

DIN EN 61158-5-9

ICS 35.100.70; 35.160

Ersatz für
DIN EN 61158-5-9:2008-09
Siehe Anwendungsbeginn

**Industrielle Kommunikationsnetze –
Fieldbusse –
Teil 5-9: Dienstfestlegungen des Application Layer
(Anwendungsschicht) – Typ 9-Elemente
(IEC 61158-5-9:2014);
Englische Fassung EN 61158-5-9:2014, nur auf CD-ROM**

Industrial communication networks –
Fieldbus specifications –
Part 5-9: Application layer service definition – Type 9 elements
(IEC 61158-5-9:2014);
English version EN 61158-5-9:2014, only on CD-ROM

Réseaux de communication industriels –
Spécifications des bus de terrain –
Partie 5-9: Définition des services de la couche application – Éléments de type 9
(CEI 61158-5-9:2014);
Version anglaise EN 61158-5-9:2014, seulement en CD-ROM

Gesamtumfang 130 Seiten

DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE

Anwendungsbeginn

Anwendungsbeginn für die von CENELEC am 2014-09-22 angenommene Europäische Norm als DIN-Norm ist 2015-09-01.

Für DIN EN 61158-5-9:2008-09 besteht eine Übergangsfrist bis 2017-09-22.

Nationales Vorwort

Vorausgegangener Norm-Entwurf: E DIN EN 61158-500:2013-08.

Für dieses Dokument ist das nationale Arbeitsgremium K 956 „Feldbus“ der DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE (www.dke.de) zuständig.

Die enthaltene IEC-Publikation wurde vom SC 65C „Industrial networks“ erarbeitet.

Das IEC-Komitee hat entschieden, dass der Inhalt dieser Publikation bis zu dem Datum (stability date) unverändert bleiben soll, das auf der IEC-Website unter „<http://webstore.iec.ch>“ zu dieser Publikation angegeben ist. Zu diesem Zeitpunkt wird entsprechend der Entscheidung des Komitees die Publikation

- bestätigt,
- zurückgezogen,
- durch eine Folgeausgabe ersetzt oder
- geändert.

Für den Fall einer undatierten Verweisung im normativen Text (Verweisung auf eine Norm ohne Angabe des Ausgabedatums und ohne Hinweis auf eine Abschnittsnummer, eine Tabelle, ein Bild usw.) bezieht sich die Verweisung auf die jeweils neueste gültige Ausgabe der in Bezug genommenen Norm.

Für den Fall einer datierten Verweisung im normativen Text bezieht sich die Verweisung immer auf die in Bezug genommene Ausgabe der Norm.

Der Zusammenhang der zitierten Normen mit den entsprechenden Deutschen Normen ergibt sich, soweit ein Zusammenhang besteht, grundsätzlich über die Nummer der entsprechenden IEC-Publikation. Beispiel: IEC 60068 ist als EN 60068 als Europäische Norm durch CENELEC übernommen und als DIN EN 60068 ins Deutsche Normenwerk aufgenommen.

Das Präsidium des DIN hat mit Präsidialbeschluss 1/2004 festgelegt, dass DIN-Normen, deren Inhalt sich auf internationale Arbeitsergebnisse der Informationsverarbeitung gründet, unter bestimmten Bedingungen allein in englischer Sprache veröffentlicht werden dürfen. Diese Bedingungen sind für die vorliegende Norm erfüllt.

Da sich die Benutzer der vorliegenden Norm der englischen Sprache als Fachsprache bedienen, wird die Englische Fassung der EN 61158-5-9 veröffentlicht. Zu deren Abschnitt 1, der den Anwendungsbereich festlegt, und Abschnitt 3, der die Begriffe festlegt, wurde eine Übersetzung angefertigt und als Nationaler Anhang NA der vorliegenden Norm hinzugefügt. Für viele der verwendeten Begriffe existieren keine gebräuchlichen deutschen Benennungen, da sich die deutschen Anwender in der Regel ebenfalls der englischen Benennungen bedienen. Diese Norm steht nicht in unmittelbarem Zusammenhang mit Rechtsvorschriften und ist nicht als Sicherheitsnorm anzusehen.

Einführung

Ein Feldbus ist ein digitaler serieller Datenbus für die Kommunikation zwischen Geräten der industriellen Automatisierungstechnik wie beispielsweise Messeinrichtungen, Stellgliedern, Reglern und speicherprogrammierbaren Steuerungen. Die Normen der Reihe DIN EN 61158 legen eine Anzahl von Feldbus-Protokolltypen fest. Die vorliegende Norm legt die Dienste des Application Layer für Typ 9 fest.

Änderungen

Gegenüber DIN EN 61158-5-9:2008-09 wurden folgende Änderungen vorgenommen:

- a) Zustandsmaschine zum Download-Dienst berichtigt.

