IEC 62769-7

Edition 1.0 2015-05

INTERNATIONAL STANDARD

NORME INTERNATIONALE

colour inside

Field Device Integration (FDI) – Part 7: FDI Communication Devices

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

IEC 62769-7:2015-05(en-fr)

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

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

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

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 It also gives information on projects, replaced and withdrawn publications.

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

Electropedia - www.electropedia.org
The world's leading online dictionary of electronic and electrical terms containing more than 30 000 terms and definitions in English and French, with equivalent terms in 15 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC Glossary - std.iec.ch/glossary
More than 60 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

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

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

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

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

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

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

Electropedia - www.electropedia.org
Le premier dictionnaire en ligne de termes électroniques et électriques. Il contient plus de 30 000 termes et définitions en anglais et en français, ainsi que les termes équivalents dans 15 langues additionnelles. Egalement appelé Vocabulaire Electrotechnique International (IEV) en ligne.

Glossaire IEC - std.iec.ch/glossary
Plus de 60 000 entrées terminologiques électrotechniques, en anglais et en français, extraites des articles Termes et Définitions des publications IEC parues depuis 2002. Plus certaines entrées antérieures extraites des publications des CE 37, 77, 86 et CISPR de l'IEC.

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

IEC 62769-7

Edition 1.0 2015-05

INTERNATIONAL STANDARD

NORME INTERNATIONALE

colour inside

Field Device Integration (FDI) – Part 7: FDI Communication Devices

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

INTERNATIONAL ELECTROTECHNICAL COMMISSION

COMMISSION ELECTROTECHNIQUE INTERNATIONALE

ICS 25.040.40; 35.100

ISBN 978-2-8322-2641-4

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

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

FIELD DEVICE INTEGRATION (FDI) –

Part 7: FDI 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.

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.

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

CDV	Report on voting
65E/350/CDV	65E/420/RVD

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

This publication has been drafted in accordance with 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 publication will remain unchanged until the stability date indicated on the IEC website under "http://webstore.iec.ch" in the data related to the specific publication. At this date, the publication will be

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

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

INTRODUCTION

The International Electrotechnical Commission (IEC) draws attention to the fact that it is claimed that compliance with this document may involve the use of patents concerning

a) Method for the Supplying and Installation of Device-Specific Functionalities, see Patent Family DE10357276;

b) Method and device for accessing a functional module of automation system, see Patent Family EP2182418;

c) Methods and apparatus to reduce memory requirements for process control system software applications, see Patent Family US2013232186;

d) Extensible Device Object Model, see Patent Family US12/893,680.

IEC takes no position concerning the evidence, validity and scope of this patent right.

The holders of these patent rights have assured the IEC that he/she is willing to negotiate licences either free of charge or under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with IEC. Information may be obtained from:

a) ABB Research Ltd
Claes Rytoft
Affolterstrasse 4
Zurich, 8050
Switzerland

b) Phoenix Contact GmbH & Co KG
Intellectual Property, Licenses & Standards
Flachsmarktstrasse 8, 32825 Blomberg
Germany

c) Fisher Controls International LLC
John Dilger, Emerson Process Management LLLP
301 S. 1st Avenue, Marshaltown, Iowa 50158
USA

d) Rockwell Automation Technologies, Inc.
1 Allen-Bradley Drive
Mayfield Heights, Ohio 44124
USA

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified above. IEC shall not be held responsible for identifying any or all such patent rights.

ISO (www.iso.org/patents) and IEC (http://patents.iec.ch) maintain on-line data bases of patents relevant to their standards. Users are encouraged to consult the data bases for the most up to date information concerning patents.

FIELD DEVICE INTEGRATION (FDI) –

Part 7: FDI 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.

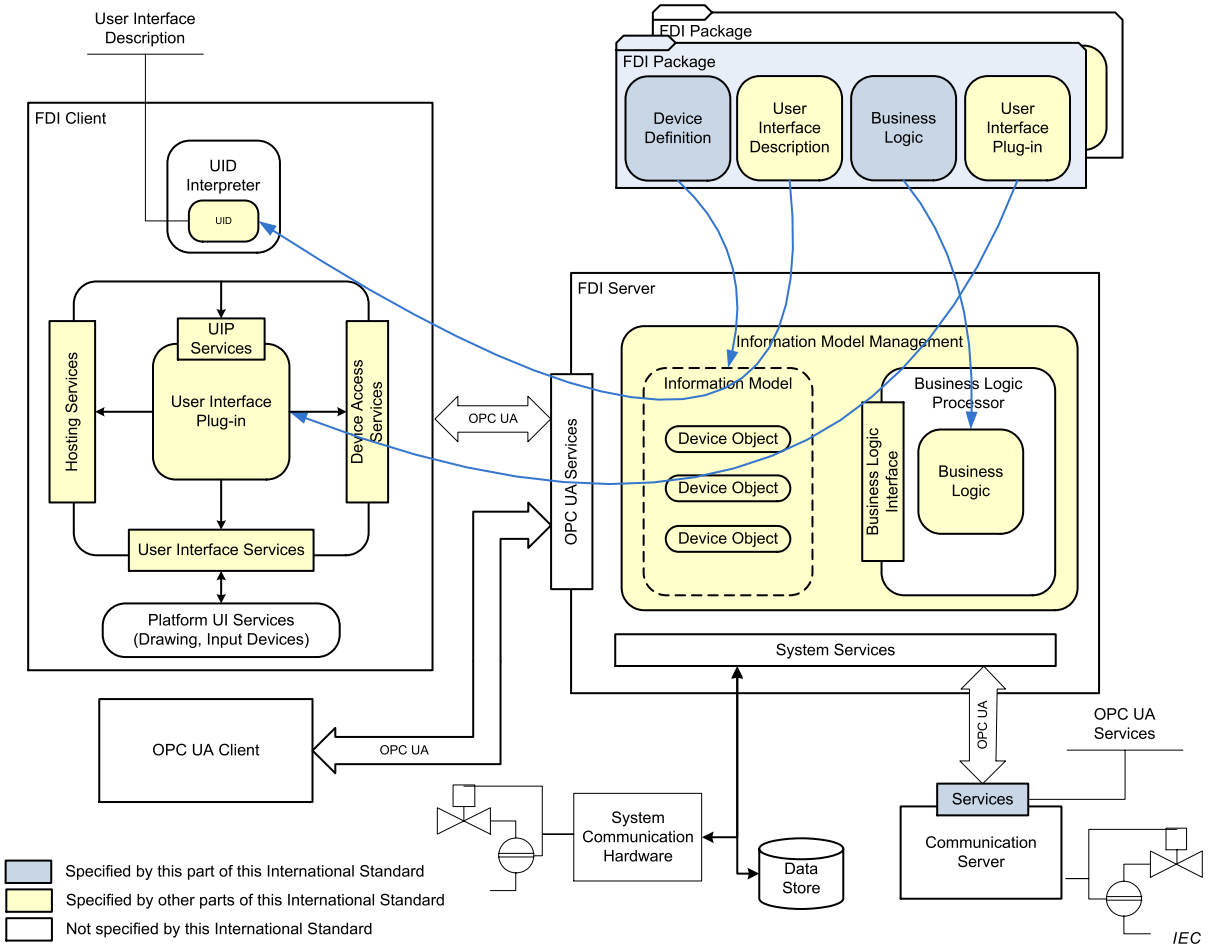


Figure 1 – FDI architecture diagram

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 TR 62541-1, *OPC Unified Architecture – Part 1: Overview and Concepts*

IEC 62541-4, *OPC Unified Architecture – Part 4: Services*

IEC 62541-6, *OPC Unified Architecture – Part 6: Mappings*

IEC 62541-7, *OPC Unified Architecture – Part 7: Profiles*

IEC 62541-100, *OPC Unified Architecture – Part 100: OPC UA for Devices*

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

NOTE IEC 62769-1 is technically identical to FDI-2021.

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

NOTE IEC 62769-2 is technically identical to FDI-2022.

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

NOTE IEC 62769-3 is technically identical to FDI-2023.

IEC 62769-4:2015, *Field Device Integration (FDI) – Part 4: FDI Packages*

NOTE IEC 62769-4 is technically identical to FDI-2024.

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

NOTE IEC 62769-5 is technically identical to FDI-2025.

3 Terms, definitions, abbreviated terms, acronyms 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.

3.1.1
gateway
communication device that enables to bridge between different physical networks or different protocols

3.2 Abbreviated terms and acronyms

For the purposes of this document, the abbreviated terms and acronyms 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 graphical notation

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 EED 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.

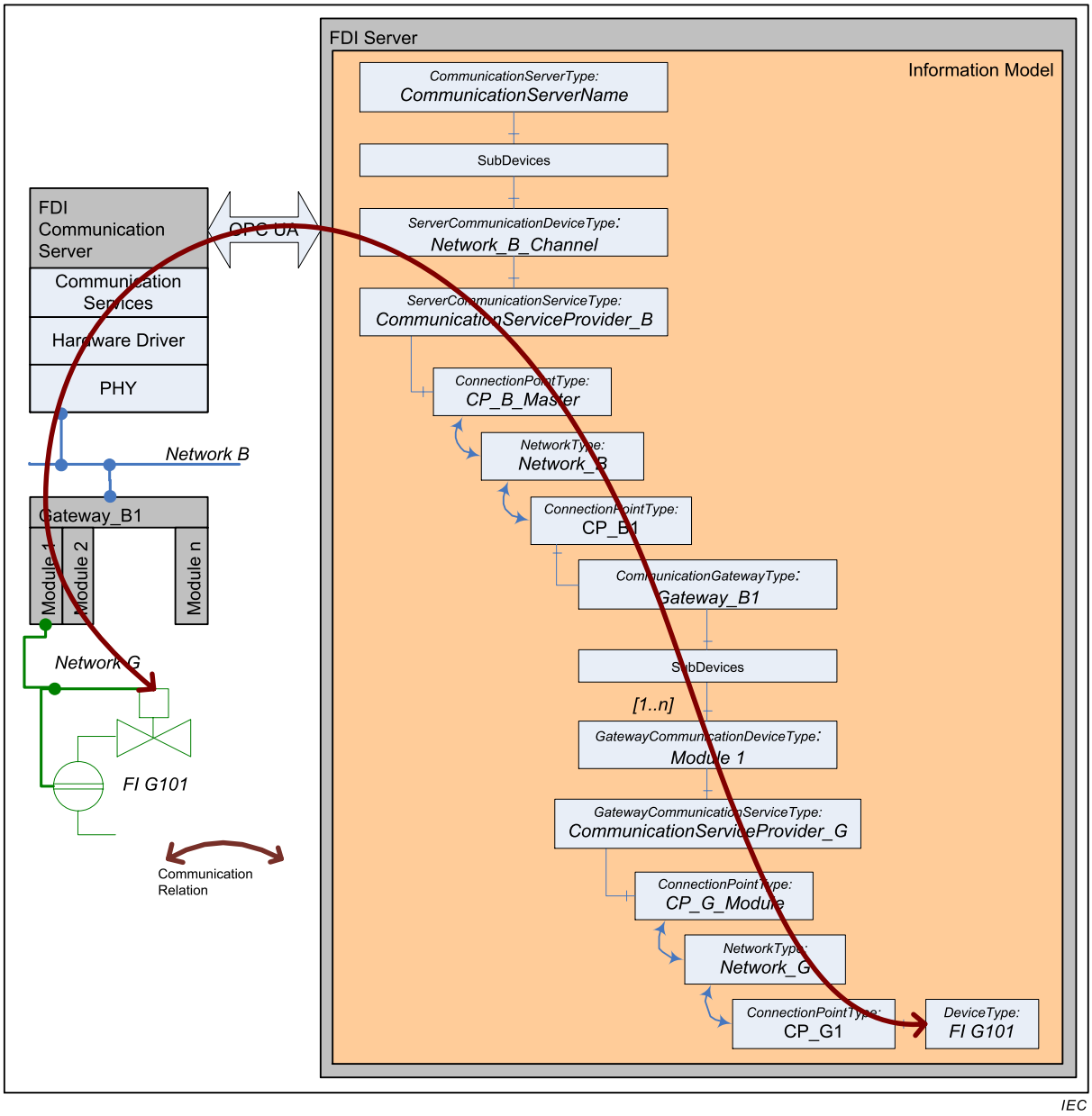


Figure 2 – FDI communication infrastructure architecture

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 Information Model used elements reflect the actual network topology.

a) 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 subdevices see IEC 62769-5).

b) 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.

c) 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.

d) 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.

e) 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.

f) 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.

g) 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 62769-4:2015, Annex F).

The EDD elements as specified in IEC 62769-4:2015, 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 62769-4:2015, Annex F. The parameter shall contain any parameter that requires adjustment for proper communication service operation.

b) Physical Layer identification. Protocol specific definitions are found in IEC 62769-4:2015, Annex F.

c) 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.

d) The COMPONENT definition shall be used to support the system implemented topology configuration. Protocol specific definitions are found in IEC 62769-4:2015, Annex F. The related COMPONENT definitions are described in 5.2.2, 5.2.3, 5.2.4, and 5.2.7.

e) 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.

f) 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.

g) 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 62769-4:2015, Annex F.

h) MENU:
The Menu structure shall follow the Menu conventions for PC based applications according to IEC 61804-4 enabling 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”. Protocol specific definitions are found in IEC 62769-4:2015, Annex F.

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 to perform generic, FDI Server driven (device) topology configuration. 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 COMPONENT_CHILD.

<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 may 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 according to 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 COMPONENT_CHILD.

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 COMPONENT_CHILD.

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 field bus 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 to perform generic, FDI Server driven network topology configuration. 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 COMPONENT_CHILD.

<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 to analyze 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 IEC 61804-3 specified error codes 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 case of an error the Action can return a problem description.

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 to analyze a set of child instances. 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 IEC 61804-3 specified error codes 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 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 e.g. 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.

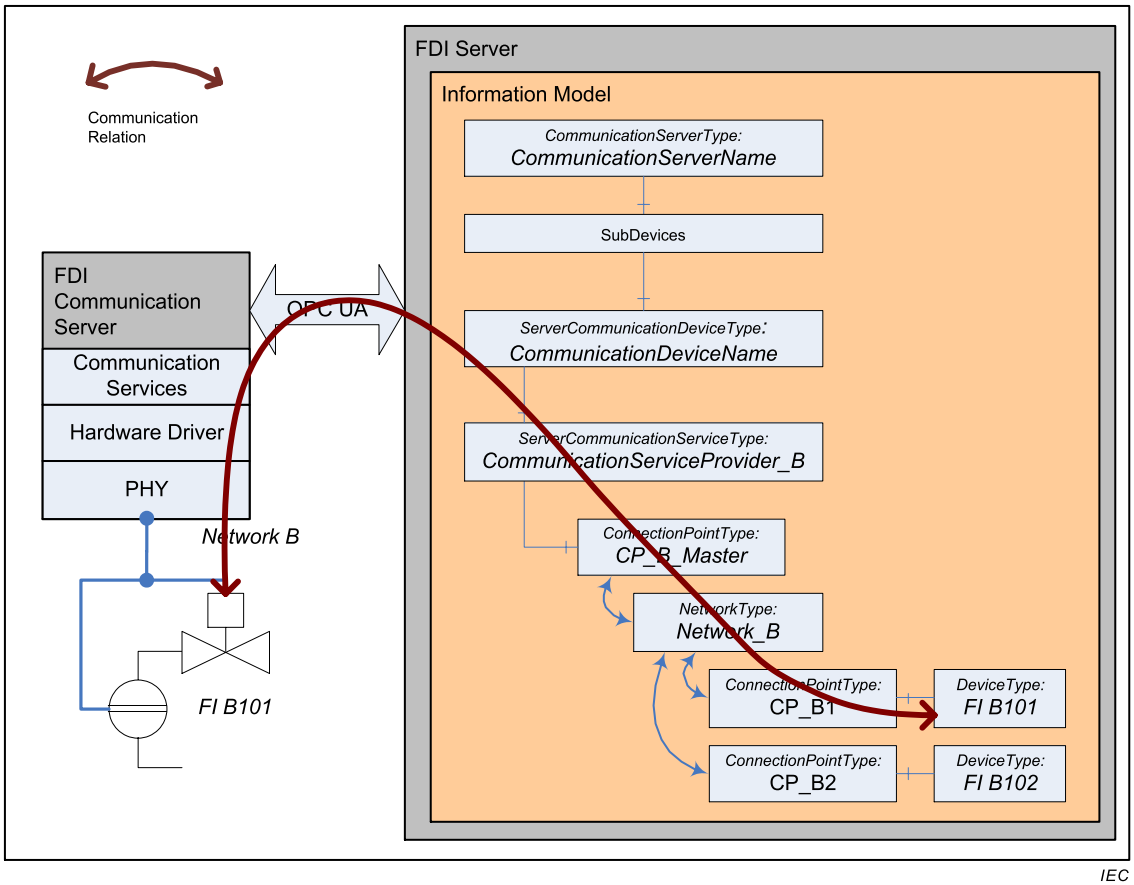


Figure 3 – Communication relation

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

The following state chart 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.

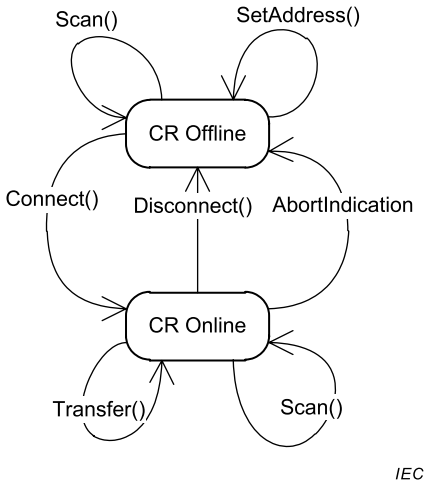


Figure 4 – Communication relation state chart

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 physical network connected device.

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 Sever 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.

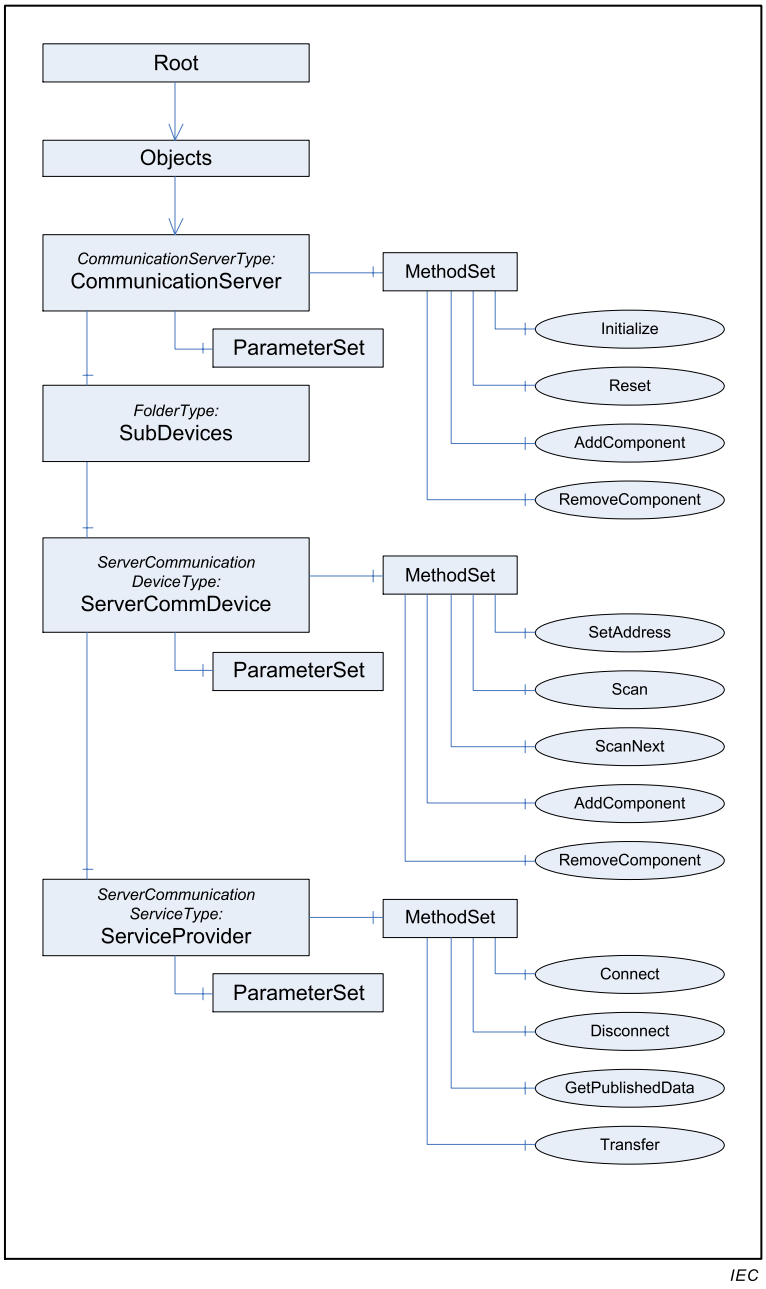


Figure 5 – FDI Communication Server AddressSpace

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.

