IEC 62769-100

Edition 2.0 2023-04
REDLINE VERSION

INTERNATIONAL STANDARD

colour
inside

Field device integration (FDI®) –
Part 100: Profiles – Generic Protocols

IEC Secretariat
3, rue de Varembé
CH-1211 Geneva 20
Switzerland

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

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

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

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 It also gives information on projects, replaced and withdrawn publications.

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

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

IEC Products & Services Portal - products.iec.ch
Discover our powerful search engine and read freely all the publications previews. With a subscription you will always have access to up to date content tailored to your needs.

Electropedia - www.electropedia.org
The world's leading online dictionary on electrotechnology, containing more than 22 300 terminological entries in English and French, with equivalent terms in 19 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC 62769-100

Edition 2.0 2023-04
REDLINE VERSION

INTERNATIONAL STANDARD

colour
inside

Field device integration (FDI®) –
Part 100: Profiles – Generic Protocols

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

ICS 25.040.40; 35.100.05

ISBN 978-2-8322-6845-2

Warning! Make sure that you obtained this publication from an authorized distributor.

® Registered trademark of the International Electrotechnical Commission

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

____

FIELD DEVICE INTEGRATION (FDI®) –

Part 100: Profiles – Generic Protocols

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.

2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.

3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.

4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.

5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.

6) All users should ensure that they have the latest edition of this publication.

7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.

8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.

9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

This redline version of the official IEC Standard allows the user to identify the changes made to the previous edition IEC 62769-100:2020. A vertical bar appears in the margin wherever a change has been made. Additions are in green text, deletions are in strikethrough red text.

IEC 62769-100 has been prepared by subcommittee 65E: Devices and integration in enterprise systems, of IEC technical committee 65: Industrial-process measurement, control and automation. It is an International Standard.

This second edition cancels and replaces the first edition published in 2020. This edition constitutes a technical revision.

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

a) added ExtendedDeviceRevision implementing the FDI® Version scheme and the method ScanExtended.

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

Draft	Report on voting
65E/865/CDV	65E/922/RVC

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

The language used for the development of this International Standard is English.

This document was drafted in accordance with ISO/IEC Directives, Part 2, and developed in accordance with ISO/IEC Directives, Part 1 and ISO/IEC Directives, IEC Supplement, available at www.iec.ch/members_experts/refdocs. The main document types developed by IEC are described in greater detail at www.iec.ch/standardsdev/publications.

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

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

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

IMPORTANT – The "colour inside" logo on the cover page of this document 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.

FIELD DEVICE INTEGRATION (FDI®) –

Part 100: Profiles – Generic Protocols

1 Scope

This part of IEC 62769 specifies an FDI®[1] profile of IEC 62769 for Generic Protocols. That means that all interfaces are defined and a host can add support for more protocols without changing its implementation. Nevertheless, there are some protocol specific definitions (PSD) that need to be specified per protocol using this profile. Annex C specifies what PSD need to be defined per protocol so that FDI® Device Packages, FDI® Communication Packages for Gateways and FDI® Communication Servers, FDI® Communication Server, Gateways and Devices supporting such a protocol can work together in a host not aware about this specific protocol.

NOTE A host not using FDI® Communication Server but a proprietary mechanism for communication ~~defines~~ needs to define its own means to deal with this profile to support several protocols without changing its implementation. This is specific to the proprietary way how the communication driver is bound to the host.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61804 (all parts), *Devices and integration in enterprise systems – Function blocks (FB) for process control and Electronic Device Description Language (EDDL)*

IEC 61804-3, *Devices and integration in enterprise systems – Function blocks (FB) for process control and electronic device description language (EDDL) – Part 3: EDDL syntax and semantics*

IEC 62541-100:2015, *OPC Unified Architecture – Part 100: ~~OPC UA for Devices~~ Device Interface*

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

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

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

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

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

[1] FDI® is a registered trademark of the non-profit organization Fieldbus Foundation, Inc. This information is given for the convenience of users of this document and does not constitute an endorsement by IEC of the trademark holder or any of its products. Compliance does not require use of the trade name. Use of the trade name requires permission of the trade name holder.

IEC 62769-7*, Field Device Integration (FDI®) – Part 7:~~ FDI~~ Communication Devices*

IEC 62769-151-1, *Field Device Integration (FDI®) – Part 151-1: Profiles – OPC UA*

3 Terms, definitions, abbreviated terms and ~~conventions~~ acronyms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 61804 series, IEC 62541-100, IEC 62769-2, IEC 62769-3, IEC 62769-4, IEC 62769-5, and IEC 62769-7 apply.

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

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

3.2 Abbreviated terms and acronyms

EDD	Electronic Device Description
EDDL	Electronic Device Description Language (see IEC 61804 series)
FDI~~™~~®	Field Device Integration~~™[2]~~
FCG	FieldComm Group
PSD	Protocol-specific definitions
XML	Extensible markup language (see ~~FCG TS62769-100, Edition 1.1, Field Device Integration Part 100: Profiles – Generic Protocols, available at https://fieldcommgroup.org)~~ REC-xml-20081126)

4 Conventions

4.1 EDDL syntax

This part of IEC 62769 specifies content for the EDD component that is part of FDI® Communication Packages. The specification content using EDDL syntax uses the font `Courier New`. The EDDL syntax is used for method signature, variable, data structure and component declarations.

4.2 XML syntax

XML syntax examples use font `Courier New`. The XML syntax is used to describe XML document schema.

Example: `<xs:simpleType name="ExampleType">`

4.3 Capitalizations

~~The IEC 62769 series uses capitalized terms to emphasize that these terms have an FDI-specific meaning.~~

~~[2] Field Device Integration (FDI) is the trademark of a product supplied by FieldComm Group. This information is given for the convenience of users of this document and does not constitute an endorsement by IEC of the product named. Equivalent products may be used if they can be shown to lead to the same results.~~

Capitalization of the first letter of words is used in the IEC 62769 series to emphasize an FDI® defined term.

EDD language elements are written with all letters in uppercase.

Some of these terms using an ~~initialism~~ acronym as a prefix for example

- FDI® Client, or
- FDI® Server.

Some of these terms are compound terms such as:

- Communication Servers, or
- Profile Package.

Parameter names or attributes are concatenated to a single term, where the original terms start in this term with a capital letter such as:

- ProtocolSupportFile, or
- ProtocolType.

Parameter names or attributes can also be constructed by using an underscore character to concatenate two or more terms such as:

- DEVICE_REV, or
- DEVICE_MODEL.

5 Profile for Generic Protocols

5.1 General

This profile document to the FDI® specification in IEC 62769 specifies the protocol specifics needed for FDI® Packages describing Communication Servers, Gateways and Devices.

~~For Communication Servers, this document defines protocol specifics as these need to be considered in the Information Model hosted by the Communication Servers.~~

This document also defines protocol specifics for Communication Servers as these need to be considered in the Communication Servers hosted Information Model.

Annex B defines the XML schema for Direct Access Services.

5.2 Catalog profile

5.2.1 Protocol support file

5.2.1.1 FDI® Device Package

Protocol specific attachments are mentioned in the Package Catalog as defined in IEC 62769-5. As this document defines a profile generically suitable for many protocols, it does not define requirements for any protocol specific attachments. However, it does also not prevent the usage of protocol specific attachments. The PSD~~s (see Annex C)~~ define the requirements on the need of ProtocolSupportFiles for a specific protocol. However, the configuration of a device using an FDI® Device Package shall not require the usage of a protocol specific attachment. Table 1 specifies the parameters of the ProtocolSupportFile in the FDI® Device Package in case one or many are provided.