Frühere Ausgaben

DIN V 19245-1: 1988-01
DIN 19245-1: 1991-04
DIN 19245-2: 1991-04
DIN EN 50170/1: 1997-07
DIN EN 50170/2: 1997-07
DIN EN 50170/3: 1997-07
DIN EN 50254: 1999-07
DIN EN 50170/A2: 2000-08
DIN EN 50170/A1: 2002-11
DIN EN 50170/A3: 2002-11
DIN EN 61158-5: 2005-05
DIN EN 61158-5-9: 2008-09

Nationaler Anhang NA (informativ)

1 Anwendungsbereich^{N1)}

1.1 Allgemeines

Der Feldbus-Application Layer (FAL) liefert Benutzerprogramme mit Hilfsmitteln für den Zugriff auf die Feldbuskommunikationsumgebung. In dieser Hinsicht kann der FAL als ein Fenster zwischen zugehörigen Anwendungsprogrammen betrachtet werden.

Diese Norm stellt für Typ 9-Feldbusse maßgebliche allgemeine Elemente für zeitkritischen und nicht zeitkritischen Datentransfer zwischen Anwendungsprogrammen in einer Automatisierungsumgebung bereit. Der Begriff „zeitkritisch“ (en: time-critical) wird verwendet, um die Präsenz eines Zeitfensters darzustellen, in dem eine oder mehrere festgelegte Aktivitäten mit einem vorbestimmten Grad an Gewissheit erledigt werden. Das Scheitern der Erledigung der festgelegten Aktivität innerhalb des Zeitfensters trägt das Risiko des Betriebsausfalls der Anwendung mit und somit das aufgeforderte Eingreifen mit dazugehörigem Risiko für Geräte, Anlage und möglicherweise Menschenleben.

Diese Norm definiert auf eine abstrakte Art und Weise den von außen sichtbaren Dienst, der vom Typ 9 des Feldbus-Application Layer bereitgestellt wird, in Form von:

- a) einem abstrakten Modell für das Beschreiben von Anwendungsressourcen (Objekten), die von Benutzern mittels FAL-Diensten verstellt werden können,
- b) Aktivitäten und Ereignissen der Dienstprimitiven,
- c) den mit den Aktivitäten und Ereignissen der Dienstprimitiven verbundenen Parametern, sowie der Form, die sie einnehmen und
- d) dem Zusammenhang zwischen diesen Aktivitäten und Ereignissen und ihren gültigen Abläufen.

Die Zielsetzung dieser Norm ist es, Dienste zu definieren, die bereitgestellt werden für:

- den FAL-Benutzer an der Grenze zwischen dem Benutzer und dem Application Layer des Feldbus-Referenzmodells und
- das Systemmanagement an der Grenze zwischen dem Application Layer und dem Systemmanagement des Feldbus-Referenzmodells.

Diese Norm beschreibt die Struktur und die Dienste des Typ 9-Feldbus-Application Layer in Übereinstimmung mit dem OSI-Basis-Referenzmodell (ISO/IEC 7498-1) und der OSI-Application Layer-Struktur (ISO/IEC 9545).

FAL-Dienste und Protokolle sind von FAL-Anwendungseinheiten (AE) bereitgestellt, welche innerhalb von Anwendungsprozessen enthalten sind. Die FAL AE besteht aus einer Menge von objektorientierten Anwendungsdienstelementen (ASEs) und Layer-Management-Einheiten (LME), welche die AE verwalten. Die ASEs liefern Kommunikationsdienste, die auf einer Menge von zugehörigen Anwendungsprozessobjekt(APO-) Klassen arbeitet. Eine der FAL ASEs ist eine Management-ASE, welche eine Menge gemeinsamer Dienste für das Management der Instanzen der FAL-Klassen bereitstellt.

Obwohl diese Dienste, aus dem Blickwinkel der Anwendung, beschreiben, wie Anfrage und Antwort ausgegeben und abgegeben werden, beinhalten sie keine Festlegung, was die anfragenden und antwortenden Anwendungen mit diesen vorzunehmen haben. Die Verhaltensaspekte der Anwendungen sind nicht festgelegt, nur eine Definition, welche Anfragen und Antworten diese senden/empfangen können, ist beschrieben. Dies erlaubt den FAL-Benutzern eine größere Flexibilität, um solch ein Objektverhalten zu

^{N1)} Die Nummerierung aus IEC 61158-5-9 wurde beibehalten.

standardisieren. Zusätzlich zu diesen Diensten sind in dieser Norm ebenfalls einige unterstützende Dienste festgelegt, um den Zugriff zum FAL bereitzustellen und somit bestimmte Aspekte des Betriebs zu steuern.