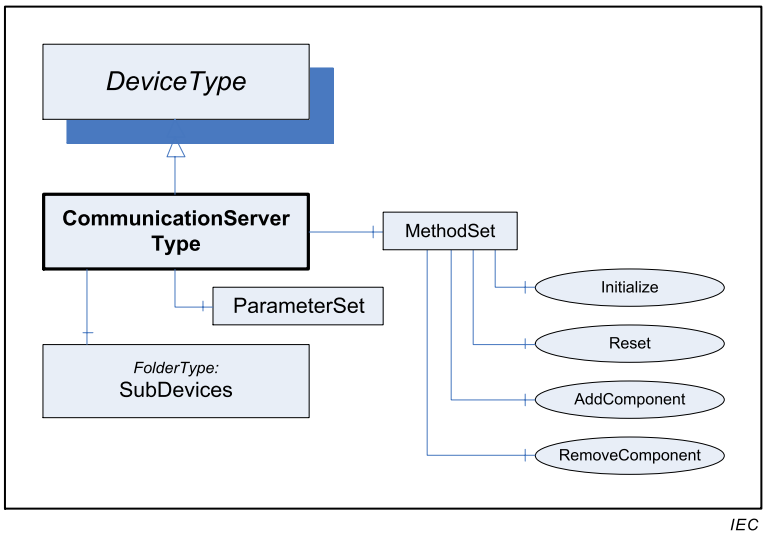


Figure 6 – CommunicationServerType

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 NodeId of these Methods will be fixed and defined in this standard. FDI Communication

Server clients therefore do not have to browse for these Methods. They can use the fixed NodeId 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 communication hardware 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 status closed.

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 FDI Communication Server 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**	**DataType**	**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 in case the FDI Communication Sever 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**	**DataType**	**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**	**DataType**	**TypeDefinition**	**ModellingRule**
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. The protocol specific, mandatory bus parameters are specified in IEC 62769-4:2015, Annex F. Figure 7 shows the ServerCommunicationDeviceType definition that is formally defined in Table 13 and Figure 14.

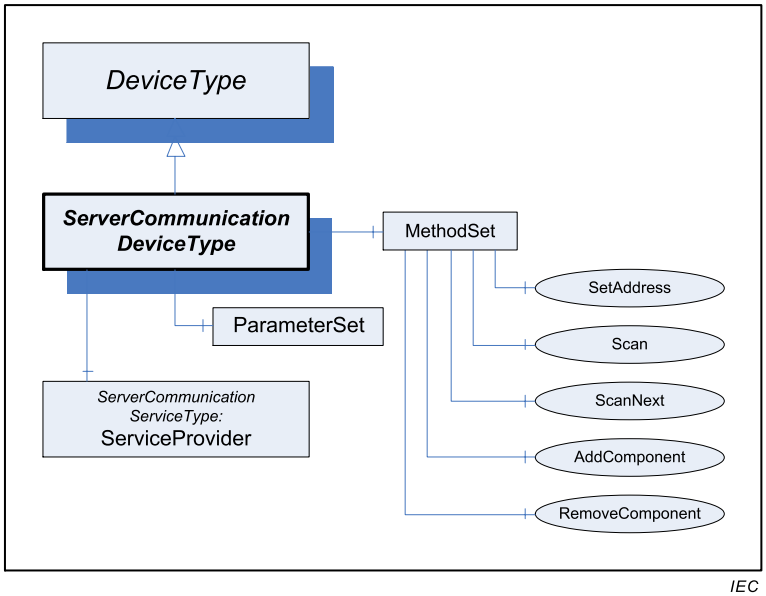


Figure 7 – ServerCommunicationDeviceType

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		ServerCommunication ServiceType	Mandatory

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 physical network connected devices. 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 In case 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 physical network connected devices. Each of the physical network connected devices 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 62769-4:2015, Annex F. Return an empty string for TopologyScanResult in 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 in case 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 physical network connected device 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 SetAdress Method are described in Table 19.

Signature

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

Table 19 – SetAddress Method arguments

Argument	Description
OldAddress	The argument represents 1..n protocol specific values representing the existing protocol specific network address of the physical network connected device. Values that represent a network address are specified in IEC 62769-4:2015, Annex F.
NewAddress	The argument represents 1..n protocol specific arguments representing the new protocol specific network address that shall be assigned to a physical network connected device. . Values that represent a network address arguments are specified in IEC 62769-4:2015, Annex F.
ServiceError	0: OK / execution finished successfully -1: SetAddress Failed / not initialized -2: SetAddress Failed / not connected to a network -3: SetAddress Failed / no device found responding to oldAddress -4: SetAddress Failed / duplicate address error -5: SetAddress Failed / device did not accept new address -6: SetAddress Failed / invalid oldAddress (in terms of syntax, data type, data format, and so on) -7: SetAddress Failed / invalid newAddress (in terms of syntax, data type, data format, and so on) -8: SetAddress Failed / not possible in status connected

7.3.4 ServerCommunicationServiceType

7.3.4.1 General

Communication services provide the means to communicate with a Device or to e.g. execute a Scan on some 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 standard. 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 including their NodeIds are uniquely defined in this standard. 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.

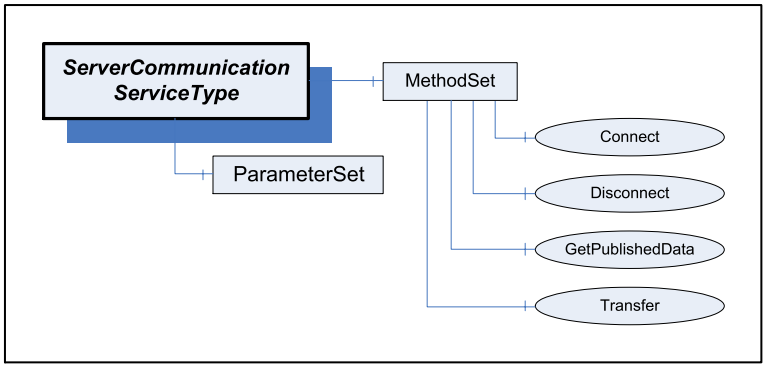


Figure 8 – ServerCommunicationServiceType

Table 20 – ServerCommunicationServiceType definition

Attribute	Value				
BrowseName	ServerCommunicationServiceType				
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 21 – MethodSet of ServerCommunicationServiceType

Attribute	Value				
BrowseName	MethodSet				
IsAbstract					
References	**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 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 according to a corresponding network protocol standard. Related details are specified in IEC 62769-4:2015, 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.

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 describing 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]   BaseDataType[]    AddressData,
   [out]  BaseDataType[]    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 to identify entries in its topology.
AddressData	A protocol specific argument list that is used for the address and optional device identification data (details described in IEC 62769-4:2015, Annex F).
DeviceInformation	A protocol specific argument list in which the connect result data are stored.
ServiceError	0: OK / execution finished, connection established successfully -1: Connect Failed / device not found -2: Connect Failed / invalid device address -3: Connect Failed / invalid device identification

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 -1: Disconnect Failed / no existing communication relation -2: Disconnect Failed / invalid communication relation identifier

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]  BaseDataType[]   SendData,
   [out] BaseDataType[]   ReceiveData,
   [out] Int32            ServiceError);
```

Table 24 – Transfer Method arguments

Argument	Description
CommunicationRelationId	See 7.3.4.2.
SendData	A protocol specific list of values as described in IEC 62769-4:2015, Annex F. The argument values represent the protocol specific communication service request that is sent to the device.
ReceiveData	A protocol specific list of values as described in IEC 62769-4:2015, 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 -1: Transfer Failed / No existing communication relation. -2: Transfer Failed / Invalid communication relation identifier -3: Transfer Failed / Invalid sendData content -4: Transfer Failed / Invalid receiveData format

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 argument SendData contained data 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]  BaseDataType[]   SendData,
   [out] BaseDataType[]   ReceiveData,
   [out] DateTime         TimeStamp
   [in]  UInt32           PublishId,
   [out] Int32            ServiceError);
```

Table 25 – GetPublishedData Method arguments

Argument	Description
CommunicationRelationId	See 7.3.4.2.
SendData	A protocol specific list of values as described in IEC 62769-4:2015, Annex F. The argument values control the exchange of unsolicited messages.
ReceiveData	A protocol specific list of values as described in IEC 62769-4:2015, 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 -2: Call Failed / unknown PublishId -3: GetPublishedData Failed / not supported -4: GetPublishedData Failed / no existing communication relation. -5: GetPublishedData Failed / invalid communication relation identifier -6: GetPublishedData Failed / invalid sendData content -7: GetPublishedData Failed / invalid receiveData format -8: GetPublishedData Failed / no data published that fits to the SendData argument

7.4 OPC UA Server Profile for FDI Communication Server

Profiles are named groupings of ConformanceUnits as defined in 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 Comunication Server Information Model	Support at least one instance of CommunicationServerType.	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.

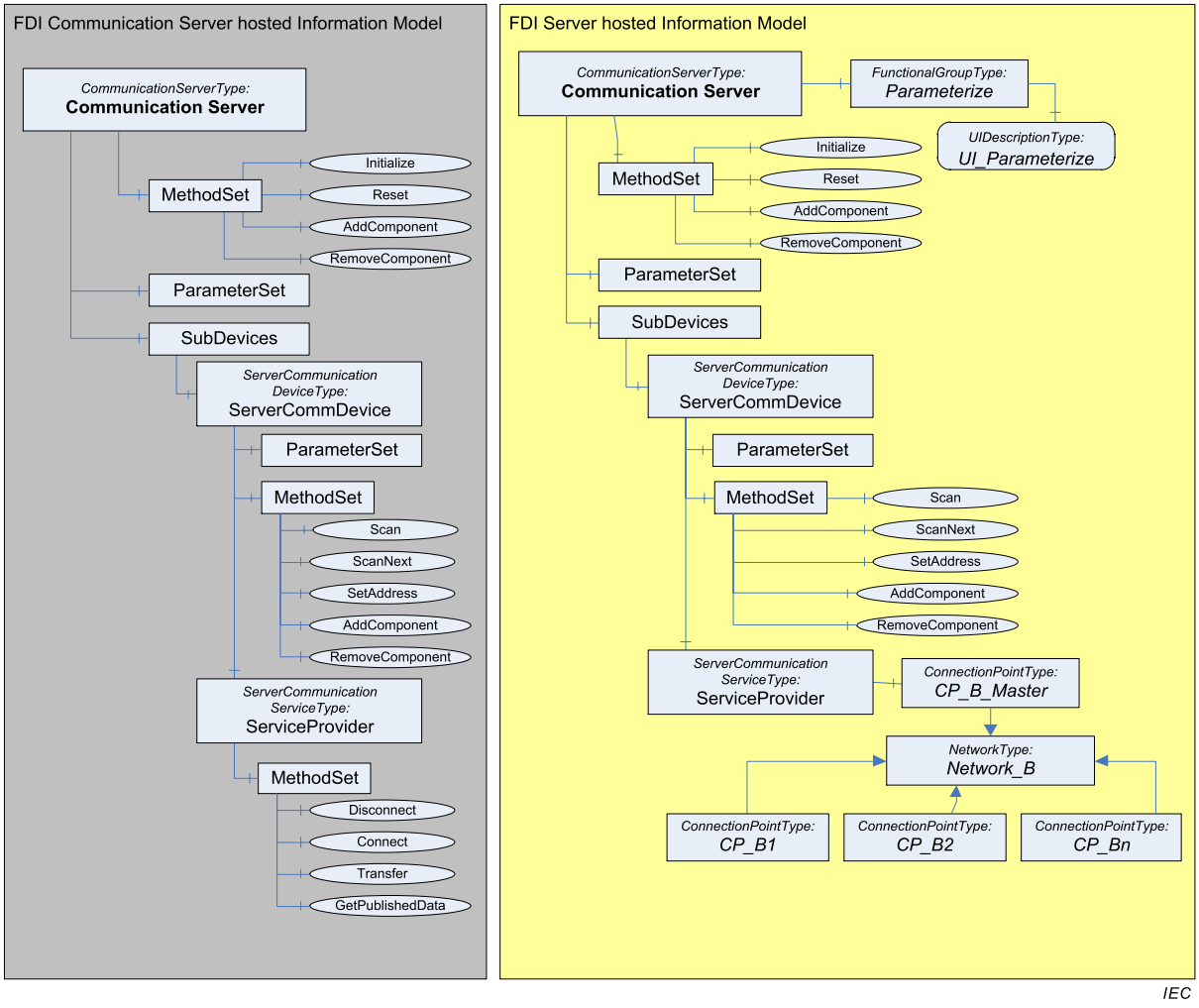


Figure 9 – Information Model differences (example)

The list of differences in the IM 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, UIDs, 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 7.7specifies 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

NOTE COMPONENT defines an attribute PRODUCT_URI that can be used for automatic discovery of the matching CommunicationServer. Since we are not using the COMPONENT construct for lightweight CommunicationServers, systems have to provide manual discovery.

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 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 define in 5.2.6 apply.

7.7.4 Documentation

The FDI Communication Package shall provide documentation describing:

i) the software installation related environment requirements and procedures

j) 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 life cycle 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.

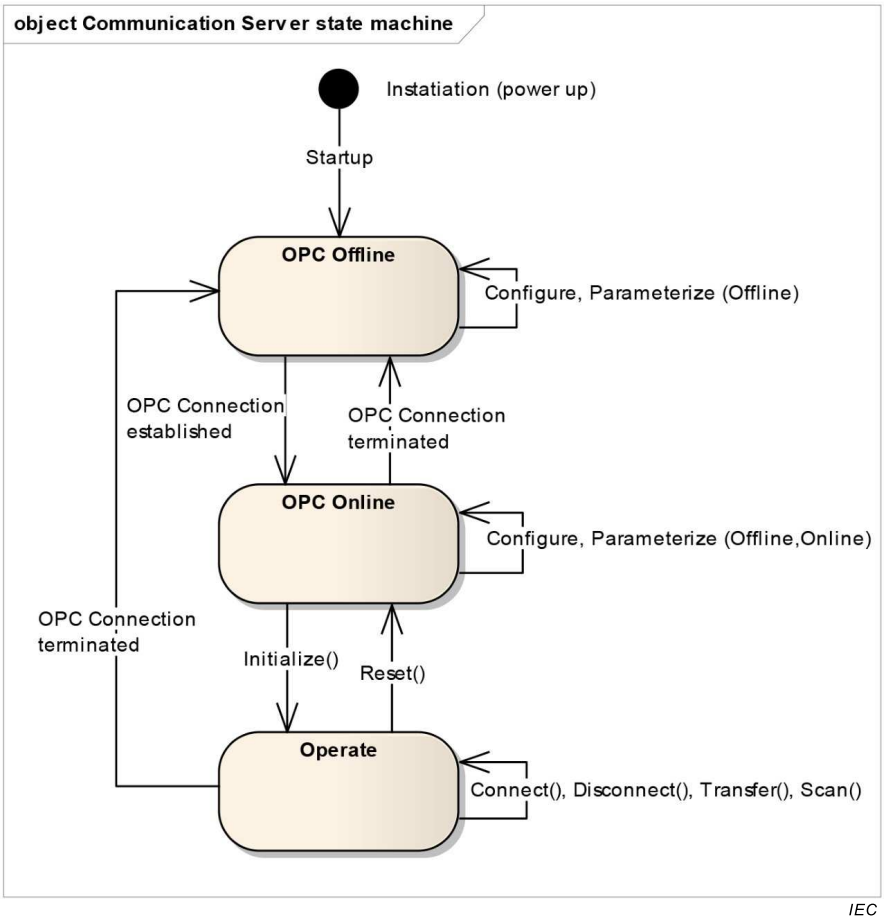


Figure 10 – FDI Communication Server state machine

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 □) 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. According to 7.8.4 the FDI Communication Server needs to know the address information about the Discovery Server. The standard 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 according to 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 start 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 according to IEC 62541.

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 then the FDI Server shall perform the synchronization using the sub device browse name matching and the invocation of the FDI Communication Server hosted module management services.

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

7.8.10 Configure 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 name matching. 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 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.

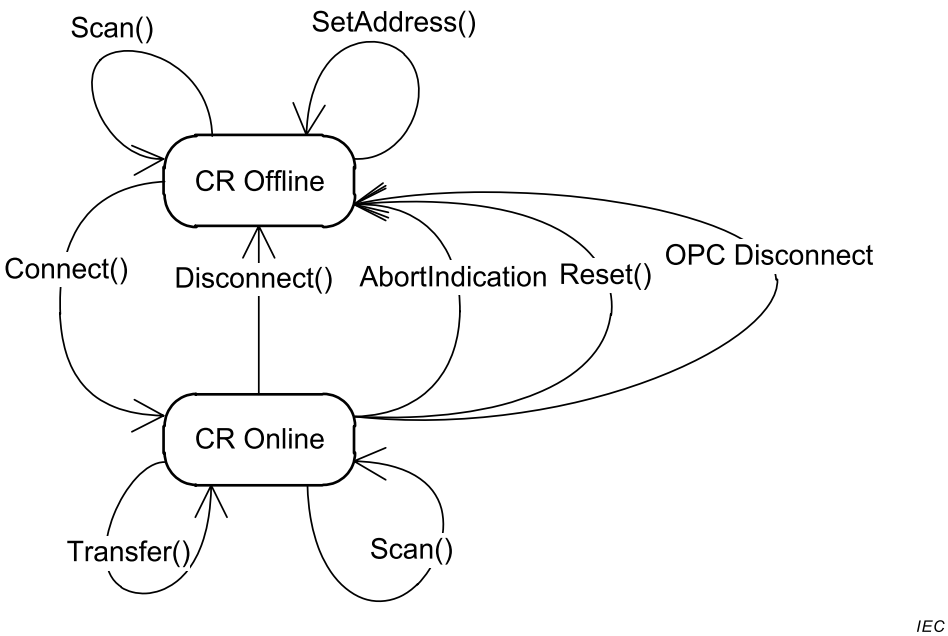


Figure 11 – Communication relation state chart

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 establishing a communication relation between the device application and the physical network connected device. 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 physical network connected device.

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 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.

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.

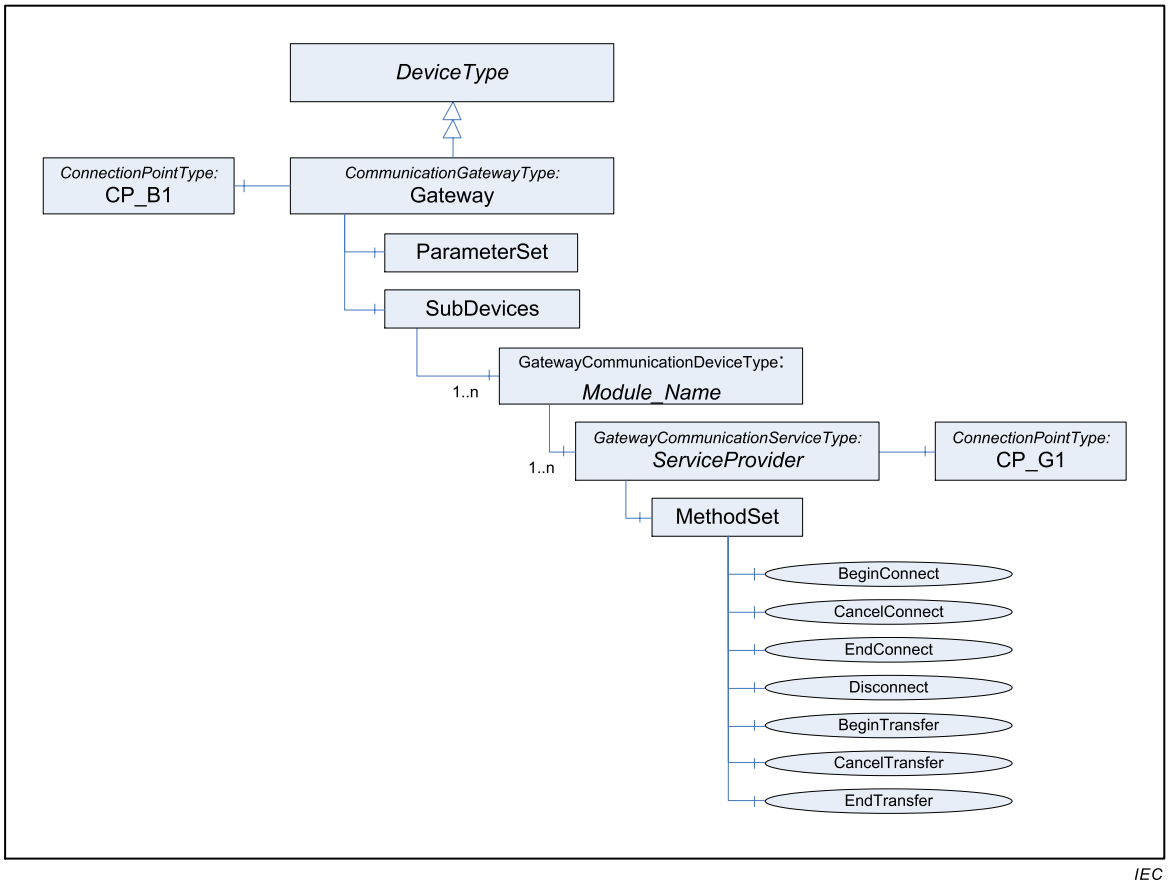


Figure 12 – Gateway information model

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.

