

INTERNATIONAL STANDARD

NORME INTERNATIONALE



**Field device integration (FDI) –
Part 7: Communication devices**

**Intégration des appareils de terrain (FDI) –
Partie 7: Appareils de communication**

IECNORM.COM : Click to view the full PDF of IEC 62769-7:2021



THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2021 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
info@iec.ch
www.iec.ch

About the IEC

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

About IEC publications

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

IEC publications search - webstore.iec.ch/advsearchform

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 It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished

Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and once a month by email.

IEC Customer Service Centre - webstore.iec.ch/csc

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

IEC online collection - oc.iec.ch

Discover our powerful search engine and read freely all the publications previews. With a subscription you will always have access to up to date content tailored to your needs.

Electropedia - www.electropedia.org

The world's leading online dictionary on electrotechnology, containing more than 22 000 terminological entries in English and French, with equivalent terms in 18 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

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.

Recherche de publications IEC -

webstore.iec.ch/advsearchform

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

IEC Just Published - webstore.iec.ch/justpublished

Restez informé sur les nouvelles publications IEC. Just Published détaille les nouvelles publications parues. Disponible en ligne et une fois par mois par email.

Service Clients - webstore.iec.ch/csc

Si vous désirez nous donner des commentaires sur cette publication ou si vous avez des questions contactez-nous: sales@iec.ch.

IEC online collection - oc.iec.ch

Découvrez notre puissant moteur de recherche et consultez gratuitement tous les aperçus des publications. Avec un abonnement, vous aurez toujours accès à un contenu à jour adapté à vos besoins.

Electropedia - www.electropedia.org

Le premier dictionnaire d'électrotechnologie en ligne au monde, avec plus de 22 000 articles terminologiques en anglais et en français, ainsi que les termes équivalents dans 16 langues additionnelles. Egalement appelé Vocabulaire Electrotechnique International (IEV) en ligne.

INTERNATIONAL STANDARD

NORME INTERNATIONALE



**Field device integration (FDI) –
Part 7: Communication devices**

**Intégration des appareils de terrain (FDI) –
Partie 7: Appareils de communication**

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

COMMISSION
ELECTROTECHNIQUE
INTERNATIONALE

ICS 25.040.40; 35.100.05

ISBN 978-2-8322-9314-0

**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	6
INTRODUCTION	8
1 Scope	9
2 Normative references	10
3 Terms, definitions, abbreviated terms and conventions	10
3.1 Terms and definitions	10
3.2 Abbreviated terms	10
3.3 Conventions	11
4 General	11
5 FDI Communication Package	13
5.1 General	13
5.2 EDD	13
5.2.1 General rules	13
5.2.2 Device component	14
5.2.3 CommunicationDevice component	15
5.2.4 Communication service provider component	16
5.2.5 Connection Point component	17
5.2.6 Connection Point collection	18
5.2.7 Network component	18
5.2.8 ValidateNetwork	20
5.2.9 ValidateModules	20
5.2.10 UIP specifics	21
5.2.11 Deployment	21
6 Communication relations	21
7 FDI Communication Server definition	23
7.1 General	23
7.2 General characteristics	23
7.3 Information Model	23
7.3.1 General	23
7.3.2 CommunicationServerType	26
7.3.3 ServerCommunicationDeviceType	30
7.3.4 ServerCommunicationServiceType	33
7.4 OPC UA Server Profile for FDI Communication Server	37
7.5 Mapping the FDI Server IM to the FDI Communication Server IM	38
7.5.1 General	38
7.5.2 Information Model differences	38
7.6 Installer	39
7.7 FDI Communication Package	39
7.7.1 General	39
7.7.2 EDD for Lightweight Communication Server	40
7.7.3 EDD for Multi-Channel Communication Server	40
7.7.4 COMMANDs in EDDs for FDI Communication Servers	41
7.7.5 Documentation	41
7.8 Handling and behavior	41
7.8.1 General	41

7.8.2	Deployment	42
7.8.3	Server configuration	42
7.8.4	Start up	43
7.8.5	Shutdown	43
7.8.6	Watchdog	43
7.8.7	Establish the OPC UA connection	43
7.8.8	Instantiate the Communication Server	44
7.8.9	Configure the communication hardware	44
7.8.10	Configuring the Network	44
7.8.11	Parameterize	44
7.8.12	Initialize	44
7.8.13	Create the communication service object	45
7.8.14	Communication relation	45
7.8.15	Connect	45
7.8.16	Disconnect	46
7.8.17	Abort Indication	46
7.8.18	Scan	46
7.8.19	SetAddress	46
8	FDI Communication Gateway definition	46
8.1	General	46
8.2	Information Model	46
8.2.1	General	46
8.2.2	CommunicationGatewayType	47
8.2.3	GatewayCommunicationDeviceType	48
8.2.4	GatewayCommunicationServiceType	51
8.3	FDI Communication Package	55
8.3.1	General	55
8.3.2	EDD	56
8.4	Handling and behavior	57
8.4.1	General	57
8.4.2	Deployment	58
8.4.3	Start up	58
8.4.4	Configure the communication hardware	58
8.4.5	Configure the Network	58
8.4.6	Parameterize	58
8.4.7	Communication relation	59
8.4.8	Connect	59
8.4.9	Disconnect	59
8.4.10	Abort indication	59
8.4.11	Scan	59
8.4.12	Communication Error Handling	60
Annex A	(informative) Layered protocols	61
A.1	General	61
A.2	Convention for protocol-specific annex creation	61
A.3	FDI Communication Package definition	62
A.3.1	Communication services	62
A.3.2	Connection Point	62
A.3.3	Network	62
A.4	Representation in the IM	63

Annex B (normative) Namespace and mappings	64
Figure 1 – FDI architecture diagram	9
Figure 2 – FDI communication infrastructure architecture	12
Figure 3 – Communication relation	22
Figure 4 – Communication relation state chart	23
Figure 5 – FDI Communication Server AddressSpace	25
Figure 6 – CommunicationServerType	26
Figure 7 – ServerCommunicationDeviceType	30
Figure 8 – ServerCommunicationServiceType	34
Figure 9 – Information Model differences (example)	38
Figure 10 – FDI Communication Server state machine	42
Figure 11 – Communication relation state chart	45
Figure 12 – Gateway information model	47
Figure 13 – CommunicationGatewayType	48
Figure 14 – GatewayCommunicationDeviceType	49
Figure 15 – GatewayCommunicationServiceType	52
Figure 16 – Nested Communication	58
Table 1 – ValidateNetwork Action arguments	20
Table 2 – ValidateModules Action arguments	21
Table 3 – CommunicationServerType definition	26
Table 4 – MethodSet of CommunicationServerType	26
Table 5 – Reset Method arguments	27
Table 6 – Reset Method AddressSpace definition	27
Table 7 – Initialize Method arguments	28
Table 8 – Initialize Method AddressSpace definition	28
Table 9 – AddComponent Method arguments	29
Table 10 – AddComponent Method AddressSpace definition	29
Table 11 – RemoveComponent Method arguments	30
Table 12 – RemoveComponent Method AddressSpace definition	30
Table 13 – ServerCommunicationDeviceType definition	31
Table 14 – MethodSet of ServerCommunicationDeviceType	31
Table 15 – Scan Method arguments	32
Table 16 – Scan Method AddressSpace definition	32
Table 17 – ResetScan Method arguments	32
Table 18 – ResetScan Method AddressSpace definition	33
Table 19 – SetAddress Method arguments	33
Table 20 – ServerCommunicationServiceType definition	34
Table 21 – MethodSet of ServerCommunicationServiceType	34
Table 22 – Connect Method arguments	35
Table 23 – Disconnect Method arguments	36
Table 24 – Transfer Method arguments	36

Table 25 – GetPublishedData Method arguments.....	37
Table 26 – FDICommunicationServer_Facet definition	37
Table 27 – CommunicationGatewayType definition	48
Table 28 – GatewayCommunicationDeviceType definition.....	49
Table 29 – MethodSet of GatewayCommunicationDeviceType	49
Table 30 – Scan Method arguments	50
Table 31 – Scan Method AddressSpace definition.....	50
Table 32 – ScanNext Method arguments.....	51
Table 33 – ScanNext Method AddressSpace definition	51
Table 34 – GatewayCommunicationServiceType definition.....	52
Table 35 – MethodSet of GatewayCommunicationServiceType	53
Table 36 – Connect Method arguments.....	54
Table 37 – Transfer Method arguments.....	55

IECNORM.COM : Click to view the full PDF of IEC 62769-7:2021

INTERNATIONAL ELECTROTECHNICAL COMMISSION

FIELD DEVICE INTEGRATION (FDI) –

Part 7: Communication devices

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.

International Standard IEC 62769-7 has been prepared by subcommittee 65E: Devices and integration in enterprise systems, of IEC technical committee 65: Industrial-process measurement, control and automation.

This second edition cancels and replaces the first edition published in 2015. This edition constitutes a technical revision.

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

- a) support for generic protocol extension for faster adoption of other technologies;
- b) support of new protocols;
- c) generic protocol extension to allow adoption of other communication protocols;
- d) based on generic protocol extension: Modbus RTU.

The text of this International Standard is based on the following documents:

FDIS	Report on voting
65E/764/FDIS	65E/774/RVD

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

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

A list of all parts in the 62769 series, published under the general title *Field Device Integration (FDI)*, can be found on the IEC website.

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

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

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

INTRODUCTION

The IEC 62769 series has the general title *Field Device Integration (FDI)* and the following parts:

- Part 1: Overview
- Part 2: FDI Client
- Part 3: FDI Server
- Part 4: FDI Packages
- Part 5: FDI Information Model
- Part 6: FDI Technology Mapping
- Part 7: FDI Communication Devices
- Part 100: Profiles – Generic Protocol Extensions
- Part 101-1: Profiles – Foundation Fieldbus H1
- Part 101-2: Profiles – Foundation Fieldbus HSE
- Part 103-1: Profiles – PROFIBUS
- Part 103-4: Profiles – PROFINET
- Part 109-1: Profiles – HART and WirelessHART
- Part 115-2: Profiles – Protocol-specific Definitions for Modbus RTU
- Part 150-1: Profiles – ISA 100.11a

IECNORM.COM : Click to view the full PDF of IEC 62769-7:2021

FIELD DEVICE INTEGRATION (FDI) –

Part 7: Communication devices

1 Scope

This part of IEC 62769 specifies the elements implementing communication capabilities called Communication Devices (IEC 62769-5).

The overall FDI architecture is illustrated in Figure 1. The architectural components that are within the scope of this document have been highlighted in this illustration. The document scope with respect to FDI Packages is limited to Communication Devices. The Communication Server shown in Figure 1 is an example of a specific Communication Device.

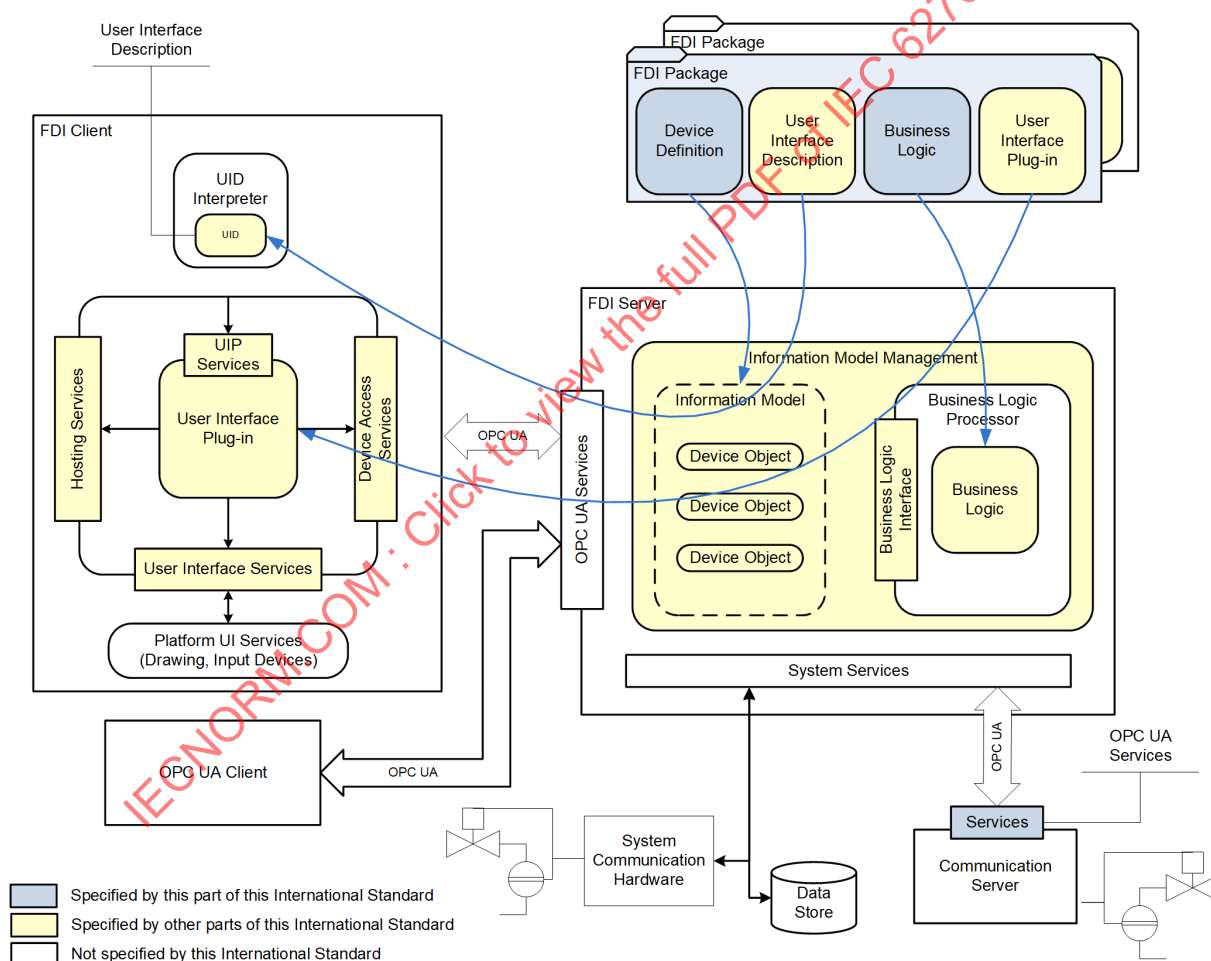


Figure 1 – FDI architecture diagram

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.

IEC 61804-3, *Function blocks (FB) for process control and Electronic Device Description Language (EDDL) – Part 3: EDDL syntax and semantics*

IEC 61804-4, *Function blocks (FB) for process control and Electronic Device Description Language (EDDL) – Part 4: EDD interpretation*

IEC 62541 (all parts), *OPC Unified Architecture*

IEC 62769-1, *Field Device Integration (FDI) – Part 1: Overview*

IEC 62769-2, *Field Device Integration (FDI) – Part 2: FDI Client*

IEC 62769-3, *Field Device Integration (FDI) – Part 3: FDI Server*

IEC 62769-4:2020, *Field Device Integration (FDI) – Part 4: FDI Packages*

IEC 62769-5, *Field Device Integration (FDI) – Part 5: FDI Information Model*

3 Terms, definitions, abbreviated terms and conventions

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 62769-1 as well as the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at <http://www.electropedia.org/>
- ISO Online browsing platform: available at <http://www.iso.org/obp>

3.1.1

gateway

communication device that enables to bridge between different physical networks or different protocols

3.2 Abbreviated terms

For the purposes of this document, the abbreviated terms given in IEC 62769-1 and the following apply.

HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
PHY	Physical communication hardware
SNMP	Simple Network Management Protocol
TCP	Transmission Control Protocol
URI	Uniform Resource Identifier

3.3 Conventions

For the purposes of this document, the conventions given in IEC 62769-1 apply.

This document uses the graphical notation defined in IEC 62769-5.

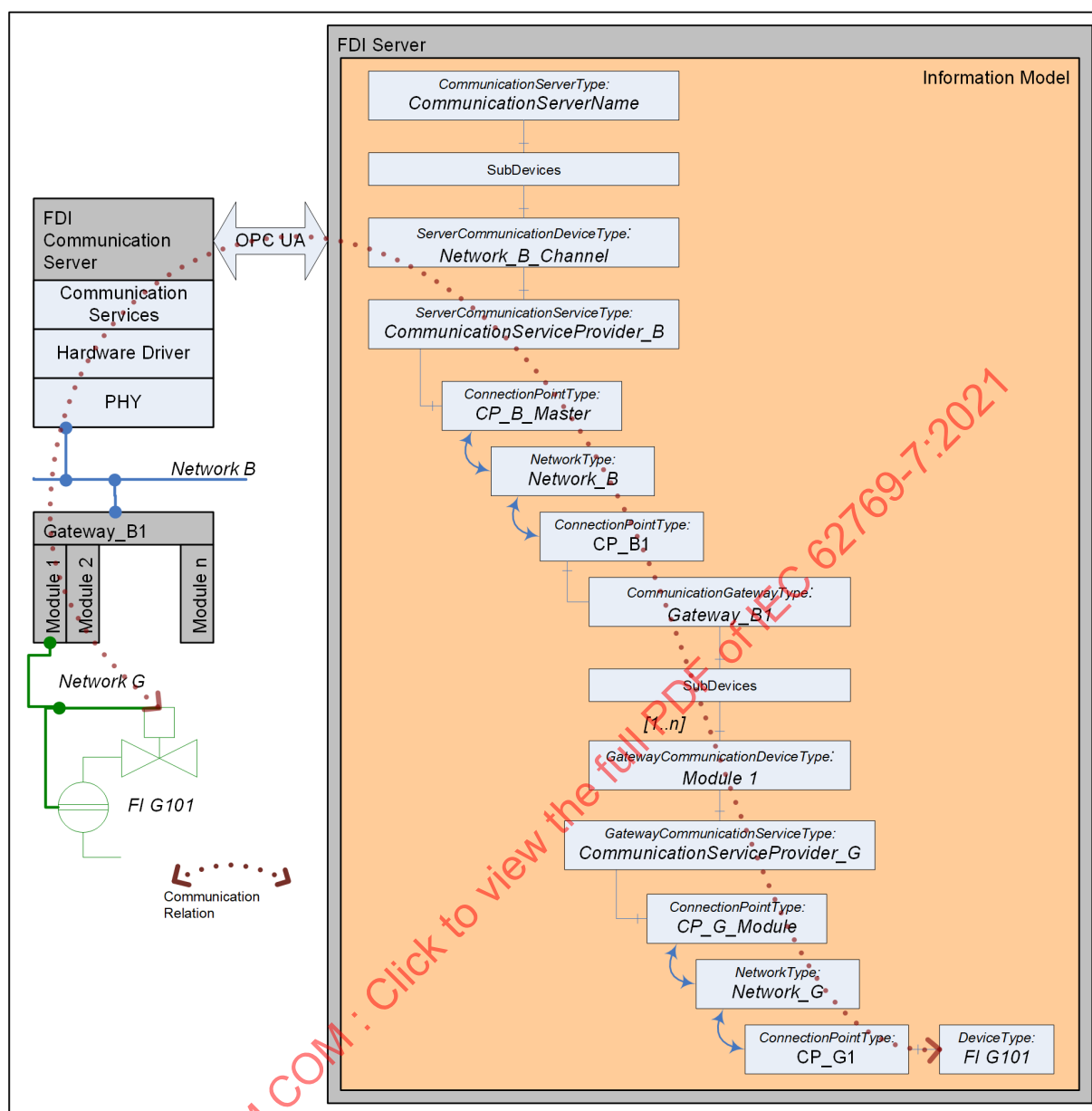
4 General

The abstract term "FDI Communication Device" represents an entity implementing communication functions over a network using a specific protocol. The group of FDI Communication Devices splits into two main groups.

- a) The FDI Communication Server is a dedicated OPC UA Server providing access to one or more field device networks. The FDI Communication Server is specified in Clause 7.
- b) The FDI Communication Gateway enables to bridge between different physical networks or different protocols. The bridging business logic is implemented in the EDD component that is provided with an FDI Communication Package. The FDI Communication Gateway is specified in Clause 8.

NOTE The main differences between a Gateway and a Communication Server are: in terms of FDI, the FDI Communication Server is a dedicated OPC UA Server providing access to one or more field device networks. A Gateway is a communication device that enables to bridge between different physical networks or different protocols. The logical representation of a Gateway device within the FDI Server hosted Information Model enables the FDI Server to process communication in heterogeneous network topologies.

IECNORM.COM : Click to view the full PDF of IEC 62769-7:2021



IEC

Figure 2 – FDI communication infrastructure architecture

The FDI Server hosted Information Model contains a representation of the network topology. (see also IEC 62769-5). The Information Model shown in Figure 2 is an example excerpt to illustrate how the used elements of the Information Model reflect the actual network topology.

- 1) The instance of CommunicationServerType (named CommunicationServerName) represents the FDI Communication Server. The FDI Communication Server implements physical communication network access (Communication hardware). Clause 7 describes related Information Model specifics, required FDI Communication Package content and handling of elements therein. For sub-devices, see IEC 62769-5.
- 2) The instance of ServerCommunicationDeviceType and ServerCommunication-ServiceType (named Network_B_Channel) maps to the FDI Communication Server implemented communication services. The ServerCommunicationDeviceType is specified in 7.3.3. The ServerCommunicationServiceType is specified in 7.3.4.

- 3) The instance of `CommunicationGatewayType` (named `Gateway_B1`) represents the physical Gateway. Clause 8 describes the related Information Model specifics, the required FDI Package content and the handling of elements therein.
- 4) The instance of `GatewayCommunicationDeviceType` (named `Module 1`) maps to a physical or logical module enabling communication to the network to which this module is connected. The `GatewayCommunicationDeviceType` is specified in 8.3.2.3. The related Gateway specifics are described in Clause 8.
- 5) The instance of `GatewayCommunicationServiceType` (named `CommunicationServiceProvider_G`) represents the Gateways' ability to process communication services. The Gateway-specific implementation of `GatewayCommunicationServiceType` is based on Business Logic that enables to run communication services in heterogeneous communication networks.
- 6) A communication relation (more details are described in Clause 6) between a physical device and the device representation managed by the FDI Server is always associated to communication service objects that are instances of a `GatewayCommunicationServiceType` or `ServerCommunicationServiceType`. The ability of instantiating multiple communication service objects supports protocols enables to operate multiple logical connections between a bus master and a device.
- 7) The Information Model represents the connections between the physical devices shown on the left side of Figure 2 based on instances of `ConnectionPointType` `NetworkType` and the depicted relations. `ConnectionPointType` and `NetworkType` are specified in IEC 62769-5.

5 FDI Communication Package

5.1 General

The FDI Server imports the FDI Communication Package like any other FDI Device Package. Clause 5 specifies the FDI Communication Package details.

5.2 EDD

5.2.1 General rules

The FDI Communication Package contained EDD is not restricted, but bound to a protocol-specific annex (IEC 62769-4:2020, Annex F).

The EDD elements as specified in IEC 62769-4:2020, Annex F, and provided with an FDI Communication Package shall describe:

- a) Parameter and parameter structures. Mandatory protocol-specific parameter definitions are found in IEC 62769-4:2020, Annex F. The parameters shall contain any parameter that requires adjustment for proper communication service operation.
- b) Physical Layer identification. Protocol-specific definitions are found in IEC 62769-4:2020, Annex F.

Communication devices' modularity: The modularity information shall be based on using the EDDL constructs `COMPONENT` (see IEC 61804-3).

FDI envisions communication device modularity to cope with communication hardware providing multiple physical or logical communication channels to access multiple logical or physical communication networks. Each module element of the whole communication device shall be described by a separate EDD element.

- c) The `COMPONENT` definition shall be used to support the system implemented topology configuration. Protocol-specific definitions are found in IEC 62769-4:2020, Annex F. The related `COMPONENT` definitions are described in 5.2.2, 5.2.3, 5.2.4, and 5.2.7.
- d) The Business Logic shall contain a method enabled to validate the network (see 5.2.8). The validation function considers the elements only directly connected to the network. The validation function shall be referred by the EDDL specified `CHECK_CONFIGURATION` attribute.

- e) The Business Logic can contain a method enabled to validate the module configuration (see 5.2.9) or the network configuration (see 5.2.8). The validation function considers the elements only directly connected to the related parent element in the topology. The validation function shall be referred by the EDDL specified CHECK_CONFIGURATION attribute.
- f) Connection Point data: The Connection Point (see 5.2.4 and 5.2.6) shall be described through EDDL constructs COMPONENT, COLLECTION and VARIABLE. The COMPONENT definition associates the Connection Point element to the Communication Device. The VARIABLE definitions represent the properties of a specific Connection Point. The COLLECTION represents the Connection Point structure as such. Protocol-specific definitions are found in IEC 62769-4:2020 Annex F.
- g) MENU:
The Menu structure shall follow the Menu conventions for PC-based applications in accordance with IEC 61804-4, which enables access to:
 - 1) FDI Communication Device Type (Bus) parameters: These parameters shall be made accessible by means of "offline_root_menu".
 - 2) Topology Configuration Dialogs shall be made available by means of the menu entry point "topology_configuration".

5.2.2 Device component

Each FDI Communication Package shall contain an EDD element describing the device.

```
COMPONENT <DeviceComponentId>
{
    LABEL "<Label>";
    CAN_DELETE TRUE;
    CHECK_CONFIGURATION <ValidateModules>;
    CLASSIFICATION NETWORK_COMPONENT;
    COMPONENT_RELATIONS
    {
        <CommunicationDeviceRelationId>
    }
}

COMPONENT_RELATION <CommunicationDeviceRelationId>
{
    LABEL "Relation type description";
    RELATION_TYPE CHILD_COMPONENT;
    ADDRESSING {<AddressVar>}
    COMPONENTS
    {
        <CommunicationDeviceComponentId>
        {
            AUTO_CREATE <autoCreate>;
            REQUIRED_RANGES
            {
                <AddressVar>{ MIN_VALUE <AddrMin>; MAX_VALUE <AddrMax>;}
            }
        }
    }
    MINIMUM_NUMBER <minNumber>;
    MAXIMUM_NUMBER <maxNumber>;
}
```

<DeviceComponentId>: The COMPONENT identifier identifies the component description for the device type.

<Label>: The string value shall contain a string that allows a human user to determine the function of the FDI Communication Server object.

<ValidateModules>: The Value refers to the METHOD implementing the module topology configuration validation function. Implementation details are specified in 5.2.9.

The attribute COMPONENT_RELATIONS allows to describe how modules can be connected. The definition of the COMPONENT_RELATIONS is optional. If used, it shall describe the relations to the CommunicationDevice definitions. The construct enables generic FDI Server-driven (device) topology configuration to be performed. Syntax details are described in IEC 61804-3. The subsequent text describes the semantic use of the COMPONENT_RELATION construct.

<CommunicationDeviceRelationId>: The attribute value identifies the COMPONENT_RELATION definition describing the relation between the device component and the CommunicationDevice component.

<CommunicationDeviceComponentId>: The attribute value has to match with a COMPONENT identifier used in a COMPONENT declaration that describes a CommunicationDevice (see 5.2.3).

<autoCreate>: The attribute value describes the number of CommunicationDevice components that can be automatically instantiated with the Device component.

<minNumber>/<maxNumber>/<autoCreate>: The attribute values define the instantiation constraints. The definition of these attributes is optional. The attribute values can contain conditional expressions.

The RELATION_TYPE shall be set to CHILD_COMPONENT.

<AddressVar>: The attribute value is a reference to a VARIABLE declaration. This VARIABLE holds the address value for a CommunicationDevice instance. The definition of this attribute is optional.

<AddrMin>/<AddrMax>: Values define the address value range for a CommunicationDevice instance. The value can for example correspond to a physical slot number. Usage of attributes ADDRESSING and REQUIRED_RANGES enables generic configuration routines.

5.2.3 CommunicationDevice component

Each FDI Communication Package shall contain at least one EDD element describing at least one CommunicationDevice component. A modular communication hardware structure shall be described by multiple CommunicationDevice COMPONENT descriptions:

```
COMPONENT <CommunicationDeviceComponentId>
{
    LABEL "<Label>";
    CAN_DELETE <CanDelete>;
    CLASSIFICATION NETWORK_COMPONENT;
    COMPONENT_RELATIONS
    {
        <CommunicationServiceProviderRelationId>
    }
}
```

```
COMPONENT_RELATION <CommunicationServiceProviderRelationId>
{
    LABEL "Relation between CommunicationDevice and communication service provider";
    RELATION_TYPE CHILD_COMPONENT;
    ADDRESSING {<AddressVar>}
    COMPONENTS
    {
        <CommunicationServiceProviderId>
        {
            AUTO_CREATE <autoCreate>;
        }
    }
}
```

```

    }
  }
  MINIMUM_NUMBER 1;
  MAXIMUM_NUMBER <maxNumber>;
}

```

<CommunicationDeviceComponentId>: The COMPONENT identifier identifies the CommunicationDevice component.