1.2 Spezifikationen

Das hauptsächliche Ziel dieser Norm ist es, für zeitkritische Kommunikation geeignete Eigenschaften der konzeptionellen Application Layer-Dienste festzulegen und folglich die Ergänzung des OSI-Basisreferenzmodells mit der Anleitung der Entwicklung der Application Layer-Protokolle für die zeitkritische Kommunikation.

Ein zweites Ziel ist das Bereitstellen von Migrationspfaden von bisherigen bestehenden industriellen Kommunikationsprotokollen. Es ist dieses letztere Ziel, welches zur Steigerung von unterschiedlichen genormten Diensten, als verschiedene Typen der IEC 61158, führt.

Diese Spezifikation darf als Basis für formelle Application-Programmierschnittstellen verwendet werden. Nichtsdestotrotz ist es keine formelle Programmierschnittstelle, und jede derartige Schnittstelle hat Implementierungsanforderungen zu erfüllen, die nicht von dieser Spezifikation abgedeckt werden und Folgendes beinhalten:

- a) die Größen und Oktett-Folgen von verschiedenen Multi-oktett-Dienstparametern,
- b) die Korrelation von gepaarten Anfrage- und Indikationsprimitiven oder Anweisungs- und Antwortprimitiven.

1.3 Konformität

Diese Norm legt keine individuellen Implementierungen oder Produkte fest, auch beschränkt sie die Implementierungen von Application Layer-Einheiten innerhalb eines industriellen Automatisierungssystems nicht.

Es gibt keine Konformität von Geräten zu dieser Application Layer-Dienstdefinitionsnorm. Stattdessen wird Konformität durch die Implementierung der entsprechenden Application Layer-Protokolle erreicht, welche den Typ 9 der Application Layer-Dienste, wie in dieser Norm beschrieben, erfüllen.

3.5 Begriffe aus IEC 61158-1

Für die Anwendung dieses Dokuments gelten die folgenden Begriffe aus IEC 61158-1.

3.5.1

Anwendung

(en: application)

Funktion oder Datenstruktur, für die Daten konsumiert oder produziert werden

3.5.2

Application Layer-Interoperabilität

(en: application layer interoperability)

Fähigkeit von Anwendungsentitäten, mithilfe der Dienste des FAL koordinierte und kooperative Operationen auszuführen

3.5.3

Anwendungsobjekt

(en: application object)

Objektklasse, die zur Laufzeit für den Nachrichtenaustausch im Netzwerk und innerhalb des Netzwerkgerätes sorgt und diesen verwaltet

Anmerkung 1 zum Begriff: Es können mehrere Typen von Anwendungsobjektklassen definiert werden.

3.5.4

Anwendungsprozess

(en: application process)

Teil einer verteilten Anwendung in einem Netzwerk, der sich auf einem Gerät befindet und eindeutig adressiert wird

3.5.5

Anwendungsprozesskennung

(en: application process identifier)

unterscheidet mehrere in einem Gerät verwendete Anwendungsprozesse

3.5.6

Anwendungsprozessobjekt

(en: application process object)

Komponente eines Anwendungsprozesses, die über eine FAL-Anwendungsbeziehung identifizierbar und zugänglich ist.

Anmerkung 1 zum Begriff: Definitionen von Anwendungsprozessobjekten setzen sich aus einem Satz von Werten für die Attribute von deren Klasse zusammen (siehe Definition von „Anwendungsprozessobjektklasse“). Auf die Definitionen von Anwendungsprozessobjekten kann mithilfe der Dienste des Objektverwaltungs-ASE der FAL von einem entfernten Standort aus zugegriffen werden. Objektverwaltungsdienste des FAL können zum Laden und Aktualisieren von Objektdefinitionen, zum Lesen von Objektdefinitionen und zur dynamischen Erstellung und Löschung von Anwendungsobjekten und deren Definitionen genutzt werden.

3.5.7

Anwendungsprozessobjektklasse

(en: application process object class)

Klasse von Anwendungsprozessobjekten, die in Bezug auf den Satz der über das Netzwerk zugänglichen Attribute und Dienste der Anwendungsprozessobjekte definiert ist

3.5.8

Anwendungsbeziehung

(en: application relationship)

kooperative Verbindung zwischen zwei oder mehr Aufrufen von Anwendungsentitäten zum Zweck des Informationsaustausches und der Koordination des gemeinsamen Betriebs

Anmerkung 1 zum Begriff: Diese Beziehung wird entweder durch den Austausch von Protokolldateneinheiten der Anwendung oder als Ergebnis von Vorkonfigurationsaktivitäten aktiviert.