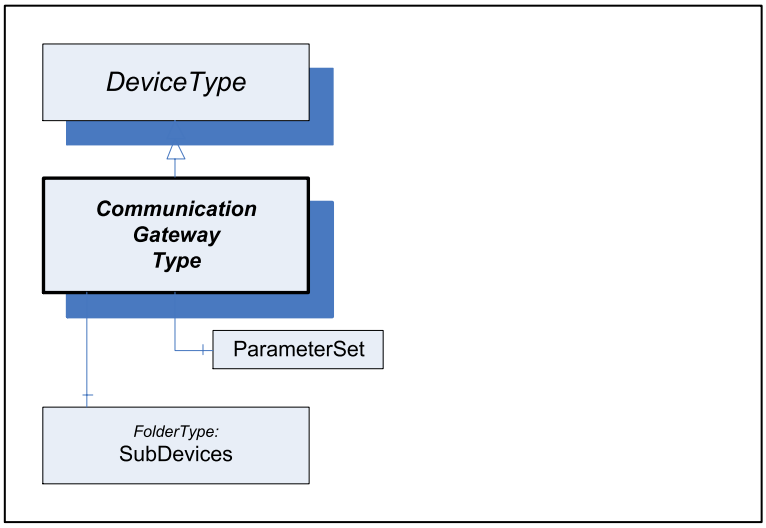


Figure 13 – CommunicationGatewayType

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 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 62769-4:2015, Annex F. Figure 14 shows the GatewayCommunicationDeviceType definition that is formally defined in Table 28 and Table 29.

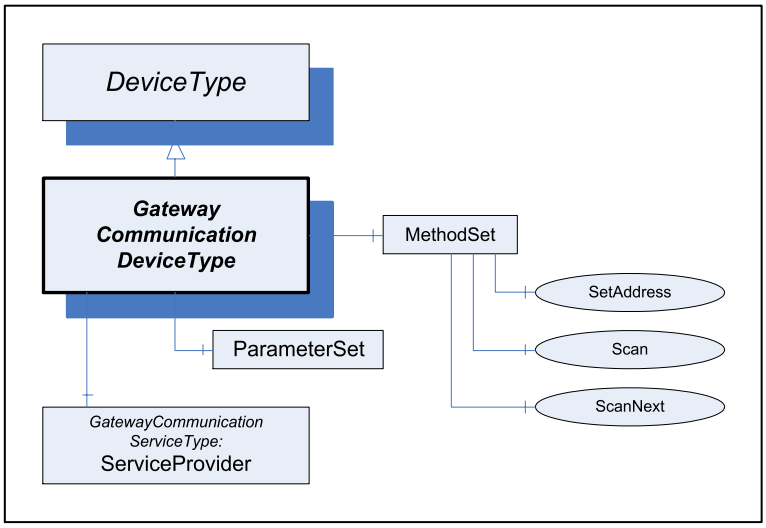


Figure 14 – GatewayCommunicationDeviceType

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 DI.					
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 physical network connected devices. 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 physical network connected devices. 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 e.g. execute a Scan on some 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 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 do 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.

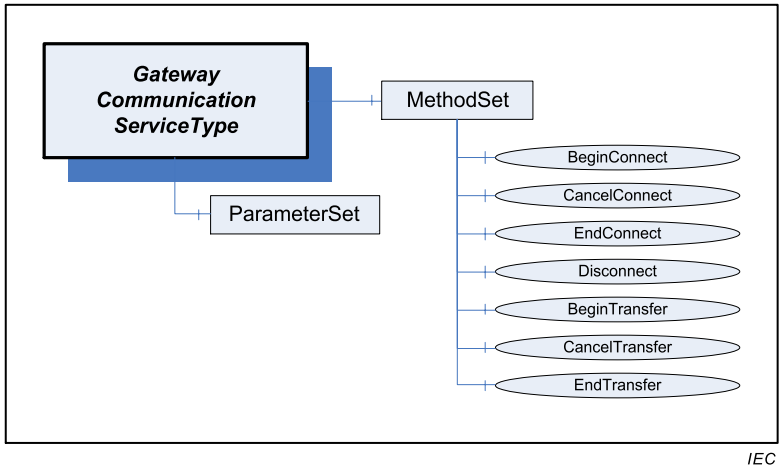


Figure 15 – GatewayCommunicationServiceType

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 according to a corresponding network protocol standard. Related details are specified in IEC 62769-4:2015, 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]   BaseDataType[]   AddressData,
   [out]  BaseDataType[]   DeviceInformation,
   [in]   UInt32           ServiceID,
   [out]  UInt32           DelayForNextCall,
   [out]  Int32            ServiceError);

EndConnect(
   [in]   ByteString       CommunicationRelationId,
   [out]  BaseDataType[]   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 to identify entries in its topology.
AddressData	A protocol specific argument list that is used for the address and optional device identification data (details are described in IEC 62769-4:2015, 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]  BaseDataType[]    SendData,
    [out] BaseDataType[]    ReceiveData,
    [in]  UInt32            ServiceId,
    [out] UInt32            DelayForNextCall,
    [out] Int32             ServiceError);

EndTransfer(
    [in]  ByteString        CommunicationRelationId,
    [out] BaseDataType[]    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	A protocol specific list of values as described in IEC 62769-4:2015, Annex F. The argument values represent the protocol specific communication service request that is sent to the device.
ReceiveData	A protocol specific list of values as described in IEC 62769-4:2015, 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 onto 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 X 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 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.

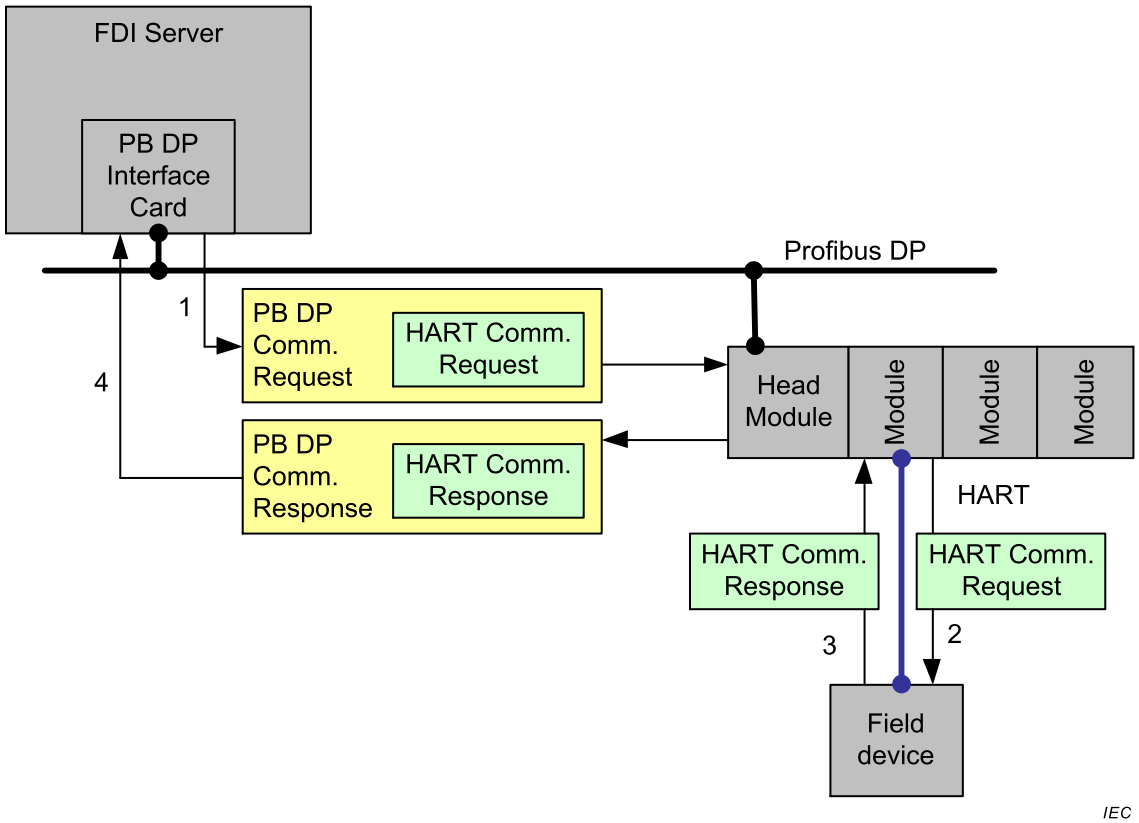


Figure 16 – Nested Communication

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

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 establishing a communication relation between the device application and the physical network connected device. This happens through 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 physical network connected device.

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 62769-4:2015, 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 abort according to 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.

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 layer 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 lower layers.

A.2 Convention for protocol specific annex creation

A.2.1 Connection Point

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 according to 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.

Annex B
(normative)

Namespace and Mappings

This appendix defines the numeric identifiers for all of the numeric *NodeIds* defined in this standard. 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 standard can be found here:
http://www.fdi-cooperation.com/tl_files/Specification/1.0/Schemas/Opc.Ua.Fdi7.NodeIds.csv

An electronic version of the complete Information Model defined in this standard 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 standard can be found here:
http://www.fdi-cooperation.com/tl_files/Specification/1.0/Schemas/Opc.Ua.Fdi7.NodeSet2.xml

Bibliography

FDI-2021, *FDI Project Technical Specification – Part 1: Overview*
<available at www.fdi-cooperation.com>

FDI-2022, *FDI Project Technical Specification – Part 2: FDI Client*
<available at www.fdi-cooperation.com>

FDI-2023, *FDI Project Technical Specification – Part 3: FDI Server*
<available at www.fdi-cooperation.com>

FDI-2024, *FDI Project Technical Specification – Part 4: FDI Packages*
<available at www.fdi-cooperation.com>

FDI-2025, *FDI Project Technical Specification – Part 5: FDI Information Model*
<available at www.fdi-cooperation.com>

FDI-2026, *FDI Project Technical Specification – Part 6: FDI Technology Mapping*
<available at www.fdi-cooperation.com>

FDI-2027, *FDI Project Technical Specification – Part 7: FDI Communication Devices*
<available at www.fdi-cooperation.com>

SOMMAIRE

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

_____

INTEGRATION DES APPAREILS DE TERRAIN (FDI) –

Partie 7: Appareils de communication FDI

AVANT-PROPOS

1) La Commission Electrotechnique 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. A 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.

La Norme internationale IEC 62769-7 a été établie par le sous-comité 65E: Les appareils 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.

Le texte de cette norme est issu des documents suivants:

CDV	Rapport de vote
65E/350/CDV	65E/420/RVD

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

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

Une liste de toutes les parties de la série IEC 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 cette publication 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 à la publication recherchée. A cette date, la publication sera

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

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

INTRODUCTION

La Commission Electrotechnique Internationale (IEC) attire l'attention sur le fait qu'il est déclaré que la conformité avec les dispositions du présent document peut impliquer l'utilisation de brevets concernant:

a) la méthode de fourniture et d'installation des fonctionnalités spécifiques aux appareils (cf. famille de brevets DE10357276);

b) la méthode et l'appareil utilisés pour l'accès à un module fonctionnel du système d'automation (cf. famille de brevets EP2182418);

c) les méthodes et les appareils utilisés pour diminuer les exigences mémoire relatives aux applications logicielles du système de commande de processus (cf. famille de brevets US2013232186);

d) modèle d'objet d'appareil extensible (cf. famille de brevets US12/893,680).

L'IEC ne prend pas position quant à la preuve, à la validité et à la portée de ces droits de propriété.

Le détenteur de ces droits de propriété a donné l'assurance à l'IEC qu'il consent à négocier des licences avec des demandeurs du monde entier, soit sans frais soit à des termes et conditions raisonnables et non discriminatoires. A ce propos, la déclaration du détenteur des droits de propriété est enregistrée à l'IEC. Des informations peuvent être demandées à:

a) ABB Research Ltd
Claes Rytoft
Affolterstrasse 4
Zurich, 8050
Suisse

b) Phoenix Contact GmbH & Co KG
Intellectual Property, Licenses & Standards
Flachsmarktstrasse 8, 32825 Blomberg
Allemagne

c) Fisher Controls International LLC
John Dilger, Emerson Process Management LLLP
301 S. 1st Avenue, Marshaltown, Iowa 50158
Etats-Unis

d) Rockwell Automation Technologies, Inc.
1 Allen-Bradley Drive
Mayfield Heights, Ohio 44124
Etats-Unis

L'attention est d'autre part attirée sur le fait que certains des éléments du présent document peuvent faire l'objet de droits de propriété autres que ceux qui ont été mentionnés ci-dessus. L'IEC ne saurait être tenue pour responsable de l'identification de ces droits de propriété en tout ou partie.

L'ISO (www.iso.org/patents) et l'IEC (http://patents.iec.ch) tiennent à jour des bases de données en ligne sur les brevets relatifs à leurs normes. Les utilisateurs sont encouragés à consulter ces bases de données pour obtenir l'information la plus récente concernant les brevets.

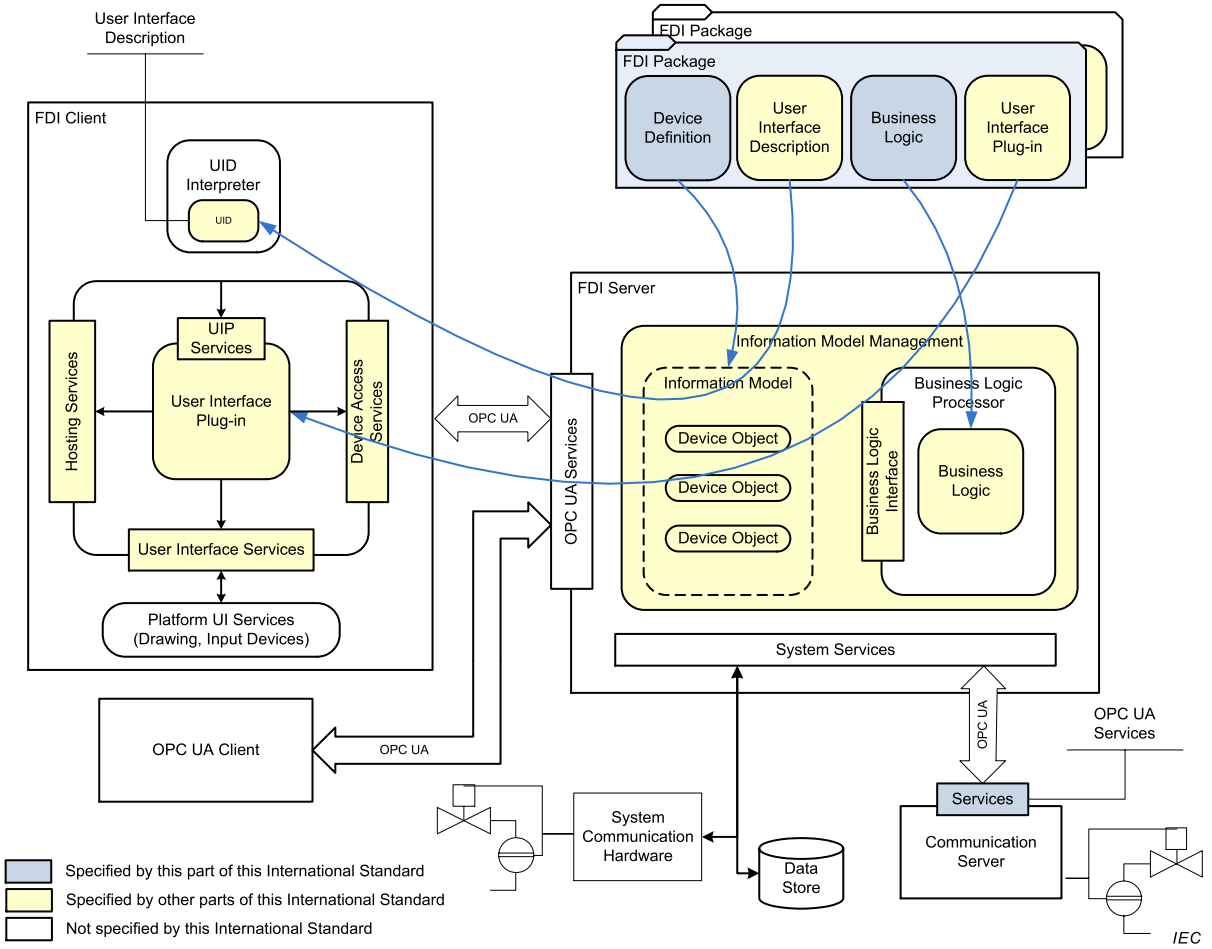
INTEGRATION DES APPAREILS DE TERRAIN (FDI) –

Partie 7: Appareils de communication FDI

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 (voir l'IEC 62769-5).

L'architecture FDI complète est présentée à la Figure 1. Les composants architecturaux relevant du domaine d'application du présent document ont été mis en évidence dans cette illustration. 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.



Anglais	Français
User Interface Description	Description d'Interface Utilisateur
FDI Client	Client FDI
UID Interpreter	Interpréteur d'UID
UID	UID
Hosting Services	Services d'hébergement
UIP Services	Service UIP (plugiciel d'interface utilisateur)

Anglais	Français
User interface Plug-in	Plugiciel d'interface utilisateur
Device Access Services	Services d'accès à l'appareil
User Interface Services	Services d'interface utilisateur
Platform UI Services (Drawing, Input Devices)	Services d'interface utilisateur (UI) de plate-forme (Dessin, appareils d'entrée)
OPC UA Client	Client OPC UA
OPC UA	OPC UA (Architecture Unifiée OPC)
FDI Package	Paquetage FDI
Device Definition	Définition d'appareil
User Interface Description	Description d'interface utilisateur
Business Logic	Logique applicative
User Interface Plug-in	Plugiciel d'interface utilisateur
FDI Server	Serveur FDI
OPC UA Services	Service OPC UA
Information Model Management	Gestion du Modèle d'information
Information Model	Modèle d'information
Device Object	Objet d'Appareil
Business Logic Interface	Interface de la logique applicative
Business Logic Processor	Processeur de la logique applicative
System Services	Services système
System Communication Hardware	Matériel de communication système
Data Store	Magasin de données
Services	Services
Communication Server	Serveur de communication
Specified by this part of this International Satandard	Spécifié par la présente partie de la présente Norme internationale
Specified by other parts of this I. S.	Spécifié par d'autres parties de la présente Norme internationale.
Not specified by this I. S.	Non spécifié par la présente Norme internationale.

Figure 1 – Diagramme de l'architecture FDI

2 Références normatives

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

IEC 61804-3, *Blocs fonctionnels (FB) pour les procédés industriels – Partie 3: Langage de description électronique de produit (EDDL)*

IEC 61804-4, *Blocs fonctionnels (FB) pour les procédés industriels – Partie 4: Langage de description électronique de produit (EDDL)*

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

IEC TR 62541-1, *OPC Unified Architecture – Part 1: Overview and Concepts* (disponible en anglais seulement)

IEC 62541-4, *Architecture unifiée OPC – Partie 4: Services*

IEC 62541-6, *Architecture unifiée OPC – Partie 6: Correspondances*

IEC 62541-7, *Architecture unifiée OPC – Partie 7: Profils*

IEC 62541-100, *OPC Unified Architecture – Part 100: OPC UA for Devices* (disponible en anglais seulement)

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

NOTE L'IEC 62769-1 est techniquement identique à la FDI-2021.