Table 1 – ProtocolSupportFile for FDI® Device Packages

Parameter	Description
Content Type	text/plain
Root Namespace	empty
Source Relationship	http://FDI-cooperation.com/2010/relationship/attachment-protocol
Filename	Not defined

5.2.1.2 FDI® Communication Packages

The same rules as for FDI® Device Packages apply.

5.2.2 CommunicationProfile definition

IEC 62769-4 defines a CommunicationProfileT string for the Catalog XML schema. The string is protocol specific and defined as ProfileIdentifier in the PSD (see Annex C).

5.2.3 Profile device

A Profile Package shall provide the catalog values for profile devices, enabling the FDI® Server to leverage a generic device description, if a specific one is not available. The definitions in Table 2 focus on catalog content that is vendor independent.

Table 2 – Catalog values for profile devices

Element	Attribute	Content
PackageType	—	Profile
Manufacturer	—	Empty
DeviceModel	—	The format of the DeviceModel is protocol specific and details on the format are defined in the PSD (see Annex C). In order to assign a scan result with a Profile Package, the ProfileId of the scan result shall be mapped to the DeviceModel of the Profile Package.

5.2.4 Protocol version information

IEC 62769-4 defines an element type named InterfaceT for the Catalog XML schema. The element type InterfaceT contains an element named Version which is supposed to provide version information about the applied communication protocol profile. The value shall follow the IEC 62769-4 defined version information schema defined in the element type VersionT. The PSD (see Annex C) define the mapping of versions of a specific protocol to this field.

5.3 Associating a Package with a device

5.3.1 Device type identification mapping

The purpose of device type identification mapping is to enable FDI® Host systems to compare the scan result against the topology representation in the Information Model. FDI® Host systems shall also be enabled to determine the FDI® Device Package that fits for a device entry contained in the scan result. This will enable the user of an FDI® Host system to synchronize the Information Model with the actual installation.

The Communication Server implemented scan service (defined in 5.6.1.7) provides the scan result through an XML document (the schema is defined in Clause A.5).

The Gateway implemented scan service (defined in 5.6.2.7) provides the scan result by means of the Information Model that contains data structures created from EDD content as specified in 5.6.2.7.

Common for both ways of presenting the scan result is that scan results contain device type identification and device instance identification.

FDI® Host systems comparing the actual network topology configuration against the topology representation in the Information Model shall be enabled to handle the following situations:

a) The physical Device instance identified at a specific device address is not logically present in the Information Model (as Instance): Enable the FDI® Host system to find the appropriate FDI® Device Package according to the device catalog information.
b) The physical Device instance identified by the device address is logically present in the Information Model (as Instance): Enable the FDI® Host system to compare device type information presented in scan result (see the identification in Clause A.5) and the device type specific information of the Instance present in the Information Model.

The FDI® Device Package contains device type identification information that can be compared to scan result based on the Catalog Schema in IEC 62769-4 defining the XML (simple) element types "DeviceModel" and "Manufacturer". Both types are used in the (complex) element types "Protocol" and "RegDeviceType".

As a result of the FDI® Package deployment, the FDI® Package information is then present in the Information Model as the specified FunctionalGroup Identification containing SerialNumber and Tag (see 5.4.3).

The mapping between different device identification data sources is described in Table 3. Since scan results provided by the Communication Server or Gateway can convey data that is produced by the device (firmware), the device type identification mapping shall be supported by providing corresponding data in the FDI® Device Package contained Catalog and Information Model.

Table 3 – Device identification information mapping

FDI® Device Package	Information Model	Communication Server provided scan result	Gateway provided scan result
Catalog specified type Manufacturer	FunctionalGroup: Identification Browse Name: Manufacturer	Element (path): ConnectionPoint/Identification Attribute: Manufacturer	COLLECTION ConnectionPoint. Identification: Manufacturer
Catalog specified type DeviceModel	FunctionalGroup: Identification Browse Name: DeviceModel	Element (path): ConnectionPoint/Identification Attribute: DeviceModel	COLLECTION ConnectionPoint. Identification. DeviceModel

Since not all protocols that are intended to be used with this profile for Generic Protocols might support a mandatory discovery mechanism allowing to identify the type of device (Manufacturer and DeviceModel), the scan results provide the capability to exclude the identification of the device and only provide the address. In that case, some host-specific mechanisms ~~can~~ might be used to assign the desired FDI® package to the device, e.g., by user interaction.

Since some protocols ~~do~~ might not even have mandatory capabilities to identify if there is a device at all for a specific protocol address hosts should provide the capability ~~for~~ that some users ~~to~~ can add devices by manually specifying address information.

5.3.2 Device type revision mapping

IEC 62769-4 envisions a concept that allows to determine the compatibility between an FDI® Device Package and a Device. IEC 62769-4 specifies a life cycle management process bearing on a single version information provided for the entire device. This is captured in the DeviceRevision (see Table 4). The DeviceRevision might be available as single number or as a string. Mapping of version information is protocol-specific and needs to be defined in the PSD (see Annex C).

Table 4 – Device revision information mapping

FDI® Device Package	Information Model	Communication Server provided scan result	Gateway provided scan result
Catalog specified type ListOfSupportedDeviceRevisions	FunctionalGroup: Identification Browse Name: DeviceRevision	Element (path): ConnectionPoint/Identification Attribute: DeviceRevision	COLLECTION ConnectionPoint. Identification. DeviceRevision

5.4 Information Model mapping

5.4.1 ProtocolType definition

In Table 5, a subtype of ProtocolType is defined to identify network communication using this profile.

Table 5 – Protocol type GenericProtocol

Attribute	Value				
BrowseName	GenericProtocol				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the ProtocolType defined in IEC 62541-100.					
HasProperty	Variable	ProtocolIdentifier	String	PropertyType	Mandatory

The mandatory Variable ProtocolIdentifier defines which concrete protocol is represented using the GenericProtocol type. It shall match the ProtocolIdentifier defined for the CommunicationProfile in 5.2.2. The string is protocol specific and defined as ProfileIdentifier in the PSD (see Annex C).

5.4.2 DeviceType mapping

Each device type inherits the properties of DeviceType. The mapping of the inherited properties from DeviceType is defined in Table 6. Note that only the attributes defined in Annex C and therefore expected by each generic protocol are used. Specific protocols ~~can~~ might provide for example a SoftwareRevsion but since this is not accessible for the host, this profile does not make use of it.

Table 6 – Inherited DeviceType property mapping

Property	Generic Protocol Mapping
SerialNumber	SerialNumber (see Annex C)
RevisionCounter	-1 (not defined)
Manufacturer	String taken from FDI® package catalog (ManufacturerName from PackageT)

Model	String taken from FDI® package catalog (Name of DeviceTypeT, which is a localized name)
DeviceManual	empty text string (not supported) [a]
DeviceRevision	DeviceRevision (see Annex C)
SoftwareRevision	empty string (not defined)
HardwareRevision	empty string (not defined)
[a] Device manuals are exposed as attachments of the FDI® Device Package.	