3.5.9

Anwendungsdienstelement für Anwendungsbeziehungen

(en: application relationship application service element)

Anwendungsdienstelement, das die alleinigen Mittel für den Aufbau und die Beendigung aller Anwendungsbeziehungen bereitstellt

3.5.10

Endpunkt der Anwendungsbeziehung

(en: application relationship endpoint)

Kontext und Verhalten einer Anwendungsbeziehung entsprechend der Sicht und Verwaltung durch einen der Anwendungsprozesse innerhalb der Anwendungsbeziehung

Anmerkung 1 zum Begriff: Jeder Anwendungsprozess, der in die Anwendungsbeziehung eingeschlossen ist, verwaltet seinen eigenen Endpunkt der Anwendungsbeziehung

3.5.11

Attribut

(en: attribute)

Beschreibung eines von außen sichtbaren Merkmals oder einer von außen sichtbaren Funktion eines Objekts. Die Attribute eines Objekts enthalten Informationen zu veränderlichen Teilen eines Objekts

Anmerkung 1 zum Begriff: Sie liefern üblicherweise Statusinformationen oder steuern den Betrieb eines Objekts. Attribute können auch das Verhalten eines Objekts beeinflussen. Attribute werden in Klassenattribute und Instanzattribute unterteilt.

3.5.12

Verhalten

(en: behaviour)

Anzeige dafür, wie das Objekt auf bestimmte Ereignisse reagiert

Anmerkung 1 zum Begriff: Seine Beschreibung schließt die Beziehung zwischen Attributwerten und Diensten ein.

3.5.13

Klasse

(en: class)

Menge von Objekten, die allesamt den gleichen Typ von Systemkomponenten repräsentieren

Anmerkung 1 zum Begriff: Eine Klasse ist eine Verallgemeinerung des Objekts; eine Vorlage für die Festlegung von Variablen und Methoden. Alle Objekte in einer Klasse sind in ihrer Form und in ihrem Verhalten identisch, enthalten aber üblicherweise unterschiedliche Daten in ihren Attributen.

3.5.14

Klassenattribute

(en: class attributes)

Attribut, welches allen Objekten innerhalb derselben Klasse gemeinsam ist

3.5.15

Klassencode

(en: class code)

eindeutige Kennung, die jeder Objektklasse zugewiesen ist

3.5.16

klassenspezifischer Dienst

(en: class specific service)

durch eine bestimmte Objektklasse definierter Dienst, um eine erforderliche Funktion auszuführen, welche nicht als allgemeiner Dienst ausgeführt wird

Anmerkung 1 zum Begriff: Ein klassenspezifisches Objekt ist für die Objektklasse, welche es definiert, eindeutig.

3.5.17

Client

(en: client)

- (a) Objekt, das die Dienste eines anderen (Server-)Objekts zur Ausführung einer Aufgabe nutzt:
- (b) Initiator einer Nachricht, auf die ein Server reagiert, wie die Rolle eines AR-Endpunktes, in der dieser bestätigte Dienstanforderungs-APDUs an einen einzelnen AR-Endpunkt, welcher als Server agiert, ausgibt

3.5.18

Übertragungspfad

(en: conveyance path)

unidirektionaler Strom von APDUs in einer Anwendungsbeziehung

3.5.19

zyklisch

(en: cyclic)

Ereignisse, die sich regelmäßig wiederholen

3.5.20

zugeordnete AR

(en: dedicated AR)

Anwendungsbeziehung, die direkt vom FAL-Benutzer verwendet wird

Anmerkung 1 zum Begriff: Bei zugeordneten ARs werden nur der FAL-Header und die Benutzerdaten übertragen.

3.5.21

Gerät

(en: device)

physische Hardware-Verbindung zum Verbindungszweig

Anmerkung 1 zum Begriff: Ein Gerät kann mehr als einen Knoten umfassen.

3.5.22

Geräteprofil

(en: device profile)

Zusammenstellung geräteabhängiger Informationen und Funktionalitäten, um Konsistenz zwischen ähnlichen Geräten desselben Gerätetyps herzustellen

3.5.23

dynamische AR

(en: dynamic AR)

Anwendungsbeziehung, die nur unter Verwendung der Prozeduren zum Aufbau von ARs in einen aktiven Zustand versetzt werden kann

3.5.24

Endpunkt

(en: endpoint)

eine der Daten übermittelnden Entitäten, die an einer Verbindung beteiligt ist

3.5.25

Fehler

(en: error)

Nichtübereinstimmung zwischen einem berechneten, beobachteten oder gemessenen Wert oder Zustand und dem festgelegten oder theoretisch richtigen Wert oder Zustand

3.5.26

Fehlerklasse

(en: error class)

allgemeine Gruppierung für Fehlerdefinitionen

Anmerkung 1 zum Begriff: Innerhalb einer Fehlerklasse sind Fehlercodes für spezifische Fehler festgelegt.