<Label>: The string value shall contain a human-readable string that allows a user to easily determine the function of the CommunicationDevice component.

<CanDelete>: Allowed values are TRUE or FALSE. It depends on whether a CommunicationDevice needs explicit configuration or whether the related communication service provider object shall be automatically instantiated with the CommunicationDevice. If the attribute CAN_DELETE is set to FALSE, the CommunicationDevice configuration is static.

The definition of the COMPONENT_RELATIONS is mandatory. It describes the relation to the communication service provider definition. The construct enables the FDI Server to instantiate communication service provider components in accordance with communication processing demands. Syntax details are described in IEC 61804-3. The subsequent text describes the semantic use of the COMPONENT_RELATION construct.

<CommunicationServiceProviderRelationId>: The attribute value identifies the COMPONENT_RELATION definition as such.

<CommunicationServiceProviderId>: The attribute value has to match with a COMPONENT identifier used in a COMPONENT declaration that describes a communication service provider (5.2.4).

<autoCreate>: The attribute value describes the number of communication service providers that can be automatically instantiated with the CommunicationDevice component.

The RELATION_TYPE shall be set to CHILD_COMPONENT.

The PROTOCOL attribute shall not be set.

5.2.4 Communication service provider component

Each FDI Communication Package describing a Communication Device shall contain at least one EDD element describing the communication service provider. The EDD component shall not define any configuration parameter.

```

COMPONENT <CommunicationServiceProviderId>
{
  LABEL "<Label>";
  BYTE_ORDER <byteOrder>;
  CAN_DELETE <CanDelete>;
  CLASSIFICATION NETWORK_COMMUNICATION_SERVICE_PROVIDER;
  COMPONENT_RELATIONS <CommunicationServiceProvidersConnectionPointRelationId>
  {
    <ConnectionPointRelationId>
  }
}

```

```

COMPONENT_RELATION <CommunicationServiceProvidersConnectionPointRelationId>
{
  LABEL "Relation between communication service provider and connection point";
  RELATION_TYPE CHILD_COMPONENT;
  ADDRESSING {<AddressVar>}
  COMPONENTS

```

```

{
    < ConnectionPointId>
    {
        AUTO_CREATE 1;
    }
}
MINIMUM_NUMBER 1;
MAXIMUM_NUMBER 1;
}

```

<CommunicationServiceProviderId>: The COMPONENT identifier identifies the communication service provider.

<Label>: The string value shall contain a human-readable string that allows a user to easily determine the function of the communication service provider object.

<CanDelete>: Allowed values are TRUE or FALSE. It depends on whether a communication service provider can be flexibly instantiated according to the communication processing demands. If the attribute CAN_DELETE is set to FALSE, the number of communication service provider component instantiations is static. The instantiation constraints declared through the attributes AUTO_CREATE, MINIMUM_NUMBER and MAXIMUM_NUMBER correspond to the capabilities of currently supported protocols.

<byteOrder>: The value enables generic integration of n-byte data types (e.g. 4-byte integer) into the communication message payload. The attribute value describes the byte order and shall be either BIG_ENDIAN or LITTLE_ENDIAN.

The definition of the COMPONENT_RELATIONS is mandatory. It describes the relation to the Connection Point definition. The construct enables to perform generic, FDI Server-driven topology configuration. Syntax details are described in IEC 61804-3. The subsequent text describes the semantic use of the COMPONENT_RELATION construct.

The Connection Point shall automatically be instantiated with the communication service provider and there shall be exactly one (1) Connection Point instance connected to the communication service provider. The instantiation constraints declared through the attributes AUTO_CREATE, MINIMUM_NUMBER and MAXIMUM_NUMBER correspond to the capabilities of currently supported protocols.

<CommunicationServiceProvidersConnectionPointRelationId>: the attribute value identifies the COMPONENT_RELATION declaration as such.

<ConnectionPointId>: the attribute value has to match with a COMPONENT identifier used for a COMPONENT declaration that describes a Connection Point (see 5.2.5).

The RELATION_TYPE shall be set to CHILD_COMPONENT.

The PROTOCOL attribute shall not be set.

5.2.5 Connection Point component

Each FDI Communication Package describing a Communication Device shall contain one EDD element describing one Connection Point for each of the protocols that are supported by the Communication Device:

```

COMPONENT <ConnectionPointId>
{
    LABEL "<Label>";
    CAN_DELETE FALSE;
    CLASSIFICATION NETWORK_CONNECTION_POINT;
}

```

```

    PROTOCOL <ProtocolId>;
    CONNECTION_POINT <ConnectionPointCollectionId>;
}

```

<ConnectionPointId>: the COMPONENT identifier identifies the Connection Point component declaration.

<Label>: the string value shall contain a string that allows a human user to determine the function of the Connection Point component.

<ProtocolId>: the value of this attribute indicates the communication capability, which allows the FDI Server to find other device types that can be connected to the network using the same type of protocol. For standardized protocols, the value is defined by the related fieldbus organization.

<ConnectionPointCollectionId>: the attribute value is a reference to a COLLECTION declaration that describes the data structure of the Connection Point as described in 5.2.6.

5.2.6 Connection Point collection

Each EDD describing the Connection Point of a communication device shall describe the COLLECTION element that describes the attributes that shall appear in the Information Model representation of the Connection Point. The protocol specific data exposed by the Connection Point identifies the device type and its network address.

```

COLLECTION <ConnectionPointCollectionId>
{
    LABEL "<Label>";
    MEMBERS
    {
        <AddressAttributeName>, <AddressAttributeVariableId>;
        VALID <VALID_VariableId>;
    }
}

```

<ConnectionPointCollectionId>: the identifier of the COLLECTION is referred by the CONNECTION_POINT attribute value defined in 7.7.3.5.

<Label>: the label identifies the Connection Point in a human readable way.

<AddressAttributeName>/<AddressAttributeVariableId>: the MEMBER section refers to the VARIABLE definitions describing the address attributes implemented by a Connection Point. The content of the MEMBER section is protocol-specific.

<VALID>/<VALID_VariableId> is a Collection member referring a Boolean VARIABLE holding the validation status that shall be set by the ValidateNetwork Action (see 5.2.8).

5.2.7 Network component

Each FDI Communication Package describing a Communication Device shall contain one EDD element describing one Network for each of the protocols that are supported by the Communication Device. The definition supports the network topology engineering:

```

COMPONENT <NetworkComponentId>
{
    LABEL "<Label>";
    CAN_DELETE TRUE;
    CHECK_CONFIGURATION <Validate>;
    CLASSIFICATION NETWORK;
    PROTOCOL <ProtocolId>;
    COMPONENT_RELATIONS
    {
        <NetworksConnectionPointRelationId>
    }
}

```

```

    }
}

COMPONENT_RELATION <NetworksConnectionPointRelationId>
{
    LABEL "Relation between network and connection point";
    RELATION_TYPE CHILD_COMPONENT;
    ADDRESSING {<AddressVar>}
    COMPONENTS
    {
        <ConnectionPointId>
        {
            REQUIRED_RANGES
            {
                <BusAddressVar>{ MIN_VALUE <BusAddrMin>; MAX_VALUE <BusAddrMax>;}
            }
        }
    }
    MINIMUM_NUMBER 1;
    MAXIMUM_NUMBER <maxNumber>;
}

```

<NetworkComponentId>: the COMPONENT identifier identifies the Network component declaration.

<Label>: the string value shall contain a human-readable string that allows a user to easily determine the function of the Network component.

<Validate>: the value refers to the METHOD implementing the network topology configuration validation function (see 5.2.8).

<ProtocolID>: the value of this attribute allows the FDI Server to find other device types that can be connected to the network using the same type of protocol. For standardized protocols, the value is defined by the related fieldbus organization.

The definition of the COMPONENT_RELATIONS is mandatory. It describes the relation to the Connection Point definition and by that the capabilities of a network. The construct enables generic FDI Server-driven network topology configuration to be performed. Syntax details are described in IEC 61804-3. The subsequent text describes the semantic use of the COMPONENT_RELATION construct.

<NetworksConnectionPointRelationId>: the attribute value identifies the COMPONENT_RELATION definition.

<ConnectionPointId>: The attribute value has to match with a COMPONENT identifier used for a COMPONENT declaration that describes a Connection Point (see 5.2.4).

<maxNumber>: The attribute value limits the number of Connection Points that can be connected to the network. The attribute values can contain conditional expressions.

The RELATION_TYPE shall be set to CHILD_COMPONENT.

<BusAddressVar>: The attribute value is a reference to a VARIABLE declaration. This VARIABLE holds the network address value for any device that is connected to the network.

<BusAddrMin>/<BusAddrMax>: Values define the network address value range.

5.2.8 ValidateNetwork

The method ValidateNetwork represents the Communication Device implemented Business Logic that validates a current network topology. The ValidateNetwork method handles any necessary dependencies related to bus parameters. The implementation of related EDDL logic is based on the EDDL Built-in function ObjectReference, which enables the analysis of a set of child instances (Connection Point instances). The validation logic shall set the <VALID> attribute of the Connection Point instance that has passed the validation.

The implementation of ValidateModules is optional if the module setup is either static or if the configuration rules defined in the COMPONENT construct are sufficient to configure the module setup.

Table 1 shows the ValidateNetwork Action Arguments.

Signature

```
ValidateNetwork (  
    [out] Integer ServiceError,  
    [out] String ErrorMessage);
```

Table 1 – ValidateNetwork Action arguments

Argument	Description
ServiceError	0: OK -1: Failed / the Connection Point that did not pass the validation is indicated by the <VALID> attribute () value set to false. Remark: The argument values correspond to the error codes specified in IEC 61804-3 named BI_SUCCESS (value = 0) and BI_ERROR (value = -1). The Action returns the ServiceError result using the "return" statement.
ErrorMessage	If the method returns an empty string (NULL), the Action call succeeded. In the case of an error, the Action can return a description of the problem.

5.2.9 ValidateModules

The method ValidateModules validates the current module setup. The implementation of the related EDDL logic is based on the EDDL Built-in function ObjectReference, which enables a set of child instances to be analyzed. The implementation of ValidateModules is optional if the module setup is either static or if the configuration rules defined in the COMPONENT construct are sufficient to configure the module setup.

NOTE The decision whether ValidateModules is needed or not is vendor-specific.

Table 2 shows the ValidateModules Action Arguments.

Signature

```
ValidateModules (  
    [out] Integer serviceError,  
    [out] String ErrorMessage);
```

Table 2 – ValidateModules Action arguments

Argument	Description
ServiceError	0: OK -1: Failed / the Connection Point that did not pass the validation is indicated by the <VALID> attribute () value set to false. Remark: The Argument values correspond to the error codes specified in IEC 61804-3 named BI_SUCCESS (value = 0) and BI_ERROR (value = -1). The Action returns the ServiceError result using the "return" statement.
ErrorMessage	If the Action returns an empty string (NULL), the method call succeeded. In the case of an error, the Action can return a problem description.

5.2.10 UIP specifics

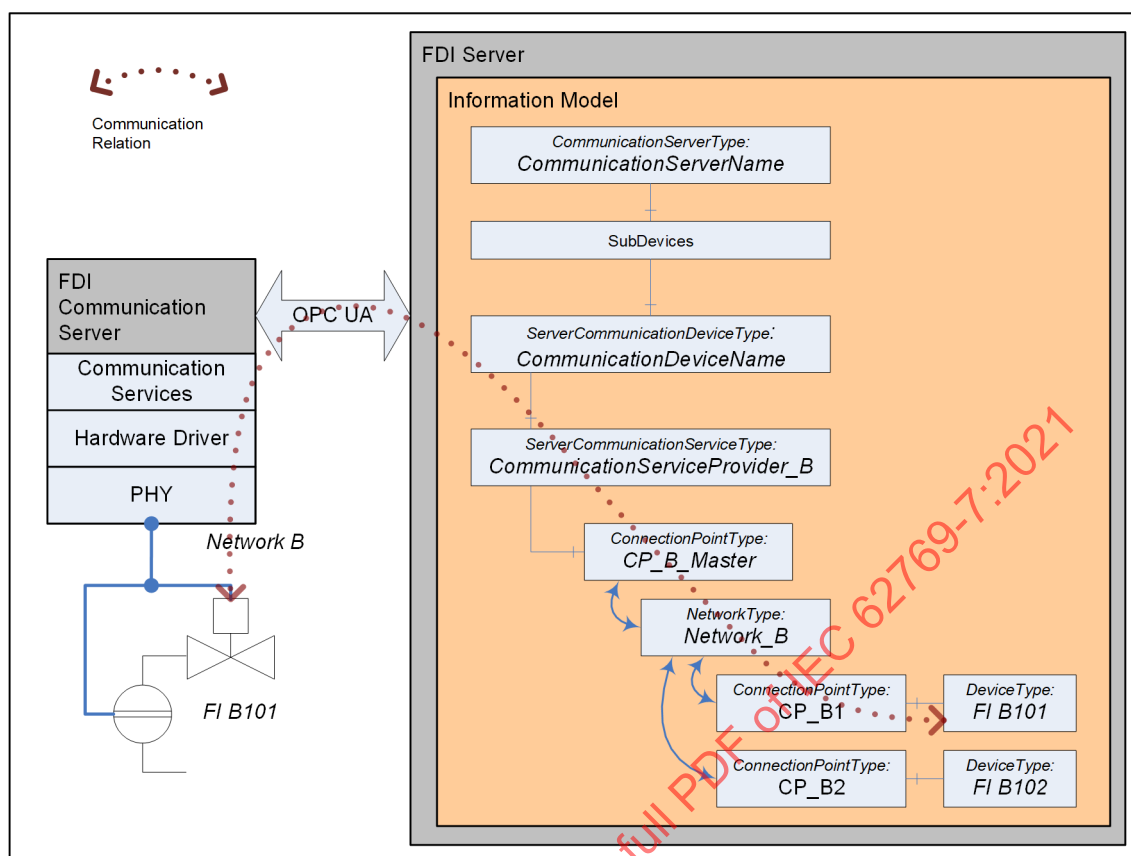
The FDI Communication Package can contain the UIP to support, for example, diagnostics and parameterization.

5.2.11 Deployment

The FDI Server imports the FDI Communication Package. The handling of EDD and UIP parts matches with the import procedure performed for the FDI Package (see IEC 62769-2 and IEC 62769-3).

6 Communication relations

The purpose of a communication device and its communication services is to exchange information between the physical device and the device representation managed by the FDI Server. The information exchange is managed via communication relations, see Figure 3. An established communication relation represents the capability to exchange information between the FDI Server-managed device representation and the physical device. The use of a Communication Relation allows abstracting from protocol specifics typically used to manage connections.



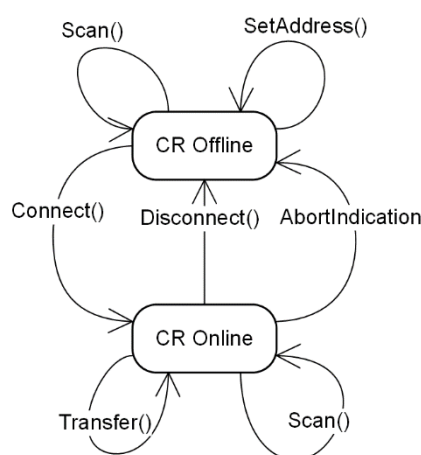
IEC

Figure 3 – Communication relation

NOTE 1 The core of information exchange happens between the device connected to the physical network and the corresponding instance within the Information Model but does not cover the complete device application.

The following state chart (Figure 4) describes the general state flow for a single communication relation. The diagram also shows which communication services can be invoked during a "CR Online" state.

The "AbortIndication" shown in Figure 4 can be detected in different protocol-specific ways. The one specified for any communication device is bound to the serviceErrors returned by the specified communication services. Even the Scan method can determine a connection loss, when the device for which a communication relation has been activated does not appear in a scan result.



IEC

Figure 4 – Communication relation state chart

NOTE 2 The management of communication relations is optional.

7 FDI Communication Server definition

7.1 General

In terms of FDI, the FDI Communication Server is a dedicated OPC UA Server providing access to one or more field device networks. Each FDI Communication Server is modelled as a Modular Device where each module (also called CommunicationDevice in the sequence) represents the access point to one network.

The Modular Device itself represents the FDI Communication Server as a whole.

7.2 General characteristics

The FDI Communication Server implements characteristics for each of its CommunicationDevices specified in 7.3.3. Additionally, an FDI Communication Server implements the following characteristics:

- The FDI Server always synchronizes (see 7.5, 7.8.8, and 7.8.11) the FDI Communication Server hosted Information Model from the FDI Server hosted Information Model content.
- CommunicationDevices can be statically instantiated or they can be created/deleted by the FDI Server.
- Communication between the FDI Server and the FDI Communication Server is based on OPC UA. OPC UA specifies a wire protocol for its services that can be implemented on arbitrary platforms and runtime environments.
- To avoid race conditions, the FDI Communication Server only allows one FDI Server being connected at a time. With this restriction an FDI Communication Server can refrain from any synchronization (locking) mechanism. The FDI specification does not enforce FDI Communication Servers implementing any interlocking mechanism to manage concurrent access to a single device connected to the physical network.

7.3 Information Model

7.3.1 General

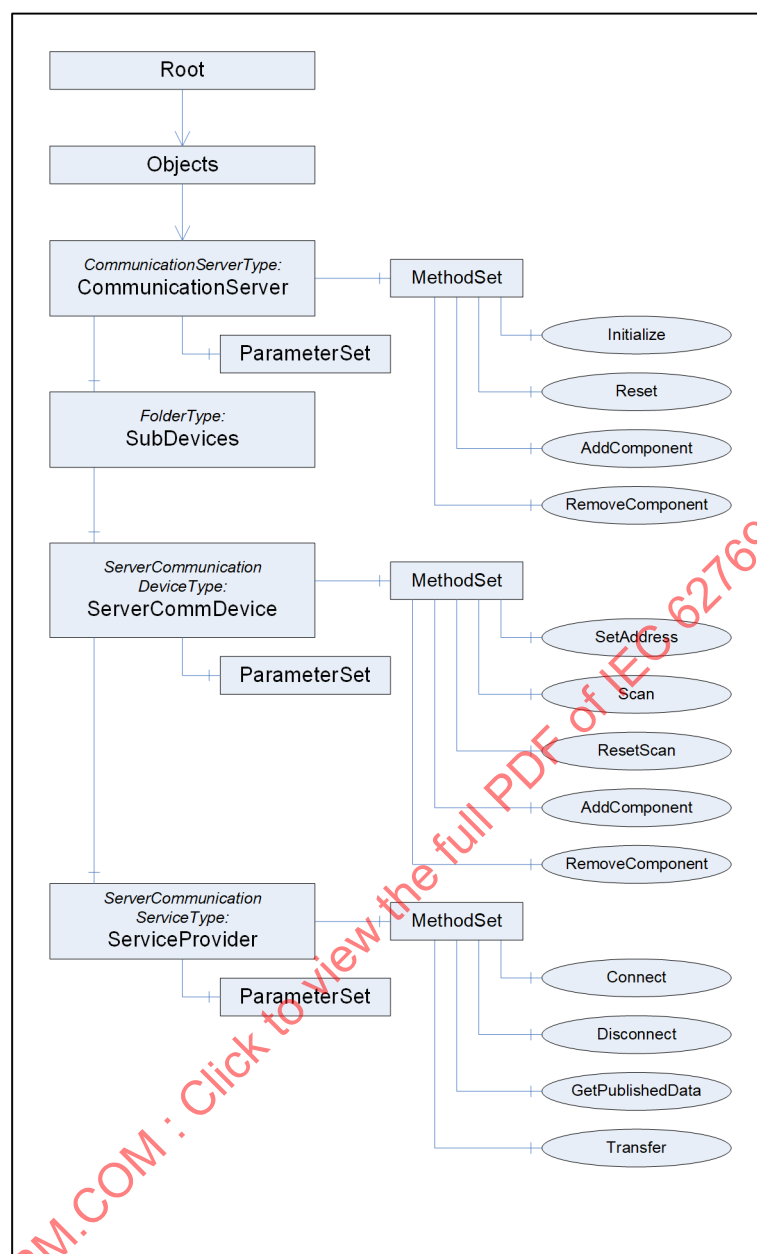
Subclause 7.3 specifies the FDI Communication Server hosted Information Model.

An FDI Communication Server is an OPC UA Server that encapsulates communication hardware and provides standardized communication ability. The FDI Server connects to the FDI Communication Server as an OPC UA Client and accesses the networks supported by the FDI Communication Server via the FDI Communication Server information model. The task of the FDI Communication Server is to expose this information model. The FDI Communication Server shall not maintain Device Instances or network topology information. All interaction with FDI devices is done through the FDI Server and just transferred by the FDI Communication Server.

For the FDI Server, an FDI Communication Server looks like a device that supports FDI Communication Services and uses OPC UA to communicate. The FDI Communication Server may run locally on the same PC as the FDI Server (loop-back adapter) or remote in the field (e.g. embedded into a controller). Like a device, each FDI Communication Server has an associated FDI Package. This FDI Package is used to create communication devices in the Information Model of the FDI Server that represent access to the networks implemented by the FDI Communication Server.

The Information Model of an FDI Communication Server is based on the Information Model defined in IEC 62769-5. Figure 5 replicates the Modular Device structure and illustrates how it maps into the overall AddressSpace. The modules represent the communication channels of the FDI Communication Server.

IECNORM.COM : Click to view the full PDF of IEC 62769-7:2021



IEC

Figure 5 – FDI Communication Server AddressSpace

The CommunicationServerType (the root of the Modular Device) is a subtype of the DeviceType. The MethodSet contains the methods Initialize, Reset, AddComponent and RemoveComponent. The methods AddComponent and RemoveComponent are optionally present if the FDI Communication Server supports the dynamic instantiation of elements in the folder SubDevices.

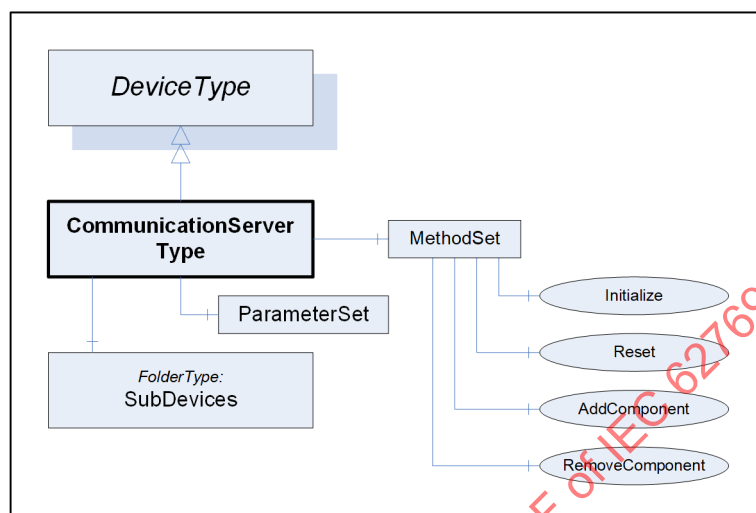
All sub-devices are instances of the ServerCommunicationDeviceType defined in 7.3.3. The instances of the ServerCommunicationDeviceType (ServerCommDevice) have a MethodSet that can implement the methods SetAddress, Scan, AddComponent, RemoveComponent. AddComponent and RemoveComponent are optionally present if the FDI Communication Server supports a variable number of instances of the ServerCommunicationServiceType.

Formal definitions are found in 7.3.2, 7.3.3 and 7.3.4.

7.3.2 CommunicationServerType

7.3.2.1 General

The CommunicationServerType is a subtype of the DeviceType and provides the methods needed to manage the instances ServerCommunicationDeviceType. Figure 6 shows the CommunicationServerType definition that is formally defined in Table 3 and Table 4.



IEC

Figure 6 – CommunicationServerType

Table 3 – CommunicationServerType definition

Attribute	Value				
BrowseName	CommunicationServerType				
IsAbstract	False				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
Subtype of the DeviceType defined in IEC 62541-100.					
HasComponent	Object	MethodSet		BaseObjectType	Mandatory
HasComponent	Object	ParameterSet		BaseObjectType	Optional
HasComponent	Object	SubDevices		FolderType	Mandatory

Table 4 – MethodSet of CommunicationServerType

Attribute	Value				
BrowseName	MethodSet				
IsAbstract	False				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
HasComponent	Method	Initialize			Mandatory
HasComponent	Method	Reset			Mandatory
HasComponent	Method	AddComponent			Optional
HasComponent	Method	RemoveComponent			Optional

The CommunicationServerType and each instance of this Type share the same Methods. The NodId of these Methods will be fixed and defined in this document (see Annex B). FDI Communication Server clients therefore do not have to browse for these Methods. They can use the fixed NodId as the MethodId of the Call Service.

The additional Methods AddComponent and RemoveComponent add the ability to add or remove instances of ServerCommunicationDeviceType according to the communication hardware's structure. These services are not applicable if the FDI Communication Server implements a static communication hardware structure.

The SubDevices folder contains instances of ServerCommunicationDeviceType that represent the communication modules.

NOTE The indication for a static communication hardware layout is indicated in the FDI Package with COMPONENT attribute CAN_DELETE set to FALSE in COMPONENT declarations.

7.3.2.2 Reset Method

Reset is used to reset the communication hardware and related driver software. Any ongoing communication will be stopped immediately. All communication channels enter the closed status.

The Method Reset shall not be present in the FDI Server-hosted Information Model. The FDI Server shall be able to handle the shut-down procedure automatically according to communication demands.

Typically, the FDI Communication Server operation includes some hardware and protocol driver handling that can be independent from any modular structure. Because of this possibility, the Reset method is arranged underneath the CommunicationServerType. For the purpose of reducing the complexity of FDI Communication Server operation, only one Reset method has been specified.

The signature of this Method is specified below. Table 5 and Table 6 specify the arguments and AddressSpace representation, respectively.

Signature

```
Reset (
    [out] Integer      serviceError);
```

Table 5 – Reset Method arguments

Argument	Description
serviceError	0: OK -1: Failed

Table 6 – Reset Method AddressSpace definition

Attribute	Value				
BrowseName	Reset				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory

7.3.2.3 Initialize Method

Initialize is used to initialize the communication hardware. The initialization function of the FDI Communication Server shall use the parameterization data hosted by the ParameterSet that is contained within the instance of the CommunicationServerType and all instances of ServerCommunicationDeviceType.

In order to enable parameter changes during operation, the Initialize method can be re-invoked. If the FDI Communication Server needs to reset its communication hardware, it shall automatically restore any communication relation that existed. A modular FDI Communication Server can flexibly initialize only those ServerCommunicationDeviceType instances for which configuration changes have been detected.

The Method Initialize shall not be present in the FDI Server-hosted Information Model. The FDI Server shall be able to handle the start procedure automatically according to human-driven communication requests.

The FDI Communication Server operation can include some hardware and protocol driver handling that can be independent from any modular structure. Because of this possibility, the Initialize method is arranged underneath the CommunicationServerType. For the purpose of reducing the complexity of the FDI Communication Server's operation, only one Initialize method has been specified.

The signature of this Method is specified below. Table 7 and Table 8 specify the arguments and AddressSpace representation, respectively.

Signature

```
Initialize(
    [out] Integer      serviceError)
```

Table 7 – Initialize Method arguments

Argument	Description
serviceError	0: OK -1: Failed

Table 8 – Initialize Method AddressSpace definition

Attribute	Value				
BrowseName	Initialize				
References	NodeClass	BrowseName	Data Type	TypeDefinition	ModellingRule
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory

7.3.2.4 AddComponent Method

AddComponent shall be used to configure the modular setup of an FDI Communication Server when the FDI Communication Server has no statically defined communication hardware setup. This method shall be used to add a module (Instance of ServerCommunicationDeviceType).

The signature of this Method is specified below. Table 9 and Table 10 specify the arguments and AddressSpace representation, respectively.