5.4.3 FunctionalGroup identification definition

As defined in IEC 62541-100:2015, 5.3, each device representation in the FDI® Server hosted Information Model shall contain a protocol specific FunctionalGroup named Identification. The Parameters of this FunctionalGroup are defined for generic protocol device types as follows:

Table 7 – Generic Protocol Device Types identification attributes

BrowseName	DataType	Mandatory/Optional
Manufacturer	String	Mandatory
DeviceModel	String	Mandatory
SerialNumber	String	Optional
Tag	String	Optional
DeviceRevision	UInt16	Optional
ExtendedDeviceRevision	String	Optional
ProfileId	String	Optional

The BaseDataVariable instances shall be created from VARIABLE declarations with identifiers that correspond to the browse names listed in Table 7.

5.5 Topology elements

5.5.1 ConnectionPoint definition

The ConnectionPoint type GenericConnectionPoint shall be used to parameterize network access points using the Generic Protocols. The ConnectionPoint type GenericConnectionPoint is a sub type of the abstract type ConnectionPointType defined in IEC 62541-100. Table 8 specifies the representation of the GenericConnectionPoint in the AddressSpace.

Table 8 – ConnectionPoint type for Generic Protocols

Attribute	Value				
BrowseName	GenericConnectionPoint				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Sub type of the ConnectionPointType defined in IEC 62541-100.					
HasProperty	Variable	Address	String	PropertyType	Mandatory
HasProperty	Variable	ProtocolIdentifier	String	PropertyType	Mandatory

The ConnectionPoint type GenericConnectionPoint shall be described by an EDD element contained in a Communication Device related FDI® Package that can drive a generic protocol network. Actual ConnectionPoint properties are declared by VARIABLE constructs grouped together in a COLLECTION named ConnectionPoint. For this profile, it shall only contain the CONNECTION_POINT_ADDRESS, mapped to the OPC UA Property Address. In addition, the PROTOCOL specified by the COMPONENT shall be mapped to the ProtocolIdentifier Property. The following EDDL source code is an example describing a Connection Point for an ExampleProtocol. The ProtocolIdentifier defined by the PSD (see Annex C) shall be used as PROTOCOL name in the EDD.

```
COMPONENT ConnectionPoint_Generic
{
  LABEL "Generic Connection Point";
  CLASSIFICATION NETWORK_CONNECTION_POINT;
  CAN_DELETE FALSE;
  PROTOCOL ExampleProtocol;
  CONNECTION_POINT ConnectionPoint;
}

VARIABLE Address
{
  LABEL "Address";
  HELP "Address of the device";
  TYPE EUC(<protocol-specific>);
  CLASS LOCAL;
}


COLLECTION ConnectionPoint
{
  LABEL "Connection Point";
  MEMBERS
  {
    CONNECTION_POINT_ADDRESS, Address;
  }
}
```

5.5.2 Communication Device definition

According to IEC 62769-7, each FDI® Communication Package shall contain an EDD element describing the communication device. The following EDDL source code is an example describing a Communication Server.

```
COMPONENT Generic_Communication_Server
{
  LABEL "Generic communication server";
  PRODUCT_URI "urn:Company Name:Product Name";
  CAN_DELETE TRUE;
  CLASSIFICATION NETWORK_COMPONENT;
  COMPONENT_RELATIONS
  {
    Generic_Communication_Device_Setup
  }
}

COMPONENT_RELATION Generic_Communication_Device_Setup
{
   LABEL "Relation between Device and communication device";
   RELATION_TYPE CHILD_COMPONENT;
   COMPONENTS
   {
       Generic_Communication_Device{AUTO_CREATE 1;}
   }
   MINIMUM_NUMBER 1;
   MAXIMUM_NUMBER 4;
}
```

According to IEC 62769-7, each FDI® Communication Package shall contain at least one EDD element describing at least one communication device component. The following EDDL source code in is an example for a generic protocol communication device:

```
COMPONENT Generic_Communication_Device
{
  LABEL "Generic communication device";
  CAN_DELETE TRUE;
  CLASSIFICATION NETWORK_COMPONENT;
  COMPONENT_RELATIONS { Generic_Service_Provider_Relation }
}

COMPONENT_RELATION Generic_Service_Provider_Relation
{
   LABEL "Relation to communication service provider";
   RELATION_TYPE CHILD_COMPONENT;
   COMPONENTS
   {
       Generic_Service_Provider{AUTO_CREATE 1;}
   }
   MINIMUM_NUMBER 1;
   MAXIMUM_NUMBER 1;
}
```

In an actual communication device, the ConnectionPoint_Generic needs to be adapted according to the supported protocol and the related connection point definitions given in 5.5. The attribute BYTE_ORDER shall not be used for this profile as the byte order handling shall be done in the gateway business logic.

5.5.3 Communication service provider definition

According to IEC 62769-7, each FDI® Communication Package shall contain at least one EDD element describing at least one communication service provider component. The following EDDL source code below is an example for a generic protocol communication service provider component.

The component reference (ConnectionPoint_Generic) corresponds to the related connection point definition in 5.5. The attribute BYTE_ORDER shall not be used for this profile as the byte order handling shall be done in the gateway business logic.

```
COMPONENT Generic_Service_Provider
{
  LABEL "Generic Protocol communication service provider";
  CAN_DELETE TRUE;
  CLASSIFICATION NETWORK_COMMUNICATION_SERVICE_PROVIDER;
  COMPONENT_RELATIONS
  {
    Generic_Service_Provider_Connection_Point_Relation
  }
}


COMPONENT_RELATION Generic_Service_Provider_Connection_Point_Relation
{
   LABEL "Relation between communication service provider and Connection Point";
   RELATION_TYPE CHILD_COMPONENT;
   ADDRESSING {Address}
   COMPONENTS
   {
       ConnectionPoint_Generic{ AUTO_CREATE 1;}
   }
   MINIMUM_NUMBER 1;
   MAXIMUM_NUMBER 1;
}
```

5.5.4 Network definition

According to IEC 62769-7, each FDI® Communication Package shall contain at least one EDD element describing network configuration constraints using the component construct.

```
COMPONENT Network_Generic
{
    LABEL "Generic Network";
    CAN_DELETE TRUE;
    CLASSIFICATION NETWORK;
    COMPONENT_RELATIONS
    {
      Generic_Network_Connection_Point_Relation
    }
}

COMPONENT_RELATION Generic_Network_Connection_Point_Relation
{
   LABEL "Relation between network and Connection Point";
   RELATION_TYPE CHILD_COMPONENT;
   ADDRESSING {Address}
   COMPONENTS
   {
       ConnectionPoint_Generic
   }
   MINIMUM_NUMBER 1;
   MAXIMUM_NUMBER 32;
}
```

5.6 Methods

5.6.1 Methods for FDI® Communication Servers

5.6.1.1 General

The Communication Server contained Information Model shall implement services according to method signatures described in 5.6.1.

5.6.1.2 Connect

Signature:

```
Connect(
    [in]   ByteString       CommunicationRelationId,
    [in]   String           Address,
    [out]  Int32            ServiceError);
```

Table 9 provides the description of the arguments.