3.5.27

Fehlercode

(en: error code)

Kennzeichnung eines bestimmten Fehlertyps innerhalb einer Fehlerklasse

3.5.28

FAL-Subnetz

(en: FAL subnet)

Netzwerke, die aus einem oder mehreren Data Link-Segmenten bestehen

Anmerkung 1 zum Begriff: Sie dürfen Bridges, jedoch keine Router umfassen. FAL-Subnetze werden durch eine Untermenge der Netzwerkadresse identifiziert.

3.5.29

logisches Gerät

(en: logical device)

legt eine bestimmte FAL-Klasse fest, die eine Softwarekomponente oder eine Firmwarekomponente als eigenständige Einrichtung eines Automatisierungsgerätes abstrahiert

3.5.30

Verwaltungsinformation

(en: management information)

über das Netzwerk zugängliche Information, die die Verwaltung des Betriebs des Feldbussystems einschließlich des Application Layer unterstützt

Anmerkung 1 zum Begriff: Zur Verwaltung gehören Funktionen wie die Steuerung, die Überwachung und die Diagnose.

3.5.31

Netzwerk

(en: network)

Reihe von Knoten, die über ein bestimmtes Kommunikationsmedium miteinander verbunden sind

Anmerkung 1 zum Begriff: Die Verbindungspfade zwischen beliebigen Paaren von Knoten können Repeater, Router und Gateways einschließen.

3.5.32

Peer

(en: peer)

Rolle eines AR-Endpunktes, in der dieser in der Lage ist, sowohl als Client als auch als Server zu agieren

3.5.33

vordefinierter AR-Endpunkt

(en: pre-defined AR endpoint)

AR-Endpunkt, der lokal innerhalb eines Gerätes ohne Nutzung des Erstellungsdienstes festgelegt ist

Anmerkung 1 zum Begriff: Vordefinierte ARs, die nicht vorher aktiviert sind, werden vor ihrer Nutzung aktiviert.

3.5.34

vorher aktivierter AR-Endpunkt

(en: pre-established AR endpoint)

AR-Endpunkt, der während der Konfiguration der AEs, die ihre Endpunkte steuern, in einen aktiven Zustand versetzt wird

3.5.35

Publisher

(en: publisher)

Rolle eines AR-Endpunktes, in der dieser APDUs auf den Feldbus für den Verbrauch durch einen oder mehrere Subscriber überträgt

Anmerkung 1 zum Begriff: Der Publisher muss die Identität oder Anzahl der Subscriber nicht kennen und kann seine APDUs mithilfe einer zugeordneten AR veröffentlichen. In dieser Norm sind zwei Typen von Publishern festgelegt, die jeweils einzeln definiert sind: Pull-Publisher und Push-Publisher.

3.5.36

Server

(en: server)

- a) Rolle eines AREP, in der dieser eine bestätigte Dienstantwort-APDU an den Client, der die Anforderung initiiert hat, zurückgibt;
- b) Objekt, das einem anderen (Client-)Objekt Dienste bereitstellt

3.5.37

Dienst

(en: service)

Operation oder Funktion, die ein Objekt und/oder eine Objektklasse auf Anforderung eines anderen Objekts und/oder einer anderen Objektklasse hin ausführt

Anmerkung 1 zum Begriff: Es ist eine Reihe von allgemeinen Diensten definiert, und es sind Maßnahmen für die Festlegung objektspezifischer Dienste vorgesehen. Objektspezifische Dienste sind Dienste, die durch eine bestimmte Objektklasse festgelegt sind, um eine erforderliche Funktion auszuführen, die nicht von einem allgemeinen Dienst ausgeführt wird.

3.5.38

Subscriber

(en: subscriber)

Rolle eines AREP, in der dieser von einem Publisher produzierte APDUs empfängt

Anmerkung 1 zum Begriff: In dieser Norm sind zwei Typen von Subscribern festgelegt, die jeweils einzeln definiert sind: Pull-Subscriber und Push-Subscriber.

3.6 Spezielle Begriffe für den Feldbus-Application Layer Typ 9

Für diesen Teil sind keine zusätzlichen Begriffe definiert.

English Version

**Industrial communication networks - Fieldbus specifications -
Part 5-9: Application layer service definition - Type 9 elements
(IEC 61158-5-9:2014)**

Réseaux de communication industriels - Spécifications des
bus de terrain - Partie 5-9: Définition des services de la
couche application - Éléments de type 9
(CEI 61158-5-9:2014)

Industrielle Kommunikationsnetze - Feldbusse -
Teil 5-9: Dienstfestlegungen des Application Layer
(Anwendungsschicht) - Typ 9-Elemente
(IEC 61158-5-9:2014)

This European Standard was approved by CENELEC on 2014-09-22. CENELEC members are bound to comply with the CEN/CENELEC Internal Regulations which stipulate the conditions for giving this European Standard the status of a national standard without any alteration.