Signature

```

AddComponent (
    [in] String      ModuleTypeName,
    [in] String      InstanceName,
    [in] String      InstanceLabel,
    [out] NodeId     InstanceNodeId,
    [out] Integer    ServiceError);

```

Table 9 – AddComponent Method arguments

Argument	Description
ModuleTypeName	Type of module to be created as defined in the FDI Package. The module type name shall correspond to one of the COMPONENT identifier definitions (see 5.2.3).
InstanceName	Non-localized name of the module's Device Node of the created element. This name has to be unique within the scope of the FDI Communication Server's Information Model.
InstanceLabel	Human-readable label for the root Node of the created module.
InstanceNodeId	Callee-assigned identifier for the module's Device Node.
ServiceError	0 – OK -1 – E_InvalidType – a module for the specified Type can not (not anymore) be added -2 – E_DuplicateName – there exists already a module with the same name as specified with the InstanceName argument -3 – E_UnknownType – an unknown ModuleTypeName has been specified -4 – E_LimitExceeded – the total number of modules is exceeded (this might be caused by power constraints or other resource limitations)

Table 10 – AddComponent Method AddressSpace definition

Attribute	Value				
BrowseName	AddComponent				
References	NodeClass	BrowseName	Data Type	TypeDefinition	ModellingRule
HasProperty	Variable	InputArguments	Argument[]	PropertyType	Mandatory
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory

7.3.2.5 RemoveComponent Method

RemoveComponent shall be used to remove a module (Instance of ServerCommunicationDeviceType). Implementation of RemoveComponent is optional if the communication hardware setup is static.

The signature of this Method is specified below. Table 11 and Table 12 specify the arguments and AddressSpace representation, respectively.

Signature

```
RemoveComponent (
    [in]  NodeId          ModuleNodeId,
    [out] Integer         ServiceError);
```

Table 11 – RemoveComponent Method arguments

Argument	Description
ModuleNodeId	The value is the identification of the existing instance in the Information Model.
ServiceError	0: OK -1: Failed, the specified node does not exist

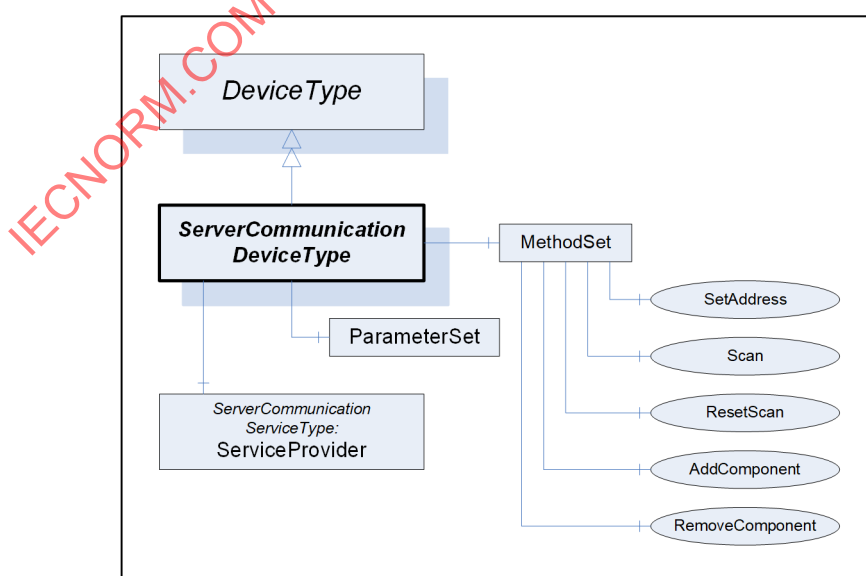
Table 12 – RemoveComponent Method AddressSpace definition

Attribute	Value				
BrowseName	RemoveComponent				
References	NodeClass	BrowseName	Data Type	Type Definition	Modelling Rule
HasProperty	Variable	InputArguments	Argument[]	PropertyType	Mandatory
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory

7.3.3 ServerCommunicationDeviceType

7.3.3.1 General

The ServerCommunicationDeviceType represents a communication channel for a particular network. The ServerCommunicationDeviceType is a subtype of the DeviceType. The ParameterSet for each instance of a ServerCommunicationDevice will contain Parameters necessary to configure the operation of the network. Figure 7 shows the ServerCommunicationDeviceType definition that is formally defined in Table 13 and Figure 14.



IEC

Figure 7 – ServerCommunicationDeviceType

Table 13 – ServerCommunicationDeviceType definition

Attribute	Value				
BrowseName	ServerCommunicationDeviceType				
IsAbstract	True				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
Subtype of the DeviceType defined in OPC UA Part DI.					
HasComponent	Object	MethodSet		BaseObjectType	Optional
HasComponent	Object	ParameterSet		BaseObjectType	Optional
HasComponent	Object	ServiceProvider		ServerCommunicationServiceType	Mandatory
HasProperty	Variable	ListOfCommunicationProfiles	String[]	PropertyType	Mandatory

The Property ListOfCommunicationProfiles contains a list of communication profiles supported by the ServerCommunicationDevice. Valid strings are defined in IEC 62769-4:2020, Annex F.

Table 14 – MethodSet of ServerCommunicationDeviceType

Attribute	Value				
BrowseName	MethodSet				
IsAbstract	True				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
HasComponent	Method	Scan			Optional
HasComponent	Method	ResetScan			Optional
HasComponent	Method	SetAddress			Optional
HasComponent	Method	AddComponent			Optional
HasComponent	Method	RemoveComponent			Optional

7.3.3.2 Scan Method

Scan shall be used to start discovering devices connected to the physical network. The associations between the method Scan and the corresponding physical network connection enables the FDI Communication Server to access the correct physical network connection. The Scan method is implemented by the Communication Server runtime module.

The signature of this Method is specified below. Table 15 and Table 16 specify the arguments and AddressSpace representation, respectively.

NOTE 1 Communication Servers can run the network scan in a background task, so the invocation of the function Scan will return cached network scan results.

NOTE 2 If the SCAN takes very long, the FDI Communication Server might return an empty TopologyScanResult and the ServiceError 1 identifying that the scan is still running.

Signature

```
Scan (
    [out] XmlElement      TopologyScanResult,
    [out] Integer         ServiceError)
```

Table 15 – Scan Method arguments

Argument	Description
TopologyScanResult	The argument value is an XML formatted string representing a list of devices connected to the physical network. Each of the devices connected to the physical network is represented by a data structure matching with a Connection Point node. Connection Point attributes are protocol-specific. The corresponding topologyScanResult schema is specified in IEC 62769-4:2020, Annex F. Returns an empty string for TopologyScanResult in the case of any error.
ServiceError	0: OK / scan completed 1: OK / get complete scan result by calling Scan again -1: Failed / not initialized -2: Failed / not connected to a network -3: Failed / no device found, the topologyScanResult is empty

Table 16 – Scan Method AddressSpace definition

Attribute	Value				
BrowseName	Scan				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory

7.3.3.3 ResetScan Method

ResetScan shall be used to reset the internal cache of scan results. It will also cancel a running scan if the FDI Communication Server scan mechanism supports this.

The signature of this Method is specified below. Table 17 and Table 18 specify the arguments and AddressSpace representation, respectively.

Signature

```
ResetScan (
    [out] Integer         ServiceError)
```

Table 17 – ResetScan Method arguments

Argument	Description
ServiceError	0: OK / scan reset -1: Failed / not initialized -2: Failed / not connected to a network

Table 18 – ResetScan Method AddressSpace definition

Attribute	Value				
BrowseName	ResetScan				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory

7.3.3.4 SetAddress Method

SetAddress shall be used to change the network address (communication address) of a device. The Communication Device shall ensure unique network address values. If the argument value of newAddress is already assigned to a device connected to the physical network, the Communication Device shall return the argument serviceError value "-4: Failed/duplicate Address error".

It depends on the protocol whether the address assignment service shall work even when a communication relation is already established.

The signature of this Method is specified below. The arguments for SetAddress Method are described in Table 19.

Signature

```
SetAddress (
    [in]                <OldAddress>,
    [in]                <NewAddress>,
    [out] Int32         ServiceError);
```

Table 19 – SetAddress Method arguments

Argument	Description
<OldAddress>	This is a placeholder for a 1..n protocol-specific arguments representing the existing protocol-specific network address of the device connected to the physical network. Values that represent a network address are specified in IEC 62769-4:2020, Annex F.
<NewAddress>	This is a placeholder for 1..n protocol-specific arguments representing the new protocol-specific network address that shall be assigned to a device connected to the physical network. Values that represent a network address's arguments are specified in IEC 62769-4:2020, Annex F.
ServiceError	0: OK/execution finished successfully Other values that represent ServiceErrors are specified in IEC 62769-4:2020, Annex F.

7.3.4 ServerCommunicationServiceType

7.3.4.1 General

Communication services provide the means to communicate with a Device or to, for example, execute a Scan on a Network. Communication services are represented through Methods in the Information Model (see IEC 62769-5).

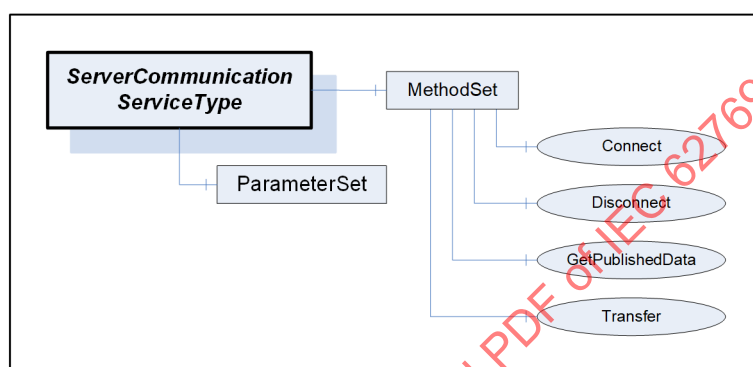
The formal definition of ServerCommunicationServiceType is found in Figure 8, Table 20 and Table 21.

The NodeId of these Methods will be fixed and defined in this document. FDI Clients therefore do not have to browse for these Methods. They can use the fixed NodeId as the MethodId of the Call Service.

Communication methods that include their NodeIds are uniquely defined in this document. FDI Clients can use the Methods directly (without browsing). The OPC UA Call Service shall be used as follows:

- the MethodId argument shall contain the fixed NodeId of the Method;
- the ObjectId argument shall contain the NodeId of the MethodSet.

The OPC UA StatusCode Bad_MethodInvalid shall be returned from the Call Service for elements where the communication methods are not supported.



IEC

Figure 8 – ServerCommunicationServiceType

Table 20 – ServerCommunicationServiceType definition

Attribute	Value				
BrowseName	ServerCommunicationServiceType				
IsAbstract					
References	NodeClass	BrowseName	Data Type	TypeDefinition	ModellingRule
Subtype of the DeviceType defined in OPC UA Part DI.					
HasComponent	Object	MethodSet		BaseObjectType	Mandatory
HasComponent	Object	ParameterSet		BaseObjectType	Optional

Table 21 – MethodSet of ServerCommunicationServiceType

Attribute	Value				
BrowseName	MethodSet				
IsAbstract					
References	NodeClass	BrowseName	Data Type	TypeDefinition	ModellingRule
HasComponent	Method	Connect			Optional
HasComponent	Method	Disconnect			Optional
HasComponent	Method	Transfer			Mandatory
HasComponent	Method	GetPublishedData			Optional

7.3.4.2 Connect Method

Connect shall be used to establish a communication relation to a device that is physically connected to the Network. Establishing the communication relation may imply checks of identification data that are part of the addressData with data inside the physical device. The Communication Device performs this DeviceType match verification in accordance with a corresponding network protocol standard. Related details are specified in IEC 62769-4:2020, Annex F.

The devices' address is contained in the Connection Point of the corresponding Device Instance within the Information Model (Device Connection Point). The communication relation between the Information Model-associated device application and the physical device is further on identified by the communication relation identifier. Details about how to manage the status of a communication relation is described in Clause 6. Connection Point descriptions may be layered as described in Annex A.

NOTE 1 As the NodeId is a unique identifier within the Information Model scope, the NodeId of the Device Connection Point can be a unique identifier for any communication relation in the scope of a communication device.

NOTE 2 The term communication relation is introduced to describe the status of an infrastructure that enables data exchange between Information Model hosted data and a physical device. If the communication relation is established, data exchange is possible.

The signature of this Method is specified below. Table 22 specifies the arguments.

Signature

```
Connect (
    [in]   ByteString      CommunicationRelationId,
    [in]   <AddressData>,
    [out]  <DeviceInformation>,
    [out]  Int32           ServiceError);
```

Table 22 – Connect Method arguments

Argument	Description
CommunicationRelationId	This is a client generated ID that is used to uniquely identify this connection. This could be an index (e.g. a NodeId) that the client (= FDI Server) needs in order to identify entries in its topology.
<AddressData>	This is a placeholder for a protocol-specific argument list that is used for the address and optional device identification data (details described in IEC 62769-4:2020, Annex F).
<DeviceInformation>	This is a placeholder for a protocol-specific argument list in which the connect result data are stored.
ServiceError	0: OK/execution finished, connection established successfully Other values that represent ServiceErrors are specified in IEC 62769-4:2020, Annex F.

7.3.4.3 Disconnect method

Disconnect shall be used to terminate a communication relation to a Device.

The signature of this Method is specified below. Attributes of the Disconnect method are specified in Table 23. Disconnect is a synchronous method call.

Signature

```

Disconnect(
    [in]   ByteString      CommunicationRelationId,
    [out]  Int32           ServiceError);

```

Table 23 – Disconnect Method arguments

Argument	Description
CommunicationRelationId	Same ID as used in method Connect specified in 7.3.4.2.
ServiceError	1: OK/disconnect finished successfully Other values that represent ServiceErrors are specified in IEC 62769-4:2020, Annex F.

7.3.4.4 Transfer method

Transfer shall be used to perform information exchange with a Device.

The signature of this Method is specified below. All arguments are specified in Table 24.

Signature

```

Transfer(
    [in]   ByteString      CommunicationRelationId,
    [in]           <SendData>,
    [out]           <ReceiveData>,
    [out]  Int32           ServiceError);

```

Table 24 – Transfer Method arguments

Argument	Description
CommunicationRelationId	See 7.3.4.2.
<SendData>	This is a placeholder for a protocol-specific list of arguments as described in IEC 62769-4:2020, Annex F. The argument values represent the protocol-specific communication service request that is sent to the device.
<ReceiveData>	This is a placeholder for a protocol-specific list of arguments as described in IEC 62769-4:2020, Annex F. The argument values represent the protocol-specific communication service response that is received from the device.
ServiceError	0: OK / execution finished, ReceivedData contains the result Other values that represent ServiceErrors are specified in IEC 62769-4:2020, Annex F.

7.3.4.5 GetPublishedData Method

The FDI Server sends GetPublishedData requests to the FDI Communication Server to receive data that is submitted by unsolicited data messages. The data contained in the argument SendData prepares the exchange of "unsolicited" data messages from the device. The content of SendData is protocol-specific. The FDI Communication Server queues GetPublishedData requests in a queue associated with the Communication Relation defined through the argument CommunicationRelationId. The argument PublishId identifies the related queue entry. Each time the FDI Communication Server receives unsolicited data messages, it saves the received data in association with the existing queue entry that has been created for the GetPublishedData. Depending on the underlying network technology (performance), the method GetPublishedData can immediately return with data coming from an "unsolicited" data message.

Subsequent pulling of data that is submitted by unsolicited data messages works through the same method `GetPublishedData`. In this case, the argument `SendData` is empty. The argument `PublishId` matches with the value that has been provided with the initial call `GetPublishedData` that has established the transmission of exchange of "unsolicited" data messages.

In order to stop the device sending the "unsolicited" data, the method `GetPublishedData` shall be used again but the argument `SendData` contained data terminates the exchange of "unsolicited" data messages from the device. Table 25 shows the `GetPublishedData` Method Arguments.

Signature

```
GetPublishedData (
    [in]   ByteString      CommunicationRelationId,
    [in]   <SendData>,
    [out]  <ReceiveData>,
    [out]  DateTime        TimeStamp
    [in]   UInt32          PublishId,
    [out]  Int32            ServiceError);
```

Table 25 – GetPublishedData Method arguments

Argument	Description
CommunicationRelationId	See 7.3.4.2.
<SendData>	This is a placeholder for a protocol-specific list of arguments as described in IEC 62769-4:2020, Annex F. The argument values control the exchange of unsolicited messages.
<ReceiveData>	This is a placeholder for a protocol-specific list of arguments as described in IEC 62769-4:2020, Annex F. The argument values convey data that comes from unsolicited messages.
TimeStamp	Time when the data was published by the device.
PublishId	The number identifies an established subscription that conveys data that comes from unsolicited messages.
ServiceError	0: OK / execution finished Other values that represent ServiceErrors are specified in IEC 62769-4:2020, Annex F.

7.4 OPC UA Server Profile for FDI Communication Server

Profiles are named groupings of `ConformanceUnits` as defined in IEC 62541-6 and IEC 62541-7. The term "Facet" in the title of a Profile indicates that this Profile is expected to be part of another larger Profile or concerns a specific aspect of OPC UA. Profiles with the term "Facet" in their title are expected to be combined with other Profiles to define the complete functionality of an OPC UA Server or Client. The minimum required OPC UA Server Profile is the "Micro Embedded Device Server Profile".

The following table specifies the facet for an OPC UA Server that acts as an FDI Communication Server. Table 26 describes `Conformance Units` included in this facet.

Table 26 – FDICommunicationServer_Facet definition

Conformance Unit	Description	Optional/Mandatory
FDI Communication Server Information Model	Support at least one instance of <code>CommunicationServerType</code> .	M

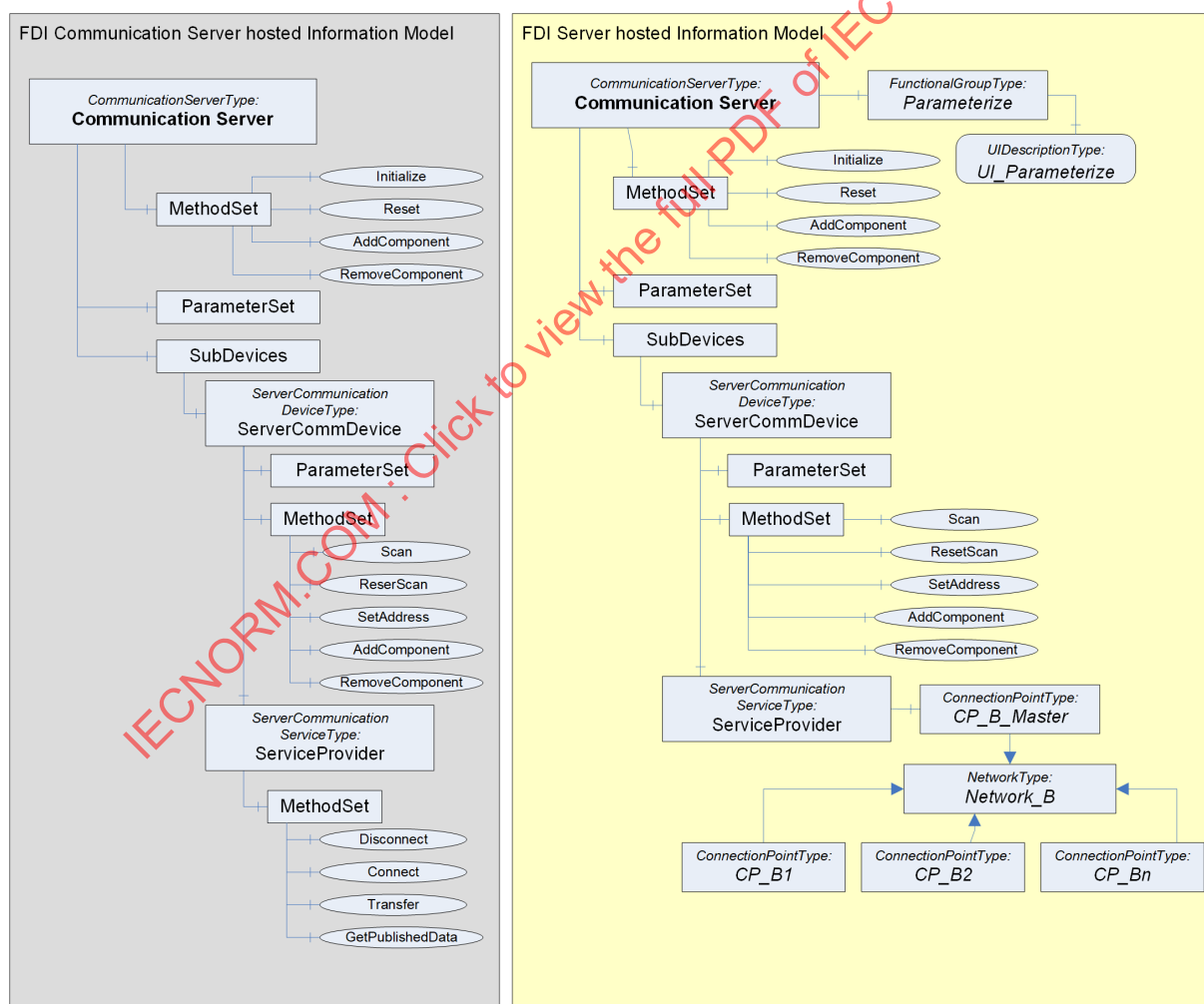
7.5 Mapping the FDI Server IM to the FDI Communication Server IM

7.5.1 General

The representation of an FDI Communication Server in the AddressSpace of an FDI Server is almost identical to the AddressSpace that exists in the FDI Communication Server. This refers in particular to the Modular Device hierarchy and the Parameters of all Devices. However, the Nodes in the FDI Server are built from the device description imported via the FDI Communication Package.

7.5.2 Information Model differences

Because of their different tasks, however, there are a few differences and a set of synchronization rules. The Information Model example shown in Figure 9 depicts the commonalities and the differences between the Information Models hosted by the FDI Communication Server and the FDI Server. In general, the FDI Communication Server-hosted Information Model is a subset of the FDI Server-hosted Information Model. The Device Instances in the FDI Server and the FDI Communication Server adhere to the same type definitions. Thus, browse names of common Information Model elements shall have the same browse name.



IEC

Figure 9 – Information Model differences (example)

The list of differences in the IMs is as follows:

- The FDI Server supports online and offline versions of the Modular Device; the FDI Communication Server supports just an online version. The online version of the FDI Server represents the version in the FDI Communication Server, i.e. if Parameter Values are read or written to the online model of the FDI Server, these operations are passed through to the FDI Communication Server. This happens both for public and for private Parameters.

NOTE The key is a match between the browse names present in the FDI Server-hosted Information Model and the FDI Communication Server-hosted Information Model. This allows generic synchronization of both information models.

- UIPs, UUIDs, Actions and Functional Groups exist only in the FDI Server.
- Modules in the FDI Server-hosted Information Model have a Connection Point component that is used to connect this module to a network when creating the Device Topology. The FDI Communication Server-hosted Information Model does not show Connection Point elements. The Device Topology is managed by the FDI Server only (see IEC 62769-5).
- The FDI Server can represent the ServiceProvider without exposing the MethodSet in order to prevent an FDI Client from invoking communication services.

The mapping of the Module Management functionality is as follows:

- AddComponent and RemoveComponent are exposed in the FDI Communication Server Information Model via the FDI Communication Server Object (the root of the Modular Device). They exist only if there are modules to be configured, i.e. they will not be available if the Communication Server does not support modular communication hardware configuration. AddComponent and RemoveComponent replace the generic Node Management service defined by OPC UA.

The FDI Server handles module topology related configuration based on the Node Management Service Set (see IEC 62769-3 and IEC 61804-4). On any module-configuration related activity, the FDI Server first calls the ValidateModules Action. The EDD Action can run through various states and even perform user dialogs (see description about Actions in IEC 62769-2 and IEC 62769-5). The EDD Methods can maintain (private) information that is global to the Modular Device. The EDD Action can access the module that shall be created.

7.6 Installer

The Installer for the FDI Communication Server executable is optional. Since the FDI specification does not prescribe the implementation platform for FDI Communication Server executables, the FDI Communication Server executable can be also preinstalled on dedicated hardware.

The installer used for the FDI Communication Server executable shall be separated from the FDI Package. Importing the FDI package is a separate procedure (see 7.7.1).

7.7 FDI Communication Package

7.7.1 General

The FDI Server imports the FDI Communication Package like any other FDI Package. Subclause 7.7 specifies the FDI Communication Package details.

With respect to the EDD element of the Package, FDI differentiates between a simple (lightweight) CommunicationServer (see 7.7.2) and a regular (multi-channel) Communication Server (see 7.7.3).

7.7.2 EDD for Lightweight Communication Server

A lightweight Communication Server provides access to a single field device network. It shall provide all configuration capabilities in its main EDD, not in the sub-modules used to expose the connection points. This allows FDI hosts not supporting modular devices to parameterize an FDI Communication Server using the standard FDI user interface mechanisms.

The EDD describing a "lightweight" Communication Server shall follow the IEC 61804-3 specified profile for the Communication Server. But it shall not use the following EDDL syntax constructs:

- COMPONENT,
- COMPONENT_RELATION,
- COMPONENT_FOLDER,
- COMPONENT_REFERENCE,
- EDDL Built-in function ObjectReference.

7.7.3 EDD for Multi-Channel Communication Server

7.7.3.1 General

The required content for an FDI Communication Package EDD element describing an FDI Communication Device is specified in Clause 5. Specific EDD element content for an FDI Communication Server is described in 7.7.3.

The rules defined in 5.2.2 apply.

The PROTOCOL attribute shall not be set.

The COMPONENT declaration shall have an additional attribute PRODUCT_URI. The attribute value holds a string describing the FDI Communication Server's product URI that enables the FDI Server to identify the FDI Communication Server based on the OPC UA Discovery service (see IEC 61804-3). The attribute value corresponds to the RegisterServer argument RegisteredServer:serverUri. The product URI shall contain the company name and the product name.

Example: PRODUCT_URI "urn:Company:ProductName".

7.7.3.2 CommunicationDevice component

The rules defined in 5.2.3 apply.

7.7.3.3 Communication Service component

The rules defined in 5.2.4 apply.

7.7.3.4 Connection Point component

The rules defined in 5.2.5 apply.

7.7.3.5 Connection Point collection

The rules defined in 5.2.6 apply.

7.7.4 COMMANDs in EDDs for FDI Communication Servers

An EDD for an FDI Communication Server shall follow the communication profile CS defined in IEC 61804-3. Because the synchronization between the VARIABLES of the EDD represented as parameters in the FDI servers information model and the actual configuration of the FDI communication server (represented as parameters in the information model of the FDI communication server) is done by the FDI server, there is no need to define COMMANDs in the EDD in order to access the data of the FDI communication server.

However, the communication profile CS allows the definition of COMMANDs in an EDD. This allows an EDD developer to explicitly trigger communication to the FDI communication server. This may be needed in complex EDD methods that require to directly write to the FDI communication server or directly read the actual value before proceeding with the method execution.

The COMMAND does not require addressing information (like NUMBER, SLOT, INDEX, see IEC 61804-3) since the addressing is done via the BrowseName of the FDI communication server and the EDD identifier of the VARIABLE. Instead, the RESPONSE of a READ COMMAND shall contain the VARIABLES to be read and the REQUEST of a WRITE COMMAND shall contain all the VARIABLES to be written.

If such a COMMAND is called by ReadCommand or WriteCommand built-ins in the EDD, the FDI server shall read or write respectively all the VARIABLES defined in the COMMAND.

Since COMMANDs also group access to VARIABLES, an FDI server shall always handle VARIABLES defined in a COMMAND as a unit. For example, when a WRITE COMMAND contains the two VARIABLES, VarA and VarB, and the FDI server needs to write VarA because it was changed by the user, the FDI server also needs to write VarB in the same OPC UA service call, although VarB might not have been changed. The same applies for reading VARIABLES.

OPC UA READ and WRITE service calls return different status codes (see IEC 62541-4). First, they return one service result code per call. Then, they return one operation level result code per parameter to be read or written. Since an EDD COMMAND only supports one RESPONSE_CODE per call, the following mapping applies.

The first status code that is not GOOD shall be used as RESPONSE_CODE with the following order:

- 1) investigate the service result code, then
- 2) investigate the operation level result codes in the order the VARIABLES appear in the COMMAND definition.

If all status codes are GOOD, the RESPONSE_CODE is GOOD.