Table 9 – Method Connect arguments

Argument	Description
CommunicationRelationId	The argument value contains the nodeId of the ConnectionPoint representing the connection between a device and a physical network which is directly connected to the Communication Server hardware. The nodeId allows finding the direct parent-child relation.
Address	The argument name shall match with the corresponding attribute name defined for the ConnectionPoint which is described by a corresponding EDD element specified in 5.5. The argument value holds the protocol-specific device’s network address that is unique within the network segment.
ServiceError	0: OK / execution finished, connection established successfully -1: Connect Failed / cancelled by caller -3: Connect Failed / device not found -4: Connect Failed / invalid device address -5: Connect Failed / invalid device identification

5.6.1.3 Disconnect

Signature:

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

Table 10 provides the description of the arguments.

Table 10 – Method Disconnect arguments

Argument	Description
CommunicationRelationId	The argument value contains the nodeId of the ConnectionPoint representing the connection between a device and a physical network which is directly connected to the Communication Server hardware. The nodeId allows finding the direct parent-child relation.
ServiceError	0: OK / disconnect finished successfully -1: Disconnect Failed / no existing communication relation -2: Disconnect Failed / invalid communication relation identifier

5.6.1.4 Transfer

Signature:

```
Transfer(
    [in]  ByteString       CommunicationRelationId,
    [in]  String           Header,
    [in]  ByteString       RequestData,
    [in]  EddDataTypeInfo[] RequestDataTypes,
    [in]  EddDataTypeInfo[] ResponseDataTypes,
    [out] ByteString       ResponseData,
    [out] ByteString       RESPONSE_CODES,
    [out] Int32            ServiceError);
```

Table 11 provides the description of the arguments.

Table 11 – Method Transfer arguments

Argument	Description
CommunicationRelationId	The argument value contains the nodeId of the ConnectionPoint representing the connection between a device and a physical network within the Information Model.
Header	The protocol-specific information on the data to be transferred. The Header is used in the COMMANDs of the EDD and the PSD (see Annex C) defines the format of this string per protocol. The first communication service treating this string shall validate the string according to its format defined in the PSD.
RequestData	The argument name shall match with the corresponding COMMAND sub-element name REQUEST. The byte stream submitted trough the argument is created from definitions provided by the REQUEST element of the corresponding COMMAND that shall be processed.
RequestDataTypes	This array contains information on what data types are used to create the RequestData ByteString The byte order used in the RequestData is EDD specific and providing the information allows the gateway or communication server to change it to a protocol-specific byte order of the data type or applying any other protocol-specific characteristics (e.g. bit order). ~~The overall length of all provided data types in the RequestDataTypes shall match the length of the RequestData.~~ An empty array can be provided if the protocol's byte order matches the EDD specific byte order. If the protocol's byte order does not match the EDD specific byte order, and the provided data types match only part of the RequestData data, the remaining data is considered to require no adaption with respect to the byte order.
ResponseDataTypes	This array contains information on what data types are expected when receiving the ResponseData ByteString. The knowledge about the expected data types is in the EDD of the device and therefore the gateway or communication server needs to get this information in order to apply protocol-specific characteristics to the response data and knows how to separate the response data into an array of ByteString. ~~The overall length of all provided data types in the ResponseDataTypes shall match the length of the ResponseData.~~ An empty array can be provided if the protocol's byte order matches the EDD specific byte order. If the protocol's byte order does not match the EDD specific byte order, and the provided data types match only part of the ResponseData, the remaining data is considered to require no adaption with respect to the byte order.
ResponseData	The argument name shall match with the corresponding COMMAND sub-element name REPLY. The byte stream returned by this argument applies to definitions provided by the REPLY element of the corresponding COMMAND that shall be processed.
RESPONSE_CODES	The argument name shall match with the COMMAND sub-element name RESPONSE_CODES. The argument value conveys the specific communication service response bytes.
ServiceError	0: OK / execution finished -1: Transfer Failed / cancelled by caller -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

The EddDataTypeInfo DataType defines the data type information of an EDD data type used in a COMMAND. Its elements are defined in Table 12.

Table 12 – EddDataTypeInfo DataType Structure

Name	Type	Description
EddDataTypeInfo	structure	This structure specifies information about a data type used in an EDD COMMAND.
eddDataType	EddDataTypeEnum	The EddDataType used.
size	UInt32	The size of the eddDataType. It shall always be provided, even if not explicitly specified by the EDD. In that case the default value for that data type shall be provided. For data types where no size can be defined (e.g. BOOLEAN), the value shall be set to 0.

The EddDataTypeEnum DataType is an enumeration that defines the possible EDD data types. Its values are defined in Table 13.

Table 13 – EddDataTypeEnum Values

Value	Description
BOOLEAN	Data type as defined by IEC 61804-3
DOUBLE	Data type as defined by IEC 61804-3
FLOAT	Data type as defined by IEC 61804-3
INTEGER	Data type as defined by IEC 61804-3
UNSIGNED_INTEGER	Data type as defined by IEC 61804-3
DATE	Data type as defined by IEC 61804-3
DATE_AND_TIME	Data type as defined by IEC 61804-3
DURATION	Data type as defined by IEC 61804-3
TIME	Data type as defined by IEC 61804-3
TIME_VALUE	Data type as defined by IEC 61804-3
BIT_ENUMERATED	Data type as defined by IEC 61804-3
ENUMERATED	Data type as defined by IEC 61804-3
ASCII	Data type as defined by IEC 61804-3
BITSTRING	Data type as defined by IEC 61804-3
EUC	Data type as defined by IEC 61804-3
OCTET	Data type as defined by IEC 61804-3
PACKED_ASCII	Data type as defined by IEC 61804-3
PASSWORD	Data type as defined by IEC 61804-3
VISIBLE	Data type as defined by IEC 61804-3

5.6.1.5 GetPublishedData

This method is not supported by the Generic Protocols profile.

5.6.1.6 SetAddress

Signature:

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

Table 14 provides the description of the arguments.

Table 14 – Method SetAddress arguments

Argument	Description
OldAddress	The argument value holds the current address of a device. The same semantic applies to this field as the Address parameter in the Connect method.
NewAddress	The argument value holds the new address for a device. The same semantic applies to this field as the Address parameter in the Connect method.
ServiceError	0: OK / execution finished successfully -1: SetAddress Failed / cancelled by caller -3: SetAddress Failed / not initialized -4: SetAddress Failed / not connected to a network -5: SetAddress Failed / no device found responding to oldAddress -6: SetAddress Failed / duplicate address error -7: SetAddress Failed / device did not accept new address -8: SetAddress Failed / invalid oldAddress (in terms of syntax, data type, data format, and so on) -9: SetAddress Failed / invalid newAddress (in terms of syntax, data type, data format, and so on) -10: SetAddress Failed / not possible in status connected

5.6.1.7 Scan

The method signature specified in IEC 62769-7 applies. The corresponding topologyScanResult schema is specified in Annex A, whereas this method returns the identification data with the DeviceRevision as unsignedShort, see Annex C.

5.6.1.8 ScanExtended

The method signature specified in IEC 62769-7 applies. The corresponding topologyScanResult schema is specified in Annex A, whereas this method returns the identification data with the ExtendedDeviceRevision, see Annex C.

5.6.1.9 ResetScan

The method signature specified in IEC 62769-7 applies.

5.6.2 Methods for Gateways

5.6.2.1 General

The methods signatures defined in 5.6.2 apply. The methods shall be implemented in the EDD element (IEC 62769-4) contained in a Gateway related FDI® Package containing the communication device definitions.