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

NOTE L'IEC 62769-2 est techniquement identique à la FDI-2022.

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

NOTE L'IEC 62769-3 est techniquement identique à la FDI-2023.

IEC 62769-4:2015, *Intégration des appareils de terrain (FDI) – Partie 4: Paquetages FDI*

NOTE L'IEC 62769-4 est techniquement identique à la FDI-2024.

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

NOTE L'IEC 62769-5 est techniquement identique à la FDI-2025.

3 Termes, définitions, abréviations, acronymes 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.

3.1.1
passerelle
appareil de communication qui permet de relier différents réseaux physiques ou différents protocoles

3.2 Abréviations et acronymes

Pour les besoins du présent document, les abréviations et acronymes 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 la notation graphique

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

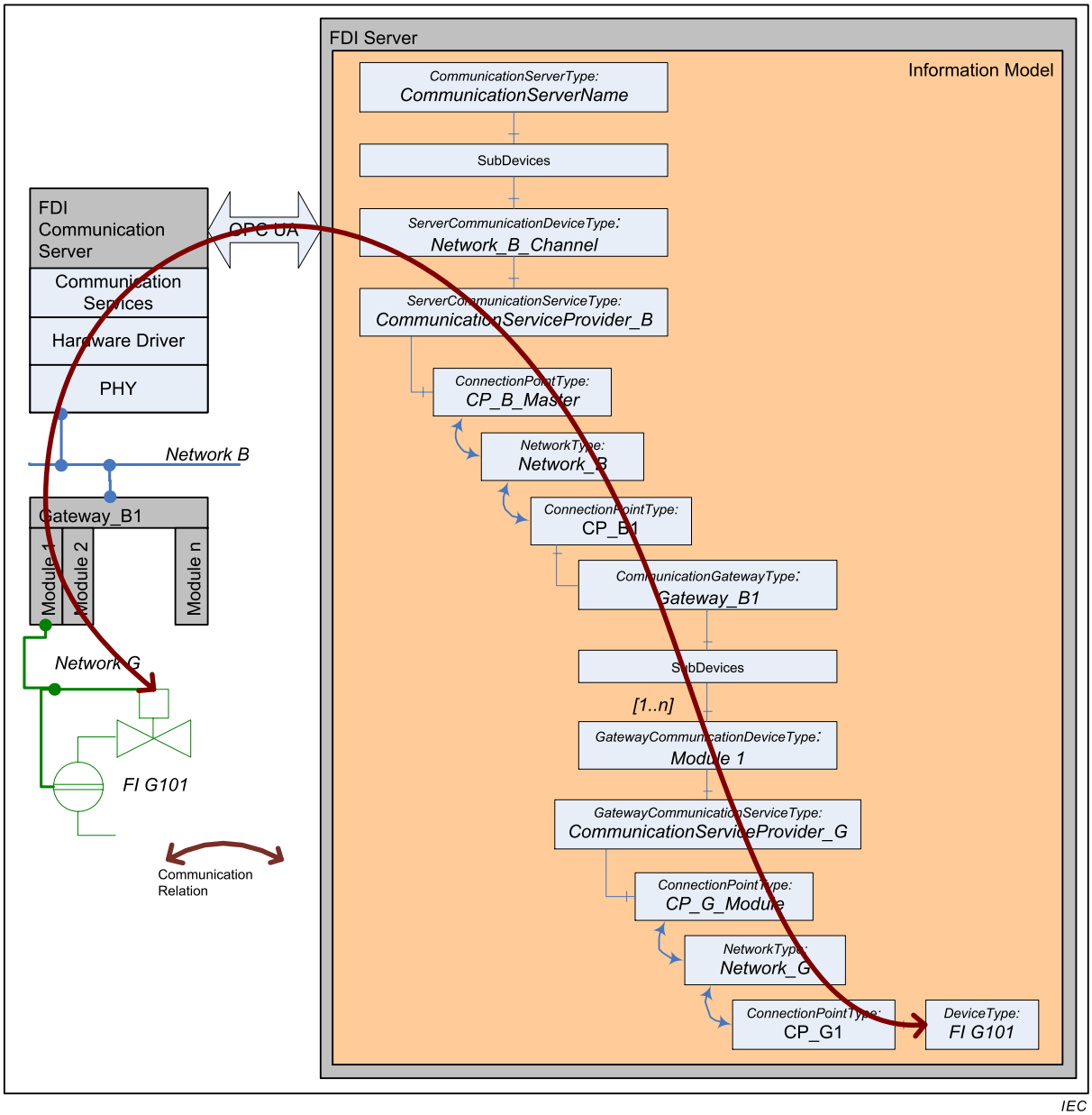
4 Généralités

Le terme abstrait Appareil de communication FDI représente une entité qui met en œuvre les fonctions de communication sur un réseau en utilisant un protocole spécifique. L'ensemble des Appareils de communication FDI se divise en deux groupes principaux.

a) 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'appareil de terrain. Le Serveur de communication FDI est spécifié à l'Article 7.

b) 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: 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'appareil 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 de réseau hétérogènes.



Anglais	Français
FDI Server	Serveur FDI
Information model	Modèle d'information
FDI Communication Server	Serveur de communication FDI
Communication Services	Services de communication
Hardware Driver	Pilote matériel
PHY	PHY
Network B	Réseau B
Gateway_B1	Passerelle_B1
Network G	Réseau G
Communication Relation	Relation de communication
FDI Server	Serveur FDI
Information Model	Modèle d'information

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 montrer comment les éléments utilisés du Modèle d'Information reflètent la topologie du réseau réelle.

a) 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 spécificités du Modèle d'Information, le contenu exigé pour le Paquetage de communication FDI et la manipulation des éléments qui s'y trouvent. Pour les sous-appareils, voir l'IEC 62769-5.

b) 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.

c) L'instance de CommunicationGatewayType (dénommée Gateway_B1) représente la Passerelle physique. L'Article 8 décrit les spécificités du Modèle d'Information, le contenu exigé pour le Paquetage FDI et la manipulation des éléments qui s'y trouvent.

d) 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 relatifs à la Passerelle sont décrits à l'Article 8.

e) L'instance de GatewayCommunicationServiceType (dénommée CommunicationServiceProvider_G) représente la capacité des Passerelles pour traiter les services de communication. La mise en œuvre spécifique de la Passerelle de GatewayCommunicationServiceType est basée sur la logique applicative qui permet d'exécuter les services de communication dans les réseaux de communication hétérogènes.

f) Une relation de communication (plus de détails sont donnés à l'Article 6) entre l'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 des protocoles de supports d'objets du service de communication multiples permet de traiter les connexions logiques multiples entre un maître bus et un appareil.

g) Le Modèle d'Information définit 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 détails 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 (voir l'IEC 62769-4:2015, Annexe F).

Les éléments EDD tels que spécifiés dans l'IEC 62769-4:2015, Annexe F et fournis avec un Paquetage de communication FDI doivent décrire:

a) Le paramètre et les structures de paramètre. Les définitions du paramètre spécifique du protocole obligatoire se trouvent dans l'IEC 62769-4:2015, Annexe F. Le paramètre doit contenir tout paramètre nécessitant un ajustement pour le fonctionnement approprié du service de communication.

b) L'identification de la couche physique. Les définitions spécifiques du protocole se trouvent dans l'IEC 62769-4:2015, Annexe F.

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

   FDI imagine la modularité d'un Appareil de communication susceptible de s'adapter au matériel de communication fournissant des canaux de communication physiques ou logiques multiples 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.

d) La définition du COMPONENT doit être utilisée pour prendre en charge la configuration de topologie mise en œuvre dans le système. Les définitions spécifiques du protocole se trouvent dans l'IEC 62769-4:2015, Annexe F. Les définitions relatives au COMPONENT sont décrites en 5.2.2, 5.2.3, 5.2.4 et 5.2.7.

e) La Logique applicative doit contenir une méthode permettant 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).

f) La Logique applicative peut contenir une méthode susceptible 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.

g) 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 et 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 du protocole se trouvent dans l'IEC 62769-4:2015, Annexe F.

h) MENU:
   La structure Menu doit suivre les conventions Menu pour les applications PC selon 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 des points d'entrée du menu "topology_configuration". Les définitions spécifiques du protocole se trouvent dans l'IEC 62769-4:2015, Annexe F.

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 mettant en œuvre la fonction de validation de la configuration de topologie du module. Les détails de mise en œuvre sont spécifiés 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 détails de syntaxe sont décrits 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 décrivant 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 mis sur COMPONENT_CHILD.

<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 du 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. Cela dépend si un CommunicationDevice nécessite une configuration explicite ou si l'objet du fournisseur de service de communication concerné doit être automatiquement instancié avec le CommunicationDevice. Si l'attribut CAN_DELETE est mis 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 service de communication. La construction permet au Serveur FDI d'instancier les composants du fournisseur de service de communication conformément aux demandes de traitement de la communication. Les détails de syntaxe sont décrits 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 service de communication (voir 5.2.4).

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

Le RELATION_TYPE doit être mis sur COMPONENT_CHILD.

L'attribut PROTOCOL ne doit pas être défini.

5.2.4 Composant fournisseur de service de communication

Chaque Paquetage de communication FDI décrivant un Appareil de communication doit contenir au moins un élément EDD qui décrit le fournisseur de service 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 service 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 service de communication.

<CanDelete>: Les valeurs permises sont TRUE ou FALSE. Cela dépend de ce que le fournisseur de service de communication peut être instancié de façon souple conformément aux demandes de traitement de la communication ou non. Si l'attribut CAN_DELETE est mis sur FALSE, le nombre d'instanciations du composant du fournisseur de service de communication est statique. Les contraintes d'instanciation déclarées par les attributs AUTO_CREATE, MINIMUM_NUMBER et MAXIMUM_NUMBER correspondent aux fonctions 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 détails de syntaxe sont décrits 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 service de communication et il doit y avoir exactement une (1) instance de Point de connexion connectée au fournisseur de service de communication. Les contraintes d'instanciation déclarées par les attributs AUTO_CREATE, MINIMUM_NUMBER et MAXIMUM_NUMBER correspondent aux fonctions 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 mis sur COMPONENT_CHILD.

L'attribut PROTOCOL ne doit pas être défini.

5.2.5 Composant Point de connexion

Chaque Paquetage de communication FDI décrivant un Appareil de communication doit contenir au moins un élément EDD qui décrit un Point de connexion pour chaque protocole 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 fonction de communication consistant à permettre au Serveur FDI de trouver d'autres types d'appareils qui peuvent être connectés au réseau en utilisant le 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 décrivant 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 décrivant 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 désignant 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 décrivant un Appareil de communication doit contenir au moins un élément EDD qui décrit un réseau pour chaque protocole 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 mettant 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 en utilisant 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, des fonctions d'un réseau. La construction permet d'effectuer la configuration de topologie réseau actionnée par le Serveur FDI. Les détails de syntaxe sont décrits 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 mis sur COMPONENT_CHILD.

<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 applicative mise en œuvre dans l'Appareil de communication qui valide une topologie réseau réelle. La méthode ValidateNetwork manipule toutes les dépendances nécessaires relatives aux paramètres du bus. La mise en œuvre de la logique EDDL concernée est basée sur la fonction ObjectReference intégrée à l'EDDL, qui permet d'analyser 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 réussi la validation.

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

Le Tableau 1 répertorie 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: Echec/le Point de connexion qui n'a pas réussi la validation est indiqué par la valeur () d'attribut <VALID> mise 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	Si 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 basée 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 que les règles de configuration définies dans la construction COMPONENT sont suffisantes pour configurer l'installation du module.

NOTE La décision de savoir si ValidateModules est nécessaire ou non incombe au vendeur.

Le Tableau 2 répertorie 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: Echec/le Point de connexion qui n'a pas réussi la validation est indiqué par la valeur () d'attribut <VALID> mise 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	Si 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 Eléments spécifiques de l'UIP

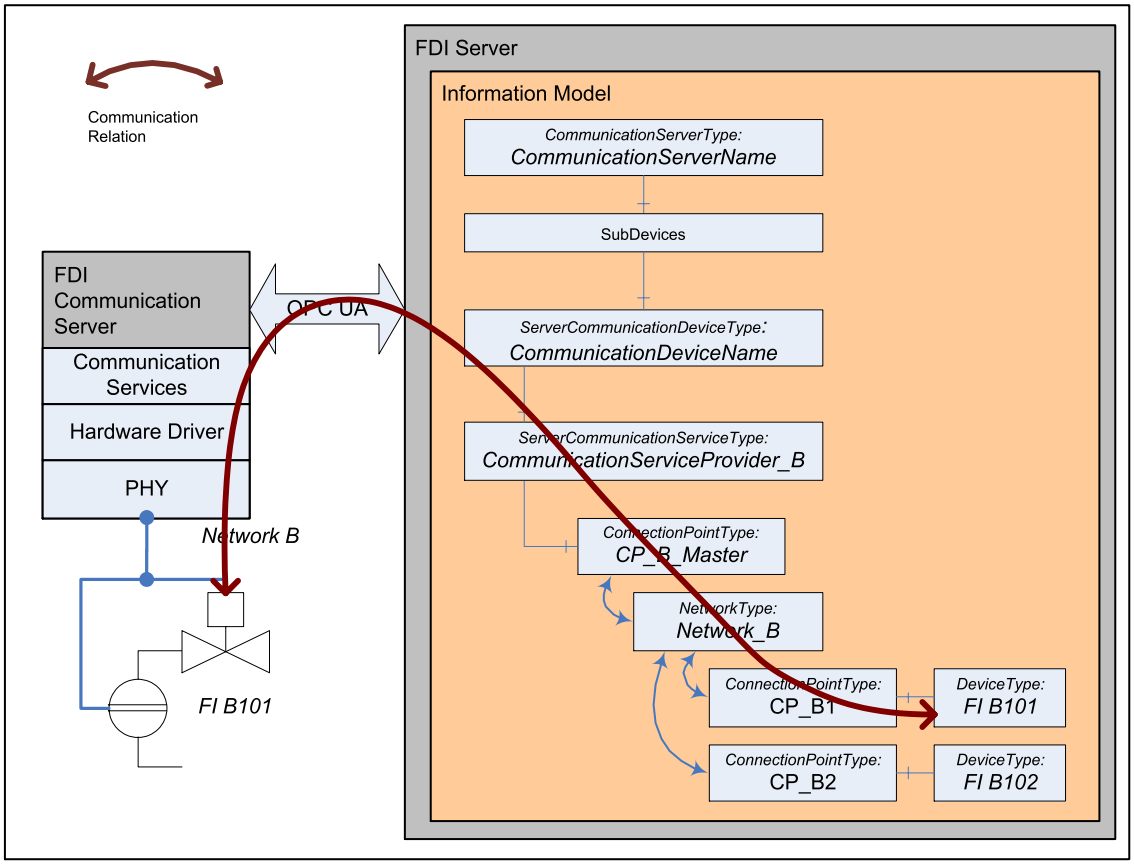
Le Paquetage de communication FDI peut contenir l'UIP pour prendre en charge les diagnostics et la paramétrisation par exemple.

5.2.11 Déploiement

Le Serveur FDI importe le Paquetage de communication FDI. La manipulation d'éléments EDD et UIP correspond à la procédure d'importation effectuée 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 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.



Anglais	Français
Communication Relation	Relation de communication
FDI Communication Server	Serveur de communication FDI
Communication Services	Services de communication
Hardware Driver	Pilote matériel
PHY	PHY
Network B	Réseau B
OPC UA	OPC UA
FDI Server	Serveur FDI
Information Model	Modèle d'information

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 décrit le flux d'état général d'une relation de communication unique. Le diagramme montre 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 serviceErrors retournées par les services de communication spécifiés. Même la méthode d'analyse peut déterminer une perte de connexion quand l'appareil pour lequel une relation de communication a été activée n'apparaît pas dans le résultat d'analyse.

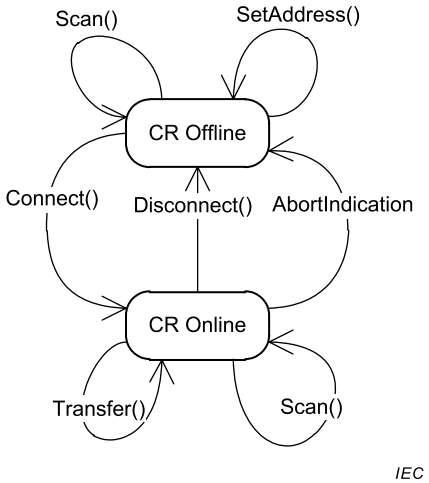


Figure 4 – Diagramme états-transitions de la relation de communication

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'appareil de terrain. Chaque Serveur de communication FDI est conçu comme un appareil modulaire où 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 CommunicationDevices 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 CommunicationDevices 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 basée 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 d'interverrouillage pour gérer l'accès concomitant à un appareil connecté à un réseau physique unique.

7.3 Modèle d'Information

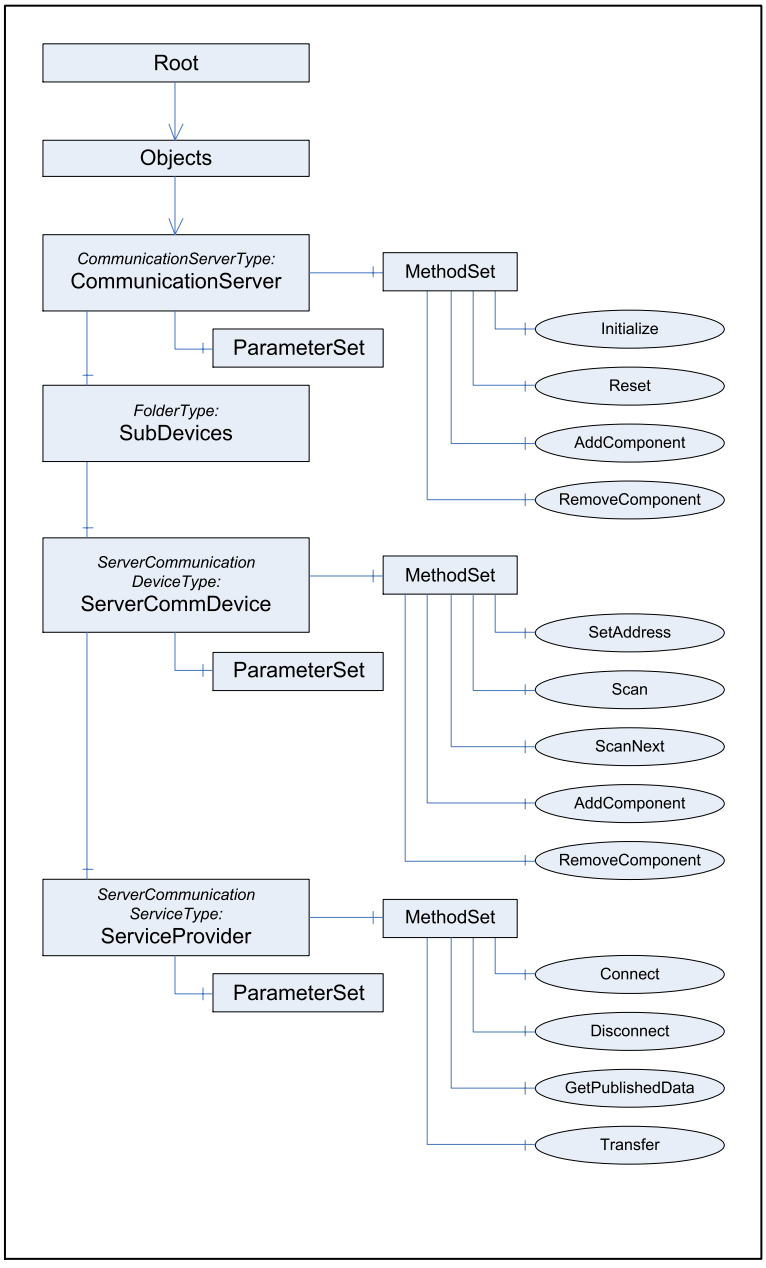
7.3.1 Généralités

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 de communication normalisée. 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 l'entremise 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. Une interaction avec les appareils FDI a lieu par le truchement 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 comme 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. Le Paquetage FDI est utilisé pour créer des Appareils de communication dans le Modèle d'Information du Serveur FDI qui représentent l'accès au réseau mis en œuvre par le Serveur de communication FDI.

Le Modèle d'Information d'un Serveur de communication FDI est basé sur le Modèle d'Information défini dans l'IEC 62769-5. La Figure 5 reproduit la structure de l'Appareil modulaire et montre comment elle est mappée dans l'AddressSpace (espace adresse) complet. Les modules représentent les canaux de communication du Serveur de communication FDI.