7.7.5 Documentation

The FDI Communication Package shall provide documentation describing:

- a) the software installation related environment requirements and procedures;
- b) the OPC UA Server configuration if needed.

7.8 Handling and behavior

7.8.1 General

Subclause 7.8 defines the FDI Communication Server handling and behavior rules along the lifecycle beginning with the deployment, start up, bus commissioning, until the communication services' processing. The diagram (see Figure 10) shows the FDI Server-maintained FDI Communication Server status.

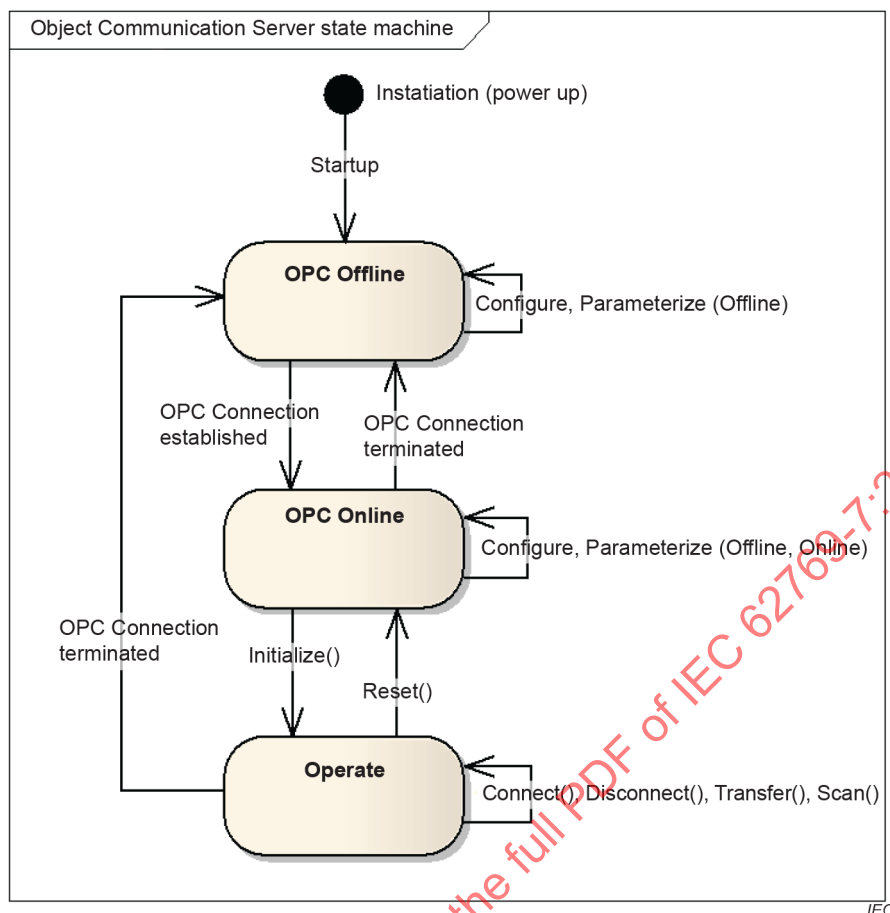


Figure 10 – FDI Communication Server state machine

7.8.2 Deployment

The FDI Server imports the FDI Communication Package. The handling of EDD and UIP parts matches with the import procedure performed for the FDI Package (see IEC 62769-2 and IEC 62769-3). The FDI Communication Package represents the Communication Server Type.

The installation procedure described in the following is optional. (An embedded FDI Communication Server need not provide an installation procedure.)

The FDI Communication Server Installer (see 7.6) is a separate element. The installation procedure is started manually.

Depending on the operating systems, the execution of installation programs could require administration rights.

7.8.3 Server configuration

The FDI Communication Server shall implement the means enabling the OPC UA Server-specific configuration setting the link to the Discovery Server and the name of the FDI Communication Server. In accordance with 7.8.4, the FDI Communication Server needs to know the address information about the Discovery Server. The document does not prescribe the way of how to do this.

The bootstrap process needed to establish the connection between an OPC UA Client and an OPC UA Server requires some administration work. A simple plug and play does not seem possible.

7.8.4 Start up

The following definitions about the Server discovery mechanism refer to the definitions found in IEC 62541-4.

The FDI Communication Server executable shall be started in accordance with one of the following described ways:

- a) The FDI Communication Server executable is loaded by means of the configured operating system function. If the FDI Communication Server is installed on a hardware separated from the FDI Server, the FDI Communication Server executable shall be loaded by means of the configured operating system function (auto-start).
- b) The FDI Server invokes the FDI Communication Server executable process. The related functions are specific to the operating system and the system vendor implementations.

The starting FDI Communication Server process shall register itself at a Discovery Server using the service RegisterServer. This enables OPC UA Clients to obtain information about the connected FDI Communication Server including the application description, existing endpoints and security information. The related OPC UA Services are FindServers and GetEndPoints. The Discovery Server is a process running outside the FDI Communication Server.

After starting up, the FDI Communication Server has the status "OPC Offline".

7.8.5 Shutdown

The following definitions about the Server discovery mechanism refer to the definitions found in IEC 62541-4.

The shutdown of the FDI Communication Server process shall unregister itself at the Discovery Server using the service RegisterServer by setting argument isOnline value FALSE.

7.8.6 Watchdog

The following definitions about the Server discovery mechanism refer to the definitions found in IEC 62541-4.

The FDI Communication Server shall periodically use the service RegisterServer to state its ability to receive a connection from the FDI Server. The frequency is 10 min. The FDI Communication Server can envision a VARIABLE for configuration purposes.

7.8.7 Establish the OPC UA connection

The FDI Server connects as an OPC UA Client to the FDI Communication Server in accordance with the IEC 62541 series.

The FDI Server (OPC UA Client) establishes a secure connection to the FDI Communication Server (OPC UA Server) using the SecureChannel service set defined in IEC TR 62541-1. The functional principles are defined in IEC TR 62541-1.

The communication between the FDI Server (OPC UA client) and the FDI Communication Server (OPC UA server) shall be based on the OPC UA TCP transport protocol with the OPC UA Binary Encoding and OPC UA Secure Conversation.

After successfully establishing the OPC UA connection, the FDI Communication Server enters the status "OPC Online".

7.8.8 Instantiate the Communication Server

The creation of a CommunicationServer instance in the FDI Server-hosted Information Model works the same way as the instantiation of a Device.

7.8.9 Configure the communication hardware

As described in 7.3.1 the FDI Communication Server can support the configuration of modular communication hardware. The modular communication hardware configuration shall be performed via the services AddComponent and RemoveComponent (7.3.2.4, 7.3.2.5).

If the Action ValidateModules (see 5.2.9) is implemented, the FDI Server shall invoke this method. If the Action result is OK, the FDI Server shall then perform the synchronization by matching the sub-device's browse name and by invoking the FDI Communication Server-hosted module's management services.

If the FDI Communication Server supports modular communication hardware configurations, the correct communication hardware configuration is a prerequisite for successful initialization, as described in 7.8.12.

7.8.10 Configuring the Network

The COMPONENT declaration defined in 5.2.2, 5.2.3, 5.2.4, and 5.2.7 enables a description-based approach for the FDI Server to configure the network connections. The FDI Communication Server's ValidateNetwork Action (see 5.2.8) can be added to support the network topology validation function as described in 5.2.8. The network topology is only present in the FDI Server-hosted Information Model (see 7.5).

7.8.11 Parameterize

The FDI Communication Server can require proper bus parameter adjustment prior to any communication service processing. The FDI Communication Package-contained user dialogs (UIP or UID) enable interactive bus parameter adjustment. The FDI Communication Package can contain additional Business Logic for bus parameterization purposes. The editing of FDI Communication Server parameters changes the content of the FDI Server-hosted Information Model. The FDI Server shall perform synchronization using the parameters' browse names. The FDI Server copies the modified values from the FDI Server-hosted Information Model to the FDI Communication Server-hosted Information Model.

A simple FDI Communication Server shall provide all its configuration capabilities in its main EDD, not in the sub-modules used to expose the Connection Points. This allows FDI hosts not supporting modular devices to parameterize an FDI Communication Server using the standard FDI user interface mechanisms.

NOTE The FDI Server can change Parameter values in an arbitrary order.

7.8.12 Initialize

On invocation of the method Initialize (see 7.3.2.3), the FDI Communication Server shall use the current parameter settings and communication hardware configuration for communication hardware initialization purposes.

After successful Initialization, the FDI Communication Server enters the status "Operate".

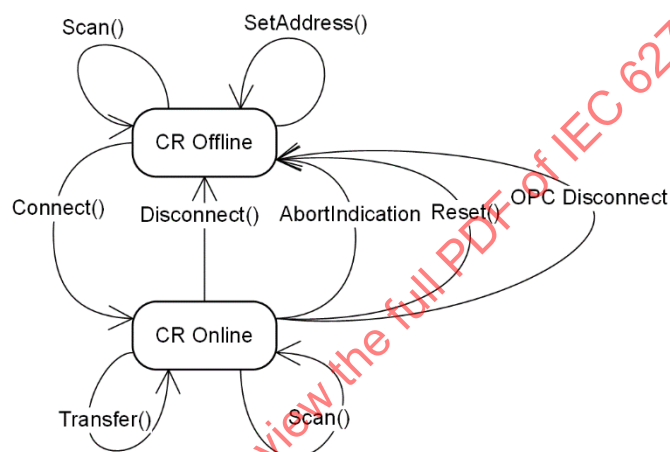
7.8.13 Create the communication service object

Prior to running any data exchange, at least one instance of type `ServerCommunicationServiceType` has to be present or created. One instance of `ServerCommunicationServiceType` shall be always present. Further instances of `ServerCommunicationServiceType` can be created if the instance of `ServerCommunicationDeviceType` implements the method `AddComponent`.

7.8.14 Communication relation

The definitions of Clause 6 apply. The FDI Communication Server specifics are defined in the subsequent text.

The state chart in Figure 11 describes the state flow of a single communication relation with added status changes that are related to OPC UA specifics. Beside the specific aspects, the definitions in Clause 6 apply as well.



IEC

Figure 11 – Communication relation state chart

On invocation of the method `Reset` or the OPC Connection termination (OPC Connection loss) the FDI Communication Server shall terminate all communication relations.

The FDI Communication Server shall reject any parameterization or configuration change attempt in the status "Operate".

The FDI Communication Server shall reject any communication relation related operation if the FDI Communication Server status is different than "Operate".

If the Communication Server supports multiple instances of `ServerCommunicationServiceType`, these instances need to share information about existing communication relations.

7.8.15 Connect

Prior to running any information exchange related communication, the FDI Communication Server requires the establishment of a communication relation between the device application and the device connected to the physical network. This happens through invocation of the method `Connect`.

The FDI Communication Server shall be able to manage multiple communication relations. After successful execution of the method Connect the corresponding communication relation enters the status "CR Online".

Because of the direct association of method Connect to a single communication service provider instance the communication device knows the corresponding physical network connection.

7.8.16 Disconnect

Invocation of the method Disconnect terminates a communication relation, which inhibits further information exchange related communication with the device connected to the physical network.

After execution of the method Disconnect, the corresponding communication relation enters the status "CR Offline". The communication relation becomes invalid.

7.8.17 Abort Indication

Depending on protocol specifics, the FDI Communication Server can detect communication aborts. Such communication abort indications are returned as communication service results during processing of the methods Transfer or Scan. After the FDI Communication Server has returned an Abort Indication, the current communication relation enters the status "CR Offline". The communication relation becomes invalid.

7.8.18 Scan

The topology scan function can be invoked independently from an existing communication relation. Scan service details are specified in 7.3.3.2.

7.8.19 SetAddress

It depends on the protocol whether the address assignment service shall work even when a communication relation is already established.

8 FDI Communication Gateway definition

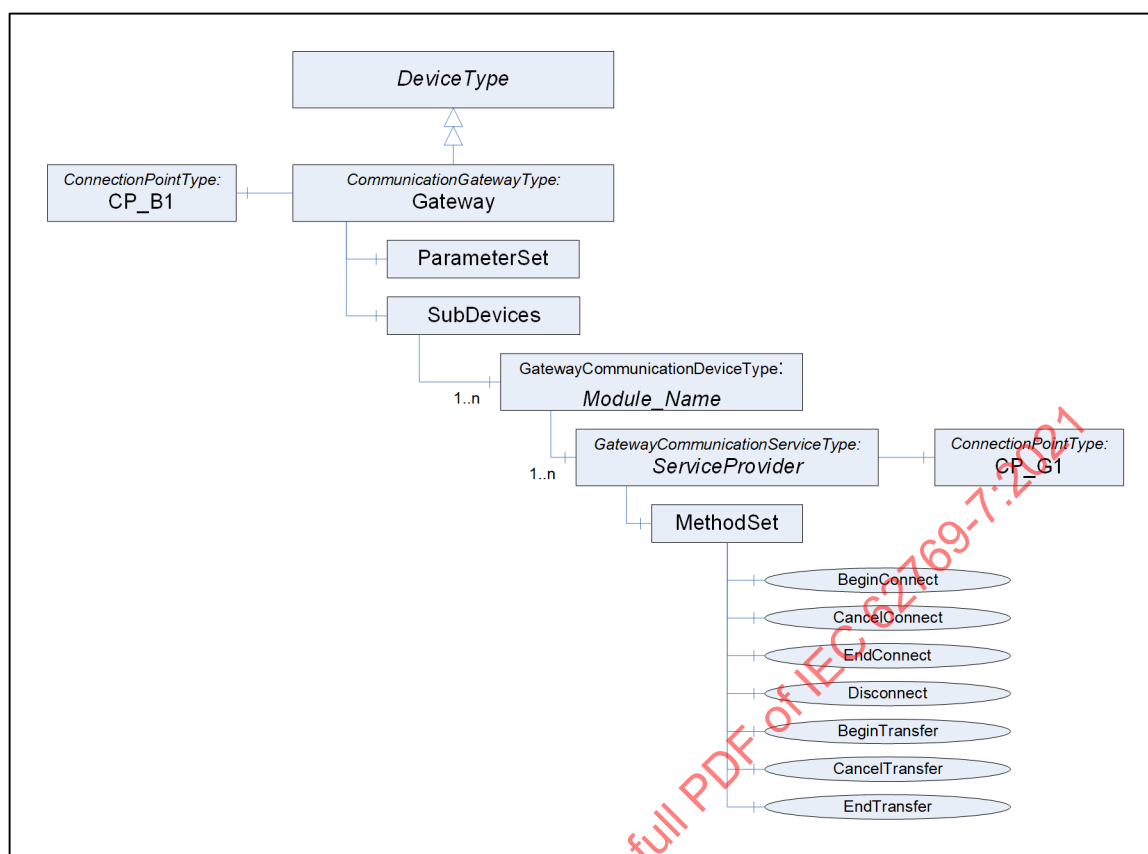
8.1 General

A Gateway is a communication device that enables bridging between different physical networks or different protocols. The logical representation of a Gateway device within the FDI Server-hosted Information Model enables the FDI Server to process communication in heterogeneous network topologies.

8.2 Information Model

8.2.1 General

The Information Model of a Gateway is based on the Information Model defined in IEC 62769-5. Figure 12 replicates the Modular Device structure and its integration in the overall FDI Server-hosted Information Model.



IEC

Figure 12 – Gateway information model

The Gateway is connected to the Network (see IEC 62769-5) through an instance of a ConnectionPointType (CP_B1). CP_B1 represents the FDI Server-assigned object (see IEC 62769-5) identification (name).

The Gateway is an instance of DeviceType. The optionally available ParameterSet (see IEC 62769-5) shall contain all device parameters that parameterize the communication interface used for Gateway-initiated communication service requests.

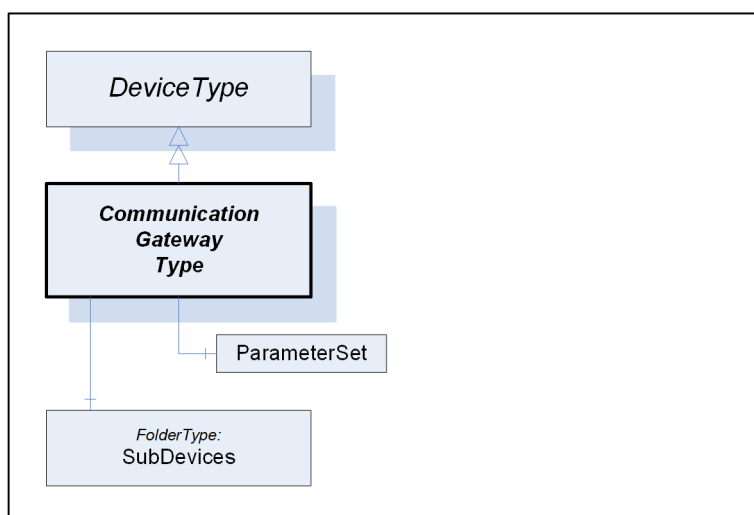
The elements underneath SubDevices represent the physical or logical access to the network media of the Gateway. The attribute Module_Name represents the FDI Server-assigned object (see IEC 62769-5) identification (browse name).

The Gateway's communication service processing capabilities are accessible through multiple communication service provider instances created from GatewayCommunicationServiceType. The Business Logic behind the service methods implements the protocol translation function that is associated with the communication service interface.

NOTE Compared to the FDI CommunicationServer, the Gateway does not support the transport of unsolicited messages, see 7.3.4.5.

8.2.2 CommunicationGatewayType

The CommunicationGatewayType is a subtype of the DeviceType. Figure 13 shows the CommunicationGatewayType definition. It is formally defined in Table 27.



IEC

Figure 13 – CommunicationGatewayType

Table 27 – CommunicationGatewayType definition

Attribute	Value				
BrowseName	CommunicationGatewayType				
IsAbstract	False				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
Subtype of the DeviceType defined in IEC 62541-100.					
HasComponent	Object	SubDevices		FolderType	Mandatory
HasComponent	Object	ParameterSet			Optional

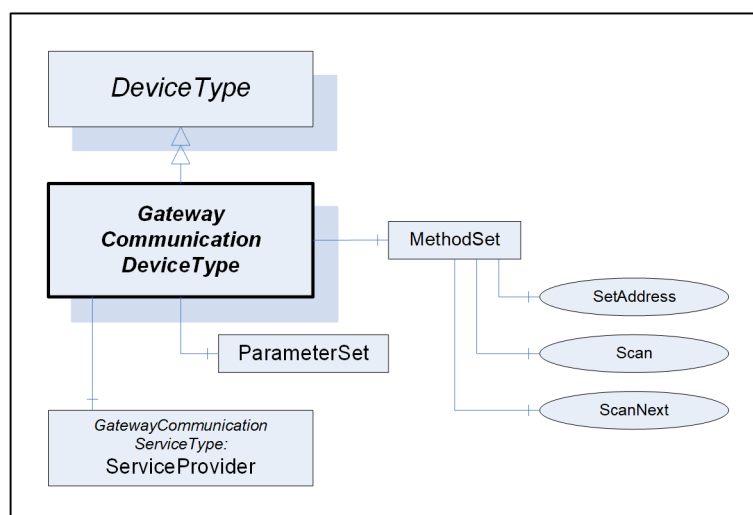
The module management needed to support a configurable communication hardware structure is based on the COMPONENT definitions for the entire CommunicationGatewayType. The COMPONENT definitions provide sufficient information to run the generic module setup configuration.

NOTE The indication for a static communication hardware layout is present with the COMPONENT attribute CAN_DELETE set to FALSE for all COMPONENT declarations related to the sub-devices of the entire device.

8.2.3 GatewayCommunicationDeviceType

8.2.3.1 General

The GatewayCommunicationDeviceType represents a communication module or channel connected to a particular network. The GatewayCommunicationDeviceType is a subtype of the DeviceType. The ParameterSet for each GatewayCommunicationDeviceType will contain Parameters necessary to configure the operation of the network. The protocol-specific, mandatory bus parameters are specified in IEC 62769-4:2020, Annex F. Figure 14 shows the GatewayCommunicationDeviceType definition that is formally defined in Table 28 and Table 29.



IEC

Figure 14 – GatewayCommunicationDeviceType

Table 28 – GatewayCommunicationDeviceType definition

Attribute	Value				
BrowseName	GatewayCommunicationDeviceType				
IsAbstract	True				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
Subtype of the DeviceType defined in OPC UA Part 11.					
HasComponent	Object	MethodSet		BaseObjectType	Optional
HasComponent	Object	ParameterSet		BaseObjectType	Optional
HasComponent	Object	ServiceProvider		Gateway Communication ServiceType	Mandatory

Table 29 – MethodSet of GatewayCommunicationDeviceType

Attribute	Value				
BrowseName	MethodSet				
IsAbstract	True				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
HasComponent	Method	Scan			Optional
HasComponent	Method	ScanNext			Optional
HasComponent	Method	SetAddress			Optional

8.2.3.2 Scan method

Scan shall be used to start discovering devices connected to the physical network. The associations between the method Scan and the corresponding physical network connection enables the FDI Communication Server to access the correct physical network connection. The Scan method is implemented by an EDDL-based method.

Because EDDL logic is not designed to handle XML documents, the Scan service signature deviates from the definition given in 7.3.3.2. The scan service implementation returns the discovered devices using the LIST construct containing the COLLECTION instances. Details are specified in 8.2.3.4. The COLLECTION instances represent the Connection Point instances.

The signature of this Method is specified below. Table 30 and Table 31 specify the arguments and AddressSpace representation, respectively.

Signature

Scan([out] Integer ServiceError)

Table 30 – Scan Method arguments

Argument	Description
ServiceError	0: OK 1: OK/get complete scan result by calling ScanNext -1: Failed/not initialized -2: Failed/not connected to a network -3: Failed/no device found the topologyScanResult is empty

Table 31 – Scan Method AddressSpace definition

Attribute	Value				
BrowseName	Scan				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory

8.2.3.3 ScanNext method

ScanNext shall be used to continue discovering devices connected to the physical network. The associations between the method Scan and the corresponding physical network connection enables the FDI Communication Server to access the correct physical network connection. The Scan method is implemented by an EDDL-based method.

Because EDDL logic is not designed to handle XML documents, the Scan service signature deviates from the definition given in 7.3.3.2. The scan service implementation returns the discovered devices using the LIST construct containing COLLECTION instances. Details are specified in 8.2.3.4. The COLLECTION instances represent the Connection Point instances.

The signature of this Method is specified below. Table 32 and Table 33 specify the arguments and AddressSpace representation, respectively.

Signature

ScanNext ([out] Integer ServiceError)

Table 32 – ScanNext Method arguments

Argument	Description
ServiceError	0: OK 1: OK/get complete scan result by recalling ScanNext -1: Failed/not initialized -2: Failed/not connected to a network -3: Failed/no device found the topologyScanResult is empty

Table 33 – ScanNext Method AddressSpace definition

Attribute	Value				
BrowseName	ScanNext				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory

8.2.3.4 Scan List

Each EDD describing the Connection Point of a Gateway shall describe the LIST element that holds the list of discovered devices after the successful execution of the Scan service.

```
LIST <Id>
{
    TYPE <ListEntry>;
}
```

<ID>: The identifier shall match with the SCAN_LIST attribute value described in 8.3.2.4.

<ListEntry>: The attribute value shall refer to the COLLECTION definition that describes the Connection Point as defined in 8.3.2.4.

8.2.3.5 SetAddress Method

For definitions, see 7.3.3.4.

8.2.4 GatewayCommunicationServiceType

8.2.4.1 General

Communication services provide the means to communicate with a Device or to, for example, execute a Scan on a Network. Communication services are represented through EDDL-based methods (business logic) provided with the FDI Device Package-contained EDD.

The implementation of all communication services except the Disconnect service follows an asynchronous pattern. This implies that each communication service is split into three methods. This allows the FDI Server to execute communication services in parallel as well as cancel operations that last too long.

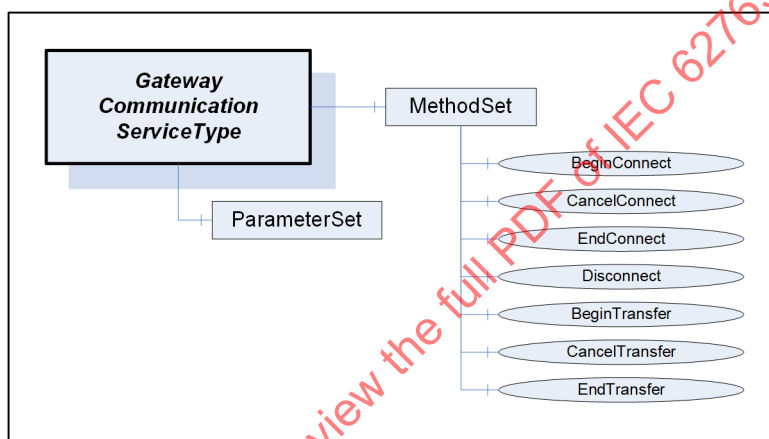
Begin<name>: this method starts the execution of the service. The method returns either the execution state of <name> or the result if it is immediately present.

End<name>: this methods checks, if the result of the service is already available, that it was started using a preceding Begin<name> call. Like Begin<name>, this method returns either the execution state or the result if available.

Cancel<name>: this method cancels a started service execution.

A service identification number (ServiceId) enables the Communication Gateway to keep track of the relation between the method calls that belong to a single communication service process. If the Communication Gateway supports multiple instances of GatewayCommunicationServiceType, these instances share information about currently used ServiceIds. Thus, a communication service has to be executed on a single instance of GatewayCommunicationServiceType.

To reduce unnecessary poll cycles both methods Begin<name> End<name> return a delay time value (DelayForNextCall). The method caller shall delay the invocation of the method End<name> according to the returned argument value.



IEC

Figure 15 – GatewayCommunicationServiceType

Table 34 – GatewayCommunicationServiceType definition

Attribute	Value				
BrowseName	GatewayCommunicationServiceType				
IsAbstract					
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
Subtype of the DeviceType defined in OPC UA Part DI.					
HasComponent	Object	MethodSet		BaseObjectType	Mandatory
HasComponent	Object	ParameterSet		BaseObjectType	Optional

Table 35 – MethodSet of GatewayCommunicationServiceType

Attribute	Value				
BrowseName	MethodSet				
IsAbstract					
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
HasComponent	Method	BeginConnect			Optional
HasComponent	Method	CancelConnect			Optional
HasComponent	Method	EndConnect			Optional
HasComponent	Method	Disconnect			Optional
HasComponent	Method	BeginTransfer			Mandatory
HasComponent	Method	CancelTransfer			Mandatory
HasComponent	Method	EndTransfer			Mandatory

8.2.4.2 Connect service

The Connect service shall be used to establish a communication relation to a device that is physically connected to the Network. Establishing the communication relation may imply checks of identification data that are part of the addressData with data inside the physical device. The service performs this DeviceType match verification in accordance with a corresponding network protocol standard. Related details are specified in IEC 62769-4:2020, Annex F.

The devices' address is contained in the ConnectionPoint of the corresponding Device Instance within the Information Model (Device Connection Point). The communication relation between the Information Model associated-device application and the physical device is further on identified by the communication-relation identifier. Details about how to manage the status of a communication relation are described in Clause 6.

NOTE 1 As the NodeId is a unique identifier within the Information Model scope, the NodeId of the Device Connection Point can be a unique identifier for any communication relation in the scope of a communication device.

NOTE 2 The term "communication relation" is introduced to describe the status of an infrastructure that enables data exchange between the information model-hosted data and a physical device. If the communication relation is established, data exchange is possible.

The signatures of the connect service methods are specified below. Table 36 specifies the arguments.

Signature

```

BeginConnect (
    [in]   ByteString      CommunicationRelationId,
    [in]
    [out]  <AddressData>,
    [in]   UInt32          ServiceID,
    [out]  UInt32          DelayForNextCall,
    [out]  Int32           ServiceError);

EndConnect (
    [in]   ByteString      CommunicationRelationId,
    [out]  <DeviceInformation>,
    [in]   UInt32          ServiceID,
    [out]  UInt32          DelayForNextCall,
    [out]  Int32           ServiceError);

CancelConnect (
    [in]   ByteString      CommunicationRelationId,
    [in]   UInt32          ServiceID,
    [out]  Int32           ServiceError);

```

Table 36 – Connect Method arguments

Argument	Description
CommunicationRelationId	This is a client generated ID that is used to uniquely identify this connection. This could be an index (e.g. a NodeId) that the client (= FDI Server) needs in order to identify entries in its topology.
<AddressData>	This is a placeholder for a protocol-specific argument list that is used for the address and optional device identification data (details are described in IEC 62769-4:2020, Annex F.
<DeviceInformation>	A protocol-specific argument list in which the connect routine stores the resulting data.
ServiceId	The service transaction code establishes the relation between the service request and the corresponding response.
DelayForNextCall	The value specifies a delay time in ms the caller shall wait before the next invocation of EndConnect.
ServiceError	0: OK/function started asynchronously, result has to be polled with EndConnect 1: OK/execution finished, connection established successfully -1: Connect Failed/cancelled by caller -2: Call Failed/unknown service ID -3: Connect Failed/device not found -4: Connect Failed/invalid device address -5: Connect Failed/invalid device identification

8.2.4.3 Disconnect method

Specified in 7.3.4.3.