5.6.2.2 Connect

Subclause 5.6.2.2 describes the generic protocol specific implementation of the service Connect specified in IEC 62769-7.

```
METHOD BeginConnect(
   DD_STRING               CommunicationRelationId,
   DD_STRING               Address,
   unsigned long           ServiceId,
   unsigned long           &DelayForNextCall,
   long                    &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}

METHOD EndConnect(
   DD_STRING               CommunicationRelationId,
   unsigned long           ServiceId,
   unsigned long           &DelayForNextCall,
   long                    &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}

METHOD CancelConnect(
   DD_STRING               CommunicationRelationId,
   unsigned long           ServiceId,
   long                    &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}
```

Table 15 provides the description of the arguments.

Table 15 – Connect service arguments

Argument	Description
CommunicationRelationId	The argument value contains the nodeId of the ConnectionPoint representing the connection between a device and a physical network which is directly connected to the Communication Server hardware. The nodeId allows finding the direct parent-child relation.
Address	The argument name shall match with the corresponding attribute name defined for the ConnectionPoint which is described by a corresponding EDD element specified in 5.5. The argument value holds the protocol-specific device's network address that is unique within the network segment.
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 to limit the EndConnect invocation cycle that shall not be faster than specified in the argument value.
ServiceError	1: OK / function started asynchronously, result has to be polled with EndConnect 0: 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

5.6.2.3 Disconnect

Subclause 5.6.2.3 describes the generic protocol specific implementation of the service Disconnect specified in IEC 62769-7.

```
METHOD Disconnect(
   DD_STRING CommunicationRelationId,
   long   &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}
```

All the arguments of the Disconnect method are described in Table 10.

5.6.2.4 Transfer

Subclause 5.6.2.4 describes the generic protocol specific implementation of the service Transfer specified in IEC 62769-7. Due to limitations in the capabilities of EDD methods (it is not possible to transfer any collections of a dynamic size), this profile requires some additional handling when using the transfer via Gateways. In addition to the calls of the EDD method, a host first needs to provide the data type information of the request and the response data. The following order needs to be executed by the host:

1) Add the request and response data type information into a list defined by the gateway EDD (called DATA_TYPE_INFO_PER_SERVICE_ID). The entry shall include the ServiceId also used in the method calls.
2) Call BeginTransfer using the same ServiceId and follow the asynchronous call pattern as defined in IEC 62769-7.

3) When the host has received the response, it is the responsibility of the host to remove the entry from the list. Thus, only the host is manipulating the list and the gateway business logic is only accessing but not changing the list in order to get the information on the data types.

```
METHOD BeginTransfer(
   DD_STRING                CommunicationRelationId,
   DD_STRING                Header,
   DD_STRING                RequestData,
   unsigned long            ServiceId,
   DD_STRING                &ResponseData,
   DD_STRING                &RESPONSE_CODES,
   long                     &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}

METHOD EndTransfer(
   DD_STRING                CommunicationRelationId,
   unsigned long            ServiceId,
   DD_STRING                &ResponseData,
   DD_STRING                &RESPONSE_CODES,
   unsigned long            &DelayForNextCall,
   long                     &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}

METHOD CancelTransfer(
   DD_STRING                CommunicationRelationId,
   DD_STRING                &ResponseData,
   DD_STRING                &RESPONSE_CODES,
   unsigned long            ServiceId,
   long                     &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}
```

Table 16 provides the description of the arguments.

Table 16 – Method Transfer arguments

Argument	Description
CommunicationRelationId	The argument value contains the nodeId of the ConnectionPoint representing the connection between a device and a physical network within the Information Model.
Header	The protocol-specific information on the data to be transferred. The Header is used in the COMMANDs of the EDD and the PSD (see Annex C) defines the format of this string per protocol. The first communication service treating this string shall validate the string according to its format defined in the PSD.
RequestData	The argument name shall match with the corresponding COMMAND sub-element name REQUEST. The byte stream submitted trough the argument is created from definitions provided by the REQUEST element of the corresponding COMMAND that shall be processed. The byte stream is separated in several byte streams for the different requested data types. This allows to consider a protocol-specific byte order of specific data types in the gateway or communication server.
ResponseData	The argument name shall match with the corresponding COMMAND sub-element name REPLY. The byte stream returned by this argument applies to definitions provided by the REPLY element of the corresponding COMMAND that shall be processed. The data are separated into several byte strings based on the ResponseDataTypes field. This allows the gateway or communication server to consider protocol-specific byte order of data types.
RESPONSE_CODES	The argument name shall match with the COMMAND sub-element name RESPONSE_CODES. The argument value conveys the specific communication service response bytes.
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 to limit the EndTransfer invocation cycle that shall not be faster than specified in the argument value.
ServiceError	1: OK / function started asynchronously, result has to be polled with EndTransfer 0: OK / execution finished -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

The List that is used to add the data type information so the business logic of the gateway can do the needed data type transfers shall be called DATA_TYPE_INFO_LIST. It is defined in the following:

```
VARIABLE EddDataTypeEnum
{
  LABEL "Edd Data Type";
  CLASS LOCAL;
    TYPE ENUMERATED
    {
                { 0x00, "BOOLEAN" }
                { 0x01, "DOUBLE" }
                { 0x02, "FLOAT" }
                { 0x03, "INTEGER" }
                { 0x04, "UNSIGNED_INTEGER" }
                { 0x05, "DATE" }
                { 0x06, "DATE_AND_TIME" }
                { 0x07, "DURATION" }
                { 0x08, "TIME" }
                { 0x09, "TIME_VALUE" }
                { 0x0a, "BIT_ENUMERATED" }
                { 0x0b, "ENUMERATED" }
                { 0x0c, "ASCII" }
                { 0x0d, "BITSTRING" }
                { 0x0e, "EUC" }
                { 0x10, "OCTET" }
                { 0x11, "PACKED_ASCII" }
                { 0x12, "PASSWORD" }
                { 0x13, "VISIBLE" }
     }
}

VARIABLE SizeV
{
  LABEL "Size of data type";
  TYPE UNSIGNED_INTEGER(4);
  CLASS LOCAL;
}

COLLECTION DataTypeInfo
{
  MEMBERS
  {
    Size, SizeV;
    EddDataType, EddDataTypeEnum;
  }
}

LIST DataTypeInfoList
{
  TYPE DataTypeInfo;
}

VARIABLE ServiceIdentification
{
  LABEL "Identification of service call";
  TYPE UNSIGNED_INTEGER(4);
  CLASS LOCAL;
}

COLLECTION DataTypeInfoPerServiceId
{
  MEMBERS
  {
    ServiceId, ServiceIdentification;
    RequestDataTypes, DataTypeInfoList;
    ResponseDataTypes, DataTypeInfoList;
  }
}
LIST DATA_TYPE_INFO_PER_SERVICE_ID
{
  TYPE DataTypeInfoPerServiceId;
}
```

5.6.2.5 GetPublishedData

This method is not supported in the generic protocol profile.

5.6.2.6 SetAddress

Subclause 5.6.2.6 describes the generic protocol specific implementation of the service SetAddress specified in IEC 62769-7.

```
BeginSetAddress(
   DD_STRING                 OldAddress,
   DD_STRING                 NewAddress,
   unsigned long             ServiceId,
   unsigned long             &DelayForNextCall,
   long                      &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}

EndSetAddress(
   unsigned long             ServiceId,
   unsigned long             &DelayForNextCall,
   long                      &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}
```

Table 17 provides the description of the arguments.