Anglais	Français
Root	Racine
Objects	Objets

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 si 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. L'instance de ServerCommunicationDeviceType (ServerCommDevice) a un MethodSet qui peut mettre en œuvre les méthodes SetAddress, Scan, AddComponent, RemoveComponent. AddComponent et RemoveComponent sont facultativement présents si le Serveur de communication FDI prend en charge un numéro 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.

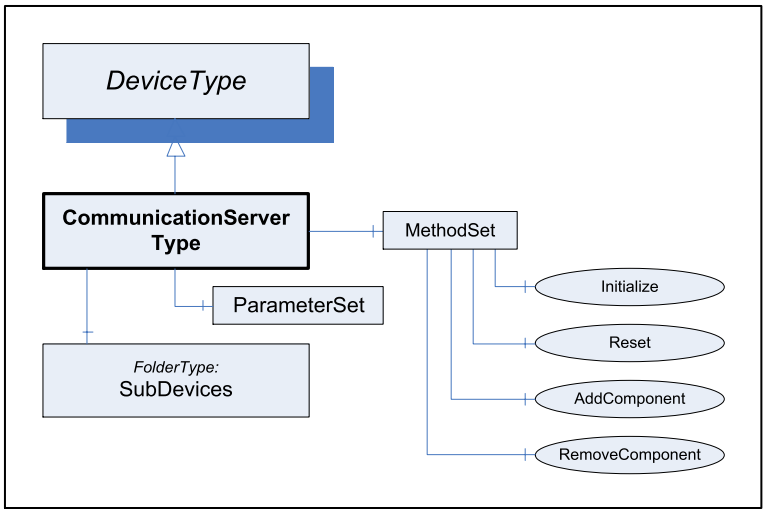


Figure 6 – CommunicationServerType

Tableau 3 – Définition de CommunicationServerType

Attribut	Valeur				
BrowseName	CommunicationServerType				
IsAbstract	False				
Références	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Sous-type de DeviceType défini dans l'IEC 62541-100.					
HasComponent	Object	MethodSet		BaseObjectType	Obligatoire
HasComponent	Object	ParameterSet		BaseObjectType	Facultatif
HasComponent	Object	SubDevices		FolderType	Obligatoire

Tableau 4 – MethodSet de CommunicationServerType

Attribut	Valeur				
BrowseName	MethodSet				
IsAbstract	False				
Références	**NodeClass**	**BrowseName**	**Data Type**	**TypeDefinition**	**ModellingRule**
HasComponent	Method	Initialize			Obligatoire
HasComponent	Method	Reset			Obligatoire
HasComponent	Method	AddComponent			Facultatif
HasComponent	Method	RemoveComponent			Facultatif

Le CommunicationServerType et chaque instance de ce type partagent les mêmes méthodes. Le NodeId de ces Méthodes sera fixé et défini dans la présente norme. Les clients du Serveur de communication FDI ne doivent donc pas explorer pour chercher ces Méthodes. Ils peuvent utiliser le NodeId fixe en tant que MethodId du service d'appel.

Les méthodes supplémentaires AddComponent et RemoveComponent ajoutent la capacité d'ajouter ou de supprimer les instances de ServerCommunicationDeviceType conformément à la structure du matériel de communication. Les services ne sont pas applicables si 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 mis 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 sera arrêtée immédiatement. Tous les canaux 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 manipuler la procédure d'arrêt automatiquement conformément aux demandes de communication.

Généralement, le fonctionnement du Serveur de communication FDI inclut la manipulation du matériel et du pilote du protocole qui peut être indépendante 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: Echec

Tableau 6 – Définition de l'AddressSpace de la méthode Reset

Attribut	Valeur				
BrowseName	Reset				
Références	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Obligatoire

7.3.2.3 Méthode Initialize

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

Afin de permettre les modifications de paramètre pendant le fonctionnement, la méthode Initialize peut être réinvoquée. Si le Serveur de communication FDI doit 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 changements de configuration ont été détectés.

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 manipuler la procédure de démarrage automatiquement conformément aux demandes de communication humaines.

Le fonctionnement du Serveur de communication FDI peut inclure la manipulation du matériel et du pilote du protocole qui peut être indépendante 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: Echec

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	Obligatoire

7.3.2.4 Méthode AddComponent

La méthode AddComponent doit être utilisée pour configurer l'installation modulaire d'un Serveur de communication FDI au cas où le Serveur de communication FDI n'aurait pas d'installation du matériel de communication définie statiquement. Cette méthode ne doit pas 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	Etiquette en langage humain pour le nœud de 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 a été spécifié -4: E_LimitExceeded – le nombre total de modules a été dépassé (cela pourrait être causé par les 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**	**DataType**	**TypeDefinition**	**ModellingRule**
HasProperty	Variable	InputArguments	Argument[]	PropertyType	Obligatoire
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Obligatoire

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 si 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: Echec, 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	Obligatoire
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Obligatoire

7.3.3 ServerCommunicationDeviceType

7.3.3.1 Généralités

Le ServerCommunicationDeviceType représente un canal de communication pour un réseau particulier. Le ServerCommunicationDeviceType est un sous-type du DeviceType. Le ParameterSet de chaque instance d'un ServerCommunicationDevice contiendra les paramètres nécessaires pour configurer le fonctionnement du réseau. Les paramètres de bus obligatoires spécifiques au protocole sont spécifiés dans l'IEC 62769-4:2015, Annexe F. La Figure 7 représente la définition de ServerCommunicationDeviceType qui est formellement décrite dans le Tableau 13 et la Figure 14.

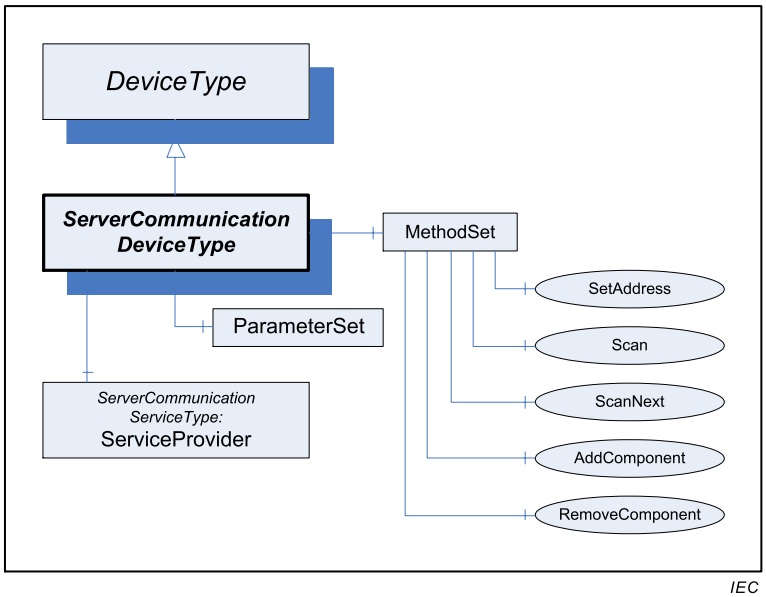


Figure 7 – ServerCommunicationDeviceType

Tableau 13 – Définition de ServerCommunicationDeviceType

Attribut	Valeur				
BrowseName	ServerCommunicationDeviceType				
IsAbstract	True				
Références	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Sous-type de DeviceType défini en OPC UA Partie DI.					
HasComponent	Object	MethodSet		BaseObjectType	Facultatif
HasComponent	Object	ParameterSet		BaseObjectType	Facultatif
HasComponent	Object	ServiceProvider		ServerCommunication ServiceType	Obligatoire

Tableau 14– MethodSet de ServerCommunicationDeviceType

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

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 retournera des résultats de balayage réseau mis en cache.

NOTE 2 Dans l'éventualité où l'opération SCAN prendrait trop de temps, le Serveur de communication FDI pourrait retourner une valeur TopologyScanResult vide et une ServiceError 1 indiquant 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 représentant une liste d'appareils connectés au réseau physique. Chaque appareil connecté 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 62769-4:2015, 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 en rappelant Scan -1: Echec/non initialisé -2: Echec/non connecté à un réseau -3: Echec/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	Obligatoire

7.3.3.3 Méthode ResetScan

La méthode ResetScan doit être utilisée pour réinitialiser le cache interne des résultats de balayage. Le balayage sera également annulé si 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: Echec/non initialisé -2: Echec/non connecté à un réseau

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

Attribut	Valeur				
BrowseName	ResetScan				
Références	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Obligatoire

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. Si 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: Echec/erreur de l'adresse en double".

Cela dépend du protocole 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 SetAdress sont décrits dans le Tableau 19.

Signature

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

Tableau 19 – Arguments de la méthode SetAddress

Argument	Description
OldAddress	L'argument représente 1..n valeurs spécifiques au protocole représentant 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 62769-4:2015, Annexe F.
NewAddress	L'argument représente 1..n arguments spécifiques au protocole représentant la nouvelle adresse réseau existante spécifique au protocole qui doit être assignée à l'appareil connecté au réseau physique. Les valeurs qui représentent les arguments d'une adresse réseau sont spécifiées dans l'IEC 62769-4:2015, Annexe F.
ServiceError	0: OK/exécution terminée avec succès -1: Echec de SetAddress/non initialisé -2: Echec de SetAddress/non connecté à un réseau -3: Echec de SetAddress/aucun appareil correspondant à oldAddress n'a été trouvé -4: Echec de SetAddress/erreur d'adresse en double -5: Echec de SetAddress/l'appareil n'a pas accepté la nouvelle adresse -6: Echec de SetAddress/oldAddress non valide (en termes de syntaxe, de type de données, de format des données, etc.) -7: Echec de SetAddress/newAddress non valide (en termes de syntaxe, de type de données, de format des données, etc.) -8: Echec de SetAddress/impossible à l'état connecté

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 sera fixé et défini dans la présente norme. Les Clients FDI ne doivent donc pas explorer pour chercher ces Méthodes. Ils peuvent utiliser le NodeId fixe en tant que MethodId du service d'appel.

Les méthodes de communication contenant leurs NodeIds sont définies de manière unique dans la présente norme. 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.

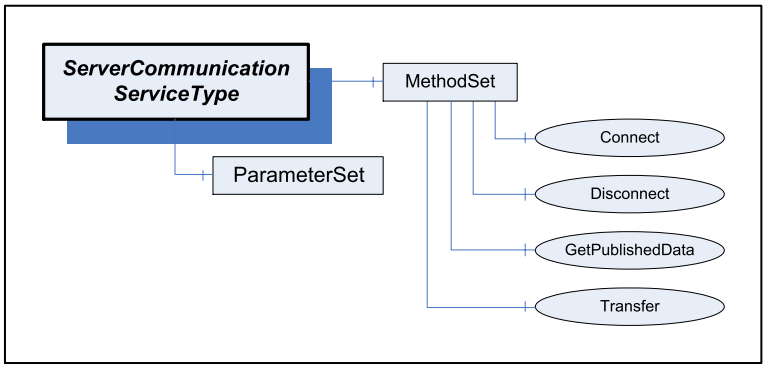


Figure 8 – ServerCommunicationServiceType

Tableau 20 – Définition de ServerCommunicationServiceType

Attribut	Valeur				
BrowseName	ServerCommunicationServiceType				
IsAbstract					
Références	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Sous-type de DeviceType défini en OPC UA Partie DI.					
HasComponent	Object	MethodSet		BaseObjectType	Obligatoire
HasComponent	Object	ParameterSet		BaseObjectType	Facultatif

Tableau 21 – MethodSet de ServerCommunicationServiceType

Attribut	Valeur				
BrowseName	MethodSet				
IsAbstract					
Références	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
HasComponent	Method	Connect			Facultatif
HasComponent	Method	Disconnect			Facultatif
HasComponent	Method	Transfer			Obligatoire
HasComponent	Method	GetPublishedData			Facultatif

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. Etablir 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 la norme du protocole réseau. Les détails pertinents sont spécifiés dans l'IEC 62769-4:2015, 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 le Modèle d'Information associé à l'application de l'appareil et l'appareil physique est identifiée avec de plus amples détails par l'identificateur de la relation de communication. Les détails sur la manière de gérer l'état d'une relation de communication sont décrits à l'Article 6.

NOTE 1 Le NodeId étant 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 la portée d'un Appareil de communication.

NOTE 2 Le terme relation de communication est introduit pour décrire l'état d'une infrastructure qui permet l'échange des données entre les données hébergées par le Modèle d'Information et un appareil physique. Si 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]   BaseDataType[]   AddressData,
   [out]  BaseDataType[]   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 pourrait s'agir d'un index (par exemple, un NodeId) dont le client (= Serveur FDI) doit identifier les entrées dans sa topologie.
AddressData	Liste d'arguments spécifique au protocole utilisée pour l'adresse et les données d'identification facultatives de l'appareil (les détails sont décrits dans l'IEC 62769-4:2015, Annexe F).
DeviceInformation	Liste d'arguments spécifique au protocole dans laquelle sont stockées les données de résultat de la méthode Connect.
ServiceError	0: OK/exécution terminée, connexion établie avec succès -1: Echec de la connexion/appareil introuvable -2: Echec de la connexion/adresse d'appareil non valide -3: Echec de la connexion/identification d'appareil non valide

7.3.4.3 Méthode Disconnect

La méthode Disconnect doit être utilisée pour déterminer la 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 -1: Echec de la déconnexion/aucune relation de communication existante -2: Echec de la déconnexion/identificateur de relation de communication non valide