8.2.4.4 Transfer service

Transfer shall be used to perform information exchange with a Device.

The signatures of the Transfer service methods are specified below. All arguments are specified in Table 37.

Signature

```

BeginTransfer (
    [in]   ByteString      CommunicationRelationId,
    [in]   <SendData>,
    [out]  <ReceiveData>,
    [in]   UInt32          ServiceId,
    [out]  UInt32          DelayForNextCall,
    [out]  Int32           ServiceError);

EndTransfer (
    [in]   ByteString      CommunicationRelationId,
    [out]  <ReceiveData>,
    [in]   UInt32          ServiceId,
    [out]  UInt32          DelayForNextCall,
    [out]  Int32           ServiceError);

CancelTransfer (
    [in]   ByteString      CommunicationRelationId,
    [in]   UInt32          ServiceId,
    [out]  Int32           ServiceError);

```

Table 37 – Transfer Method arguments

Argument	Description
CommunicationRelationId	See 8.2.4.2.
<SendData>	This is a placeholder for a protocol-specific list of arguments as described in IEC 62769-4:2020, Annex F. The argument values represent the protocol-specific communication service request that is sent to the device.
<ReceiveData>	This is a placeholder for a protocol-specific list of arguments as described in IEC 62769-4:2020, Annex F. The argument values represent the protocol-specific communication service response that is received from the device.
ServiceId	The service transaction code establishes the relation between the service request and the corresponding response.
DelayForNextCall	The value specifies a delay time in ms the caller shall wait before the next invocation of EndTransfer.
ServiceError	0: OK/function started asynchronously, result has to be polled with EndTransfer 1: OK/execution finished, ReceivedData contains the result -1: Transfer Failed/cancelled by caller -2: Call Failed/unknown service ID -3: Transfer Failed/no existing communication relation -4: Transfer Failed/invalid communication relation identifier -5: Transfer Failed/invalid sendData content -6: Transfer Failed/invalid receiveData format

8.3 FDI Communication Package

8.3.1 General

Subclause 8.3 specifies the FDI Communication Package details that are specific for Gateways. The definitions given in 5.1 apply also.

8.3.2 EDD

8.3.2.1 General

The definitions in 5.2 apply. Additionally, the EDD elements as specified in IEC 62769-4 and provided with a Gateway-specific FDI Communication Package shall contain:

- a) A PROFILE (IEC 61804-3): the PROFILE Definition shall be chosen according to the protocol used for communication service requests.
- b) A Business Logic: the communication service provider related EDD COMPONENTs shall implement the methods specified in IEC 61804-3. These methods implement the protocol bridging logic. The translation procedures open out into outbound communication requests via the EDDL Built-in library function invocation or the writing of data into an online node. The set of usable EDDL Built-in library functions is bound to the PROFILE.
- c) A Module management: the Gateway related Component can implement the ValidateModules (see 5.2.9). The implemented logic shall validate individual changes and handle any parameter-related dependencies for the whole Gateway device. The implementation of ValidateModules is optional when the actual product-specific COMPONENT declaration is sufficient to configure the module setup without any additional Business Logic.

8.3.2.2 Gateway Component

Each FDI Communication Package describing a Gateway shall contain an EDD element describing the Gateway, as defined in 5.2.2. Gateway specifics are described in the following:

```
COMPONENT <DeviceComponentId>
{
    LABEL "<Label>";
    CAN_DELETE TRUE;
    CHECK_CONFIGURATION <ValidateModules>;
    CLASSIFICATION NETWORK_COMPONENT;
    COMPONENT_RELATIONS
    {
        <CommunicationDeviceRelationId>
    }
    PROTOCOL <ProtocolId>;
}
```

<ProtocolID>: the existence of this attribute indicates the connectivity of the Gateway regarding outbound communication. It allows the FDI Server to find the network using the same type of protocol to which this Gateway can be connected. For standardized protocols, the value is defined by the related fieldbus organization.

8.3.2.3 Gateway CommunicationDevice Component

Each FDI Communication Package describing a Gateway shall contain at least one EDD element describing at least one CommunicationDevice component, as defined in 5.2.3.

NOTE A Gateway is sometimes referred to as "Remote IO". Remote IO is a Modular Device supporting multiple various module types that can be flexibly assigned to any slot. Thus it is possible to create multiple different Remote IO configurations (n – slots $\times$ m – module types).

The rules about COMPONENT attribute settings shown in 5.2.3 apply. Gateway specifics are described in the following:

```
COMPONENT <CommunicationDeviceComponentId>
{
    LABEL "<Label>";
    CAN_DELETE <CanDelete>;
    CLASSIFICATION NETWORK_COMPONENT;
    COMPONENT_RELATIONS
    {
        <CommunicationServiceProviderRelationId>
    }
    SCAN <Scan>;
    SCAN_LIST <ScanList>;
}
```

<Scan>: the attribute refers to the METHOD implementing the device discovery service. The reference works because the identifier value of the METHOD matches with the attribute value. Implementation details are specified in 8.2.3.2.

<ScanList>: the attribute value refers to the LIST describing the topology scan results. This list shall contain all devices discovered during execution of the device discovery service. Implementation details are specified in 8.2.3.4.

8.3.2.4 Communication Service component

The rules defined in 5.2.4 apply.

Additionally, the file describing the component contains the implementations of the methods defined in 8.2.4.2, 8.2.4.3 and 8.2.4.4. The identifier used for the related METHOD constructs matches with the method names specified in Table 35.

8.3.2.5 Connection Point Component

The rules defined in 5.2.5 apply.

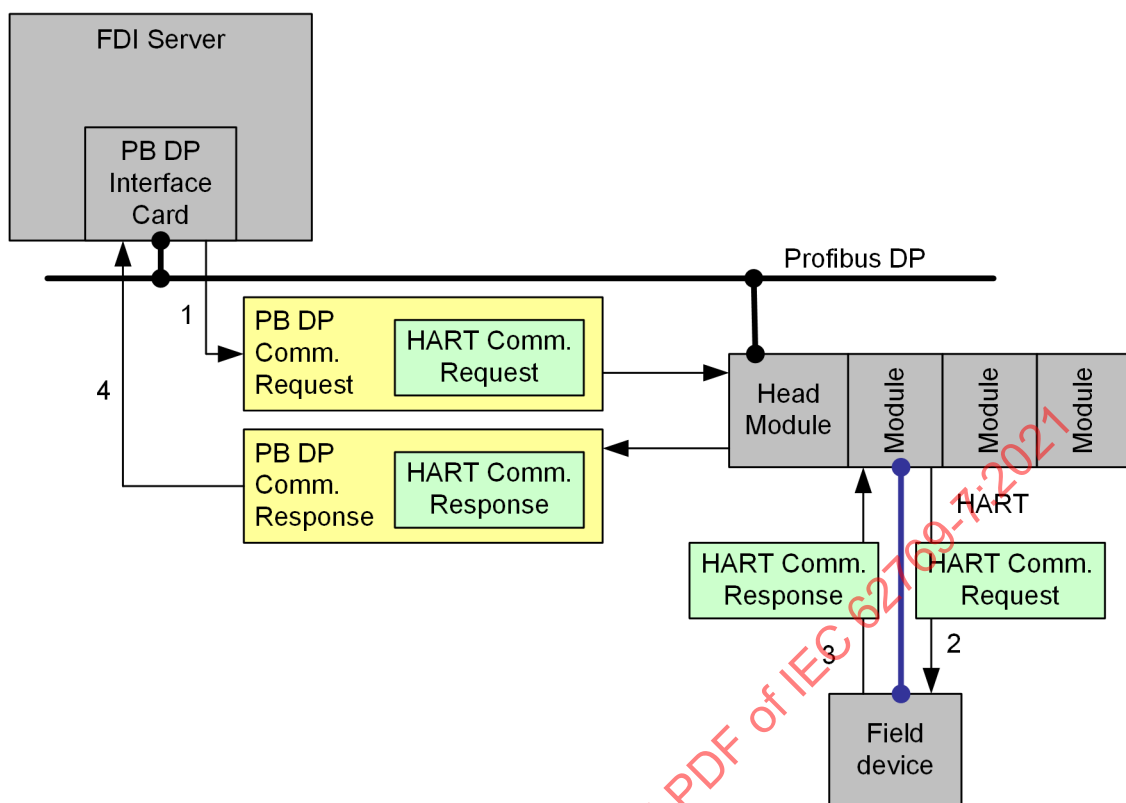
8.3.2.6 Connection Point Collection

The rules defined in 5.2.6 apply.

8.4 Handling and behavior

8.4.1 General

A Gateway provides functionality to communicate between two communication protocols. Gateways are used to communicate from one network of the automation system to another (subordinated) network. Figure 16 shows a typical example where a HART device is connected to a PROFIBUS Remote I/O. In order to communicate to the HART device, the Remote I/O receives a communication request via the PROFIBUS network (see Figure 16, key 1). The communication request contains the necessary information that allows the gateway to create the according HART Command and send it to the HART device (see Figure 16, key 2). The way the HART Command is wrapped into the PROFIBUS communication request may be standard or Gateway-specific. The HART response (see Figure 16, key 3) from the device is embedded as a PROFIBUS communication response (see Figure 16, key 4). The way the Gateway wraps the response may either be standard or Gateway-specific.



IEC

Figure 16 – Nested Communication

Subclause 8.4 defines the Gateway handling and behavior rules along the lifecycle beginning with the deployment, start up, bus commissioning, until the processing of communication services.

8.4.2 Deployment

The definitions of 5.2.11 apply.

8.4.3 Start up

The Information Model and the FDI Package EDD element-based Gateway representation does not require any start-up procedures.

8.4.4 Configure the communication hardware

The handling and behavior matches with the specification in 7.8.9. The only difference is in the channel setup that is represented in the FDI Server-hosted Information Model only.

8.4.5 Configure the Network

The handling and behavior matches with the specification in 7.8.10.

8.4.6 Parameterize

The Gateway can require proper parameter adjustment prior to any communication service processing. The FDI Communication Package-contained user dialogs (UID or UIP) enable interactive bus parameter adjustment. The FDI Communication Package can provide additional Business Logic for bus parameterization purposes.

8.4.7 Communication relation

The status machine definitions in Clause 6 apply.

If the Communication Gateway supports multiple instances of GatewayCommunicationServiceType, these instances need to share information about existing communication relations.

8.4.8 Connect

Prior to running any information exchange-related communication, the Gateway requires the establishment of a communication relation between the device application and the device connected to the physical network. This happens through the invocation of the method Connect. This enables the Gateway to perform an optional outbound communication service request, which might be needed for a specific Gateway device to establish a communication relation.

The Gateway shall be able to manage multiple communication relations. After successful execution of the method Connect the corresponding communication relation enters the status "CR Online".

Invocation of the services Transfer and Scan is allowed in the status "Online" only.

8.4.9 Disconnect

Invocation of the method Disconnect terminates a communication relation, which inhibits further information exchange related communication with the device connected to the physical network.

After execution of the method Disconnect, the corresponding communication relation enters the status "CR Offline". The communication relation becomes invalid.

8.4.10 Abort indication

Depending on protocol specifics, the Gateway can detect communication aborts. Such communication abort indications are returned as communication service results during the processing of the methods Transfer or Scan. After the Gateway has returned an Abort Indication, the current communication relation enters the status "CR Offline". The communication relation becomes invalid.

8.4.11 Scan

Gateways declare their device discovery service capability based on the EDDL construct COMPONENT – the attributes SCAN and SCAN_LIST. Related definitions are defined in IEC 61804-3. The SCAN attribute setting is mandatory within the EDD element describing the Gateway CommunicationDevice. The attribute value refers to the METHOD executing the topology scan function. The SCAN_LIST attribute setting is mandatory within the Gateway CommunicationDevice COMPONENT. The attribute value refers to the element that contains the topology scan result created by the method referenced by the attribute value SCAN. The SCAN_LIST shall refer to a LIST containing COLLECTION elements describing the detected devices (Scan-List-Item). The protocol-specific content of a Scan-List-Item is described in IEC 62769-4:2020, Annex F.

The invocation of the functions Scan and ScanNext results in outbound communication service requests.

8.4.12 Communication Error Handling

The communication service processing shall use the EDDL Built-in function provided for aborting in accordance with the EDDL Profiles:

- during the creation of communication messages,
- during the processing of the response from the communication request.

The communication service processing does not trigger communication errors based on communication timeouts.

IECNORM.COM : Click to view the full PDF of IEC 62769-7:2021

Annex A (informative)

Layered protocols

A.1 General

Ethernet-based protocols commonly consist of a stack of different protocols based on the ISO/OSI model. Looking at the growing number of Ethernet-based fieldbus protocols, it is crucial to have a common layered modeling concept for Connection Points based on the ISO/OSI model also.

Connection Points are the elements that contain the address information accessed by the Communication Devices to collect the information needed for communication. The semantics of Connection Point attributes need to be standardized.

The PROFINET device may concurrently support PROFINET, SNMP and HTTP. The information stored in the Connection Point of a device is different for each protocol due to different application layers. The information for layers 1 to 4 shall consistently hold the same information. Therefore, Connection Points shall inherit Connection Point information from lower network layers.

The problem is about how to ensure that address information from lower layers is named in a consistent way throughout all protocols that are built upon the lower layers.

A.2 Convention for protocol-specific annex creation

Since Connection Point description is based on EDDL, an actual inheritance approach known from object-oriented programming languages doesn't seem applicable. The approach described in Clause A.2 is based more on conventions. The naming convention shall ensure that the address value attribute names defined for a "lower-level" protocol are reused in the Connection Point definitions for higher-level protocols. The same holds true for the COLLECTION referred VARIABLES. VARIABLE declarations for higher-level protocols shall be copies from the VARIABLE declarations for lower-level protocols. The convention described here is applicable for the creation of protocol-specific annexes. The following shows some EDDL examples of how Connection Point declarations follow this naming convention.

```
COLLECTION ConnectionPoint_MAC
{
    LABEL "<Label>";
    MEMBERS
    {
        MAC,
        VALID
    }
}
```

```
COLLECTION ConnectionPoint_IPv4
{
    LABEL "<Label>";
    MEMBERS
    {
        MAC,
        IPv4,
        VALID
    }
}
```

```
COLLECTION ConnectionPoint_TCPUDP
```

```

{
    LABEL "<Label>";
    MEMBERS
    {
        MAC,
        IP,
        PORT,
        VALID
    }
}

COLLECTION ConnectionPoint_PROFINET
{
    LABEL "<Label>";
    MEMBERS
    {
        MAC,
        IP,
        PORT,
        DNSNAME,
        VALID
    }
}

COLLECTION ConnectionPoint_HTTP
{
    LABEL "<Label>";
    MEMBERS
    {
        MAC,
        IP,
        PORT,
        URL,
        VALID
    }
}

```

A.3 FDI Communication Package definition

A.3.1 Communication services

The actual communication service is always implemented in accordance with the specific protocol. This is reflected by the different semantic of the communication service arguments specified by the protocol-specific annexes. So, the actual implementation of a ServerCommunicationDeviceType or GatewayCommunicationDeviceType can support just one protocol. Thus, if an FDI Communication Server or Gateway is able to support multiple different protocols, it shall describe separate GatewayCommunicationDeviceTypes or ServerCommunicationDeviceTypes. The need to define protocol-specific service sets represents the demand to separate the Connection Point and Network-related COMPONENT definitions described in A.3.2 and A.3.3.

A.3.2 Connection Point

The FDI Communication Package shall contain separate Connection Point descriptions for each supported protocol.

A.3.3 Network

The relation between the COMPONENT describing the network and the COMPONENT describing the Connection Point enables generic communication path detection and generic topology configuration. Thus, the FDI Communication Package shall contain a separate COMPONENT definition for each supported Network (protocol).

A.4 Representation in the IM

Connection Points sharing a certain set of address formation may contain redundant address information, for example the IP address is the same for the SNMP and PROFINET I/O.

If a Device and an FDI Communication Server share a set of protocols, then that Device and FDI Communication Server are associated through multiple separate networks.

A Device supporting multiple protocols can be connected to different FDI Communication Devices that support only one protocol.

IECNORM.COM : Click to view the full PDF of IEC 62769-7:2021

Annex B (normative)

Namespace and mappings

This annex defines the numeric identifiers for all of the numeric *NodeIds* defined in this document. The identifiers are specified in a CSV file with the following syntax:

<SymbolName>, <Identifier>, <NodeClass>

Where the *SymbolName* is either the *BrowseName* of a *Type Node* or the *BrowsePath* for an *Instance Node* that appears in the specification and the *Identifier* is the numeric value for the *NodeId*.

The *BrowsePath* for an *Instance Node* is constructed by appending the *BrowseName* of the instance *Node* to the *BrowseName* for the containing instance or type. An underscore character is used to separate each *BrowseName* in the path.

The *NamespaceUri* <http://fdi-cooperation.com/OpcUa/FDI7/> is applied to *NodeIds* defined here.

The CSV released with this version of the document is provided by FieldComm Group™, see <https://www.fieldcommgroup.org>. An electronic version of the complete Information Model defined in this document is also provided. It follows the XML Information Model Schema syntax defined in IEC 62541-6.

The Information Model Schema released with this version of the document is provided by FieldComm Group™.

IECNORM.COM : Click to view the full PDF of IEC 62769-7:2021

SOMMAIRE

AVANT-PROPOS	70
INTRODUCTION	72
1 Domaine d'application	73
2 Références normatives	74
3 Termes, définitions, termes abrégés et conventions	74
3.1 Termes et définitions	74
3.2 Termes abrégés	74
3.3 Conventions	74
4 Généralités	75
5 Paquetage de Communication FDI	76
5.1 Généralités	76
5.2 Description d'Appareil Électronique (EDD)	76
5.2.1 Règles générales	76
5.2.2 Composant Appareil	77
5.2.3 Composant CommunicationDevice	79
5.2.4 Composant fournisseur de services de communication	80
5.2.5 Composant Point de Connexion	81
5.2.6 Collection du Point de Connexion	82
5.2.7 Composant réseau	82
5.2.8 ValidateNetwork	83
5.2.9 ValidateModules	84
5.2.10 Éléments spécifiques de l'UIP	84
5.2.11 Déploiement	84
6 Relations de communication	85
7 Définition du Serveur de Communication FDI	86
7.1 Généralités	86
7.2 Caractéristiques générales	86
7.3 Modèle d'Information	86
7.3.1 Généralités	86
7.3.2 CommunicationServerType	89
7.3.3 ServerCommunicationDeviceType	93
7.3.4 ServerCommunicationServiceType	96
7.4 Profil de Serveur OPC UA pour le Serveur de Communication FDI	100
7.5 Mapping du Modèle d'Information du Serveur FDI au Modèle d'Information du Serveur de Communication FDI	101
7.5.1 Généralités	101
7.5.2 Différences des Modèles d'Information	101
7.6 Programme d'installation	103
7.7 Paquetage de Communication FDI	103
7.7.1 Généralités	103
7.7.2 EDD pour le Serveur de Communication léger	103
7.7.3 EDD pour le Serveur de Communication multivoie	103
7.7.4 COMMAND dans les EDD pour les Serveurs de Communication FDI	104
7.7.5 Documentation	105

7.8	Traitement et comportement	105
7.8.1	Généralités	105
7.8.2	Déploiement	106
7.8.3	Configuration du Serveur	106
7.8.4	Démarrage	107
7.8.5	Arrêt	107
7.8.6	Chien de garde	107
7.8.7	Établissement de la connexion OPC UA	107
7.8.8	Instanciation du Serveur de Communication	107
7.8.9	Configuration du matériel de communication.....	108
7.8.10	Configuration du Réseau	108
7.8.11	Paramétrage.....	108
7.8.12	Initialisation	108
7.8.13	Création de l'objet de service de communication	108
7.8.14	Relation de communication.....	108
7.8.15	Connect.....	109
7.8.16	Disconnect	109
7.8.17	Indication d'abandon	109
7.8.18	Scan.....	110
7.8.19	SetAddress.....	110
8	Définition de la Passerelle de Communication FDI.....	110
8.1	Généralités	110
8.2	Modèle d'Information.....	110
8.2.1	Généralités	110
8.2.2	CommunicationGatewayType	111
8.2.3	GatewayCommunicationDeviceType	112
8.2.4	GatewayCommunicationServiceType	114
8.3	Paquetage de Communication FDI	118
8.3.1	Généralités	118
8.3.2	EDD	118
8.4	Traitement et comportement	120
8.4.1	Généralités	120
8.4.2	Déploiement	121
8.4.3	Démarrage	121
8.4.4	Configuration du matériel de communication.....	121
8.4.5	Configuration du Réseau	121
8.4.6	Paramétrage.....	121
8.4.7	Relation de communication.....	121
8.4.8	Connect.....	122
8.4.9	Disconnect	122
8.4.10	Indication d'abandon	122
8.4.11	Scan.....	122
8.4.12	Traitement des Erreurs de Communication	122
Annexe A (informative)	Protocoles hiérarchisés	123
A.1	Généralités	123
A.2	Convention relative à la création de l'annexe spécifique au protocole	123

A.3	Définition du Paquetage de Communication FDI	124
A.3.1	Services de communication	124
A.3.2	Point de Connexion	124
A.3.3	Réseau	124
A.4	Représentation dans le Modèle d'Information	125
Annexe B (normative)	Espace de noms et mappings	126
Figure 1	– Diagramme de l'architecture FDI	73
Figure 2	– Architecture de l'infrastructure de communication FDI	75
Figure 3	– Relation de communication	85
Figure 4	– Diagramme états-transitions de la relation de communication	86
Figure 5	– AddressSpace du Serveur de Communication FDI	88
Figure 6	– CommunicationServerType	89
Figure 7	– ServerCommunicationDeviceType	93
Figure 8	– ServerCommunicationServiceType	97
Figure 9	– Différences des Modèles d'Information (exemple)	101
Figure 10	– Diagramme d'états du Serveur de Communication FDI	106
Figure 11	– Diagramme états-transitions de la relation de communication	109
Figure 12	– Modèle d'Information de la Passerelle	110
Figure 13	– CommunicationGatewayType	111
Figure 14	– GatewayCommunicationDeviceType	112
Figure 15	– GatewayCommunicationServiceType	115
Figure 16	– Communication imbriquée	121
Tableau 1	– Arguments de l'Action ValidateNetwork	84
Tableau 2	– Arguments de l'Action ValidateModules	84
Tableau 3	– Définition de CommunicationServerType	89
Tableau 4	– MethodSet de CommunicationServerType	89
Tableau 5	– Arguments de la Méthode Reset	90
Tableau 6	– Définition de l'AddressSpace de la Méthode Reset	90
Tableau 7	– Arguments de la Méthode Initialize	91
Tableau 8	– Définition de l'AddressSpace de la Méthode Initialize	91
Tableau 9	– Arguments de la Méthode AddComponent	92
Tableau 10	– Définition de l'AddressSpace de la Méthode AddComponent	92
Tableau 11	– Arguments de la Méthode RemoveComponent	93
Tableau 12	– Définition de l'AddressSpace de la Méthode RemoveComponent	93
Tableau 13	– Définition de ServerCommunicationDeviceType	94
Tableau 14	– MethodSet de ServerCommunicationDeviceType	94
Tableau 15	– Arguments de la Méthode Scan	95
Tableau 16	– Définition de l'AddressSpace de la Méthode Scan	95
Tableau 17	– Arguments de la Méthode ResetScan	95
Tableau 18	– Définition de l'AddressSpace de la Méthode ResetScan	96
Tableau 19	– Arguments de la Méthode SetAddress	96
Tableau 20	– Définition de ServerCommunicationServiceType	97

Tableau 21 – MethodSet de ServerCommunicationServiceType	97
Tableau 22 – Arguments de la Méthode Connect	98
Tableau 23 – Arguments de la Méthode Disconnect	99
Tableau 24 – Arguments de la Méthode Transfer	99
Tableau 25 – Arguments de la Méthode GetPublishedData	100
Tableau 26 – Définition de <i>FDICommunicationServer_Facet</i>	100
Tableau 27 – Définition de CommunicationGatewayType	111
Tableau 28 – Définition de GatewayCommunicationDeviceType	112
Tableau 29 – MethodSet de GatewayCommunicationDeviceType	113
Tableau 30 – Arguments de la Méthode Scan	113
Tableau 31 – Définition de l'AddressSpace de la Méthode Scan	113
Tableau 32 – Arguments de la Méthode ScanNext	114
Tableau 33 – Définition de l'AddressSpace de la Méthode ScanNext	114
Tableau 34 – Définition de GatewayCommunicationServiceType	116
Tableau 35 – MethodSet de GatewayCommunicationServiceType	116
Tableau 36 – Arguments de la Méthode Connect	117
Tableau 37 – Arguments de la Méthode Transfer	118

IECNORM.COM : Click to view the full PDF of IEC 62769-7:2021

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

INTÉGRATION DES APPAREILS DE TERRAIN (FDI) –

Partie 7: Appareils de communication

AVANT-PROPOS

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

La Norme internationale IEC 62769-7 a été établie par le sous-comité 65E: Les dispositifs et leur intégration dans les systèmes de l'entreprise, du comité d'études 65 de l'IEC: Mesure, commande et automation dans les processus industriels.

Cette deuxième édition annule et remplace la première édition parue en 2015. Cette édition constitue une révision technique.

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

- a) prise en charge de l'extension de protocoles génériques qui visent à accélérer l'adoption d'autres technologies;
- b) prise en charge de nouveaux protocoles;
- c) extension de protocoles génériques qui permettent l'adoption d'autres protocoles de communication;
- d) d'après l'extension de protocoles génériques: Modbus RTU.

Le texte de cette Norme internationale est issu des documents suivants:

FDIS	Rapport de vote
65E/764/FDIS	65E/774/RVD

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

Ce document a été rédigé selon les Directives ISO/IEC, Partie 2.

Une liste de toutes les parties de la série 62769, publiées sous le titre général *Intégration des appareils de terrain (FDI)*, peut être consultée sur le site web de l'IEC.

Le comité a décidé que le contenu de ce document ne sera pas modifié avant la date de stabilité indiquée sur le site web de l'IEC sous "<http://webstore.iec.ch>" dans les données relatives au document recherché. À cette date, le document sera

- reconduit,
- supprimé,
- remplacé par une édition révisée, ou
- amendé.

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

INTRODUCTION

La série IEC 62769 est publiée sous le titre général "*Intégration des appareils de terrain (FDI)*" et comporte les parties suivantes:

- Partie 1: Vue d'ensemble
- Partie 2: Client FDI
- Partie 3: Serveur FDI
- Partie 4: Paquetages FDI
- Partie 5: Modèle d'Information FDI
- Partie 6: Mapping de technologies FDI
- Partie 7: Appareils de Communication FDI
- Partie 100: Profils – Extensions de protocoles génériques
- Partie 101-1: Profils – Foundation Fieldbus H1
- Partie 101-2: Profils – Foundation Fieldbus HSE
- Partie 103-1: Profils – PROFIBUS
- Partie 103-4: Profils – PROFINET
- Partie 109-1: Profils – HART et WirelessHART
- Partie 115-2: Profils – Définitions spécifiques au protocole pour Modbus-RTU
- Partie 150-1: Profils – ISA 100.11a

IECNORM.COM : Click to view the full PDF of IEC 62769-7:2021

INTÉGRATION DES APPAREILS DE TERRAIN (FDI) –

Partie 7: Appareils de communication

1 Domaine d'application

La présente partie de l'IEC 62769 spécifie les éléments de mise en œuvre des fonctions de communication, appelés Appareils de Communication (IEC 62769-5).

