedure

The Confirmed Service Procedure specified in Clause 4 applies to this service.

#### 6.3.2.3.6 Put OD services state machine

The state machine is shown in Figure 5.

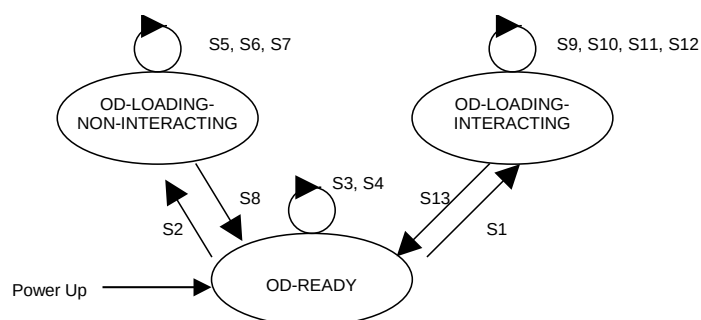


**Figure 5 – Put OD state machine**

**6.3.2.3.6.1 States****OD-LOADING-NON-INTERACTING**

In this state the execution of the service InitiatePutOD is not permitted. In this state the service PutOD on the DV-OD or the DP-OD is not permitted.

**OD-LOADING-INTERACTING**

Except for the connection over which the InitiatePutOD service was received, all connections are locked and the establishment of a further connection should be rejected. On this connection only the following services are allowed:

- Initiate
- Abort
- Reject
- Status
- Identify
- PhysRead
- PhysWrite
- GetOD
- InitiatePutOD
- PutOD
- TerminatePutOD

**OD-READY**

In this state all connections and services may be used normally.

**6.3.2.3.6.2 State transitions**

Table 16 specifies the state transitions related to PutOD.

**Table 16 – Put OD state transitions**

| #   | Current State              | Events                                                             | Actions                                                          | Next State                 |
|-----|----------------------------|--------------------------------------------------------------------|------------------------------------------------------------------|----------------------------|
| S1  | OD-READY                   | InitiatePutOD.ind<br>&& consequence <> loading-free-of-interaction | InitiatePutOD.rsp(+){<br>}                                       | OD-LOADING-INTERACTING     |
| S2  | OD-READY                   | InitiatePutOD.ind<br>&& consequence = loading-free-of-interaction  | InitiatePutOD.rsp(+){<br>}                                       | OD-LOADING-NON-INTERACTING |
| S3  | OD-READY                   | PutOD.ind                                                          | PutOD.rsp(-){<br>ErrorType = object-state-conflict<br>}          | OD-READY                   |
| S4  | OD-READY                   | TerminatePutOD.ind                                                 | TerminatePutOD.rsp(-){<br>ErrorType = object-state-conflict<br>} | OD-READY                   |
| S5  | OD-LOADING-NON-INTERACTING | InitiatePutOD.ind                                                  | InitiatePutOD.rsp(-){<br>ErrorType = object- state-conflict<br>} | OD-LOADING-NON-INTERACTING |
| S6  | OD-LOADING-NON-INTERACTING | PutOD.ind                                                          | PutOD.rsp(+){<br>}                                               | OD-LOADING-NON-INTERACTING |
| S7  | OD-LOADING-NON-INTERACTING | TerminatePutOD.ind<br>&& OD is not usable                          | TerminatePutOD.rsp(-){<br>ErrorType = operational-problem<br>}   | OD-LOADING-NON-INTERACTING |
| S8  | OD-LOADING-NON-INTERACTING | TerminatePutOD.ind<br>&& OD is okay                                | TerminatePutOD.rsp(+){<br>}                                      | OD-READY                   |
| S9  | OD-LOADING-INTERACTING     | InitiatePutOD.ind<br>&& consequence <> loading-free-of-interaction | InitiatePutOD.rsp(+){<br>}                                       | OD-LOADING-INTERACTING     |
| S10 | OD-LOADING-INTERACTING     | InitiatePutOD.ind<br>&& consequence = loading-free-of-interaction  | InitiatePutOD.rsp(-){<br>ErrorType = object-state-conflict<br>}  | OD-LOADING-INTERACTING     |
| S11 | OD-LOADING-INTERACTING     | PutOD.ind                                                          | PutOD.rsp(+){<br>}                                               | OD-LOADING-INTERACTING     |
| S12 | OD-LOADING-INTERACTING     | TerminatePutOD.ind<br>&& OD is not usable                          | TerminatePutOD.rsp(-){<br>ErrorType = operational-problem<br>}   | OD-LOADING-INTERACTING     |
| S13 | OD-LOADING-INTERACTING     | TerminatePutOD.ind<br>&& OD is okay                                | TerminatePutOD.rsp(+){<br>}                                      | OD-READY                   |

### 6.3.3 Context management ASE

#### 6.3.3.1 Overview

The context management services are used to explicitly establish a VCR, to release a VCR and to reject improper services. The data necessary for this are stored in the OD object description and in the VCR object. Transaction objects are provided for each VCR. The transaction objects are used for the management of confirmed service primitives.

#### 6.3.3.2 Context management class specifications

##### 6.3.3.2.1 VCR list class specification

##### 6.3.3.2.1.1 VCR list formal model

The Virtual Communication Relationship List (VCR List) contains the descriptions of all VCRs of a device. A VCR consists of static part and dynamic part. This subclause gives the definition to FMS specific part of VCR. Refer Network Management Specification for the definition of other part of VCR.