7.3.4.4 Méthode Transfer

La méthode Transfer doit être utilisée pour effectuer l'échange 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]  BaseDataType[]   SendData,
   [out] BaseDataType[]   ReceiveData,
   [out] Int32            ServiceError);
```

Tableau 24 – Arguments de la méthode Transfer

Argument	Description
CommunicationRelationId	Voir 7.3.4.2.
SendData	Liste de valeurs spécifiques au protocole telles que décrites dans l'IEC 62769-4:2015, Annexe F. Les valeurs d'argument représentent la demande de services de communication spécifique au protocole envoyée à l'appareil.
ReceiveData	Liste de valeurs spécifiques au protocole telles que décrites dans l'IEC 62769-4:2015, 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 -1: Echec du transfert/aucune relation de communication existante -2: Echec du transfert/identificateur de relation de communication non valide -3: Echec du transfert/contenu sendData non valide -4: Echec du transfert/format receiveData non valide

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 des données non sollicitées. L'argument SendData contenu dans les données prépare l'échange des messages des données "non sollicitées" depuis 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 sollicités, 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 (performance), la méthode GetPublishedData peut immédiatement être retournée avec les données provenant d'un message de données "non sollicité".

L'extraction suivante des données envoyées par les messages de données non sollicités 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 des messages de données "non sollicitées".

Afin d'arrêter l'appareil envoyant les données "non sollicitées", la méthode GetPublishedData doit être utilisée à nouveau, mais l'argument SendData contenu dans les données met fin à l'échange des messages de données "non sollicitées" depuis l'appareil. Le Tableau 25 répertorie les arguments de la méthode GetPublishedData.

Signature

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

Tableau 25 – Arguments de la méthode GetPublishedData

Argument	Description
CommunicationRelationId	Voir 7.3.4.2.
SendData	Liste de valeurs spécifiques au protocole, telles que décrites dans l'IEC 62769-4:2015, Annexe F. Les valeurs d'argument contrôlent l'échange de messages non sollicités.
ReceiveData	Liste de valeurs spécifiques au protocole, telles que décrites dans l'IEC 62769-4:2015, Annexe F. Les valeurs d'argument transmettent les données en provenance des messages non sollicités.
TimeStamp	Temps auquel les données ont été publiées par l'appareil.
PublishId	Le numéro identifie un abonnement qui transmet les données provenant des messages non sollicités.
ServiceError	0: OK/exécution terminée -2: Echec de l'appel/PublishId inconnu -3: Echec de GetPublishedData/non pris en charge -4: Echec de GetPublishedData/aucune relation de communication existante -5: Echec de GetPublishedData/identificateur de relation de communication non valide -6: Echec de GetPublishedData/contenu sendData non valide -7: Echec de GetPublishedData/format receiveData non valide -8: Echec de GetPublishedData/aucune donnée publiée ne correspond à l'argument SendData

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 62541-7. La présence du terme Facet dans l'intitulé d'un profil signifie que ce profil est censé faire partie d'un autre Profil plus vaste ou qu'il gère un aspect spécifique de l'OPC UA. Les profils comportant le mot Facet dans leur intitulé sont censés 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 "Micro Embedded Device Server Profile".

Le tableau suivant spécifie la facette d'un Serveur OPC UA jouant 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 CommunicationServerType.	M

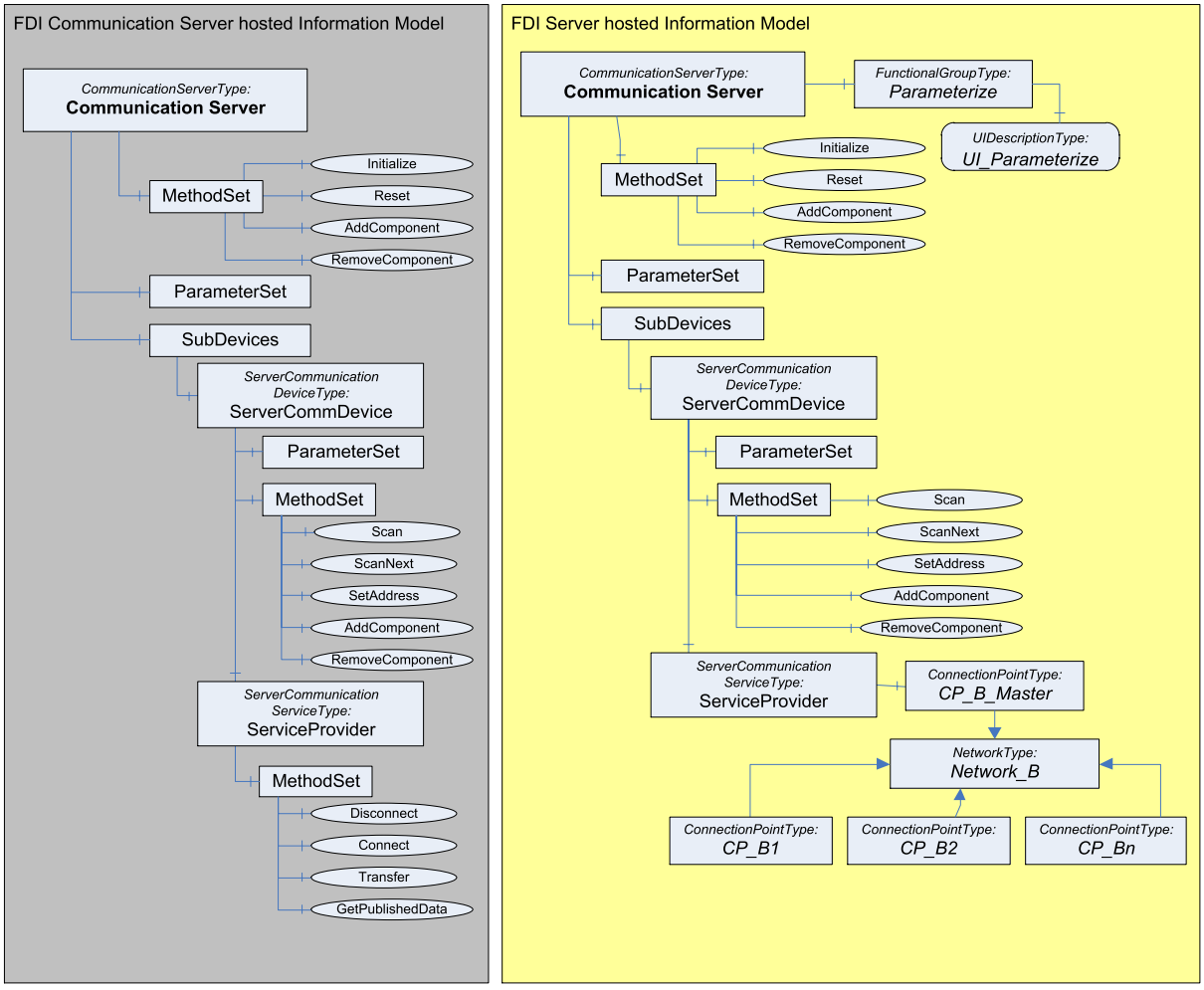
7.5 Mapping de l'IM du Serveur FDI à l'IM du Serveur de communication FDI

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 l'entremise 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 des éléments du Modèle d'Information communs doivent avoir le même nom d'exploration.



Anglais	Français
FDI Communication Server hosted Information Model	Modèle d'information hébergé par le Serveur de communication FDI
FDI Server hosted Information Model	Modèle d'information hébergé par le Serveur FDI
Communication Server	Serveur de communication

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 passent au Serveur de communication FDI. Cela 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. Cela 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.

Le mapping de la fonctionnalité de la gestion du module est comme suit:

- AddComponent et RemoveComponent sont exposés dans le Modèle d'Information du Serveur de communication FDI par l'entremise de l'objet du Serveur de communication FDI (la racine de l'appareil modulaire). Ils n'existent que s'il y a des modules à configurer, ce qui veut dire qu'ils ne seront pas disponibles si le Serveur de communication ne prend pas en charge la configuration du matériel de communication modulaire. AddComponent et RemoveComponent remplacent le service de gestion du nœud générique défini par OPC UA.
- Le Serveur FDI manipule la configuration relative à la topologie du module en fonction de l'ensemble des services de gestion du nœud (voir l'IEC 62769-3 et l'IEC 61804-4).
- Pour toute activité relative à la configuration du module, le Serveur FDI appelle d'abord l'action ValidateModules. L'action EDD peut s'exécuter par différents états, voire effectuer les dialogues de l'utilisateur (voir la description des actions dans l'IEC 62769-2 et l'IEC 62769-5). Les méthodes EDD peuvent conserver les informations (privées) qui sont globales pour l'appareil modulaire. L'action EDD peut accéder au module qui doit être créé.

7.6 Programme d'installation

Le programme d'installation de l'exécutable du Serveur de communication FDI est facultatif. Puisque la spécification FDI ne prescrit pas la mise en œuvre d'une plate-forme pour les programmes exécutables du Serveur de communication FDI, le programme exécutable du Serveur de communication FDI peut également être préinstallé sur le matériel dédié.

Le programme d'installation utilisé pour l'exécutable du Serveur de communication FDI doit être distinct du Paquetage FDI. L'importation du Paquetage FDI est une procédure distincte (voir 7.7.1).

7.7 Paquetage de communication FDI

7.7.1 Généralités

Le Serveur FDI importe le Paquetage de communication FDI comme tout autre Paquetage FDI. Le Paragraphe 7.7 spécifie les détails du Paquetage de communication FDI.

En fonction de l'élément EDD du Paquetage, FDI distingue un Serveur de communication simple (léger) (voir 7.7.2) et un Serveur de communication régulier (multicanaux) (voir 7.7.3).

7.7.2 EDD pour le Serveur de communication léger

Un Serveur de communication léger donne accès à un seul réseau d'appareil de terrain. Il doit indiquer toutes les fonctions de configuration dans son élément EDD principal, non dans les sous-modules utilisés pour exposer les Points de connexion. Cela permet aux Hôtes FDI ne prenant pas en charge les appareils modulaires de paramétrer un Serveur de communication FDI en utilisant les mécanismes d'interface d'utilisateur FDI normalisés.

L' élément EDD décrivant un Serveur de communication "léger" doit suivre le profil spécifié par l'IEC 61804-3 pour le Serveur de communication. Mais il ne doit pas utiliser les constructions de la syntaxe EDDL suivantes:

- COMPONENT
- COMPONENT_RELATION
- COMPONENT_FOLDER
- COMPONENT_REFERENCE
- Fonction ObjectReference intégrée dans l'EDDL

NOTE COMPONENT définit un attribut PRODUCT_URI qui peut être utilisé pour la découverte automatique du CommunicationServer correspondant. Puisque nous n'utilisons pas la construction COMPONENT pour les CommunicationServers légers, les systèmes doivent fournir la découverte manuelle.

7.7.3 EDD pour le Serveur de communication multicanaux

7.7.3.1 Généralités

Le contenu exigé pour l'élément EDD d'un Paquetage de communication FDI décrivant un Appareil de communication FDI est spécifié à l'Article 5. Le contenu de l'élément EDD spécifique pour un Serveur de communication FDI est décrit en 7.7.3.

Les règles définies en 5.2.2 s'appliquent.

L'attribut PROTOCOL ne doit pas être défini.

La déclaration COMPONENT doit avoir un attribut supplémentaire PRODUCT_URI. La valeur d'attribut contient une chaîne décrivant le produit URI du Serveur de communication FDI qui permet au Serveur FDI d'identifier le Serveur de communication FDI basé sur le service de découverte OPC UA (voir l'IEC 61804-3). La valeur d'attribut correspond à l'argument RegisterServer RegisteredServer:serverUri. Le produit URI doit contenir le nom de la compagnie et le nom du produit.

Exemple: PRODUCT_URI "urn:Company:ProductName".

7.7.3.2 Composant CommunicationDevice

Les règles définies en 5.2.3 s'appliquent.

7.7.3.3 Composant du service de communication

Les règles définies en 5.2.4 s'appliquent.

7.7.3.4 Composant Point de connexion

Les règles définies en 5.2.5 s'appliquent.

7.7.3.5 Collection du Point de connexion

Les règles définies en 5.2.6 s'appliquent.

7.7.4 Documentation

Le Paquetage de communication FDI doit fournir la documentation décrivant:

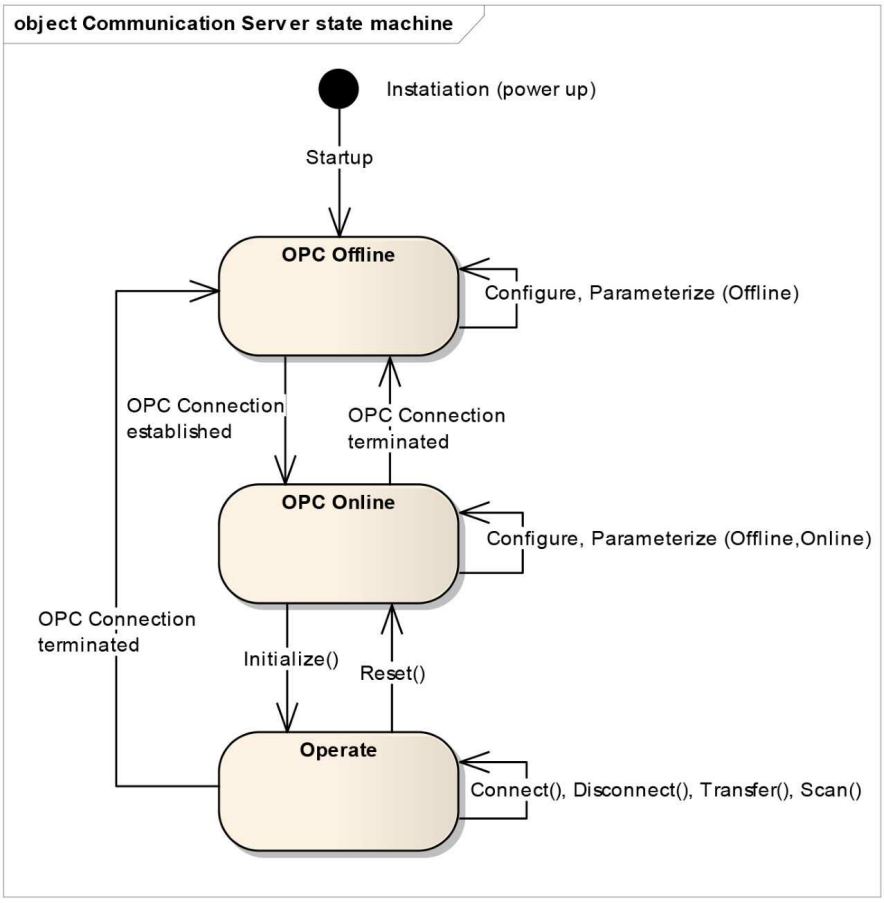
i) les exigences d'environnement et les procédures relatives à l'installation du logiciel;

j) la configuration du Serveur OPC UA, si nécessaire.

7.8 Manipulation et comportement

7.8.1 Généralités

Le Paragraphe 7.8 définit les règles de manipulation et de comportement du Serveur de communication FDI tout au long du cycle de vie (déploiement, démarrage, mise en service du bus, traitement des services de communication). Le diagramme (voir Figure 10) affiche l'état du Serveur de communication FDI géré par le Serveur FDI.



Anglais	Français
Object Communication Server state machine	Diagramme d'états du serveur de communication
Instatiation (power up)	Instanciation (sous tension)
Startup	Démarrage
OPC Offline	OPC hors ligne
Configure, Parameterize (Offline)	Configurer, paramétrer (hors ligne)
OPC Connection established	Connexion OPC établie
OPC Connection terminated	Connexion OPC interrompue
OPC Online	OPC en ligne
Configure, Parameterize (Offline,Online)	Configurer, paramétrer (hors ligne, en ligne)

Anglais	Français
Initialize	Initialiser
Reset	Réinitialiser
Operate	Actionner
Connect(), Disconnect(), Transfer(), Scan()	Connect(), Disconnect(), Transfer(), Scan()

Figure 10 – Diagramme d'états du Serveur de communication FDI

7.8.2 Déploiement

Le Serveur FDI importe le Paquetage de communication FDI. La manipulation d'éléments EDD et UIP correspond à la procédure d'importation effectuée pour le Paquetage FDI (voir l'IEC 62769-2 et l'IEC 62769-3). Le Paquetage de communication FDI représente le type du Serveur de communication.

La procédure d'installation décrite ci-dessous est facultative. (Un Serveur de communication FDI intégré peut ne pas fournir une procédure d'installation.)

Le programme d'installation du Serveur de communication FDI (voir □) est un élément distinct. La procédure d'installation est démarrée manuellement.

Selon les systèmes d'exploitation, l'exécution des programmes d'installation pourrait nécessiter des droits d'administrateur.

7.8.3 Configuration du Serveur

Le Serveur de communication FDI doit mettre en œuvre les moyens permettant la configuration spécifique du Serveur OPC UA qui établit le lien au Serveur de découverte et le nom du Serveur de communication FDI. Conformément à 7.8.4, le Serveur de communication FDI doit connaître les informations d'adresse du Serveur de découverte. La norme ne prescrit pas la manière de le faire.

Le processus d'amorce nécessaire pour établir la connexion entre un Client OPC UA et un Serveur OPC UA nécessite un certain travail d'administration. Un simple processus prêt à l'emploi (plug and play) ne semble pas possible.

7.8.4 Démarrage

Les définitions suivantes concernant le mécanisme de découverte du Serveur se rapportent aux définitions qui se trouvent dans l'IEC 62541-4.

Le programme exécutable du Serveur de communication FDI doit être lancé conformément à l'une des méthodes suivantes:

a) Le programme exécutable du Serveur de communication FDI est chargé au moyen de la fonction du système d'exploitation configuré. Si le Serveur de communication FDI est installé sur un matériel distinct du Serveur FDI, le programme exécutable du Serveur de communication FDI doit être chargé au moyen de la fonction du système d'exploitation configuré (autodémarrage).
b) Le Serveur FDI invoque le processus du programme exécutable du Serveur de communication FDI. Les fonctions concernées sont spécifiques au système d'exploitation et aux mises en œuvre du vendeur du système.

Le processus de démarrage du Serveur de communication FDI doit s'enregistrer lui-même dans un Serveur de découverte en utilisant le service RegisterService. Cela permet aux Clients OPC UA d'obtenir les informations sur le Serveur de communication FDI concerné, y compris la description de l'application, les points d'extrémité existants et les informations de

sécurité. Les services OPC UA concernés sont FindServers et GetEndPoints. Le Serveur de découverte est un processus qui s'exécute à l'extérieur du Serveur de communication FDI.

Après le démarrage, le Serveur de communication FDI a le statut "OPC Offline".

7.8.5 Arrêt

Les définitions suivantes concernant le mécanisme de découverte du Serveur se rapportent aux définitions qui se trouvent dans l'IEC 62541-4.

L'arrêt du processus du Serveur de communication FDI doit se désenregistrer lui-même du Serveur de découverte en utilisant le service RegisterService et en mettant la valeur de l'argument isOnline sur FALSE.

7.8.6 Surveillance

Les définitions suivantes concernant le mécanisme de découverte du Serveur se rapportent aux définitions qui se trouvent dans l'IEC 62541-4.

Le Serveur de communication FDI doit périodiquement utiliser le service RegisterServer pour indiquer sa capacité à recevoir une connexion du Serveur FDI. La fréquence est de 10 min. Le Serveur de communication FDI peut prévoir une VARIABLE aux fins de la configuration.

7.8.7 Etablissement d'une connexion OPC UA

Le Serveur FDI se connecte comme un Client OPC UA au Serveur de communication FDI conformément à l'IEC 62541.

Le Serveur FDI (Client OPC UA) établit une connexion sécurisée au Serveur de communication FDI (Serveur OPC UA) en utilisant l'ensemble de services SecureChannel défini dans l'IEC TR 62541-1. Les principes fonctionnels sont définis dans l'IEC TR 62541-1.

La communication entre le Serveur FDI (client OPC UA) et le Serveur de communication FDI (Serveur OPC UA) doit être basée sur le protocole de transport OPC UA TCP, avec l'encodage binaire OPC UA et la conversation sécurisée OPC UA.

Une fois la connexion OPC UA établie avec succès, le Serveur de communication FDI acquiert le statut "OPC Online".

7.8.8 Instanciation du Serveur de communication

La création de l'instance CommunicationServer dans le Modèle d'Information hébergé dans le Serveur FDI fonctionne de la même manière que l'instanciation d'un appareil.

7.8.9 Configuration du matériel de communication

Comme décrit en 7.3.1, le Serveur de communication FDI peut prendre en charge la configuration du matériel de communication modulaire. La configuration du matériel de communication modulaire doit être effectuée par le truchement des services AddComponent et RemoveComponent (voir 7.3.2.4 et 7.3.2.5).

Si l'action ValidateModules (voir 5.2.9) est mise en œuvre, le Serveur FDI doit invoquer cette méthode. Le résultat de l'action est OK, puis le Serveur FDI doit effectuer la synchronisation en utilisant le nom d'exploration du sous-appareil correspondant et l'invocation des services de gestion du module hébergés par le Serveur de communication FDI.

Si le Serveur de communication FDI prend en charge la configuration du matériel de communication modulaire, la configuration du matériel de communication correct est une condition préalable pour l'initialisation réussie, comme décrit en 7.8.12.

7.8.10 Configuration du réseau

La déclaration du COMPONENT définie en 5.2.2, 5.2.3, 5.2.4 et 5.2.7 permet une approche basée sur la description pour le Serveur FDI en ce qui concerne la configuration des connexions réseau. L'action ValidateNetwork du Serveur de communication FDI (voir 5.2.8) peut être ajoutée pour prendre en charge la fonction de validation de topologie du réseau, comme décrit en 5.2.8. La topologie du réseau est présente uniquement dans le Modèle d'Information hébergé par le Serveur FDI (voir 7.5).

7.8.11 Paramétrisation