L'architecture FDI complète est représentée à la Figure 1. Les composants architecturaux qui relèvent du domaine d'application du présent document ont été mis en évidence dans cette représentation. Le domaine d'application du document relatif aux Paquetages FDI est limité aux Appareils de Communication. Le Serveur de communication représenté à la Figure 1 est un exemple d'Appareil de Communication spécifique.

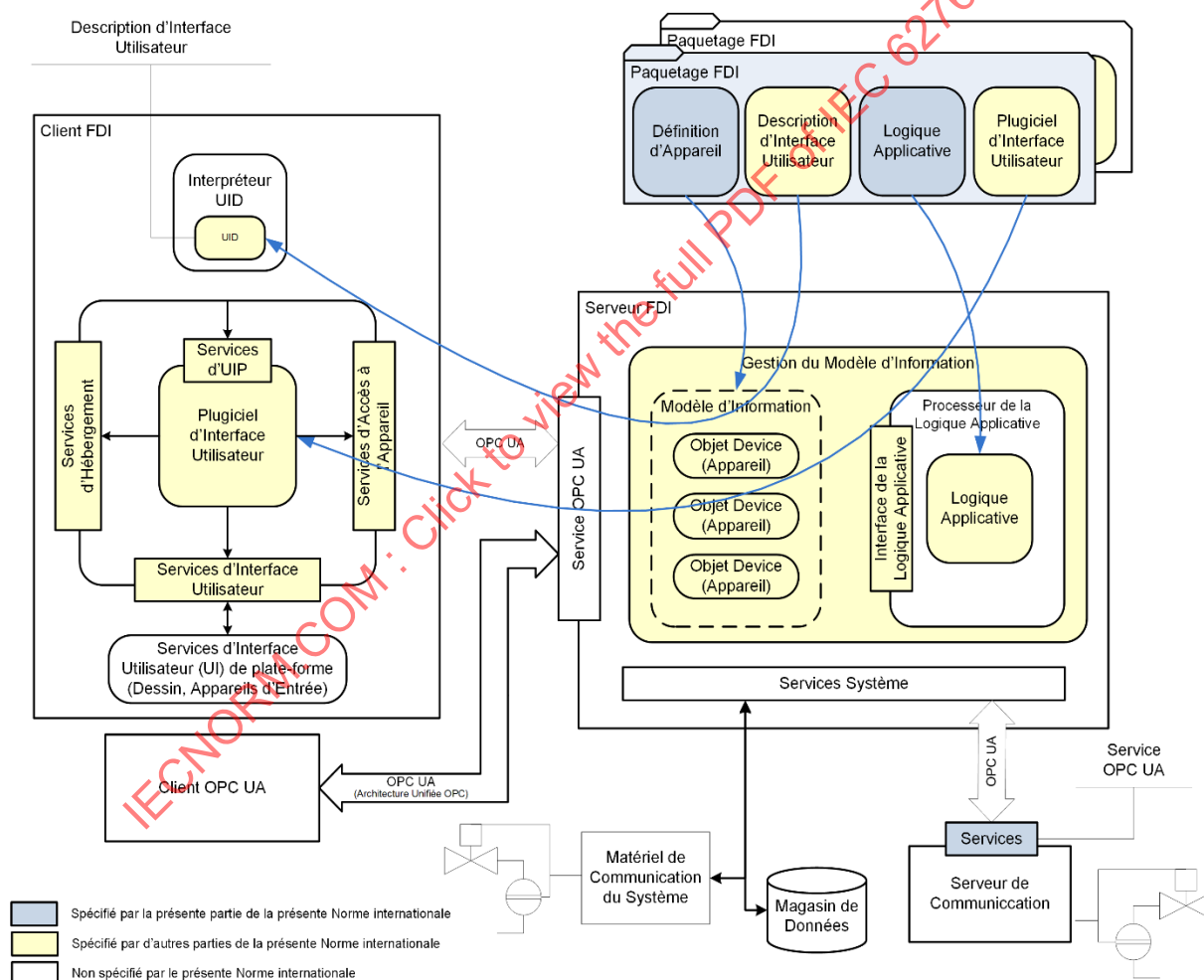


Figure 1 – Diagramme de l'architecture FDI

2 Références normatives

Les documents ci-après, dans leur intégralité ou non, sont des références normatives indispensables à l'application du présent document. Pour les références datées, seule l'édition citée s'applique. Pour les références non datées, la dernière édition du document de référence s'applique (y compris les éventuels amendements).

IEC 61804-3, *Blocs Fonctionnels (FB) pour les procédés industriels et le Langage de Description Electronique de Produit (EDDL) – Partie 3: Sémantique et syntaxe EDDL*

IEC 61804-4, *Blocs fonctionnels (FB) pour les procédés industriels et le langage de description électronique de produit (EDDL) – Partie 4: Interprétation EDD*

IEC 62541 (toutes les parties), *Architecture unifiée OPC*

IEC 62769-1, *Intégration des appareils de terrain (FDI) – Partie 1: Vue d'ensemble*

IEC 62769-2, *Intégration des appareils de terrain (FDI) – Partie 2: Client FDI*

IEC 62769-3, *Intégration des appareils de terrain (FDI) – Partie 3: Serveur FDI*

IEC 62769-4:2020, *Intégration des appareils de terrain (FDI) – Partie 4: Paquetages FDI*

IEC 62769-5, *Intégration des appareils de terrain (FDI) – Partie 5: Modèle d'Information FDI*

3 Termes, définitions, termes abrégés et conventions

3.1 Termes et définitions

Pour les besoins du présent document, les termes et définitions de l'IEC 62769-1, ainsi que les suivants s'appliquent.

L'ISO et l'IEC tiennent à jour des bases de données terminologiques destinées à être utilisées en normalisation, consultables aux adresses suivantes:

- IEC Electropedia: disponible à l'adresse <http://www.electropedia.org/>
- ISO Online browsing platform: disponible à l'adresse <http://www.iso.org/obp>

3.1.1

passerelle

appareil de communication qui permet de relier différents réseaux physiques ou différents protocoles

3.2 Termes abrégés

Pour les besoins du présent document, les termes abrégés de l'IEC 62769-1, ainsi que les suivants s'appliquent.

HTTP	Hypertext Transfer Protocol (protocole de transfert hypertexte)
IP	Internet Protocol (protocole Internet)
PHY	Matériel de communication physique
SNMP	Simple Network Management Protocol (protocole de gestion de réseau simple)
TCP	Transmission Control Protocol (protocole de contrôle de transmission)
URI	Uniform Resource Identifier (identificateur uniforme de ressource)

3.3 Conventions

Pour les besoins du présent document, les conventions de l'IEC 62769-1 s'appliquent.

Le présent document utilise la notation graphique définie dans l'IEC 62769-5.

4 Généralités

Le terme abstrait "Appareil de Communication FDI" représente une entité qui met en œuvre les fonctions de communication sur un réseau au moyen d'un protocole spécifique. Le groupe des Appareils de Communication FDI se divise en deux groupes principaux.

- Le Serveur de Communication FDI est un Serveur dédié de l'architecture unifiée OPC (OPC UA) qui fournit l'accès à un ou plusieurs réseaux d'appareils de terrain. Le Serveur de Communication FDI est spécifié à l'Article 7.
- La Passerelle de Communication FDI permet de relier différents réseaux physiques ou différents protocoles. La logique applicative du pontage est mise en œuvre dans le composant EDD qui est fourni avec un Paquetage de Communication FDI. La Passerelle de Communication FDI est spécifiée à l'Article 8.

NOTE Les principales différences entre une Passerelle et un Serveur de Communication sont les suivantes: En matière de FDI, le Serveur de Communication FDI est un Serveur OPC UA dédié qui fournit l'accès à un ou plusieurs réseaux d'appareils de terrain. Une Passerelle est un Appareil de Communication qui permet de relier différents réseaux physiques ou différents protocoles. La représentation logique d'un appareil Passerelle dans le Modèle d'Information hébergé par le Serveur FDI permet au Serveur FDI de traiter la communication dans des topologies hétérogènes de réseau.

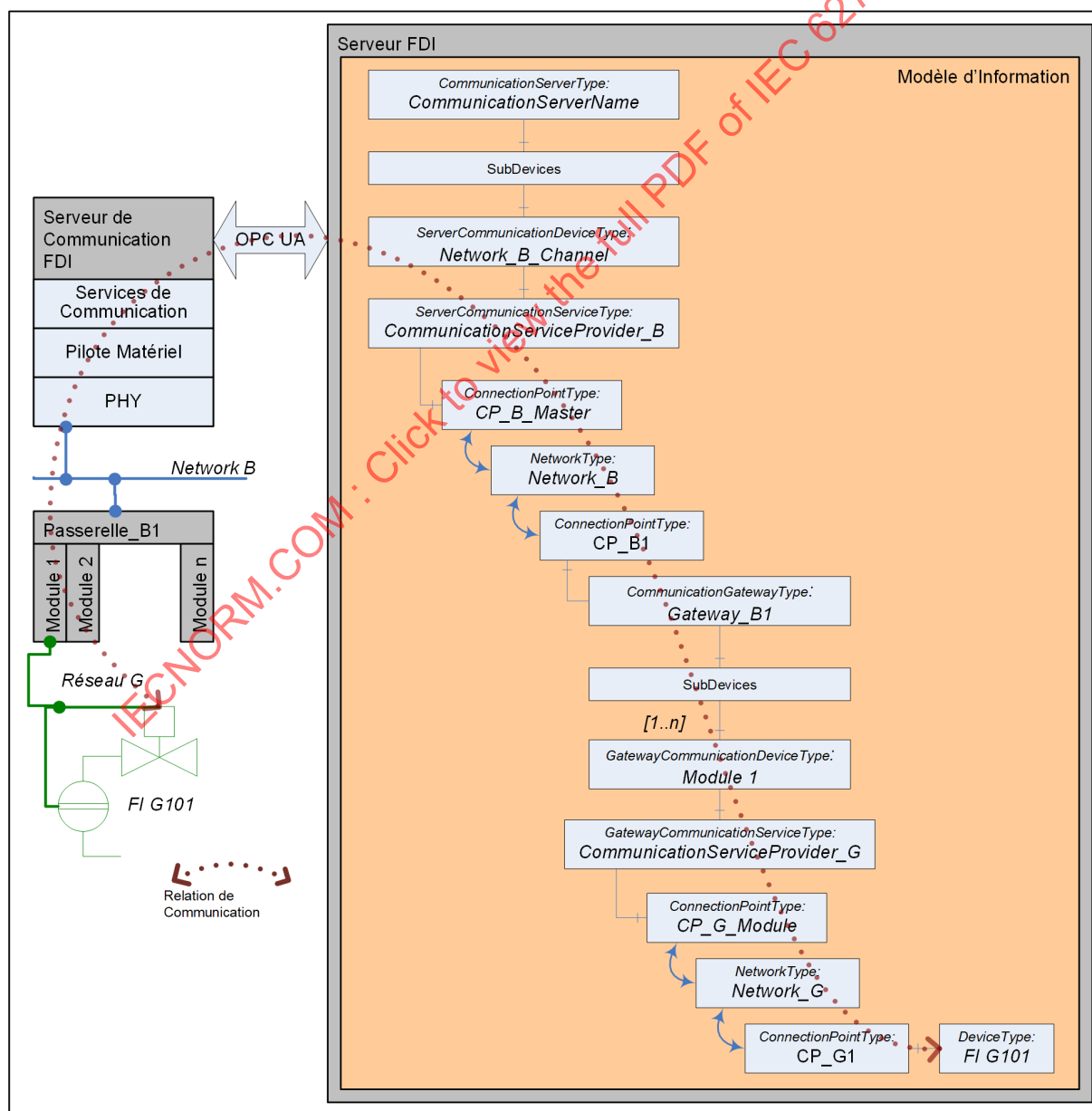


Figure 2 – Architecture de l'infrastructure de communication FDI

Le Modèle d'Information hébergé par le Serveur FDI contient une représentation de la topologie du réseau (voir également l'IEC 62769-5). Le Modèle d'Information représenté à la Figure 2 est un exemple extrait pour représenter la manière dont les éléments utilisés du Modèle d'Information reflètent la topologie réelle du réseau.

- 1) L'instance de `CommunicationServerType` (dénommée `CommunicationServerName`) représente le Serveur de Communication FDI. Le Serveur de Communication FDI met en œuvre l'accès au réseau de communication physique (matériel de communication). L'Article 7 décrit les éléments spécifiques du Modèle d'Information, le contenu exigé pour le Paquetage de Communication FDI et le traitement des éléments qui s'y trouvent. Pour les sous-appareils, voir l'IEC 62769-5.
- 2) L'instance de `ServerCommunicationDeviceType` et de `ServerCommunication-ServiceType` (dénommée `Network_B_Channel`) est mappée aux services de communication mis en œuvre par le Serveur de Communication FDI. Le `ServerCommunicationDeviceType` est spécifié en 7.3.3. Le `ServerCommunicationServiceType` est spécifié en 7.3.4.
- 3) L'instance de `CommunicationGatewayType` (dénommée `Gateway_B1`) représente la Passerelle physique. L'Article 8 décrit les éléments spécifiques associés du Modèle d'Information, le contenu exigé pour le Paquetage FDI et le traitement des éléments qui s'y trouvent.
- 4) L'instance de `GatewayCommunicationDeviceType` (dénommée `Module 1`) est mappée à un module physique ou logique, ce qui permet la communication au réseau auquel ce module est connecté. Le `GatewayCommunicationDeviceType` est spécifié en 8.3.2.3. Les éléments spécifiques associés de la Passerelle sont décrits à l'Article 8.
- 5) L'instance de `GatewayCommunicationServiceType` (dénommée `CommunicationServiceProvider_G`) représente la capacité des Passerelles à traiter les services de communication. La mise en œuvre spécifique de la Passerelle de `GatewayCommunicationServiceType` est fondée sur la Logique Appllicative qui permet d'exécuter les services de communication dans les réseaux hétérogènes de communication.
- 6) Une relation de communication (de plus amples informations sont données à l'Article 6) entre un appareil physique et la représentation de l'appareil gérée par le Serveur FDI est toujours associée aux objets du service de communication qui sont des instances d'un `GatewayCommunicationServiceType` ou d'un `ServerCommunicationServiceType`. La capacité d'instanciation de plusieurs protocoles de supports d'objets du service de communication permet de traiter plusieurs connexions logiques entre un maître bus et un appareil.
- 7) Le Modèle d'Information représente les connexions entre les appareils physiques représentés sur le côté gauche de la Figure 2 d'après les instances du `NetworkType` `ConnectionPointType` et les relations décrites. `ConnectionPointType` et `NetworkType` sont spécifiés dans l'IEC 62769-5.

5 Paquetage de Communication FDI

5.1 Généralités

Le Serveur FDI importe le Paquetage de Communication FDI comme tout autre Paquetage d'Appareil FDI. L'Article 5 spécifie les informations détaillées du Paquetage de Communication FDI.

5.2 Description d'Appareil Électronique (EDD)

5.2.1 Règles générales

L'élément EDD contenu dans le Paquetage de Communication FDI n'est pas limité, mais est associé à une annexe spécifique de protocole (IEC 62769-4:2020, Annexe F).

Les éléments EDD tels que spécifiés dans l'IEC 62769-4:2020, Annexe F et fournis avec un Paquetage de Communication FDI doivent décrire:

- a) Le paramètre et les structures de paramètre. Les définitions des paramètres obligatoires spécifiques au protocole se trouvent dans l'IEC 62769-4:2020, Annexe F. Les paramètres doivent contenir tout paramètre qui exige un ajustement pour le fonctionnement approprié du service de communication.
- b) L'identification de la Couche Physique. Les définitions spécifiques au protocole se trouvent dans l'IEC 62769-4:2020, Annexe F.

La modularité des Appareils de communication: Les informations de modularité doivent être fondées sur l'utilisation du COMPONENT des constructions EDDL (voir l'IEC 61804-3).

La FDI représente la modularité d'un Appareil de Communication susceptible de s'adapter au matériel de communication qui fournit plusieurs voies de communication physiques ou logiques pour accéder aux multiples réseaux de communication logiques ou physiques. Chaque élément de module de l'ensemble de l'Appareil de Communication doit être décrit par un élément EDD distinct.

- c) La définition du COMPONENT doit être utilisée pour prendre en charge la configuration de topologie mise en œuvre dans le système. Les définitions spécifiques au protocole se trouvent dans l'IEC 62769-4:2020, Annexe F. Les définitions du COMPONENT associées sont décrites en 5.2.2, 5.2.3, 5.2.4 et 5.2.7.
- d) La Logique Applicative doit contenir une méthode qui permet de valider le réseau (voir 5.2.8). La fonction de validation ne prend en considération que les éléments directement connectés au réseau. Elle doit être référencée par l'attribut CHECK_CONFIGURATION spécifié par le langage de description d'appareil électronique (EDDL).
- e) La Logique Applicative peut contenir une méthode qui permet de valider la configuration du module (voir 5.2.9) ou la configuration du réseau (voir 5.2.8). La fonction de validation ne prend en considération que les éléments directement connectés à l'élément parent concerné dans la topologie. Elle doit être référencée par l'attribut CHECK_CONFIGURATION spécifié par l'EDDL.
- f) Données relatives au Point de Connexion: Le Point de Connexion (voir 5.2.4 et 5.2.6) doit être décrit par les constructions EDDL COMPONENT, COLLECTION et VARIABLE. La définition du COMPONENT associe l'élément Point de Connexion à l'Appareil de Communication. Les définitions de la VARIABLE représentent les propriétés d'un Point de Connexion spécifique. La COLLECTION représente la structure du Point de Connexion comme telle. Les définitions spécifiques au protocole se trouvent dans l'IEC 62769-4:2020, Annexe F.
- g) MENU:
La structure Menu doit suivre les conventions Menu pour les applications PC conformément à l'IEC 61804-4 qui permettent l'accès aux éléments suivants:
 - 1) Paramètres de type d'Appareil de Communication FDI (bus): Ces paramètres doivent être rendus accessibles par le biais de "offline_root_menu".
 - 2) Les Dialogues de Configuration de la topologie doivent être rendus accessibles par le biais du point d'entrée du menu "topology_configuration".

5.2.2 Composant Appareil

Chaque Paquetage de Communication FDI doit contenir un élément EDD qui décrit l'appareil.

```
COMPONENT <DeviceComponentId>
{
  LABEL "<Label>";
  CAN_DELETE TRUE;
  CHECK_CONFIGURATION <ValidateModules>;
  CLASSIFICATION NETWORK_COMPONENT;
  COMPONENT_RELATIONS
  {
    <CommunicationDeviceRelationId>
  }
}
```

```

}

COMPONENT_RELATION <CommunicationDeviceRelationId>
{
    LABEL "Relation type description";
    RELATION_TYPE CHILD_COMPONENT;
    ADDRESSING {<AddressVar>}
    COMPONENTS
    {
        <CommunicationDeviceComponentId>
        {
            AUTO_CREATE <autoCreate>;
            REQUIRED_RANGES
            {
                <AddressVar>{ MIN_VALUE <AddrMin>; MAX_VALUE <AddrMax>;}
            }
        }
    }
    MINIMUM_NUMBER <minNumber>;
    MAXIMUM_NUMBER <maxNumber>;
}

```

<DeviceComponentId>: L'identificateur du COMPONENT identifie la description du composant pour le type d'appareil.

<Label>: La chaîne de valeur doit contenir une chaîne qui permet à un utilisateur humain de déterminer la fonction de l'objet du Serveur de Communication FDI.

<ValidateModules>: La Valeur désigne la METHOD qui met en œuvre la fonction de validation de la configuration de topologie du module. Les informations détaillées de mise en œuvre sont spécifiées en 5.2.9.

L'attribut COMPONENT_RELATIONS permet de décrire comment les modules peuvent être connectés. La définition de COMPONENT_RELATIONS est facultative. Si elle est utilisée, elle doit décrire les relations avec les définitions de CommunicationDevice. La construction permet d'effectuer la configuration de topologie actionnée par le Serveur FDI (appareil). Les informations détaillées de syntaxe sont décrites dans l'IEC 61804-3. Le texte suivant décrit l'usage sémantique de la construction COMPONENT_RELATION.

<CommunicationDeviceRelationId>: La valeur d'attribut identifie la définition COMPONENT_RELATION qui décrit la relation entre le composant Device et le composant CommunicationDevice.

<CommunicationDeviceComponentId>: La valeur d'attribut doit correspondre à l'identificateur du COMPONENT utilisé dans une déclaration COMPONENT qui décrit un Appareil de Communication (voir 5.2.3).

<autoCreate>: La valeur d'attribut décrit le nombre de composants CommunicationDevice qui peuvent être automatiquement instanciés avec le composant Device.

<minNumber>/<maxNumber>/<autoCreate>: Les valeurs des attributs définissent les contraintes d'instanciation. La définition de ces attributs est facultative. Les valeurs des attributs peuvent contenir des expressions conditionnelles.

Le RELATION_TYPE doit être défini sur CHILD_COMPONENT.

<AddressVar>: La valeur d'attribut est une référence à une déclaration de VARIABLE. Cette VARIABLE contient la valeur d'adresse pour une instance CommunicationDevice. La définition de cet attribut est facultative.

<AddrMin>/<AddrMax>: Les valeurs définissent la plage de valeurs d'adresse pour une instance CommunicationDevice. La valeur peut par exemple correspondre à un numéro de position physique. L'utilisation des attributs ADDRESSING et REQUIRED_RANGES active les routines de configuration générique.

5.2.3 Composant CommunicationDevice

Chaque Paquetage de Communication FDI doit contenir au moins un élément EDD qui décrit au moins un composant CommunicationDevice. La structure d'un matériel de communication modulaire doit être décrite par des descriptions COMPONENT CommunicationDevice multiples:

```

COMPONENT <CommunicationDeviceComponentId>
{
    LABEL "<Label>";
    CAN_DELETE <CanDelete>;
    CLASSIFICATION NETWORK_COMPONENT;
    COMPONENT_RELATIONS
    {
        <CommunicationServiceProviderRelationId>
    }
}

COMPONENT_RELATION <CommunicationServiceProviderRelationId>
{
    LABEL "Relation between CommunicationDevice and communication service provider";
    RELATION_TYPE CHILD_COMPONENT;
    ADDRESSING {<AddressVar>}
    COMPONENTS
    {
        <CommunicationServiceProviderId>
        {
            AUTO_CREATE <autoCreate>;
        }
    }
    MINIMUM_NUMBER 1;
    MAXIMUM_NUMBER <maxNumber>;
}

```

<CommunicationDeviceComponentId>: L'identificateur du COMPONENT identifie le composant CommunicationDevice.

<Label>: La chaîne de valeur doit contenir une chaîne en langage humain qui permet à un utilisateur de déterminer facilement la fonction du composant CommunicationDevice.

<CanDelete>: Les valeurs permises sont TRUE ou FALSE selon qu'un CommunicationDevice nécessite une configuration explicite ou que l'objet du fournisseur de services de communication concerné doit être automatiquement instancié avec le CommunicationDevice. Lorsque l'attribut CAN_DELETE est défini sur FALSE, la configuration de CommunicationDevice est statique.

La définition de `COMPONENT_RELATIONS` est obligatoire. Elle décrit la relation à la définition du fournisseur de services de communication. La construction permet au Serveur FDI d'instancier les composants du fournisseur de services de communication conformément aux demandes de traitement de la communication. Les informations détaillées de syntaxe sont décrites dans l'IEC 61804-3. Le texte suivant décrit l'usage sémantique de la construction `COMPONENT_RELATION`.

`<CommunicationServiceProviderRelationId>`: La valeur d'attribut identifie la définition de `COMPONENT_RELATION` comme telle.

`<CommunicationServiceProviderId>`: La valeur d'attribut doit correspondre à l'identificateur du `COMPONENT` utilisé dans une déclaration `COMPONENT` qui décrit un fournisseur de services de communication (5.2.4).

`<autoCreate>`: La valeur d'attribut décrit le nombre de fournisseurs de services de communication qui peuvent être automatiquement instanciés avec le composant `CommunicationDevice`.

Le `RELATION_TYPE` doit être défini sur `CHILD_COMPONENT`.

L'attribut `PROTOCOL` ne doit pas être défini.

5.2.4 Composant fournisseur de services de communication

Chaque Paquetage de Communication FDI qui décrit un Appareil de Communication doit contenir au moins un élément EDD qui décrit le fournisseur de services de communication. Le composant EDD ne doit définir aucun paramètre de configuration.

```
COMPONENT <CommunicationServiceProviderId>
{
    LABEL "<Label>";
    BYTE_ORDER <byteOrder>;
    CAN_DELETE <CanDelete>;
    CLASSIFICATION NETWORK_COMMUNICATION_SERVICE_PROVIDER;
    COMPONENT_RELATIONS <CommunicationServiceProvidersConnectionPointRelationId>
    {
        <ConnectionPointRelationId>
    }
}

COMPONENT_RELATION <CommunicationServiceProvidersConnectionPointRelationId>
{
    LABEL "Relation between communication service provider and connection point";
    RELATION_TYPE CHILD_COMPONENT;
    ADDRESSING {<AddressVar>}
    COMPONENTS
    {
        < ConnectionPointId>
        {
            AUTO_CREATE 1;
        }
    }
    MINIMUM_NUMBER 1;
    MAXIMUM_NUMBER 1;
}
```

`<CommunicationServiceProviderId>`: L'identificateur du `COMPONENT` identifie le fournisseur de services de communication.

`<Label>`: La chaîne de valeur doit contenir une chaîne en langage humain qui permet à un utilisateur de déterminer facilement la fonction de l'objet du fournisseur de services de communication.

<CanDelete>: Les valeurs permises sont TRUE ou FALSE selon qu'un fournisseur de services de communication peut être instancié de façon souple conformément aux demandes de traitement de la communication ou non. Lorsque l'attribut CAN_DELETE est défini sur FALSE, le nombre d'instanciations du composant du fournisseur de services de communication est statique. Les contraintes d'instanciation déclarées par les attributs AUTO_CREATE, MINIMUM_NUMBER et MAXIMUM_NUMBER correspondent aux capacités des protocoles actuellement pris en charge.

<byteOrder>: La valeur permet l'intégration générique des types de données de n octets (par exemple, un nombre entier de 4 octets) dans les données utiles du message de communication. La valeur d'attribut décrit l'ordre des octets et doit être BIG_ENDIAN ou LITTLE_ENDIAN.

La définition de COMPONENT_RELATIONS est obligatoire. Elle décrit la relation à la définition du Point de Connexion. La construction permet d'effectuer la configuration de topologie actionnée par le Serveur FDI. Les informations détaillées de syntaxe sont décrites dans l'IEC 61804-3. Le texte suivant décrit l'usage sémantique de la construction COMPONENT_RELATION.

Le Point de Connexion doit automatiquement être instancié avec le fournisseur de services de communication et il doit y avoir exactement une (1) instance de Point de Connexion connectée au fournisseur de services de communication. Les contraintes d'instanciation déclarées par les attributs AUTO_CREATE, MINIMUM_NUMBER et MAXIMUM_NUMBER correspondent aux capacités des protocoles actuellement pris en charge.

<CommunicationServiceProvidersConnectionPointRelationId>: La valeur d'attribut identifie la déclaration de COMPONENT_RELATION comme telle.

<ConnectionPointId>: la valeur d'attribut doit correspondre à l'identificateur du COMPONENT utilisé dans une déclaration COMPONENT qui décrit un Point de Connexion (voir 5.2.5).

Le RELATION_TYPE doit être défini sur CHILD_COMPONENT.

L'attribut PROTOCOL ne doit pas être défini.

5.2.5 Composant Point de Connexion

Chaque Paquetage de Communication FDI qui décrit un Appareil de Communication doit contenir un élément EDD qui décrit un Point de Connexion pour chacun des protocoles pris en charge par l'Appareil de Communication:

```
COMPONENT <ConnectionPointId>
{
  LABEL "<Label>";
  CAN_DELETE FALSE;
  CLASSIFICATION NETWORK_CONNECTION_POINT;
  PROTOCOL <ProtocolId>;
  CONNECTION_POINT <ConnectionPointCollectionId>;
}
```

<ConnectionPointId>: l'identificateur du COMPONENT identifie la déclaration de composant du Point de Connexion.

<Label>: La chaîne de valeur doit contenir une chaîne qui permet à un utilisateur humain de déterminer la fonction du composant de Point de Connexion.

<ProtocolId>: la valeur de cet attribut indique la capacité de communication qui permet au Serveur FDI de trouver d'autres types d'appareils qui peuvent être connectés au réseau au moyen du même type de protocole. Pour les protocoles normalisés, la valeur est définie par l'organisation de bus de terrain concernée.