Table 17 – Method SetAddress arguments

Argument	Description
OldAddress	The argument value holds the current address of a device. The same semantic applies to this field as the Address parameter in the Connect method.
NewAddress	The argument value holds the new address for a device. The same semantic applies to this field as the Address parameter in the Connect method.
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 to limit the EndTransfer invocation cycle that shall not be faster than specified in the argument value.
ServiceError	1: OK / function started asynchronously, result has to be polled with EndSetAddress 0: OK / execution finished successfully -1: SetAddress Failed / cancelled by caller -2: Call Failed / unknown service ID -3: SetAddress Failed / not initialized -4: SetAddress Failed / not connected to a network -5: SetAddress Failed / no device found responding to oldAddress -6: SetAddress Failed / duplicate address error -7: SetAddress Failed / device did not accept new address -8: SetAddress Failed / invalid oldAddress (in terms of syntax, data type, data format, and so on) -9: SetAddress Failed / invalid newAddress (in terms of syntax, data type, data format, and so on) -10: SetAddress Failed / not possible in status connected

5.6.2.7 Scan

The method signature specified in IEC 62769-7 applies. The generic protocol gateway business logic shall create the scan result following IEC 62769-7. The following definitions shall be present in the COMPONENT declaration that holds the definitions for a communication device. The data structure corresponds to the data structure defined in the XML schema in Annex A. The SCAN_LIST attribute inside the COMPONENT declaration shall refer to LIST TopologyScanResult.

```
VARIABLE GenericAddress
{
  LABEL "Station address";
  TYPE EUC(<protocol-specific>);
  CLASS LOCAL;
}

VARIABLE Manufacturer
{
  LABEL "Manufacturer ID";
  TYPE EUC(<protocol-specific>);
  CLASS LOCAL;
}

VARIABLE DeviceModel
{
  LABEL "Device ID";
  TYPE EUC(<protocol-specific>);
  CLASS LOCAL;
}
```

```
VARIABLE DeviceRevision
{
  LABEL "Device revision";
  TYPE UNSIGNED_INTEGER(2);
  CLASS LOCAL;
}

VARIABLE SerialNumber
{
  LABEL "Serial number";
  TYPE EUC(<protocol-specific>);
  CLASS LOCAL;
}

VARIABLE Tag
{
  LABEL "Tag";
  TYPE EUC(<protocol-specific>);
  CLASS LOCAL;
}

VARIABLE ProfileId
{
  LABEL "ProfileId";
  TYPE EUC(<protocol-specific>);
  CLASS LOCAL;
}


COLLECTION ScanItemType
{
  MEMBERS
  {
    ADDRESS, GenericAddress;
    MANUFACTURER_ID, Manufacturer;
    DEVICE_MODEL, DeviceModel;
    DEVICE_REV, DeviceRevision;
    SERIAL_NUMBER, SerialNumber;
    TAG, Tag;
    PROFILEID, ProfileId;
  }
}
LIST TopologyScanResult
{
  TYPE ScanItemType;
}
```

5.6.2.8 ScanNext

The method signature specified in IEC 62769-7 applies. The generic gateway business logic shall create the scan result following IEC 62769-7. The method ScanNext stores the result into data structures described for the method Scan (5.6.2.7).

~~4.6.3 Transfer service parameters~~

~~Transfer service parameters are defined in Annex B.~~

Annex A
(normative)

Topology scan result schema

A.1 General

The topology scan result schema specified in Annex A describes the Generic Protocols specific format Method Scan argument `topologyScanResult`. The XML document content and structure shall correspond to the Information Model designed concept to describe a topology in order to enable generic matching between physical devices connected to the network and the FDI® Server hosted Information Model.

A.2 Network

The subsequent element is used to return the scan result corresponding to the Information Model described in IEC 62769-5.

The XML schema for a Network element is:

```
<xs:element name="Network" type="GP:GenericNetworkT"/>
```

A.3 GenericNetworkT

The element type describes the complete scan result for a single network because of the scan method that is provided per instance of a "Communication Device" which exists in 1:1 relation to a network instance.

The XML schema for a GenericNetworkT type is:

```
<xs:complexType name="GenericNetworkT">
   <xs:sequence>
      <xs:element name="ConnectionPoint" type="GP:GenericConnectionPointT"
         maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

The elements of a GenericNetworkT type are described in Table A.1.

Table A.1 – Elements of GenericNetworkT

Element	Description
ConnectionPoint	The ConnectionPoint element holds the address and identification of the network connected device that has been found during bus scan operations.

A.4 GenericConnectionPointT

The XML schema for a GenericConnectionPointT type is:

```
<xs:complexType name="GenericConnectionPointT">
   <xs:sequence>
      <xs:choice minOccurs=”0”>
         <xs:element name="Identification" type="GP:GenericIdentificationT~~"~~
         ~~minOccurs="0~~"/>
         <xs:element name="IdentificationExt"
         type="GP:GenericIdentificationExtendedT"/>
      </xs:choice>
   </xs:sequence>
   <xs:attribute name="Address" type="GP:GenericAddressT" use="required"/>
</xs:complexType>
```

The attributes of a GenericConnectionPointT type are described in Table A.2.

Table A.2 – Attributes of GenericConnectionPointT

Attribute	Description
Address	The attribute value holds the address of the network connected device used to uniquely address the device in the network.

The elements of a GenericConnectionPointT type are described in Table A.3.

Table A.3 – Elements of GenericConnectionPointT

Element	Description
Identification	The element data holds the device type identification data. Compared to the Information Model (IEC 62541-100), the ConnectionPoint does not contain or refer to the device type identification data. But in order to support the FDI® Host system in finding the package that matches the connected device, this schema associates the device type identification with the ConnectionPoint. This identification may contain the DeviceRevision as unsignedShort or as String in the minor.major.revision format. Note that this element is optional since ~~it is possible that~~ a protocol ~~does~~ might not support identification of a device.

A.5 GenericIdentificationT

The element content corresponds to the "FunctionalGroup Identification".

The XML schema for a GenericIdentificationT type is:

```
<xs:complexType name="GenericIdentificationT">
   <xs:attribute name="Manufacturer" type="xs:string" use="required"/>
   <xs:attribute name="DeviceModel" type="xs:string" use="required"/>
   <xs:attribute name="SerialNumber" type="xs:string" use="optional"/>
   <xs:attribute name="Tag" type="xs:string" use="optional"/>
   <xs:attribute name="DeviceRevision" type="xs:unsignedShort"
      use="optional"/>
   <xs:attribute name="ProfileId" type="xs:string" use="optional"/>
</xs:complexType>
```

The attributes of a GenericIdentificationT type are described in Table A.4.

Table A.4 – Attributes of GenericIdentificationT

Attribute	Description
Manufacturer	See Table 7
DeviceModel	See Table 7
SerialNumber	See Table 7
Tag	See Table 7
DeviceRevision	See Table 7
ProfileId	See Table 7

A.6 GenericAddressT

The protocol-specific address used to uniquely identify a device in a network segment.

The XML schema for a GenericAddressT type is:

```
<xs:simpleType name="GenericAddressT">
  <xs:restriction base="xs:string"/>
</xs:simpleType>
```

A.7 GenericIdentificationExtendedT

The element content corresponds to the "FunctionalGroup Identification". This type includes the extended DeviceRevision information in the major.minor.revision format as defined in IEC 62769-151-1.

The XML schema for a GenericIdentificationExtendedT type is:

```
<xs:complexType name="GenericIdentificationT">
  <xs:attribute name="Manufacturer" type="xs:string" use="required"/>
  <xs:attribute name="DeviceModel" type="xs:string" use="required"/>
  <xs:attribute name="SerialNumber" type="xs:string" use="optional"/>
  <xs:attribute name="Tag" type="xs:string" use="optional"/>
  <xs:attribute name="ExtendedDeviceRevision" type="FDI:VersionT"
    use="optional"/>
  <xs:attribute name="ProfileId" type="xs:string" use="optional"/>
</xs:complexType>
```

The attributes of a GenericIdentificationExtendedT type are described in Table A.5.

Table A.5 – Attributes of GenericIdentificationExtendedT

Attribute	Description
Manufacturer	See Table 7
DeviceModel	See Table 7
SerialNumber	See Table 7
Tag	See Table 7
ExtendedDeviceRevision	See Table 7, ExtendedDeviceRevision in major.minor.revision format as defined in IEC 62769-151-1
ProfileId	See Table 7

Annex B
(normative)

Transfer service parameters

B.1 General

Direct Access Services specified in IEC 62769-2 enable the User Interface Plug-in (UIP) to directly exchange data with the device. Direct data exchange means that data exchanged between a device and a UIP ~~may not~~ cannot be reflected in the Information Model. The IEC 62769-5 defined interface IDirectAccess corresponds to the IEC 62769-2 specified Direct Access Services. Interface IDirectAccess defined functions BeginTransfer and EndTransfer need to convey protocol specific information. The protocol specifics shall be ~~captured~~ encapsulated in the XML document defined by this schema.

B.2 sendData

The element described in the following contains data to be submitted through the IDirectAccess function BeginTransfer defined argument sendData.

The XML schema for a sendData element is:

```
<xs:element name="sendData" type="GP:TransferSendDataT"/>
```

B.3 receiveData

The element described in the following contains data that is returned through the ~~return value defined in EndTransfer in the~~ IDirectAccess function EndTransfer defined return value.

The XML schema for a receiveData element is:

```
<xs:element name="receiveData" type="GP:TransferResultDataT"/>
```

B.4 TransferSendDataT

A complex type that defines the service parameter data format that shall be applied to Transfer defined argument sendData.

The XML schema for a TransferSendDataT type is:

```
<xs:complexType name="TransferSendDataT">
   <xs:sequence>
      <xs:element name="RequestDataTypes" type="GP:EddDataTypeInfoListT"/>
      <xs:element name="ResponseDataTypes" type="GP:EddDataTypeInfoListT"/>
   </xs:sequence>
   <xs:attribute name="Header" type="xs:string" use="required"/>
   <xs:attribute name="RequestData" type="xs:hexBinary" use="required"/>
</xs:complexType>
```

The attributes of a TransferSendDataT type are described in Table B.1.

Table B.1 – Attributes of TransferSendDataT

Attribute	Description
Header	The attribute corresponds to the Transfer method argument Header.
RequestData	The attribute corresponds to the Transfer method argument RequestData.

The elements of a TransferSendDataT type are described in Table B.2.

Table B.2 – Elements of TransferSendDataT

Element	Description
RequestDataTypes	The element corresponds to the Transfer method argument RequestDataTypes
ResponseDataTypes	The element corresponds to the Transfer method argument ResponseDataTypes

B.5 EddDataTypeInfoListT

List of EDD data type information.

The XML schema for a EddDataTypeInfoListT type is:

```
<xs:complexType name="EddDataTypeInfoListT">
  <xs:sequence>
    <xs:element name="EddDataTypeInfo" type="GP:EddDataTypeInfoT"
      maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

The elements of a EddDataTypeInfoListT type are described in Table B.3.

Table B.3 – Elements of EddDataTypeInfoListT

Element	Description
EddDataTypeInfo	Describes an individual EDD data type and its size.

B.6 EddDataTypeInfoT

Describes an individual EDD data type and its size.

The XML schema for a EddDataTypeInfoT type is:

```
<xs:complexType name="EddDataTypeInfoT">
  <xs:attribute name="EddDataType" type="GP:EddDataTypeT" use="required"/>
  <xs:attribute name="Size" type="xs:nonNegativeInteger" use="required"/>
</xs:complexType>
```

The attributes of a EddDataTypeInfoT type are described in Table B.4.

Table B.4 – Attributes of EddDataTypeInfoT

Attribute	Description
EddDataType	The EddDataType used.
Size	The size of the eddDataType. It shall always be provided, even if not explicitly specified by the EDD. In that case the default value for that data type shall be provided. For data types where no size can be defined (e.g. BOOLEAN), the value shall be set to 0.

B.7 EddDataTypeT

A simple type that defines the possible EDD data types.

The XML schema for a EddDataTypeT enumeration type is:

```
<xs:simpleType name="EddDataTypeT">
   <xs:restriction base="xs:string">
      <xs:enumeration value="BOOLEAN"/>
      <xs:enumeration value="DOUBLE"/>
      <xs:enumeration value="FLOAT"/>
      <xs:enumeration value="INTEGER"/>
      <xs:enumeration value="UNSIGNED_INTEGER"/>
      <xs:enumeration value="DATE"/>
      <xs:enumeration value="DATE_AND_TIME"/>
      <xs:enumeration value="DURATION"/>
      <xs:enumeration value="TIME"/>
      <xs:enumeration value="TIME_VALUE"/>
      <xs:enumeration value="BIT_ENUMERATED"/>
      <xs:enumeration value="ENUMERATED"/>
      <xs:enumeration value="ASCII"/>
      <xs:enumeration value="BITSTRING"/>
      <xs:enumeration value="EUC"/>
      <xs:enumeration value="OCTET"/>
      <xs:enumeration value="PACKED_ASCII"/>
      <xs:enumeration value="PASSWORD"/>
      <xs:enumeration value="VISIBLE"/>
   </xs:restriction>
</xs:simpleType>
```

The enumeration values of a EddDataTypeT enumeration type are described in Table B.5.

Table B.5 – Enumerations of EddDataTypeT

Enumeration	Description
BOOLEAN	Data type as defined in IEC 61804-3
DOUBLE	Data type as defined in IEC 61804-3
FLOAT	Data type as defined in IEC 61804-3
INTEGER	Data type as defined in IEC 61804-3
UNSIGNED_INTEGER	Data type as defined in IEC 61804-3
DATE	Data type as defined in IEC 61804-3
DATE_AND_TIME	Data type as defined in IEC 61804-3
DURATION	Data type as defined in IEC 61804-3
TIME	Data type as defined in IEC 61804-3
TIME_VALUE	Data type as defined in IEC 61804-3
BIT_ENUMERATED	Data type as defined in IEC 61804-3
ENUMERATED	Data type as defined in IEC 61804-3
ASCII	Data type as defined in IEC 61804-3
BITSTRING	Data type as defined in IEC 61804-3
EUC	Data type as defined in IEC 61804-3
OCTET	Data type as defined in IEC 61804-3
PACKED_ASCII	Data type as defined in IEC 61804-3
PASSWORD	Data type as defined in IEC 61804-3
VISIBLE	Data type as defined in IEC 61804-3

B.8 TransferResultDataT

A complex type that defines the service parameter data format that shall be applied to Transfer defined receivedData return value.

```
<xs:complexType name="TransferResultDataT">
   <xs:attribute name="ResponseData" type="xs:hexBinary" use="required"/>
   <xs:attribute name="RESPONSE_CODES" type="xs:hexBinary" use="required"/>
</xs:complexType>
```

The attributes of a TransferResultDataT type are described in Table B.6.