Le Serveur de communication FDI peut nécessiter le réglage des paramètres de bus appropriés avant tout traitement du service de communication. Les dialogues d'utilisateur (UIP ou UID) contenus dans le Paquetage de communication FDI permettent un réglage interactif des paramètres de bus. Le Paquetage de communication FDI peut contenir une logique applicative supplémentaire aux fins de paramétrisation du bus. La modification des paramètres du Serveur de communication FDI change le contenu du Modèle d'Information hébergé par le Serveur FDI. Le Serveur FDI doit effectuer la synchronisation en utilisant le nom d'exploration correspondant des paramètres. Le Serveur FDI copie les valeurs modifiées du Modèle d'Information hébergé par le Serveur FDI au Modèle d'Information hébergé par le Serveur de communication FDI.

Un Serveur de communication FDI simple doit indiquer toutes les fonctions de configuration dans son élément EDD principal, non dans les sous-modules utilisés pour exposer les Points de connexion. Cela permet aux Hôtes FDI ne prenant pas en charge les appareils modulaires de paramétrer un Serveur de communication FDI en utilisant les mécanismes d'interface d'utilisateur FDI normalisés.

NOTE Le Serveur FDI peut modifier les valeurs des paramètres dans un ordre arbitraire.

7.8.12 Initialize

Lors de l'invocation de la méthode Initialize (voir 7.3.2.3), le Serveur de communication FDI doit utiliser les paramètres actuels et la configuration du matériel de communication pour les besoins de l'initialisation du matériel de communication.

Après l'initialisation réussie, le Serveur de communication FDI a le statut "Operate".

7.8.13 Création de l'objet de service de communication

Avant d'exécuter un échange de données, au moins une instance de type ServerCommunicationServiceType doit être présente ou créée. Une instance de ServerCommunicationServiceType doit toujours être présente. D'autres instances de ServerCommunicationServiceType peuvent être créées si l'instance de ServerCommunicationDeviceType met en œuvre la méthode AddComponent.

7.8.14 Relation de communication

Les définitions de l'Article 6 s'appliquent. Les éléments spécifiques du Serveur de communication sont définis dans le texte suivant.

Le diagramme états-transitions représenté à la Figure 11 décrit le flux d'état général d'une relation de communication unique avec les changements de statut supplémentaires relatifs aux éléments spécifiques OPC UA. Outre les aspects spécifiques, les définitions de l'Article 6 s'appliquent également.

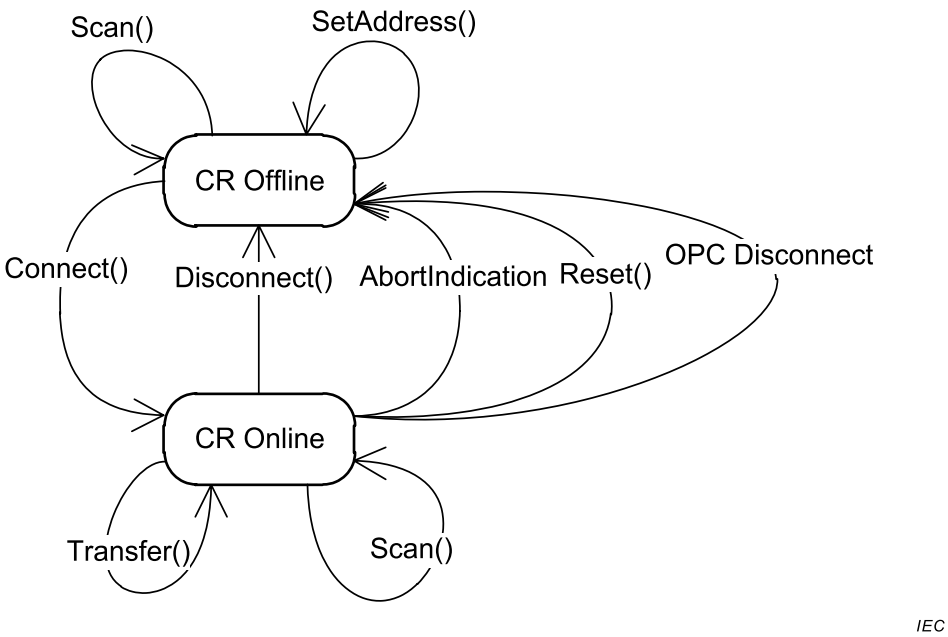


Figure 11 – Diagramme états-transitions de la relation de communication

Lors de l'invocation de la méthode Reset ou de la cessation de la connexion OPC (perte de connexion OPC), le Serveur de communication FDI doit mettre fin à toutes les relations de communication.

Le Serveur de communication FDI doit rejeter toute tentative de changement de paramétrisation ou de configuration dans le statut "Operate".

Si le statut du Serveur de communication FDI est différent de "Operate", le Serveur de communication FDI doit rejeter toute opération relative à la relation de communication.

Si le Serveur de communication prend en charge les instances multiples de ServerCommunicationServiceType, ces instances doivent partager les informations concernant les relations de communication existantes.

7.8.15 Connect

Avant d'exécuter une communication relative à l'échange d'informations, le Serveur de communication FDI nécessite l'établissement d'une relation de communication entre l'application de l'appareil et l'appareil connecté au réseau physique. Cela se produit par l'invocation de la méthode Connect.

Le Serveur de communication FDI doit être capable de gérer les relations de communication multiples. Après l'exécution réussie de la méthode Connect, la relation de communication correspondante a le statut "CR Online".

En raison de l'association directe de la méthode Connect à une instance de fournisseur de service de communication unique, l'Appareil de communication connaît la connexion réseau physique correspondante.

7.8.16 Disconnect

L'invocation de la méthode Disconnect met fin à une relation de communication, ce qui interrompt la communication relative à l'échange d'information avec l'appareil connecté au réseau physique.

Après l'exécution réussie de la méthode Disconnect, la relation de communication correspondante a le statut "CR Offline". La relation de communication devient non valide.

7.8.17 Indication d'interruption

Selon les éléments spécifiques du protocole, le Serveur de communication FDI peut détecter les interruptions de communication. Ces indications d'interruption de la communication sont retournées comme résultats du service de communication pendant le traitement des méthodes Transfer ou Scan. Une fois que le Serveur de communication FDI a retourné une indication d'interruption, la relation de communication actuelle a le statut "CR Offline". La relation de communication devient non valide.

7.8.18 Scan

La fonction de balayage de la topologie peut être invoquée indépendamment d'une relation de communication existante. Les détails du service de balayage sont spécifiés en 7.3.3.2.

7.8.19 SetAddress

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

8 Définition de la Passerelle de communication FDI

8.1 Généralités

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 de réseau hétérogènes.

8.2 Modèle d'Information

8.2.1 Généralités

Le Modèle d'Information d'une Passerelle est basé sur le Modèle d'Information défini dans l'IEC 62769-5. La Figure 12 représente la structure de l'appareil modulaire et son intégration dans le Modèle d'Information hébergé par le Serveur FDI total.

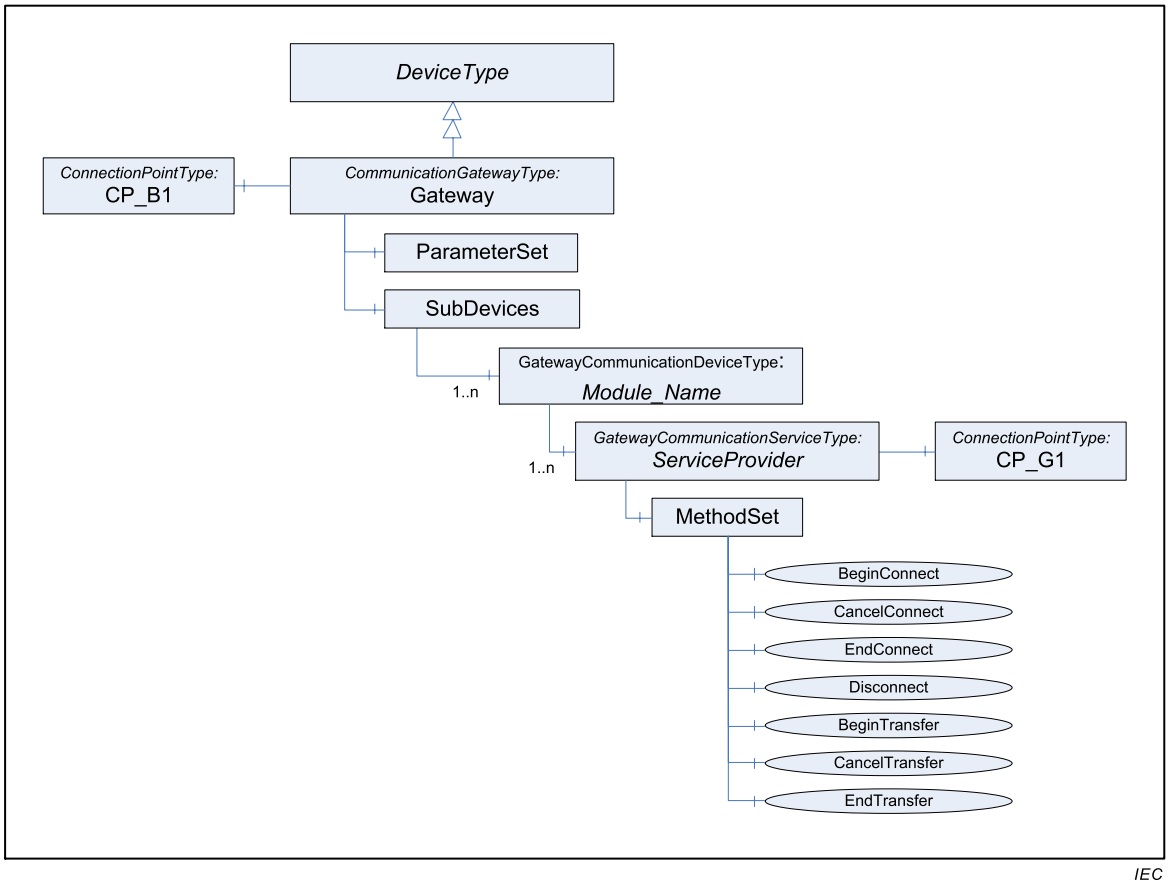


Figure 12 – Modèle d'Information de la Passerelle

La Passerelle est connectée au réseau (voir l'IEC 62769-5) par l'entremise d'une instance d'un ConnectionPointType (CP_B1). CP_B1 représente l'identification (nom) de l'objet assigné au Serveur FDI (voir l'IEC 62769-5).

La Passerelle est une instance de DeviceType. Le ParameterSet facultativement disponible (voir l'IEC 62769-5) doit contenir tous les paramètres d'appareil qui permettent de paramétrer l'interface de communication utilisée pour les demandes de service de communication lancées par la Passerelle.

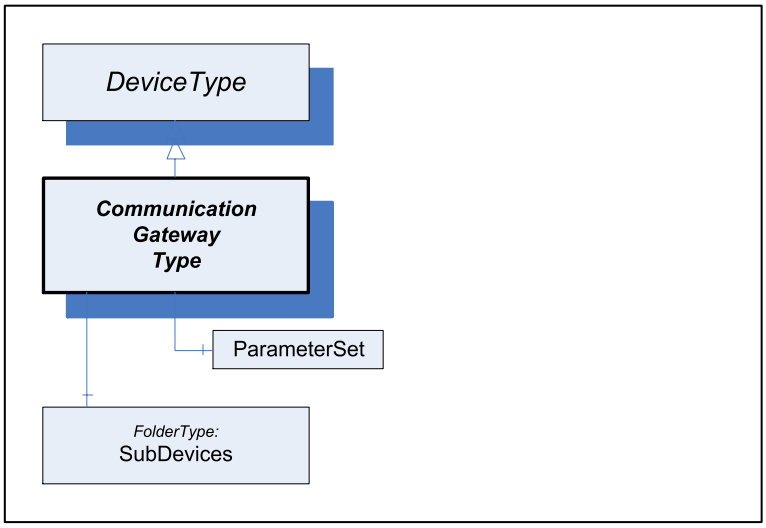
Les éléments sous SubDevices représentent l'accès physique ou logique aux médias réseau de la Passerelle. L'attribut Module_Name représente l'identification (nom d'exploration) de l'objet assigné au Serveur FDI (voir l'IEC 62769-5).

Les fonctions de traitement du service de communication de la Passerelle sont accessibles par des instances de fournisseur de service de communication multiples créées à partir de GatewayCommunicationServiceType. La logique applicative à l'origine des méthodes de service met en œuvre la fonction de traduction du protocole associée à l'interface du service de communication.

NOTE En comparaison avec le CommunicationServer du FDI, la Passerelle ne prend pas en charge les messages non sollicités, voir 7.3.4.5.

8.2.2 CommunicationGatewayType

Le CommunicationGatewayType est un sous-type du DeviceType. La Figure 13 présente la définition de CommunicationGatewayType. Elle est formellement définie dans le Tableau 27.



Anglais	Français
Communication Gateway Type	Type de passerelle de communication

Figure 13 – CommunicationGatewayType

Tableau 27 – Définition de CommunicationGatewayType

Attribut	Valeur				
BrowseName	CommunicationGatewayType				
IsAbstract	False				
Références	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Sous-type de DeviceType défini dans l'IEC 62541-100.					
HasComponent	Object	SubDevices		FolderType	Obligatoire
HasComponent	Object	ParameterSet			Facultatif

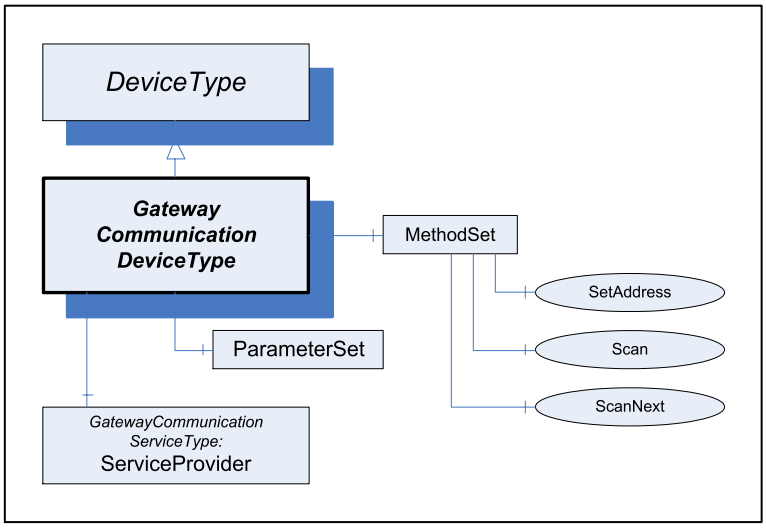
La gestion du module nécessaire pour prendre en charge une structure de matériel de communication configurable est basée sur les définitions du COMPONENT pour le CommunicationGatewayType entier. Les définitions du COMPONENT fournissent les informations suffisantes pour exécuter la configuration de l'installation du module générique.

NOTE L'indication pour une disposition de matériel de communication statique est présente avec l'attribut du COMPONENT CAN_DELETE mis sur FALSE pour toutes les déclarations relatives aux sous-appareils de l'ensemble de l'appareil.

8.2.3 GatewayCommunicationDeviceType

8.2.3.1 Généralités

Le GatewayCommunicationDeviceType représente un module de communication ou un canal connecté à un réseau particulier. Le GatewayCommunicationDeviceType est un sous-type du DeviceType. Le ParameterSet de chaque GatewayCommunicationDeviceType contiendra les paramètres nécessaires pour configurer le fonctionnement du réseau. Les paramètres de bus obligatoires spécifiques au protocole sont spécifiés dans l'IEC 62769-4:2015, Annexe F. La Figure 14 représente la définition de GatewayCommunicationDeviceType qui est formellement décrite dans le Tableau 28 et le Tableau 29.



Anglais	Français
Gateway Communication Device Type	Type d'appareil de communication de la passerelle

Figure 14 – GatewayCommunicationDeviceType

Tableau 28 – Définition de GatewayCommunicationDeviceType

Attribut	Valeur				
BrowseName	GatewayCommunicationDeviceType				
IsAbstract	True				
Références	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Sous-type de DeviceType défini en OPC UA Partie DI.					
HasComponent	Object	MethodSet		BaseObjectType	Facultatif
HasComponent	Object	ParameterSet		BaseObjectType	Facultatif
HasComponent	Object	ServiceProvider		ServiceType de la communication de Passerelle	Obligatoire

Tableau 29 – MethodSet de GatewayCommunicationDeviceType

Attribut	Valeur				
BrowseName	MethodSet				
IsAbstract	True				
Références	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
HasComponent	Method	Scan			Facultatif
HasComponent	Method	ScanNext			Facultatif
HasComponent	Method	SetAddress			Facultatif

8.2.3.2 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 une méthode basée sur EDDL.

La logique EDDL n'étant pas conçue pour manipuler les documents XML, la signature du service Scan n'est pas conforme à la définition donnée en 7.3.3.2. La mise en œuvre du service de balayage retourne les appareils découverts en utilisant la construction LIST qui contient les instances COLLECTION. Les détails sont spécifiés en 8.2.3.4. Les instances COLLECTION représentent les instances du Point de connexion.

La signature de cette Méthode est spécifiée ci-dessous. Le Tableau 30 et le Tableau 31 spécifient respectivement les arguments et la représentation de l'AddressSpace.

Signature

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

Tableau 30 – Arguments de la méthode Scan

Argument	Description
ServiceError	0: OK 1: OK/obtenir le résultat de balayage complet en appelant ScanNext -1: Echec/non initialisé -2: Echec/non connecté à un réseau -3: Echec/aucun appareil n'a été trouvé, topologyScanResult est vide

Tableau 31 – Définition de l'AddressSpace de la méthode Scan

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

8.2.3.3 Méthode ScanNext

La méthode ScanNext doit être utilisée pour continuer 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 une méthode basée sur EDDL.

La logique EDDL n'étant pas conçue pour manipuler les documents XML, la signature du service Scan n'est pas conforme à la définition donnée en 7.3.3.2. La mise en œuvre du service de balayage retourne les appareils découverts en utilisant la construction LIST qui contient les instances COLLECTION. Les détails sont spécifiés en 8.2.3.4. Les instances COLLECTION représentent les instances du Point de connexion.

La signature de cette Méthode est spécifiée ci-dessous. Le Tableau 32 et le Tableau 33 spécifient respectivement les arguments et la représentation de l'AddressSpace.

Signature

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

Tableau 32 – Arguments de la méthode ScanNext

Argument	Description
ServiceError	0: OK 1: OK/obtenir le résultat de balayage complet en rappelant ScanNext -1: Echec/non initialisé -2: Echec/non connecté à un réseau -3: Echec/aucun appareil n'a été trouvé, topologyScanResult est vide

Tableau 33 – Définition de l'AddressSpace de la méthode ScanNext

Attribut	Valeur				
BrowseName	ScanNext				
Références	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Obligatoire

8.2.3.4 Liste Scan

Chaque EDD décrivant le Point de connexion d'une Passerelle doit décrire l'élément LIST qui contient la liste des appareils découverts après l'exécution réussie du service Scan.

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

<ID>: L'identificateur doit correspondre à la valeur d'attribut SCAN_LIST décrite en 8.3.2.4.

<ListEntry>: La valeur d'attribut doit se rapporter à la définition de COLLECTION qui décrit le Point de connexion, comme défini en 8.3.2.4.

8.2.3.5 Méthode SetAddress

Pour les définitions, voir 7.3.3.4.

8.2.4 GatewayCommunicationServiceType

8.2.4.1 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 basées sur l'EDDL (logique applicative) fournies avec l'EDD contenu dans le Paquetage d'appareil FDI.

La mise en œuvre des services de communication, excepté le service Disconnect, suit un modèle asynchrone. Ceci implique que chaque service de communication est divisé en trois méthodes. Cela permet au Serveur FDI d'exécuter les services de communication de manière parallèle et d'annuler également les opérations qui durent trop longtemps.

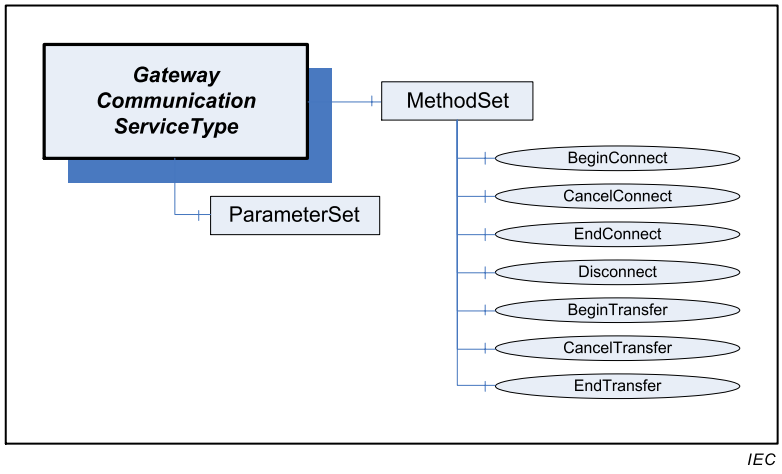
Begin<name>: Cette méthode lance l'exécution du service. La méthode retourne l'état de l'exécution de <name> ou le résultat s'il est immédiatement présent.

End<name>: Si le résultat du service est déjà disponible, cette méthode vérifie qu'il a commencé avec un appel Begin<name> précédent. Comme Begin<name>, cette méthode retourne l'état d'exécution ou le résultat s'il est disponible.

Cancel<name>: Cette méthode annule une exécution de service lancé.

Un numéro d'identification du service (ServiceId) permet à la Passerelle de communication de garder une trace de la relation entre les appels de la méthode qui appartiennent à un processus de service de communication unique. Si la Passerelle de communication prend en charge des instances multiples de GatewayCommunicationServiceType, ces instances partagent les informations concernant les ServiceIds actuellement utilisés. Par conséquent, un service de communication doit être exécuté sur une instance unique du GatewayCommunicationServiceType.

Pour réduire les cycles de sondage, les deux méthodes Begin<name> et End<name> retournent une valeur de délai (DelayForNextCall). L'appelant de la méthode doit différer l'invocation de la méthode End<name> conformément à la valeur d'argument retournée.



Anglais	Français
Gateway Communication Service Type	Type de service de communication de la passerelle

Figure 15 – GatewayCommunicationServiceType

Tableau 34 – Définition de GatewayCommunicationServiceType

Attribut	Valeur				
BrowseName	GatewayCommunicationServiceType				
IsAbstract					
Références	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Sous-type de DeviceType défini en OPC UA Partie DI.					
HasComponent	Object	MethodSet		BaseObjectType	Obligatoire
HasComponent	Object	ParameterSet		BaseObjectType	Facultatif

Tableau 35 – MethodSet de GatewayCommunicationServiceType

Attribut	Valeur				
BrowseName	MethodSet				
IsAbstract					
Références	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
HasComponent	Method	BeginConnect			Facultatif
HasComponent	Method	CancelConnect			Facultatif
HasComponent	Method	EndConnect			Facultatif
HasComponent	Method	Disconnect			Facultatif
HasComponent	Method	BeginTransfer			Obligatoire
HasComponent	Method	CancelTransfer			Obligatoire
HasComponent	Method	EndTransfer			Obligatoire

8.2.4.2 Service Connect

Le service Connect doit être utilisé pour établir une relation de communication avec un appareil physiquement connecté au réseau. Etablir 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. Le service effectue la vérification de correspondance de ce DeviceType conformément à la norme du protocole réseau. Les détails pertinents sont spécifiés dans l'IEC 62769-4:2015, 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 le Modèle d'Information associé à l'application de l'appareil et l'appareil physique est identifiée avec de plus amples détails par l'identificateur de la relation de communication. Les détails sur la manière de gérer l'état d'une relation de communication sont décrits à l'Article 6.

NOTE 1 Le NodeId étant 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 la portée d'un Appareil de communication.

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

Les signatures des méthodes de service Connect sont spécifiées ci-dessous. Le Tableau 36 spécifie les arguments.

Signature