<ConnectionPointCollectionId>: la valeur d'attribut est une référence à la déclaration de COLLECTION qui décrit la structure des données du Point de Connexion, comme décrit en 5.2.6.

5.2.6 Collection du Point de Connexion

Chaque EDD qui décrit le Point de Connexion d'un Appareil de Communication doit décrire l'élément COLLECTION qui décrit les attributs qui doivent apparaître dans la représentation du Modèle d'Information du Point de Connexion. Les données spécifiques à un protocole exposées par le Point de Connexion identifient le type d'appareil et son adresse réseau.

```
COLLECTION <ConnectionPointCollectionId>
{
    LABEL "<Label>";
    MEMBERS
    {
        <AddressAttributeName>, <AddressAttributeVariableId>;
        VALID <VALID_VariableId>;
    }
}
```

<ConnectionPointCollectionId>: l'identificateur de la COLLECTION est référencé par la valeur d'attribut CONNECTION_POINT définie en 7.7.3.5.

<Label>: l'étiquette identifie le Point de Connexion dans un langage humain.

<AddressAttributeName>/<AddressAttributeVariableId>: la section MEMBRE désigne les définitions de VARIABLE qui décrivent les attributs d'adresse mis en œuvre par un Point de Connexion. Le contenu de la section MEMBRE est spécifique au protocole.

<VALID>/<VALID_VariableId> est un membre de Collection qui désigne une VARIABLE booléenne qui contient l'état de validation qui doit être défini par l'Action ValidateNetwork (voir 5.2.8).

5.2.7 Composant réseau

Chaque Paquetage de Communication FDI qui décrit un Appareil de Communication doit contenir un élément EDD qui décrit un Réseau pour chacun des protocoles pris en charge par l'Appareil de Communication. La définition prend en charge l'ingénierie topologique du réseau.

```
COMPONENT <NetworkComponentId>
{
    LABEL "<Label>";
    CAN_DELETE TRUE;
    CHECK_CONFIGURATION <Validate>;
    CLASSIFICATION NETWORK;
    PROTOCOL <ProtocolId>;
    COMPONENT_RELATIONS
    {
        <NetworksConnectionPointRelationId>
    }
}

COMPONENT_RELATION <NetworksConnectionPointRelationId>
{
    LABEL "Relation between network and connection point";
    RELATION_TYPE CHILD_COMPONENT;
    ADDRESSING {<AddressVar>}
    COMPONENTS
    {
        <ConnectionPointId>
        {
            REQUIRED_RANGES
            {
                <BusAddressVar>{ MIN_VALUE <BusAddrMin>; MAX_VALUE <BusAddrMax>;}
            }
        }
    }
    MINIMUM_NUMBER 1;
    MAXIMUM_NUMBER <maxNumber>;
}
```

<NetworkComponentId>: l'identificateur du COMPONENT identifie la déclaration du composant Réseau.

<Label>: la chaîne de valeur doit contenir une chaîne en langage humain qui permet à un utilisateur de déterminer facilement la fonction du composant Réseau.

<Validate>: la valeur désigne la METHOD qui met en œuvre la fonction de validation de la configuration de topologie du réseau (voir 5.2.8).

<ProtocolID>: la valeur de cet attribut permet au Serveur FDI de trouver d'autres types d'appareils qui peuvent être connectés au réseau par le même type de protocole. Pour les protocoles normalisés, la valeur est définie par l'organisation de bus de terrain concernée.

La définition de COMPONENT_RELATIONS est obligatoire. Elle décrit la relation à la définition du Point de Connexion et, par conséquent, les capacités d'un réseau. La construction permet d'effectuer la configuration générique de topologie réseau actionnée par le Serveur FDI. Les informations détaillées de syntaxe sont décrites dans l'IEC 61804-3. Le texte suivant décrit l'usage sémantique de la construction COMPONENT_RELATION.

<NetworksConnectionPointRelationId>: la valeur d'attribut identifie la définition de COMPONENT_RELATION.

<ConnectionPointId>: La valeur d'attribut doit correspondre à l'identificateur du COMPONENT utilisé dans une déclaration COMPONENT qui décrit un Point de Connexion (voir 5.2.4).

<maxNumber>: La valeur d'attribut limite le nombre de Points de Connexion qui peuvent être connectés au réseau. Les valeurs des attributs peuvent contenir des expressions conditionnelles.

Le RELATION_TYPE doit être défini sur CHILD COMPONENT.

<BusAddressVar>: La valeur d'attribut est une référence à une déclaration de VARIABLE. Cette VARIABLE contient la valeur de l'adresse réseau pour tout appareil connecté au réseau.

<BusAddrMin>/<BusAddrMax>: Les valeurs définissent la plage de valeurs d'adresse réseau.

5.2.8 ValidateNetwork

La méthode ValidateNetwork représente la Logique Appllicative mise en œuvre dans l'Appareil de Communication qui valide une topologie réseau réelle. La méthode ValidateNetwork gère toutes les dépendances nécessaires relatives aux paramètres du bus. La mise en œuvre de la logique EDDL concernée est fondée sur la fonction ObjectReference intégrée à l'EDDL, qui permet l'analyse d'un ensemble d'instances enfants (instances du Point de Connexion). La logique de validation doit définir l'attribut <VALID> de l'instance du Point de Connexion qui a concluante la validation.

La mise en œuvre de ValidateModules est facultative lorsque l'installation du module est statique ou lorsque les règles de configuration définies dans la construction COMPONENT sont suffisantes pour configurer l'installation du module.

Le Tableau 1 présente les arguments de l'Action ValidateNetwork.

Signature

```
ValidateNetwork (
    [out] Integer ServiceError,
    [out] String ErrorMessage);
```

Tableau 1 – Arguments de l'Action ValidateNetwork

Argument	Description
ServiceError	0: OK -1: Échec/le Point de Connexion qui n'a pas réussi la validation est indiqué par la valeur () d'attribut <VALID> définie sur false. Remarque: Les valeurs d'argument correspondent aux codes d'erreur spécifiés dans l'IEC 61804-3, dénommés BI_SUCCESS (valeur = 0) et BI_ERROR (valeur = -1). L'Action retourne le résultat ServiceError au moyen de la déclaration de "return".
ErrorMessage	Lorsque la méthode retourne une chaîne vide (NULL), l'appel de l'Action a réussi. En cas d'erreur, l'Action peut retourner une description du problème.

5.2.9 ValidateModules

La méthode ValidateModules valide l'installation actuelle du module. La mise en œuvre de la logique EDDL concernée est fondée sur la fonction ObjectReference intégrée à l'EDDL, qui permet d'analyser un ensemble d'instances enfants. La mise en œuvre de ValidateModules est facultative si l'installation du module est statique ou si les règles de configuration définies dans la construction COMPONENT sont suffisantes pour configurer l'installation du module.

NOTE La décision qui consiste à déterminer si ValidateModules est nécessaire ou non dépend du fournisseur.

Le Tableau 2 présente les arguments de l'Action ValidateModules.

Signature

```
ValidateModules (
    [out] Integer serviceError,
    [out] String ErrorMessage);
```

Tableau 2 – Arguments de l'Action ValidateModules

Argument	Description
ServiceError	0: OK -1: Échec/le Point de Connexion qui n'a pas réussi la validation est indiqué par la valeur () d'attribut <VALID> définie sur false. Remarque: Les valeurs d'argument correspondent aux codes d'erreur spécifiés dans l'IEC 61804-3 dénommés BI_SUCCESS (valeur = 0) et BI_ERROR (valeur = -1). L'Action retourne le résultat ServiceError en utilisant la déclaration de "return".
ErrorMessage	Lorsque l'Action retourne une chaîne vide (NULL), l'appel de la méthode a réussi. En cas d'erreur, l'Action peut retourner une description du problème.

5.2.10 Éléments spécifiques de l'UIP

Le Paquetage de Communication FDI peut contenir l'UIP pour prendre en charge les diagnostics et le paramétrage, par exemple.

5.2.11 Déploiement

Le Serveur FDI importe le Paquetage de Communication FDI. Le traitement d'éléments EDD et UIP correspond à la procédure d'importation exécutée pour le Paquetage FDI (voir l'IEC 62769-2 et l'IEC 62769-3).

6 Relations de communication

L'objectif d'un Appareil de communication et de ses services de communication est d'échanger les informations entre l'appareil physique et la représentation de l'appareil gérée par le Serveur FDI. L'échange d'informations est géré par les relations de communication (voir la Figure 3). Une relation de communication établie représente la capacité d'échanger des informations entre la représentation de l'appareil gérée par le Serveur FDI et l'appareil physique. L'utilisation de la Relation de Communication permet de faire abstraction des éléments spécifiques du protocole généralement utilisés pour gérer les connexions.

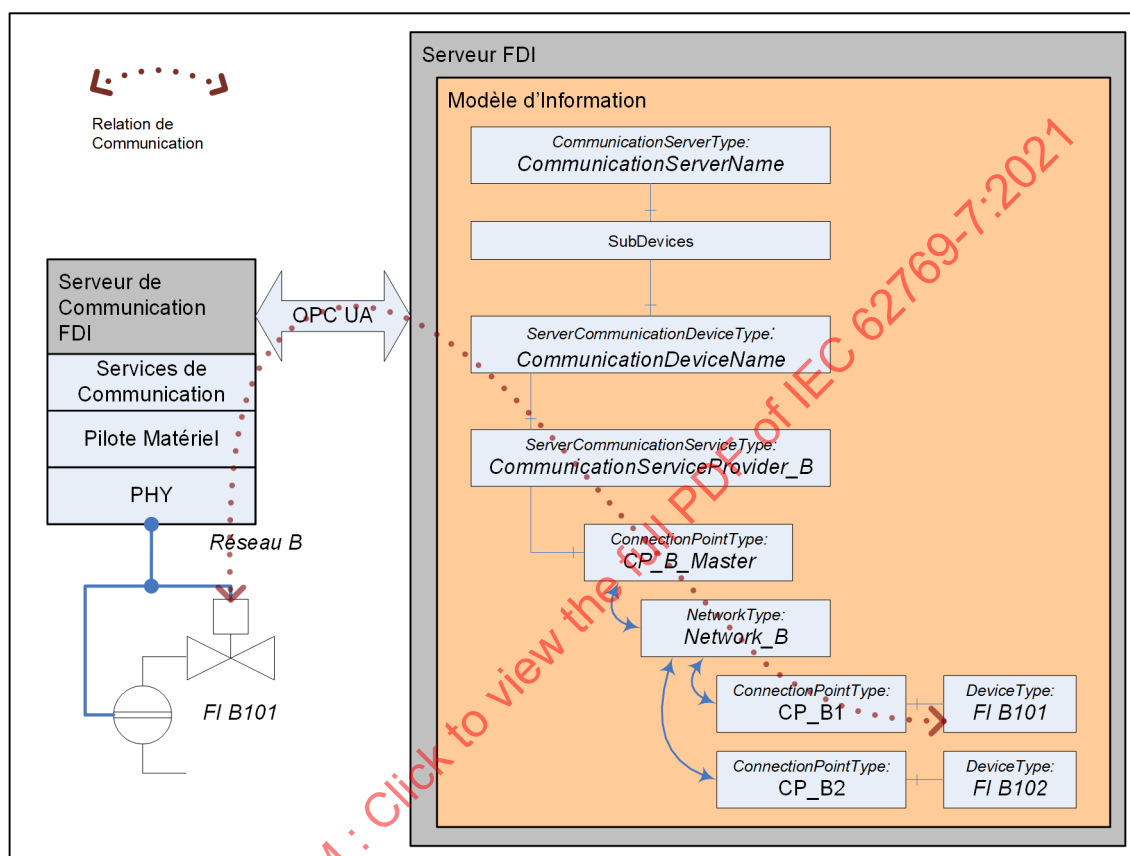
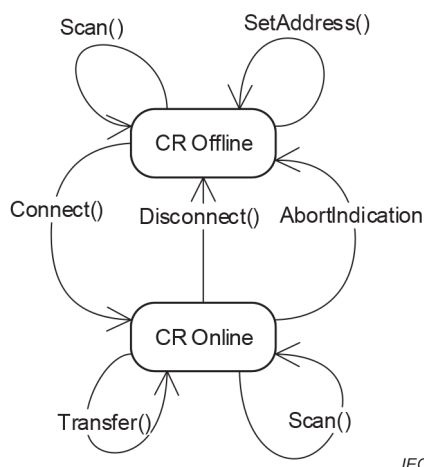


Figure 3 – Relation de communication

NOTE 1 L'essentiel de l'échange d'informations a lieu entre l'appareil connecté au réseau physique et l'instance correspondante dans le Modèle d'Information, mais ne couvre pas l'application complète de l'appareil.

Le diagramme états-transitions suivant (Figure 4) décrit le flux d'état général d'une relation de communication unique. Le diagramme représente également les services de communication qui peuvent être invoqués pendant un état "CR Online".

L'état "AbortIndication" représenté à la Figure 4 peut être détecté de différentes façons spécifiques au protocole. La façon spécifiée pour un appareil de communication donné est liée aux serviceError retournées par les services de communication spécifiés. Même la méthode de balayage (Scan) peut déterminer une perte de connexion quand l'appareil pour lequel une relation de communication a été activée n'apparaît pas dans le résultat du balayage.



IEC

Figure 4 – Diagramme états-transitions de la relation de communication

NOTE 2 La gestion des relations de communication est facultative.

7 Définition du Serveur de Communication FDI

7.1 Généralités

En matière de FDI, le Serveur de Communication FDI est un Serveur OPC UA dédié qui fournit l'accès à un ou plusieurs réseaux d'appareils de terrain. Chaque Serveur de Communication FDI est conçu comme un Appareil Modulaire dans lequel chaque module (appelé également CommunicationDevice dans la séquence) représente le point d'accès à un réseau.

L'Appareil Modulaire lui-même représente le Serveur de Communication FDI comme un ensemble.

7.2 Caractéristiques générales

Le Serveur de Communication FDI met en œuvre les caractéristiques de chacun de ses CommunicationDevice spécifiés en 7.3.3. En outre, le Serveur de Communication FDI met en œuvre les caractéristiques suivantes:

- Le Serveur FDI synchronise en permanence (voir 7.5, 7.8.8 et 7.8.11) le Modèle d'Information hébergé par le Serveur de Communication FDI depuis le contenu du Modèle d'Information hébergé par le Serveur FDI.
- Les CommunicationDevice peuvent être instanciés statiquement ou peuvent être créés/supprimés par le Serveur FDI.
- La communication entre le Serveur FDI et le Serveur de Communication FDI est fondée sur OPC UA. OPC UA spécifie un protocole filaire pour ses services qui peuvent être mis en œuvre sur les plates-formes arbitraires et les environnements d'exécution.
- Pour éviter les accrochages, le Serveur de Communication FDI permet la connexion d'un seul Serveur FDI à la fois. Avec cette restriction, un Serveur de Communication FDI peut s'abstenir de tout mécanisme de synchronisation (verrouillage). La spécification FDI n'exécute pas les Serveurs de Communication FDI qui mettent en œuvre un mécanisme de verrouillage pour gérer l'accès concomitant à un appareil unique connecté au réseau physique.

7.3 Modèle d'Information

7.3.1 Généralités

Le paragraphe 7.3 spécifie le Modèle d'Information hébergé par le Serveur de Communication FDI.

Un Serveur de Communication FDI est un Serveur OPC UA qui encapsule le matériel de communication et fournit la capacité normalisée de communication. Le Serveur FDI se connecte au Serveur de Communication FDI comme Client OPC UA et accède aux réseaux pris en charge

par le Serveur de Communication FDI par le biais du Modèle d'Information du Serveur de Communication FDI. La tâche du Serveur de Communication FDI consiste à exposer ce Modèle d'Information. Le Serveur de Communication FDI ne doit pas conserver les Instances d'Appareil ni les informations de topologie du réseau. Toute interaction avec les appareils FDI a lieu par le biais du Serveur FDI et est juste transférée par le Serveur de Communication FDI.

Pour le Serveur FDI, un Serveur de Communication FDI ressemble à un appareil qui prend en charge les services de Communication FDI et utilise OPC UA pour communiquer. Le Serveur de Communication FDI peut s'exécuter localement sur le même PC que le Serveur FDI (adaptateur de boucle de retour) ou à distance sur le terrain (par exemple, intégré à un contrôleur). Comme un appareil, chaque Serveur de Communication FDI a un Paquetage FDI associé. Ce Paquetage FDI est utilisé pour créer des appareils de communication dans le Modèle d'Information du Serveur FDI qui représentent l'accès aux réseaux mis en œuvre par le Serveur de Communication FDI.

Le Modèle d'Information d'un Serveur de Communication FDI est fondé sur le Modèle d'Information défini dans l'IEC 62769-5. La Figure 5 reproduit la structure de l'Appareil Modulaire et représente la manière dont elle est mappée dans l'AddressSpace (espace d'adressage) complet. Les modules représentent les voies de communication du Serveur de Communication FDI.

IECNORM.COM : Click to view the full PDF of IEC 62769-7:2021

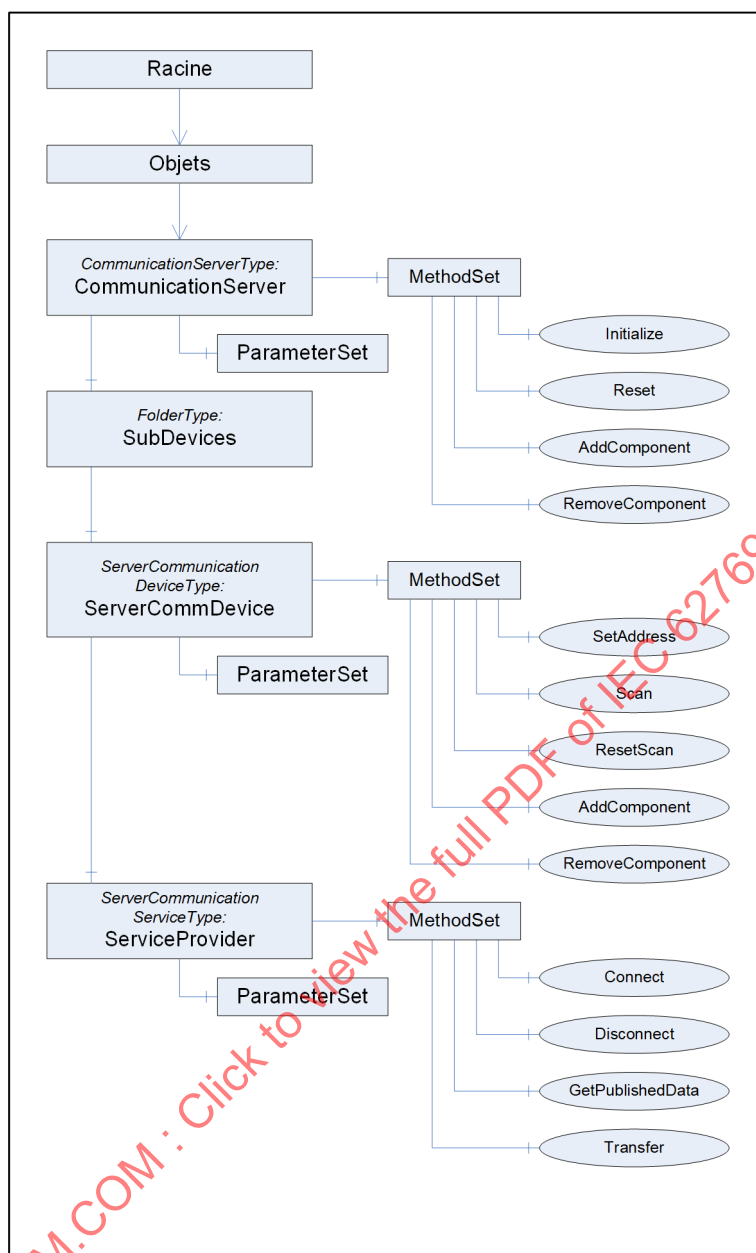


Figure 5 – AddressSpace du Serveur de Communication FDI

Le CommunicationServerType (la racine de l'Appareil Modulaire) est un sous-type de DeviceType. Le MethodSet contient les méthodes Initialize, Reset, AddComponent et RemoveComponent. Les méthodes AddComponent et RemoveComponent sont facultativement présentes lorsque le Serveur de Communication FDI prend en charge l'instanciation dynamique des éléments dans le dossier SubDevices.

Tous les sous-appareils sont des instances de ServerCommunicationDeviceType définies en 7.3.3. Les instances de ServerCommunicationDeviceType (ServerCommDevice) ont un MethodSet qui peut mettre en œuvre les méthodes SetAddress, Scan, AddComponent, RemoveComponent. AddComponent et RemoveComponent sont facultativement présents lorsque le Serveur de Communication FDI prend en charge un nombre variable des instances de ServerCommunicationServiceType.

Les définitions formelles se trouvent en 7.3.2, 7.3.3 et 7.3.4.

7.3.2 CommunicationServerType

7.3.2.1 Généralités

Le CommunicationServerType est un sous-type du DeviceType et fournit les méthodes nécessaires pour gérer les instances ServerCommunicationDeviceType. La Figure 6 représente la définition de CommunicationServerType qui est formellement décrite dans le Tableau 3 et le Tableau 4.

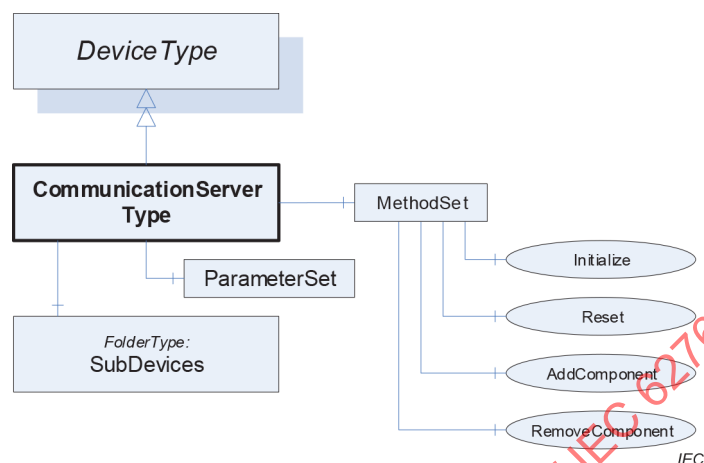


Figure 6 – CommunicationServerType

Tableau 3 – Définition de CommunicationServerType

Attribut	Valeur				
BrowseName	CommunicationServerType				
IsAbstract	False				
Références	NodeClass	BrowseName	Data Type	TypeDefinition	ModellingRule
Le sous-type de DeviceType est défini dans IEC 62541-100.					
HasComponent	Object	MethodSet		BaseObjectType	Mandatory
HasComponent	Object	ParameterSet		BaseObjectType	Optional
HasComponent	Object	SubDevices		FolderType	Mandatory

Tableau 4 – MethodSet de CommunicationServerType

Attribut	Valeur				
BrowseName	MethodSet				
IsAbstract	False				
Références	NodeClass	BrowseName	Data Type	TypeDefinition	ModellingRule
HasComponent	Method	Initialize			Mandatory
HasComponent	Method	Reset			Mandatory
HasComponent	Method	AddComponent			Optional
HasComponent	Method	RemoveComponent			Optional

Le `CommunicationServerType` et chaque instance de ce Type partagent les mêmes Méthodes. Le `NodeId` de ces Méthodes est fixé et défini dans le présent document (voir l'Annexe B). Les clients du Serveur de Communication FDI ne doivent donc pas rechercher ces Méthodes. Ils peuvent utiliser le `NodeId` fixe en tant que `MethodId` du Service Call.

Les Méthodes supplémentaires `AddComponent` et `RemoveComponent` ajoutent la capacité d'ajouter ou de supprimer les instances de `ServerCommunicationDeviceType` selon la structure du matériel de communication. Ces services ne sont pas applicables lorsque le Serveur de Communication FDI met en œuvre une structure de matériel de communication statique.

Le dossier `SubDevices` contient les instances de `ServerCommunicationDeviceType` qui représentent les modules de communication.

NOTE L'indication pour une disposition de matériel de communication statique est indiquée dans le Paquetage FDI avec l'attribut `COMPONENT CAN_DELETE` défini sur `FALSE` dans les déclarations du `COMPONENT`.

7.3.2.2 Méthode Reset

La méthode `Reset` est utilisée pour réinitialiser le matériel de communication et le logiciel du pilote concerné. Toute communication courante est arrêtée immédiatement. Toutes les voies de communication entrent dans l'état fermé.

La Méthode `Reset` ne doit pas être présente dans le Modèle d'Information hébergé par le Serveur FDI. Le Serveur FDI doit être capable de gérer la procédure d'arrêt automatiquement selon les demandes de communication.

Généralement, le fonctionnement du Serveur de Communication FDI inclut le traitement du matériel et du pilote du protocole qui peut être indépendant de toute structure modulaire. En raison de cette possibilité, la méthode `Reset` est organisée sous le `CommunicationServerType`. Afin de réduire la complexité de fonctionnement du Serveur de Communication FDI, une seule méthode `Reset` a été spécifiée.

La signature de cette Méthode est spécifiée ci-dessous. Le Tableau 5 et le Tableau 6 spécifient respectivement les arguments et la représentation de l'`AddressSpace`.

Signature

```
Reset (
    [out] Integer    serviceError);
```

Tableau 5 – Arguments de la Méthode Reset

Argument	Description
serviceError	0: OK --1: Échec

Tableau 6 – Définition de l'`AddressSpace` de la Méthode Reset

Attribut	Valeur				
BrowseName	Reset				
Références	NodeClass	BrowseName	Data Type	TypeDefinition	ModellingRule
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory

7.3.2.3 Méthode Initialize

La méthode `Initialize` sert à initialiser le matériel de communication. La fonction d'initialisation du Serveur de Communication FDI doit utiliser les données de paramétrage hébergées par le `ParameterSet` qui est contenu dans l'instance du `CommunicationServerType` et toutes les instances de `ServerCommunicationDeviceType`.

Afin de permettre les modifications de paramètres pendant le fonctionnement, la méthode Initialize peut être ré invoquée. Lorsque le Serveur de Communication FDI nécessite de réinitialiser son matériel de communication, il doit automatiquement restaurer la relation de communication qui existait auparavant. Un Serveur de Communication FDI modulaire ne peut initialiser de manière souple que les instances de ServerCommunicationDeviceType pour lesquelles les modifications de configuration ont été détectées.

La Méthode Initialize ne doit pas être présente dans le Modèle d'Information hébergé par le Serveur FDI. Le Serveur FDI doit être capable de gérer la procédure de démarrage automatiquement selon les demandes de communication humaines.

Le fonctionnement du Serveur de Communication FDI peut inclure le traitement du matériel et du pilote du protocole qui peut être indépendant de toute structure modulaire. En raison de cette possibilité, la méthode Initialize est organisée sous le CommunicationServerType. Afin de réduire la complexité de fonctionnement du Serveur de Communication FDI, une seule méthode Initialize a été spécifiée.

La signature de cette Méthode est spécifiée ci-dessous. Le Tableau 7 et le Tableau 8 spécifient respectivement les arguments et la représentation de l'AddressSpace.

Signature

```
Initialize (
    [out] Integer    serviceError)
```

Tableau 7 – Arguments de la Méthode Initialize

Argument	Description
serviceError	0: OK -1: Échec

Tableau 8 – Définition de l'AddressSpace de la Méthode Initialize

Attribut	Valeur				
BrowseName	Initialize				
Références	NodeClass	BrowseName	Data Type	TypeDefinition	ModellingRule
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory

7.3.2.4 Méthode AddComponent

La Méthode AddComponent doit être utilisée pour configurer l'installation modulaire d'un Serveur de Communication FDI lorsque le Serveur de Communication FDI n'a pas d'installation du matériel de communication définie statiquement. Cette méthode doit être utilisée pour ajouter un module (instance de ServerCommunicationDeviceType).

La signature de cette Méthode est spécifiée ci-dessous. Le Tableau 9 et le Tableau 10 spécifient respectivement les arguments et la représentation de l'AddressSpace.