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

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

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



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

CEN-CENELEC Management Centre: Avenue Marnix 17, B-1000 Brussels

Foreword

The text of document 65C/763/FDIS, future edition 2 of IEC 61158-5-9, prepared by SC 65C "Industrial networks" of IEC/TC 65 "Industrial-process measurement, control and automation" was submitted to the IEC-CENELEC parallel vote and approved by CENELEC as EN 61158-5-9:2014.

The following dates are fixed:

- latest date by which the document has to be (dop) 2015-06-22
implemented at national level by
publication of an identical national
standard or by endorsement
- latest date by which the national (dow) 2017-09-22
standards conflicting with the
document have to be withdrawn

This document supersedes EN 61158-5-9:2008.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CENELEC [and/or CEN] shall not be held responsible for identifying any or all such patent rights.

This document has been prepared under a mandate given to CENELEC by the European Commission and the European Free Trade Association.

Endorsement notice

The text of the International Standard IEC 61158-5-9:2014 was approved by CENELEC as a European Standard without any modification.

In the official version, for Bibliography, the following notes have to be added for the standards indicated:

IEC 61158-3-1	NOTE	Harmonized as EN 61158-3-1.
IEC 61158-4-1	NOTE	Harmonized as EN 61158-4-1.
IEC 61158-5-5	NOTE	Harmonized as EN 61158-5-5.
IEC 61158-6-9	NOTE	Harmonized as EN 61158-6-9.
IEC 61784-1	NOTE	Harmonized as EN 61784-1.
IEC 61784-2	NOTE	Harmonized as EN 61784-2.

CONTENTS

INTRODUCTION.....	6
1 Scope	7
1.1 General.....	7
1.2 Specifications	8
1.3 Conformance	8
2 Normative references	8
3 Terms, definitions, symbols, abbreviations and conventions	9
3.1 ISO/IEC 7498-1 terms.....	9
3.2 ISO/IEC 8822 terms.....	9
3.3 ISO/IEC 9545 terms.....	9
3.4 ISO/IEC 8824-1 terms.....	9
3.5 IEC 61158-1 terms.....	9
3.6 Type 9 fieldbus application-layer specific definitions	13
3.7 Abbreviations and symbols	13
3.8 Conventions	14
4 Concepts.....	17
5 Data type ASE.....	17
5.1 Overview	17
5.2 Formal definition of data type objects	19
5.3 FAL defined data types	21
5.4 Data type ASE service specification	24
5.5 Summary of data types	24
6 Communication model specification	25
6.1 Concepts.....	25
6.2 Common parameters	25
6.3 ASEs.....	25
6.4 ARs.....	113
6.5 Summary of classes	116
6.6 Permitted services by AREP role.....	117
Bibliography	119
Annex ZA (normative) Normative references to international publications with their corresponding European publications.....	120
Figure 1 – Data type class hierarchy	18
Figure 2 – VFD model	26
Figure 3 – Abstract model of an automation system (VFD).....	26
Figure 4 – Source OD/remote OD.....	32
Figure 5 – Put OD state machine.....	45
Figure 6 – Transaction object state machine.....	52
Figure 7 – Context test of two features-supported with different bitstring length.....	59
Figure 8 – Overview of event	79
Figure 9 – Event state machine	84
Figure 10 – Domain genericdownload/download state machine (server)	98
Figure 11 – Domain upload state machine (server)	100

Figure 12 – State diagram.....	111
Table 1 – Data type summary	25
Table 2 – Logical status.....	27
Table 3 – Status	28
Table 4 – Unsolicited status	29
Table 5 – Identify.....	30
Table 6 – Structure of the object dictionary	33
Table 7 – Structure of the static list of types	33
Table 8 – Structure of the static object dictionary	34
Table 9 – Structure of the dynamic list of variable lists	34
Table 10 – Structure of the dynamic list of program invocations.....	35
Table 11 – Empty object dictionary.....	38
Table 12 – Get OD service parameters	41
Table 13 – Initiate put OD service parameters	43
Table 14 – Put OD service parameters	44
Table 15 – Terminate put OD service parameters	45
Table 16 – Put OD state transitions.....	47
Table 17 – Attribute FMS features supported.....	49
Table 18 – Transaction object state transitions.....	52
Table 19 – Initiate service parameters.....	53
Table 20 – Failure reasons	54
Table 21 – Abort service parameters.....	55
Table 22 – User abort reasons	56
Table 23 – APO ASE abort reasons	56
Table 24 – Reject service parameters	57
Table 25 – Reject APDU reasons	57
Table 26 – Compatibility of the local context to the remote context	58
Table 27 – Unconfirmed send service parameters	61
Table 28 – Confirmed send service parameters	62
Table 29 – AR-Abort service parameters	62
Table 30 – Compel service parameters	63
Table 31 – Get buffered message service parameters	64
Table 32 – AR-Status service parameters	65
Table 33 – Simple variable access group membership.....	66
Table 34 – Simple variable access rights membership	66
Table 35 – Array variable access group membership	68
Table 36 – Array variable access rights membership	68
Table 37 – Variable list access group membership	70
Table 38 – Variable list access rights membership	71
Table 39 – Read service parameters	74
Table 40 – Write service parameters	75