```
BeginConnect(
   [in]   ByteString       CommunicationRelationId,
   [in]   BaseDataType[]   AddressData,
   [out]  BaseDataType[]   DeviceInformation,
   [in]   UInt32           ServiceID,
   [out]  UInt32           DelayForNextCall,
   [out]  Int32            ServiceError);

EndConnect(
   [in]   ByteString       CommunicationRelationId,
   [out]  BaseDataType[]   DeviceInformation,
   [in]   UInt32           ServiceID,
   [out]  Uint32           DelayForNextCall,
   [out]  Int32            ServiceError);

CancelConnect(
   [in]   ByteString       CommunicationRelationId,
   [in]   Uint32           ServiceID,
   [out]  Int32            ServiceError);
```

Tableau 36 – 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 pourrait s'agir d'un index (par exemple, un NodeId) dont le client (= Serveur FDI) doit identifier les entrées dans sa topologie.
AddressData	Liste d'arguments spécifique au protocole utilisée pour l'adresse et les données d'identification facultatives de l'appareil (les détails sont décrits dans l'IEC 62769-4:2015, Annexe F).
DeviceInformation	Liste d'arguments spécifique au protocole dans laquelle la routine de connexion stocke les données résultantes.
ServiceId	Le code de transaction de service établit la relation entre la demande de service et la réponse correspondante.
DelayForNextCall	La valeur spécifie un délai en ms pendant lequel l'appelant doit attendre avant l'invocation suivante de "EndConnect".
ServiceError	0: OK/fonction démarrée de façon asynchrone, le résultat doit être interrogé avec EndConnect 1: OK/exécution terminée, connexion établie avec succès -1: Echec de la connexion/annulée par l'appelant -2: Echec de l'appel/ID de service inconnu -3: Echec de la connexion/appareil introuvable -4: Echec de la connexion/adresse d'appareil non valide -5: Echec de la connexion/identification d'appareil non valide

8.2.4.3 Méthode Disconnect

Spécifié en 7.3.4.3.

8.2.4.4 Service Transfer

La méthode Transfer doit être utilisée pour effectuer l'échange avec un appareil.

Les signatures des méthodes de service Transfer sont spécifiées ci-dessous. Tous les arguments sont spécifiés dans le Tableau 37.

Signature

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

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

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

Tableau 37 – Arguments de la méthode Transfer

Argument	Description
CommunicationRelationId	Voir 8.2.4.2.
SendData	Liste de valeurs spécifiques au protocole telles que décrites dans l'IEC 62769-4:2015, Annexe F. Les valeurs d'argument représentent la demande de services de communication spécifique au protocole envoyée à l'appareil.
ReceiveData	Liste de valeurs spécifiques au protocole telles que décrites dans l'IEC 62769-4:2015, Annexe F. Les valeurs d'argument représentent la réponse de services de communication spécifique au protocole reçue de l'appareil.
ServiceId	Le code de transaction de service établit la relation entre la demande de service et la réponse correspondante.
DelayForNextCall	La valeur spécifie un délai pendant lequel l'appelant doit attendre avant l'invocation suivante d'EndTransfer.
ServiceError	0: OK/fonction démarrée de façon asynchrone, le résultat doit être interrogé avec EndTransfer 1: OK/exécution terminée, ReceivedData contient le résultat -1: Echec du transfert/annulé par l'appelant -2: Echec de l'appel/ID de service inconnu -3: Echec du transfert/aucune relation de communication existante -4: Echec du transfert/identificateur de relation de communication non valide -5: Echec du transfert/contenu sendData non valide -6: Echec du transfert/format receiveData non valide

8.3 Paquetage de communication FDI

8.3.1 Généralités

Le Paragraphe 8.3 spécifie les détails du Paquetage de communication FDI qui sont spécifiques pour les Passerelles. Les définitions indiquées en 5.1 s'appliquent également.

8.3.2 EDD

8.3.2.1 Généralités

Les définitions indiquées en 5.2 s'appliquent. En outre, les éléments EDD spécifiés dans l'IEC 62769-4 et fournis avec le Paquetage de communication FDI spécifique à une Passerelle doivent contenir:

a) Un PROFILE (IEC 61804-3): La définition de PROFILE doit être choisie selon un protocole utilisé pour les demandes de services de communication.

b) Une Logique applicative: Les COMPONENTS EDD relatifs au fournisseur de service de communication doivent mettre en œuvre les méthodes spécifiées dans l'IEC 61804-3. Ces méthodes mettent en œuvre la logique de pontage du protocole. Les procédures de traduction s'ouvrent en demandes de communication sortantes par l'invocation de la fonction de la bibliothèque intégrée EDDL ou l'écriture des données dans un nœud en ligne. L'ensemble des fonctions utilisables de la bibliothèque intégrée EDDL est lié au PROFILE.

c) Une Gestion du module: Le composant relatif à la Passerelle peut mettre en œuvre les ValidateModules (voir 5.2.9). La logique mise en œuvre doit valider les changements individuels et manipuler les dépendances relatives aux paramètres pour l'appareil Passerelle entier. La mise en œuvre de ValidateModules est facultative lorsque la déclaration COMPONENT spécifique au produit actuel est suffisante pour configurer l'installation du module sans aucune autre logique applicative.

8.3.2.2 Composant de la Passerelle

Chaque Paquetage de communication FDI décrivant une Passerelle doit contenir un élément EDD décrivant la Passerelle, comme défini en 5.2.2. Les éléments spécifiques de la Passerelle sont décrits dans ce qui suit:

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

<ProtocolID>: L'existence de cet attribut indique la connectivité de la Passerelle concernant la communication sortante. Il permet au Serveur FDI de trouver le réseau en utilisant le même type de protocole auquel cette Passerelle peut être connectée. Pour les protocoles normalisés, la valeur est définie par l'organisation de bus de terrain concernée.

8.3.2.3 Composant CommunicationDevice de la Passerelle

Chaque Paquetage de communication FDI décrivant une Passerelle doit contenir au moins un élément EDD qui décrit au moins un composant CommunicationDevice, comme défini en 5.2.3.

NOTE Une Passerelle est parfois désignée comme "Remote IO". Remote IO est un appareil modulaire prenant en charge plusieurs types de modules différents qui peut être assigné de manière souple à n'importe quelle position. Par conséquent, plusieurs configurations Remote IO différentes (n – positions X, m – types de modules) peuvent être créées.

Les règles concernant les paramètres de l'attribut COMPONENT décrits en 5.2.3 s'appliquent. Les éléments spécifiques de la Passerelle sont décrits dans ce qui suit:

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

<Scan>: L'attribut se rapporte à la METHOD mettant en œuvre le service de découverte de l'appareil. La référence fonctionne parce que la valeur de l'identificateur de la METHOD correspond à la valeur d'attribut. Les détails de mise en œuvre sont spécifiés en 8.2.3.2.

<ScanList>: La valeur d'attribut se rapporte à la LIST décrivant les résultats de balayage de la topologie. Cette liste doit contenir tous les appareils découverts pendant l'exécution du service de découverte de l'appareil. Les détails de mise en œuvre sont spécifiés en 8.2.3.4.

8.3.2.4 Composant du service de communication

Les règles définies en 5.2.4 s'appliquent.

En outre, le fichier décrivant le composant contient les mises en œuvre des méthodes définies en 8.2.4.2, 8.2.4.3 et 8.2.4.4. L'identificateur utilisé pour les constructions METHOD correspond aux noms de méthode spécifiés dans le Tableau 35.

8.3.2.5 Composant Point de connexion

Les règles définies en 5.2.5 s'appliquent.

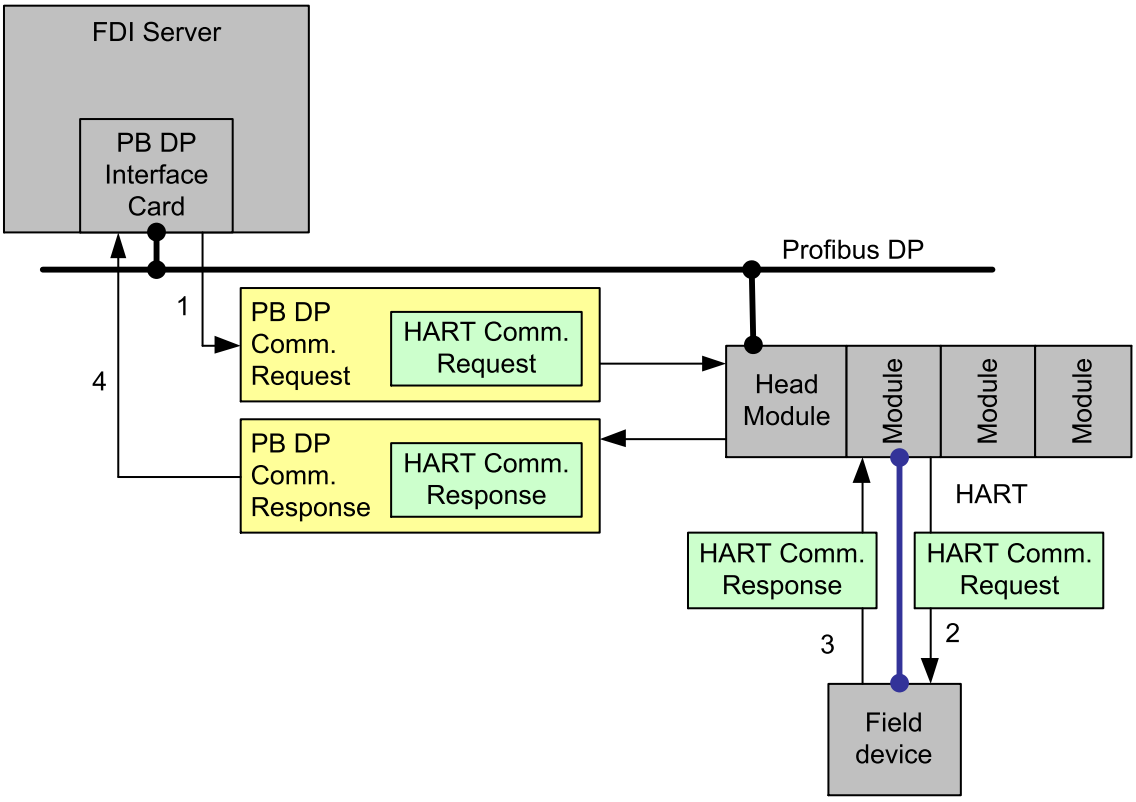
8.3.2.6 Collection du Point de connexion

Les règles définies en 5.2.6 s'appliquent.

8.4 Manipulation et comportement

8.4.1 Généralités

Une Passerelle fournit la fonctionnalité permettant de communiquer entre deux protocoles de communication. Les Passerelles sont utilisées pour la communication entre un réseau du système d'automatisation et un autre réseau (subordonné). La Figure 16 représente un exemple typique où un appareil HART est connecté à l'E/S distante PROFIBUS. Afin de communiquer avec l'appareil HART, l'E/S distante reçoit une demande de communication par le réseau PROFIBUS (voir Figure 16, légende 1). La demande de communication contient les informations nécessaires pour permettre à la Passerelle de créer la commande HART et de l'envoyer à l'appareil HART (voir Figure 16, légende 2). La manière dont la commande HART est contenue dans la demande de communication PROFIBUS peut être normalisée ou spécifique à la Passerelle. La réponse HART (voir Figure 16, légende 3) en provenance de l'appareil est intégrée comme réponse de communication PROFIBUS (voir Figure 16, légende 4). La manière dont la Passerelle contient la réponse peut être normalisée ou spécifique à la Passerelle.



Anglais	Français
FDI Server	Serveur FDI
PB DP Interface Card	Carte interface PB DP
PB DP Comm. Request	Demande de communication PB DP
HART Comm. Request	Demande de communication HART
PB DP Comm. Response	Réponse de communication PB DP
HART Comm. Response	Réponse de communication HART
Head Module	Module de tête
Module	Module
Field device	Appareil de terrain
Profibus DP	Profibus DP

Figure 16 – Communication imbriquée

Le Paragraphe 8.4 définit les règles de manipulation et de comportement de la Passerelle tout au long du cycle de vie (déploiement, démarrage, mise en service du bus, traitement des services de communication).

8.4.2 Déploiement

Les définitions indiquées en 5.2.11 s'appliquent.

8.4.3 Démarrage

Le Modèle d'Information et la représentation de la Passerelle basée sur l'élément EDD du Paquetage FDI ne nécessitent aucune procédure de démarrage.

8.4.4 Configuration du matériel de communication

La manipulation et le comportement sont conformes à la spécification décrite en 7.8.9. La seule différence concerne l'installation du canal représentée uniquement dans le Modèle d'Information hébergé par le Serveur FDI.

8.4.5 Configuration du réseau

La manipulation et le comportement sont conformes à la spécification décrite en 7.8.10.

8.4.6 Paramétrisation

La Passerelle peut nécessiter le réglage des paramètres appropriés avant tout traitement du service de communication. Les dialogues d'utilisateur (UID ou UIP) contenus dans le Paquetage de communication FDI permettent un réglage interactif des paramètres de bus. Le Paquetage de communication FDI peut fournir une logique applicative supplémentaire aux fins de paramétrisation du bus.

8.4.7 Relation de communication

Les définitions de diagrammes d'états données à l'Article 6 s'appliquent.

Si la Passerelle de communication prend en charge les instances multiples de GatewayCommunicationServiceType, ces instances doivent partager les informations concernant les relations de communication existantes.

8.4.8 Connect

Avant d'exécuter une communication relative à l'échange d'informations, la Passerelle exige l'établissement d'une relation de communication entre l'application de l'appareil et l'appareil connecté au réseau physique. Cela se produit par l'invocation de la méthode Connect. La Passerelle peut ainsi effectuer une demande de service de communication sortante facultative, ce qui pourrait être nécessaire pour qu'un appareil Passerelle spécifique établisse une relation de communication.

La Passerelle doit être capable de gérer les relations de communication multiples. Après l'exécution réussie de la méthode Connect, la relation de communication correspondante a le statut "CR Online".

L'invocation des services Transfer et Scan est permise dans le statut "Online" seulement.

8.4.9 Disconnect

L'invocation de la méthode Disconnect met fin à une relation de communication, ce qui interrompt la communication relative à l'échange d'information avec l'appareil connecté au réseau physique.

Après l'exécution réussie de la méthode Disconnect, la relation de communication correspondante a le statut "CR Offline". La relation de communication devient non valide.

8.4.10 Indication d'interruption

Selon les éléments spécifiques du protocole, la Passerelle peut détecter les interruptions de communication. Ces indications d'interruption de la communication sont retournées comme résultats du service de communication pendant le traitement des méthodes Transfer ou Scan. Une fois que la Passerelle a retourné une indication d'interruption, la relation de communication actuelle a le statut "CR Offline". La relation de communication devient non valide.

8.4.11 Scan

Les Passerelles déclarent leur fonction de service de découverte d'appareils selon la construction EDDL COMPONENT et les attributs SCAN et SCAN_LIST. Les définitions concernées sont définies dans l'IEC 61804-3. Le paramètre de l'attribut SCAN est obligatoire dans l'élément EDD décrivant la Passerelle CommunicationDevice. La valeur d'attribut se rapporte à la METHOD exécutant la fonction de balayage de la topologie. Le paramètre de l'attribut SCAN_LIST est obligatoire dans le COMPONENT CommunicationDevice de la Passerelle. La valeur d'attribut se rapporte à l'élément qui contient le résultat de balayage de la topologie créé par la méthode indiquée par la valeur d'attribut SCAN. La SCAN_LIST doit se rapporter à une LIST contenant les éléments COLLECTION décrivant les appareils détectés (Scan-List-Item). Le contenu spécifique au protocole d'un Scan-List-Item est décrit dans l'IEC 62769-4:2015, Annexe F.

L'invocation des fonctions Scan et ScanNext entraîne des demandes de services de communication sortante.

8.4.12 Manipulation des erreurs de communication

Le traitement du service de communication doit utiliser la fonction EDDL intégrée d'interruption conformément aux profils EDDL:

- pendant la création de messages de communication;
- pendant le traitement de la réponse provenant de la demande de communication.

Le traitement du service de communication ne déclenche pas les erreurs de communication selon les expirations de délai de communication.

Annexe A
(informative)

Protocoles hiérarchisés

A.1 Généralités

Les protocoles basés sur Ethernet consistent en une pile de différents protocoles basés sur le modèle ISO/OSI. Vu le nombre croissant de protocoles de bus de terrain basés sur Ethernet, il est crucial d'avoir un concept de modélisation hiérarchisé commun pour les Points de connexion basés sur le modèle ISO/OSI.

Les Points de connexion sont les éléments qui contiennent les informations d'adresse auxquelles les Appareils de communication accèdent pour recueillir les informations nécessaires pour la communication. La sémantique des attributs du Point de connexion doit être normalisée.

L'appareil PROFINET peut prendre en charge simultanément PROFINET, SNMP et HTTP. Les informations stockées dans le Point de connexion d'un appareil sont différentes pour chaque protocole, et ce en raison des différentes couches d'application. Les informations de couches 1 à 4 doivent contenir systématiquement les mêmes informations. Par conséquent, le Point de connexion doit hériter des informations du Point de connexion des couches de réseau inférieures.

Le problème consiste à savoir comment garantir que les informations d'adresse des couches inférieures soient dénommées de manière cohérente à travers les protocoles construits sur les couches inférieures.

A.2 Convention relative à la création de l'annexe spécifique au protocole

A.2.1 Point de connexion

Puisque la description du Point de connexion est basée sur l'EDDL, une approche de la relation d'héritage réelle connue à partir des langages de programmation orientés objet ne semble pas applicable. L'approche décrite à l'Article A.2 est davantage basée sur les conventions. La convention de dénomination doit assurer que les noms d'attribut de la valeur d'adresse définis pour un protocole de "niveau inférieur" soient réutilisés dans les définitions du Point de connexion pour les protocoles de niveau supérieur. Il en va de même pour les VARIABLES qui se rapportent à la COLLECTION. Les déclarations de VARIABLE pour les protocoles de niveau supérieur doivent être des copies des déclarations de VARIABLE pour les protocoles de niveau inférieur. La convention décrite ici est applicable pour la création des annexes spécifiques au protocole. Voici quelques exemples EDDL de la manière dont les déclarations de Point de connexion suivent cette convention de dénomination.

```
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 Définition du Paquetage de communication FDI

A.3.1 Services de communication

Le service de communication réel est toujours mis en œuvre conformément au protocole spécifique. Ceci est reflété par la sémantique différente des arguments du service de communication spécifiés par les annexes spécifiques du protocole. La mise en œuvre réelle d'un ServerCommunicationDeviceType ou d'un GatewayCommunicationDeviceType ne peut donc prendre en charge qu'un seul protocole. Par conséquent, si un Serveur de communication FDI ou une Passerelle est capable de prendre en charge plusieurs protocoles différents, il doit décrire des GatewayCommunicationDeviceTypes ou des ServerCommunicationDeviceTypes distincts. Le besoin de définir les ensembles de services spécifiques au protocole représente la demande de séparer les définitions du COMPONENT relatives au Point de connexion et au réseau décrites en A.3.2 et A.3.3.

A.3.2 Point de connexion

Le Paquetage de communication FDI doit contenir des descriptions distinctes du Point de connexion pour chaque protocole pris en charge.

A.3.3 Réseau

La relation ente le COMPONENT décrivant le réseau et le COMPONENT décrivant le Point de connexion permet la détection du chemin de communication générique et la configuration de la topologie générique. Ainsi, le Paquetage de communication FDI doit contenir une définition de COMPONENT distincte pour chaque réseau pris en charge (protocole).

A.4 Représentation dans le modèle d'information

Les Points de connexion partageant une certaine formation d'ensemble d'adresses peuvent contenir des informations d'adresse redondantes; par exemple, l'adresse IP est la même pour le SNMP et le PROFINET I/O.

Si un appareil et un Serveur de communication FDI partagent un ensemble de protocoles, alors cet appareil et ce Serveur de communication FDI sont associés par des réseaux distincts multiples.

Un appareil prenant en charge des protocoles multiples peut être connecté à différents Appareils de communication FDI qui prennent en charge un seul protocole.

Annexe B
(normative)

Espace de noms et mappings

La présente Annexe définit les identificateurs numériques de tous les *NodeIds* numériques définis dans la présente norme. Les identificateurs sont spécifiés dans un fichier CSV avec la syntaxe suivante:

`<SymbolName>, <Identifier>, <NodeClass>`

où *SymbolName* est soit le *BrowseName* d'un *Nœud de Type*, doit le *BrowsePath* d'un *Nœud d'Instance* qui apparaît dans la spécification, et où *Identifier* est la valeur numérique du *NodeId*.

Le *BrowsePath* d'un *Nœud d'Instance* est construit par ajout du *BrowseName* du *Nœud* d'instance au *BrowseName* de l'instance ou du type qui le contient. Un caractère de soulignement est utilisé pour séparer chaque *BrowseName* du chemin.

Le *NamespaceUri* http://fdi-cooperation.com/OpcUa/FDI7/ s'applique aux *NodeIds* définis ici.

Le CSV fourni avec la présente version de la norme peut être trouvé ici:
http://www.fdi-cooperation.com/tl_files/Specification/1.0/Schemas/Opc.Ua.Fdi7.NodeIds.csv

Une version électronique du Modèle d'Information défini dans la présente norme est également fournie. Elle suit la syntaxe du Schéma de Modèle d'Information XML défini dans l'IEC 62541-6.

Le Schéma de Modèle d'Information fourni avec la présente version de la norme peut être trouvé ici:
http://www.fdi-cooperation.com/tl_files/Specification/1.0/Schemas/Opc.Ua.Fdi7.NodeSet2.xml

Bibliographie

FDI-2021, *FDI Project Technical Specification – Part 1: Overview*
<disponible en anglais seulement à www.fdi-cooperation.com>

FDI-2022, *FDI Project Technical Specification – Part 2: FDI Client*
<disponible en anglais seulement à www.fdi-cooperation.com>

FDI-2023, *FDI Project Technical Specification – Part 3: FDI Server*
<disponible en anglais seulement à www.fdi-cooperation.com>

FDI-2024, *FDI Project Technical Specification – Part 4: FDI Packages*
<disponible en anglais seulement à www.fdi-cooperation.com>

FDI-2025, *FDI Project Technical Specification – Part 5: FDI Information Model*
<disponible en anglais seulement à www.fdi-cooperation.com>

FDI-2026, *FDI Project Technical Specification – Part 6: FDI Technology Mapping*
< disponible en anglais seulement www.fdi-cooperation.com>

FDI-2027, *FDI Project Technical Specification – Part 7: FDI Communication Devices*
<disponible en anglais seulement à www.fdi-cooperation.com>

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

3, rue de Varembé
PO Box 131
CH-1211 Geneva 20
Switzerland

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