Signature

```
AddComponent (
    [in] String      ModuleTypeName,
    [in] String      InstanceName,
    [in] String      InstanceLabel,
    [out] NodeId     InstanceNodeId,
    [out] Integer     ServiceError);
```

Tableau 9 – Arguments de la Méthode AddComponent

Argument	Description
ModuleTypeName	Type de module à créer conformément à la définition dans le Paquetage FDI. Le nom du type de module doit correspondre à l'une des définitions de l'identificateur du COMPONENT (voir 5.2.3).
InstanceName	Nom non localisé du Nœud d'Appareil du module de l'élément créé. Ce nom doit être unique avec le domaine d'application du Modèle d'Information du Serveur de Communication FDI.
InstanceLabel	Étiquette en langage humain pour le Nœud racine du module créé.
InstanceNodeId	Identificateur assigné à l'appelé pour le Nœud d'Appareil du module.
ServiceError	0 – OK -1 – E_InvalidType – aucun module pour le Type spécifié ne peut (plus) être ajouté -2 – E_DuplicateName – il existe déjà un module avec le même nom, tel que spécifié avec l'argument InstanceName -3 – E_UnknownType – un ModuleTypeName inconnu a été spécifié -4 – E_LimitExceeded – le nombre total de modules est dépassé (ce dépassement peut être dû aux contraintes d'alimentation ou d'autres limitations de ressources)

Tableau 10 – Définition de l'AddressSpace de la Méthode AddComponent

Attribut	Valeur				
BrowseName	AddComponent				
Références	NodeClass	BrowseName	Data Type	TypeDefinition	ModellingRule
HasProperty	Variable	InputArguments	Argument[]	PropertyType	Mandatory
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory

7.3.2.5 Méthode RemoveComponent

La méthode RemoveComponent doit être utilisée pour supprimer un module (Instance de ServerCommunicationDevice). La mise en œuvre de RemoveComponent est facultative lorsque l'installation du matériel de communication est statique.

La signature de cette Méthode est spécifiée ci-dessous. Le Tableau 11 et le Tableau 12 spécifient respectivement les arguments et la représentation de l'AddressSpace.

Signature

```
RemoveComponent (
    [in] NodeId      ModuleNodeId,
    [out] Integer    ServiceError);
```

Tableau 11 – Arguments de la Méthode RemoveComponent

Argument	Description
ModuleNodeId	La valeur est l'identification de l'instance existante dans le Modèle d'Information.
ServiceError	0: OK -1: Échec, le nœud spécifié n'existe pas

Tableau 12 – Définition de l'AddressSpace de la Méthode RemoveComponent

Attribut	Valeur				
BrowseName	RemoveComponent				
Références	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
HasProperty	Variable	InputArguments	Argument[]	PropertyType	Mandatory
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory

7.3.3 ServerCommunicationDeviceType

7.3.3.1 Généralités

Le ServerCommunicationDeviceType représente une voie de communication pour un réseau particulier. Le ServerCommunicationDeviceType est un sous-type du DeviceType. Le ParameterSet de chaque instance d'un ServerCommunicationDevice contient les Paramètres nécessaires pour configurer le fonctionnement du réseau. La Figure 7 représente la définition de ServerCommunicationDeviceType qui est formellement décrite dans le Tableau 13 et à la Figure 14.

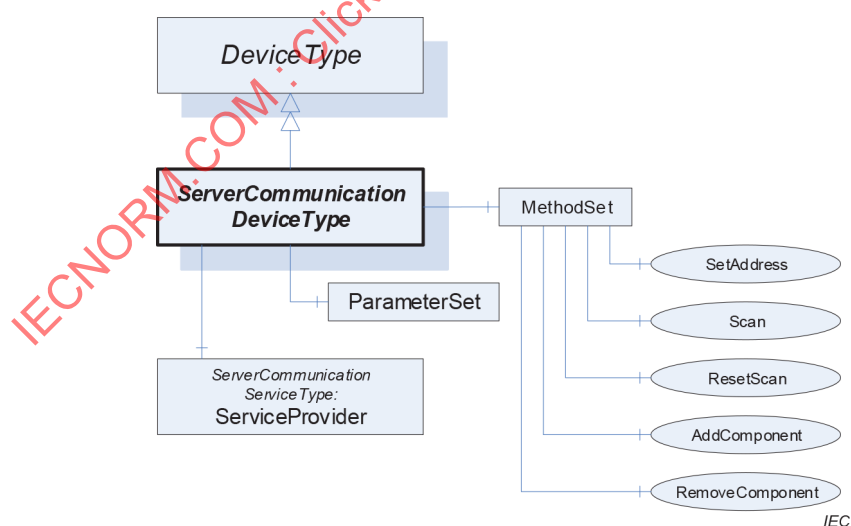


Figure 7 – ServerCommunicationDeviceType

Tableau 13 – Définition de ServerCommunicationDeviceType

Attribut	Valeur				
BrowseName	ServerCommunicationDeviceType				
IsAbstract	True				
Références	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
Le sous-type de DeviceType est défini dans l'OPC UA Partie DI.					
HasComponent	Object	MethodSet		BaseObjectType	Optional
HasComponent	Object	ParameterSet		BaseObjectType	Optional
HasComponent	Object	ServiceProvider		ServerCommunicationServiceType	Mandatory
HasProperty	Variable	ListOfCommunicationProfiles	String[]	PropertyType	Mandatory

La Propriété ListOfCommunicationProfiles contient une liste de profils de communication pris en charge par le ServerCommunicationDevice. Les chaînes valides sont définies dans l'IEC 62769-4:2020, Annexe F.

Tableau 14 – MethodSet de ServerCommunicationDeviceType

Attribut	Valeur				
BrowseName	MethodSet				
IsAbstract	True				
Références	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
HasComponent	Method	Scan			Optional
HasComponent	Method	ResetScan			Optional
HasComponent	Method	SetAddress			Optional
HasComponent	Method	AddComponent			Optional
HasComponent	Method	RemoveComponent			Optional

7.3.3.2 Méthode Scan

La méthode Scan doit être utilisée pour commencer à découvrir les appareils connectés au réseau physique. Les associations entre la méthode Scan et la connexion au réseau physique correspondante permettent au Serveur de Communication FDI d'accéder à la connexion au réseau physique correcte. La méthode Scan est mise en œuvre par le module d'exécution du Serveur de Communication.

La signature de cette Méthode est spécifiée ci-dessous. Le Tableau 15 et le Tableau 16 spécifient respectivement les arguments et la représentation de l'AddressSpace.

NOTE 1 Les Serveurs de Communication peuvent exécuter le balayage réseau dans une tâche d'arrière-plan. Par conséquent, l'invocation de la fonction Scan retourne des résultats de balayage réseau mis en cache.

NOTE 2 Lorsque l'opération SCAN dure trop longtemps, le Serveur de Communication FDI peut retourner une valeur TopologyScanResult vide et la ServiceError 1 qui indique que le balayage est toujours en cours d'exécution.

Signature

```

Scan (
    [out] XmlElement      TopologyScanResult,
    [out] Integer         ServiceError)

```

Tableau 15 – Arguments de la Méthode Scan

Argument	Description
TopologyScanResult	La valeur d'argument est une chaîne au format XML qui représente une liste d'appareils connectés au réseau physique. Chacun des appareils connectés au réseau physique est représenté par une structure de données correspondant avec un nœud de Point de Connexion. Les attributs du Point de Connexion sont spécifiques au protocole. Le schéma topologyScanResult correspondant est spécifié dans l'IEC 62769-4:2020, Annexe F. En cas d'erreur, une chaîne vide est retournée pour TopologyScanResult.
ServiceError	0: OK / balayage terminé 1: OK / obtenir le résultat de balayage complet par le rappel de Scan -1: Échec / non initialisé -2: Échec / non connecté à un réseau -3: Échec / aucun appareil n'a été trouvé, topologyScanResult est vide

Tableau 16 – Définition de l'AddressSpace de la Méthode Scan

Attribut	Valeur				
BrowseName	Scan				
Références	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory

7.3.3.3 Méthode ResetScan

La Méthode ResetScan doit être utilisée pour réinitialiser le cache interne des résultats de balayage. Elle sert également à annuler un balayage en cours d'exécution lorsque le mécanisme de balayage du Serveur de Communication FDI le prend en charge.

La signature de cette Méthode est spécifiée ci-dessous. Le Tableau 17 et le Tableau 18 spécifient respectivement les arguments et la représentation de l'AddressSpace.

Signature

```

ResetScan (
    [out] Integer         ServiceError)

```

Tableau 17 – Arguments de la Méthode ResetScan

Argument	Description
ServiceError	0: OK / balayage réinitialisé -1: Échec / non initialisé -2: Échec / non connecté à un réseau

Tableau 18 – Définition de l'AddressSpace de la Méthode ResetScan

Attribut	Valeur				
BrowseName	ResetScan				
Références	NodeClass	BrowseName	Data Type	TypeDefinition	ModellingRule
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory

7.3.3.4 Méthode SetAddress

La Méthode SetAddress doit être utilisée pour modifier l'adresse réseau (adresse de communication) d'un appareil. L'Appareil de Communication doit assurer des valeurs d'adresse réseau uniques. Lorsque la valeur d'argument de newAddress est déjà assignée à un appareil connecté au réseau physique, l'Appareil de Communication doit retourner l'argument serviceError, valeur "-4: Échec/erreur de l'adresse en double".

Le protocole détermine si le service d'assignation de l'adresse doit fonctionner même quand une relation de communication est déjà établie.

La signature de cette Méthode est spécifiée ci-dessous. Les arguments de la Méthode SetAddress sont décrits dans le Tableau 19.

Signature

```
SetAddress (
    [in]          <OldAddress>,
    [in]          <NewAddress>,
    [out] Int32    ServiceError);
```

Tableau 19 – Arguments de la Méthode SetAddress

Argument	Description
<OldAddress>	Il s'agit d'un paramètre fictif pour 1..n arguments spécifiques au protocole qui représentent l'adresse réseau existante spécifique au protocole de l'appareil connecté au réseau physique. Les valeurs qui représentent une adresse réseau sont spécifiées dans l'IEC 62769-4:2020, Annexe F.
<NewAddress>	Il s'agit d'un paramètre fictif pour 1..n arguments spécifiques au protocole qui représentent la nouvelle adresse réseau existante spécifique au protocole qui doit être assignée à un appareil connecté au réseau physique. Les valeurs qui représentent les arguments d'une adresse réseau sont spécifiées dans l'IEC 62769-4:2020, Annexe F.
ServiceError	0: OK / exécution terminée avec succès Les autres valeurs qui représentent ServiceError sont spécifiées dans l'IEC 62769-4:2020, Annexe F.

7.3.4 ServerCommunicationServiceType

7.3.4.1 Généralités

Les services de communication fournissent les moyens pour communiquer avec un Appareil ou exécuter, par exemple, un Scan sur un Réseau. Les services de communication sont représentés par les Méthodes du Modèle d'Information (voir l'IEC 62769-5).

Pour la définition formelle de ServerCommunicationServiceType, se reporter à la Figure 8, au Tableau 20 et au Tableau 21.

Le NodeId de ces Méthodes est fixé et défini dans le présent document. Les Clients FDI ne doivent donc pas rechercher ces Méthodes. Ils peuvent utiliser le NodeId fixe en tant que MethodId du Service Call.

Les méthodes de communication qui contiennent leurs NodeId sont définies de manière unique dans le présent document. Les Clients FDI peuvent utiliser les Méthodes directement (sans exploration). Le Service d'Appel OPC UA doit être utilisé comme suit:

- l'argument MethodId doit contenir le NodeId fixe de la Méthode;
- l'argument ObjectId doit contenir le NodeId fixe de MethodSet.

Le StatusCode de l'OPC UA, Bad_MethodInvalid, doit être retourné depuis le Service Call pour les éléments pour lesquels les méthodes de communication ne sont pas prises en charge.

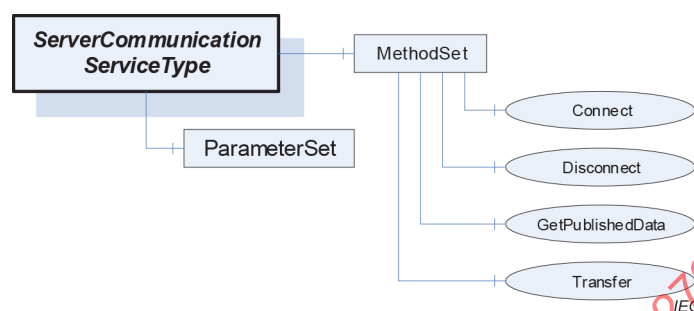


Figure 8 – ServerCommunicationServiceType

Tableau 20 – Définition de ServerCommunicationServiceType

Attribut	Valeur				
BrowseName	ServerCommunicationServiceType				
IsAbstract					
Références	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
Le sous-type de DeviceType est défini dans l'OPC UA Partie DI.					
HasComponent	Object	MethodSet		BaseObjectType	Mandatory
HasComponent	Object	ParameterSet		BaseObjectType	Optional

Tableau 21 – MethodSet de ServerCommunicationServiceType

Attribut	Valeur				
BrowseName	MethodSet				
IsAbstract					
Références	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
HasComponent	Method	Connect			Optional
HasComponent	Method	Disconnect			Optional
HasComponent	Method	Transfer			Mandatory
HasComponent	Method	GetPublishedData			Optional

7.3.4.2 Méthode Connect

La Méthode Connect doit être utilisée pour établir une relation de communication avec un appareil physiquement connecté au Réseau. L'établissement de la relation de communication peut impliquer des vérifications des données d'identification qui font partie de l'addressData avec les données contenues dans l'appareil physique. L'Appareil de Communication effectue la vérification de correspondance de ce DeviceType conformément à une norme de protocole

réseau correspondante. Des informations détaillées connexes sont spécifiées dans l'IEC 62769-4:2020, Annexe F.

L'adresse des appareils est contenue dans le Point de Connexion de l'Instance d'Appareil correspondante dans le Modèle d'Information (Point de Connexion de l'Appareil). La relation de communication entre l'application de l'appareil associée au Modèle d'Information et l'appareil physique est identifiée avec de plus amples informations détaillées par l'identificateur de la relation de communication. Des informations détaillées sur la manière de gérer l'état d'une relation de communication sont décrites à l'Article 6. Les descriptions du Point de Connexion peuvent être hiérarchisées comme cela est décrit à l'Annexe A.

NOTE 1 Etant donné que le NodeId est un identificateur unique dans le domaine d'application du Modèle d'Information, le NodeId du Point de Connexion de l'Appareil peut être un identificateur unique pour toute relation de communication dans le domaine d'application d'un appareil de communication.

NOTE 2 Le terme "relation de communication" est introduit pour décrire le statut d'une infrastructure qui permet l'échange des données entre les données hébergées par le Modèle d'Information et un appareil physique. Lorsque la relation de communication est établie, l'échange des données est possible.

La signature de cette Méthode est spécifiée ci-dessous. Le Tableau 22 spécifie les arguments.

Signature

```
Connect (
    [in]   ByteString      CommunicationRelationId,
    [in]   <AddressData>,
    [out]  <DeviceInformation>,
    [out]  Int32           ServiceError);
```

Tableau 22 – Arguments de la Méthode Connect

Argument	Description
CommunicationRelationId	Il s'agit d'un identificateur généré par le client qui permet d'identifier cette connexion de manière unique. Il peut s'agir d'un index (par exemple, un NodeId) dont a besoin le client (= Serveur FDI) pour identifier les entrées dans sa topologie.
<AddressData>	Il s'agit d'un paramètre fictif pour une liste d'arguments spécifiques au protocole utilisée pour l'adresse et les données facultatives d'identification de l'appareil (des informations détaillées sont décrites dans l'IEC 62769-4:2020, Annexe F).
<DeviceInformation>	Il s'agit d'un paramètre fictif pour une liste d'arguments spécifiques au protocole dans laquelle sont stockées les données de résultat de connexion.
ServiceError	0: OK / exécution terminée, connexion établie avec succès Les autres valeurs qui représentent ServiceError sont spécifiées dans l'IEC 62769-4:2020, Annexe F.

7.3.4.3 Méthode Disconnect

La Méthode Disconnect doit être utilisée pour mettre fin à une relation de communication avec un Appareil.

La signature de cette Méthode est spécifiée ci-dessous. Les attributs de la Méthode Disconnect sont spécifiés dans le Tableau 23. La Méthode Disconnect est une Méthode d'appel synchrone.

Signature

```
Disconnect (
    [in] ByteString CommunicationRelationId,
    [out] Int32 ServiceError);
```

Tableau 23 – Arguments de la Méthode Disconnect

Argument	Description
CommunicationRelationId	Même identificateur utilisé dans la méthode Connect spécifiée en 7.3.4.2.
ServiceError	1: OK/déconnexion effectuée avec succès Les autres valeurs qui représentent ServiceError sont spécifiées dans l'IEC 62769-4:2020, Annexe F.

7.3.4.4 Méthode Transfer

La Méthode Transfer doit être utilisée pour effectuer l'échange d'informations avec un Appareil.

La signature de cette Méthode est spécifiée ci-dessous. Tous les arguments sont spécifiés dans le Tableau 24.

Signature

```
Transfer (
    [in] ByteString CommunicationRelationId,
    [in] <SendData>,
    [out] <ReceiveData>,
    [out] Int32 ServiceError);
```

Tableau 24 – Arguments de la Méthode Transfer

Argument	Description
CommunicationRelationId	Voir 7.3.4.2.
<SendData>	Il s'agit d'un paramètre fictif pour une liste d'arguments spécifique au protocole comme cela est décrit dans l'IEC 62769-4:2020, Annexe F. Les valeurs d'argument représentent la demande de services de communication spécifique au protocole envoyée à l'appareil.
<ReceiveData>	Il s'agit d'un paramètre fictif pour une liste d'arguments spécifique au protocole comme cela est décrit dans l'IEC 62769-4:2020, Annexe F. Les valeurs d'argument représentent la réponse de services de communication spécifique au protocole reçue de l'appareil.
ServiceError	0: OK / exécution terminée, ReceivedData contient le résultat Les autres valeurs qui représentent ServiceError sont spécifiées dans l'IEC 62769-4:2020, Annexe F.

7.3.4.5 Méthode GetPublishedData

Le Serveur FDI envoie les demandes GetPublishedData au Serveur de Communication FDI pour recevoir les données envoyées par les messages de données non sollicitées. Les données contenues dans l'argument SendData préparent l'échange des messages de données "non sollicitées" de l'appareil. Le contenu de SendData est spécifique au protocole. Le Serveur de Communication FDI met les demandes GetPublishedData dans une file d'attente associée à la relation de communication définie par l'argument CommunicationRelationId. L'argument PublishId identifie l'entrée de la file d'attente concernée. Chaque fois que le Serveur de Communication FDI reçoit des messages de données non sollicitées, il enregistre les données reçues en association avec l'entrée de la file d'attente existante qui a été créée pour GetPublishedData. Selon la technologie réseau sous-jacente (performances), la

méthode `GetPublishedData` peut immédiatement être retournée avec les données qui proviennent d'un message de données "non sollicitées".

L'extraction suivante des données envoyées par les messages de données non sollicitées fonctionne selon la même méthode `GetPublishedData`. Dans ce cas, l'argument `SendData` est vide. L'argument `PublishId` correspond à la valeur qui a été fournie avec l'appel initial `GetPublishedData` qui a établi l'émission de l'échange de messages de données "non sollicitées".

Afin d'arrêter l'appareil qui envoie les données "non sollicitées", la méthode `GetPublishedData` doit être utilisée à nouveau, mais les données contenues dans l'argument `SendData` mettent fin à l'échange des messages de données "non sollicitées" de l'appareil. Le Tableau 25 présente les arguments de la Méthode `GetPublishedData`.

Signature

```
GetPublishedData (
    [in]   ByteString      CommunicationRelationId,
    [in]   <SendData>,
    [out]  <ReceiveData>,
    [out]  DateTime        TimeStamp
    [in]   UInt32          PublishId,
    [out]  Int32            ServiceError);
```

Tableau 25 – Arguments de la Méthode `GetPublishedData`

Argument	Description
<code>CommunicationRelationId</code>	See 7.3.4.2.
<code><SendData></code>	Il s'agit d'un paramètre fictif pour une liste d'arguments spécifique au protocole comme cela est décrit dans l'IEC 62769-4:2020. Les valeurs d'argument contrôlent l'échange de messages non sollicités.
<code><ReceiveData></code>	Il s'agit d'un paramètre fictif pour une liste d'arguments spécifique au protocole comme cela est décrit dans l'IEC 62769-4:2020. Les valeurs d'argument transmettent les données en provenance des messages non sollicités.
<code>TimeStamp</code>	Moment auquel les données ont été publiées par l'appareil.
<code>PublishId</code>	Le numéro identifie un abonnement établi qui transmet les données qui proviennent de messages non sollicités.
<code>ServiceError</code>	0: OK/exécution terminée Les autres valeurs qui représentent <code>ServiceError</code> sont spécifiées dans l'IEC 62769-4:2020, Annexe F.

7.4 Profil de Serveur OPC UA pour le Serveur de Communication FDI

Les profils sont des groupes dénommés Unités de conformité tels que définis dans l'IEC 62541-6 et l'IEC 62541-7. La présence du terme "Facet" dans l'intitulé d'un Profil signifie que ce Profil est réputé faire partie d'un autre Profil plus vaste ou qu'il traite un aspect spécifique de l'OPC UA. Les Profils qui comportent le terme "Facet" dans leur intitulé sont réputés être combinés à d'autres Profils afin de définir les fonctionnalités complètes d'un Serveur ou Client OPC UA. Le profil Serveur OPC UA minimal exigé est le "Micro Embedded Device Server Profile".

Le tableau suivant spécifie la facette d'un Serveur OPC UA qui joue le rôle de Serveur de Communication FDI. Le Tableau 26 décrit les Unités de Conformité incluses dans cette facette.

Tableau 26 – Définition de `FDICommunicationServer_Facet`

Unité de conformité	Description	Facultatif/ Obligatoire
Serveur de communication FDI Modèle d'Information	Prend en charge au moins une instance de <code>CommunicationServerType</code> .	O

7.5 Mapping du Modèle d'Information du Serveur FDI au Modèle d'Information du Serveur de Communication FDI

7.5.1 Généralités

La représentation d'un Serveur de Communication FDI dans l'AddressSpace d'un Serveur FDI est presque identique à l'AddressSpace qui existe dans le Serveur de Communication FDI. Elle désigne en particulier la hiérarchie des Appareils Modulaires et les Paramètres de tous les Appareils. Toutefois, les Nœuds du Serveur FDI sont construits à partir de la description de l'appareil importée par le biais du Paquetage de Communication FDI.

7.5.2 Différences des Modèles d'Information

En raison de leurs différentes tâches, il existe néanmoins quelques différences et un ensemble de règles de synchronisation. L'exemple de Modèle d'Information représenté à la Figure 9 décrit les points communs et les différences entre les Modèles d'Information hébergés par le Serveur de Communication FDI et le Serveur FDI. En général, le Modèle d'Information hébergé par le Serveur de Communication FDI est un sous-ensemble du Modèle d'Information hébergé par le Serveur FDI. Les Instances d'Appareil dans le Serveur FDI et le Serveur de Communication FDI obéissent aux mêmes définitions de type. Par conséquent, les noms d'exploration des éléments communs du Modèle d'Information doivent être identiques.

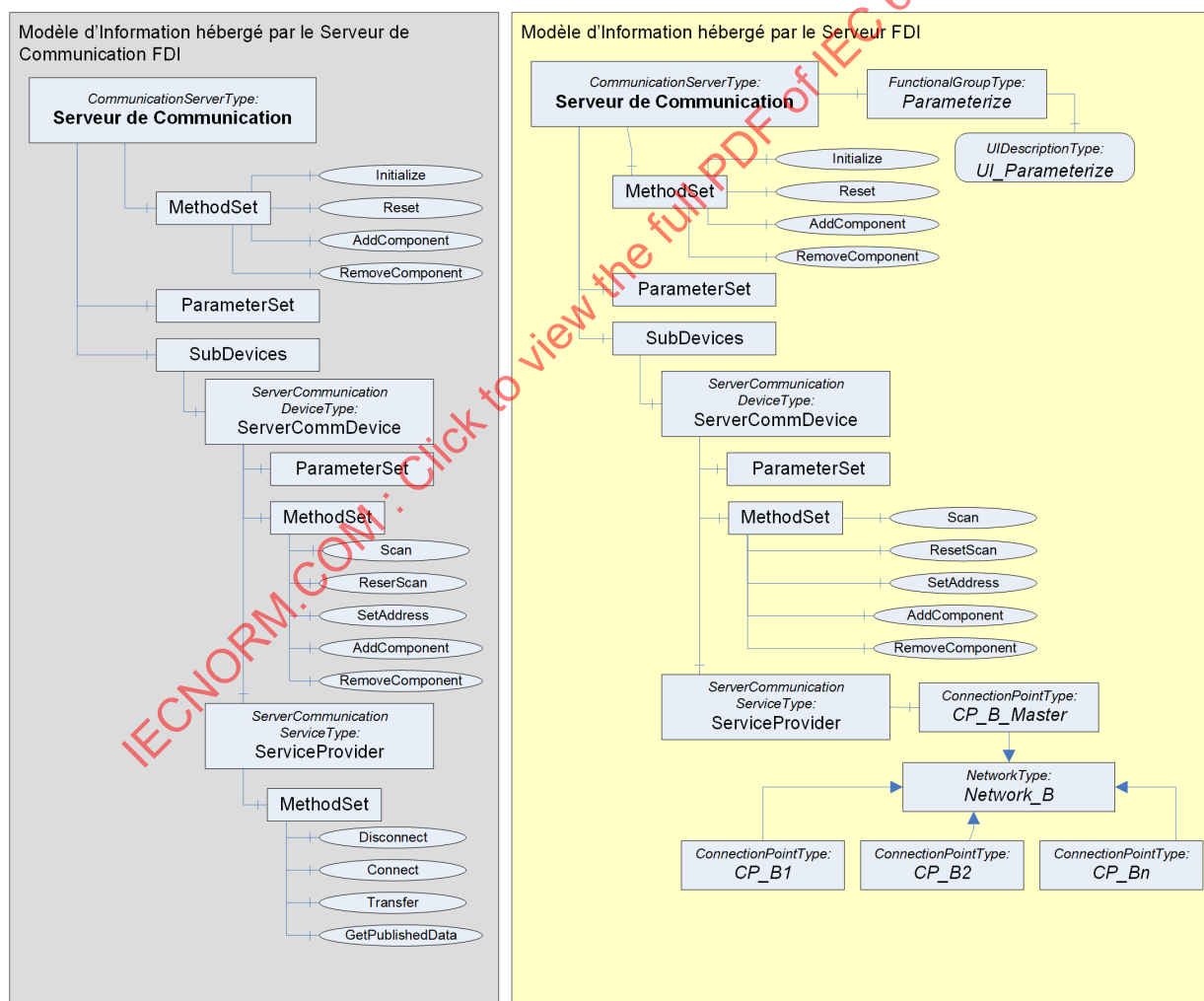


Figure 9 – Différences des Modèles d'Information (exemple)

La liste des différences dans les Modèles d'Information est la suivante:

- Le Serveur FDI prend en charge les versions en ligne et hors ligne de l'Appareil Modulaire; le Serveur de Communication FDI ne prend en charge que la version en ligne. La version en ligne du Serveur FDI représente la version dans le Serveur de Communication FDI, c'est-

à-dire que si les valeurs de Paramètre sont en lecture ou en écriture pour le modèle en ligne du Serveur FDI, ces opérations sont transmises au Serveur de Communication FDI. Ce phénomène se produit à la fois pour les Paramètres publics et privés.

NOTE La clé est une correspondance entre les noms d'exploration présents dans le Modèle d'Information hébergé par le Serveur FDI et le Modèle d'Information hébergé par le Serveur de Communication FDI. Cette clé permet la synchronisation générique des deux Modèles d'Information.

- Les UIP, les UID, les Actions et les Groupes Fonctionnels existent seulement dans le Serveur FDI.
- Les modules dans le Modèle d'Information hébergé par le Serveur FDI ont un composant de Point de Connexion utilisé pour connecter ce module à un réseau lors de la création de la Topologie d'Appareils. Le Modèle d'Information hébergé par le Serveur de Communication FDI n'affiche pas les éléments du Point de Connexion. La Topologie d'Appareils est gérée par le Serveur FDI seulement (voir l'IEC 62769-5).
- Le Serveur FDI peut représenter le ServiceProvider sans exposer le MethodSet afin d'empêcher un Client FDI d'invoquer les services de communication.

IECNORM.COM : Click to view the full PDF of IEC 62769-7:2021