Table 41 – Information report service parameters	75
Table 42 – Define variable list service parameters	77
Table 43 – Delete variable list service parameters	78
Table 44 – Event access group membership.....	80
Table 45 – Event access rights membership	80
Table 46 – Event notification service parameters	81
Table 47 – Acknowledge event notification service parameters	82
Table 48 – Alter event condition monitoring service parameters	83
Table 49 – Event state transitions	84
Table 50 – Domain access group membership.....	86
Table 51 – Domain access rights membership	86
Table 52 – GenericInitiateDownloadSequence	88
Table 53 – GenericDownloadSegment.....	88
Table 54 – GenericTerminateDownloadSequence	89
Table 55 – InitiateDownloadSequence.....	90
Table 56 – DownloadSegment.....	91
Table 57 – TerminateDownloadSequence.....	92
Table 58 – RequestDomainDownload	93
Table 59 – InitiateUploadSequence	94
Table 60 – UploadSegment	95
Table 61 – TerminateUploadSequence	96
Table 62 – RequestDomainUpload	97
Table 63 – Domain genericDownload/download state machine (server).....	98
Table 64 – Domain upload state machine (server)	100
Table 65 – Program invocation access group membership	102
Table 66 – Program invocation access group membership	102
Table 67 – Create program invocation service parameters	104
Table 68 – Delete program invocation service parameters	105
Table 69 – Start service parameters	106
Table 70 – Stop service parameters	107
Table 71 – Resume service parameters.....	108
Table 72 – Reset service parameters	109
Table 73 – Kill service parameters	110
Table 74 – Program invocation state machine.....	112
Table 75 – Class summary.....	117
Table 76 – Services by AREP role.....	117

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:

FAL ASE:	ASE Name
CLASS:	Class Name
CLASS ID:	#
PARENT CLASS:	Parent Class Name
ATTRIBUTES:	

1	(o)	Key Attribute:	numeric identifier
2	(o)	Key Attribute:	name
3	(m)	Attribute:	attribute name(values)
4	(m)	Attribute:	attribute name(values)
4.1	(s)	Attribute:	attribute name(values)
4.2	(s)	Attribute:	attribute name(values)
4.3	(s)	Attribute:	attribute name(values)
5.	(c)	Constraint:	constraint expression
5.1	(m)	Attribute:	attribute name(values)
5.2	(o)	Attribute:	attribute name(values)
6	(m)	Attribute:	attribute name(values)
6.1	(s)	Attribute:	attribute name(values)
6.2	(s)	Attribute:	attribute name(values)

SERVICES:

1	(o)	OpsService:	service name
2.	(c)	Constraint:	constraint expression
2.1	(o)	OpsService:	service name
3	(m)	MgtService:	service name

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

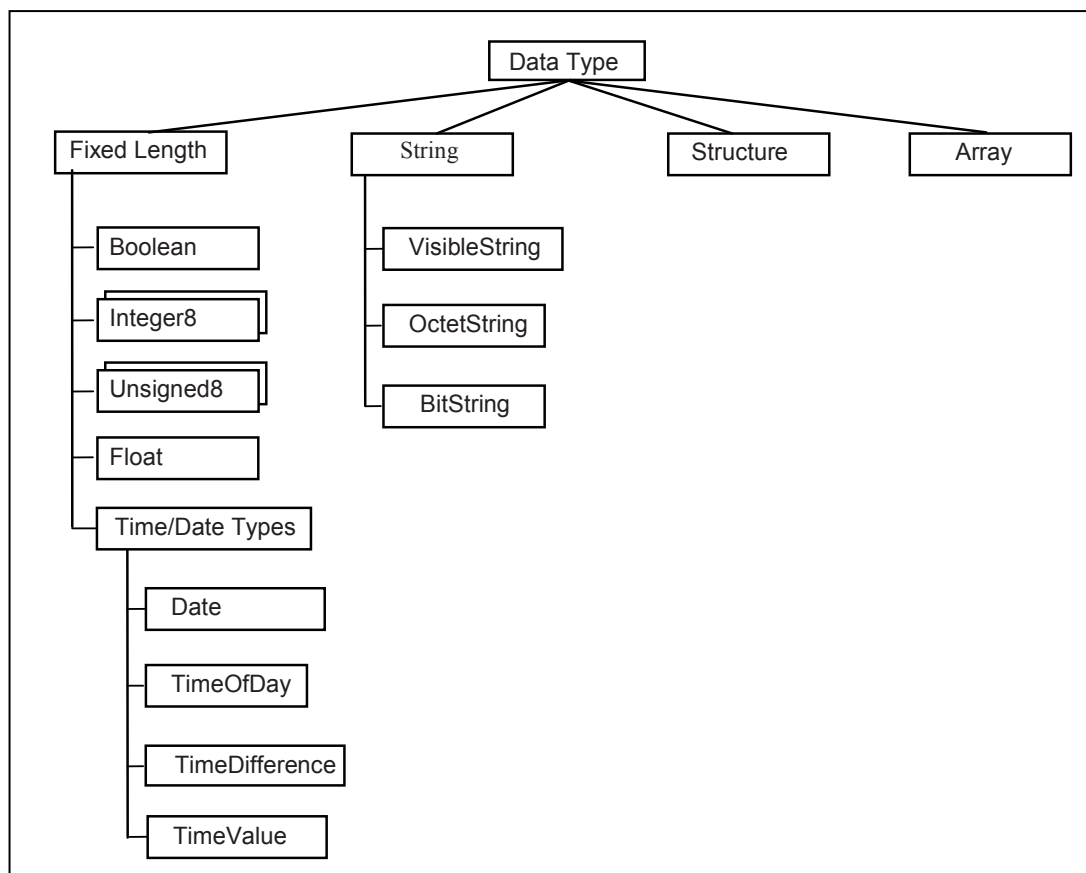


Figure 1 – Data type class hierarchy

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
CLASS: DATA TYPE

CLASS ID:		5 (FIXED LENGTH & STRING), 6 (STRUCTURE), 12 (ARRAY)
PARENT CLASS:		TOP
ATTRIBUTES:		
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
ATTRIBUTES:	
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
ATTRIBUTES:	
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

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.

5.3.2 String types

5.3.2.1 VisibleString

CLASS: Data type

ATTRIBUTES:

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

CLASS: Data type

ATTRIBUTES:

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

CLASS: Data type

ATTRIBUTES:

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.

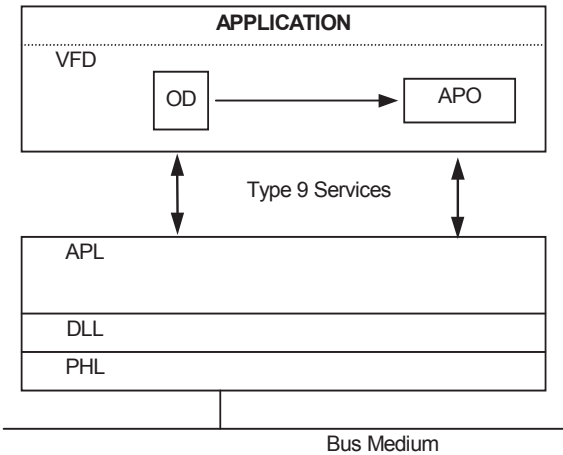


Figure 2 – VFD model

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.

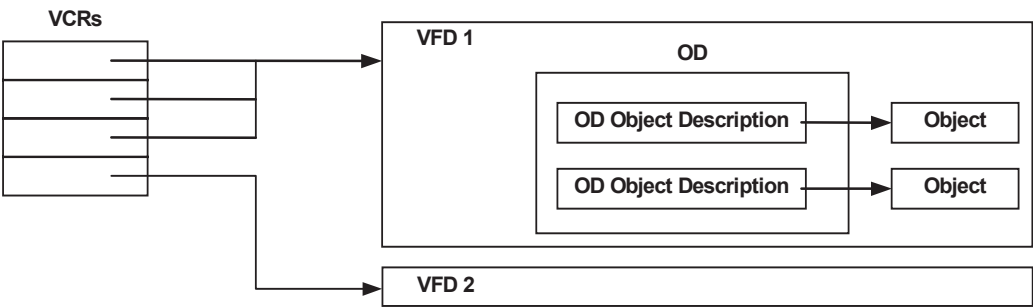


Figure 3 – Abstract model of an automation system (VFD)

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.

ASE:	VFD ASE
CLASS:	VFD
CLASS ID:	—
PARENT CLASS:	TOP
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	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: <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