Table B.6 – Attributes of TransferResultDataT

Attribute	Description
ResponseData	The attribute corresponds to the Transfer method argument ResponseData.
RESPONSE_CODES	The attribute corresponds to the Transfer method argument RESPONSE_CODES.

Annex C
(normative)

Protocol specific definitions

C.1 General

This annex defines what the PSD need to define per protocol.

C.2 Header

The HEADER attribute is used in an EDD to define the addressing of a COMMAND. When using this annex, each EDD ~~needs to~~ shall comply with the profile for generic protocol extensions defined in IEC 61804-3 and define COMMANDs in the EDD using the HEADER as addressing in order to access the corresponding VARIABLEs in the device. The HEADER consists of a string. Per protocol~~, the way in which~~ it needs to be defined how the protocol-specific addressing is mapped to the HEADER attribute ~~needs to be defined~~. Examples of protocol-specific addressing is the COMMAND number in HART®[3] or SLOT and INDEX in PROFIBUS®[4].

The first communication service treating this string shall validate the string according to its format defined in the PSD.

C.3 ProtocolIdentifier

The ProtocolIdentifier is used in the Package Catalog defined in IEC 62769-4 (called CommunicationProfile) as well as in the EDD for Gateways defined in IEC 61804-3 (called PROTOCOL) to identify the protocol. It is represented as a string in the EDD and in the Package Catalog. In order to have a unique identifier per protocol, the ProtocolIdentifier needs to be defined once per protocol and a URI syntax should be used in order to guarantee the uniqueness of the string.

C.4 Address

The address uniquely identifies a device in a network segment and is provided in the scan result as well as connect and SetAddress methods of an FDI® Communication Server or a Gateway. This profile uses an OPC UA String, XML String or an EDD EUC or DD_STRING in EDD methods as data structure in order to allow high flexibility on the actual used protocol-specific address as well as an easy way to edit an address by the user. For each protocol, the size of the EUC needs to be defined, the details on how to structure the data in the data type as well as what corresponding protocol-specific addressing is used and whether the SetAddress method can be supported. If SetAddress is not supported, Gateways and Communication Server shall not provide this method at all.

[3] HART® is a registered trademark of the non-profit organization FieldComm Group, Inc. This information is given for convenience of users of this document and does not constitute an endorsement by IEC of the trademark holder or any of its products. Compliance does not require use of the trademark. Use of the trademark requires permission of the trademark holder.

[4] PROFIBUS® is the registered trademark of the non-profit organization PROFIBUS Nutzerorganisation e.V. (PNO). This information is given for the convenience of users of this document and does not constitute an endorsement by IEC of the trademark holder or any of its products. Compliance does not require use of the trademark. Use of the trademark requires permission of the trademark holder.

C.5 Manufacturer

The Manufacturer identifies the manufacturer of a device and is provided in the scan result and in the OPC UA information model and in the Package Catalog. This profile uses an XML String, an EDD EUC, or an OPC UA String data type as data structure. For each protocol, the mapping to a protocol-specific representation needs to be defined as well as how to get the information when scanning a network. The Manufacturer is used to identify corresponding FDI® Device Packages which also contain the Manufacturer information.

An EDD also contains manufacturer information, in MANUFACTURER. Unless defined differently in the PSD, MANUFACTURER shall ~~have~~ be set to all bits in the integer value ~~set~~ to one, and the MANUFACTURER_EXT shall contain the manufacturer string as defined in the PSD.

C.6 DeviceModel

The DeviceModel identifies the type of device and is provided in the scan result and in the OPC UA information model and in the Package Catalog. This profile uses an XML String, an EDD EUC, or an OPC UA String data type as data structure. For each protocol, the mapping to a protocol-specific representation needs to be defined as well as how to get the information when scanning a network. The DeviceModel is used to identify corresponding FDI® Device Packages which also contain the DeviceModel information.

An EDD also contains information about the type of device, in DEVICE_TYPE. Unless defined differently in the PSD, the DEVICE_TYPE shall ~~have~~ be set to all bits in the integer value ~~set~~ to one.

C.7 DeviceRevision

The DeviceRevision allows to determine the compatibility between an FDI® Device Package and a Device and is optionally provided in the scan result and in the OPC UA information model. This profile uses an XML unsignedShort, an EDD UNSIGNED_INTEGER (2), or an OPC UA UInt16 data type as data structure. For each protocol, it needs to be specified whether the DeviceRevision is supported and if it is supported, the mapping to a protocol-specific representation needs to be defined as well as how to get the information when scanning a network.

C.8 SerialNumber

The SerialNumber uniquely identifies a device. It is optionally provided in the scan result and in the OPC UA information model. This profile uses an XML string, an EDD EUC or an OPC UA String data type as data structure. For each protocol, it needs to be specified whether the SerialNumber is supported and if it is supported, the mapping to a protocol-specific representation needs to be defined as well as how to get the information when scanning a network and the size of the EUC. The SerialNumber can be used by Host applications to identify a device even if it has been moved in the network or is accessible via different network paths.

C.9 Tag

The Tag provides a unique name for a device and is optionally provided in the scan result and in the OPC UA information model. This profile uses an XML string, an EDD EUC, or an OPC UA String data type as data structure. For each protocol, it needs to be specified whether the Tag is supported and if it is supported, the mapping to a protocol-specific representation needs to be defined as well as how to get the information when scanning a network and the size of the EUC.

C.10 ProfileId

The ProfileId identifies the supported profile of a device. It is optionally provided in the scan result and in the OPC UA information model. This profile uses an XML string, an EDD EUC, or an OPC UA String data type as data structure. For each protocol, it needs to be specified whether the ProfileId is supported and if it is supported, the mapping to a protocol-specific representation needs to be defined as well as how to get the information when scanning a network and the size of the EUC.

C.11 Version

The protocol version is used in the FDI® Package Catalog to define which version of a protocol is supported by the FDI® Package. It contains a major, minor and revision number. The mapping of protocol versions to this schema needs to be defined per protocol.

C.12 ProtocolSupportFile

The ProtocolSupportFiles can be provided by an FDI® Package. Whether such files are required needs to be defined per protocol, including what type of files need to be supported.

C.13 ExtendedDeviceRevision

The ExtendedDeviceRevision allows to determine the compatibility between an FDI® Device Package and a Device and is optionally provided in the scan result and in the OPC UA information model. ExtendedDeviceRevision uses the FDI® Versioning Format major.minor.revision. For each protocol, it needs to be specified whether the ExtendedDeviceRevision is supported and if it is supported, the mapping to a protocol-specific representation needs to be defined as well as how to get the information when scanning a network. If the protocol supports only a mapping for DeviceRevision, the ExtendedDeviceRevision will contain the DeviceRevision as major. Minor and revision is "00" in this case.

Bibliography

~~FCG TS62769-100, Edition 1.1, *Field Device Integration Part 100: Profiles – Generic Protocols*, available at https://fieldcommgroup.org~~

REC-xml-20081126, Extensible Markup Language (XML) 1.0 (Fifth Edition) – W3C Recommendation 26 November 2008, available at http://www.w3.org/TR/xml/ [viewed 2023-01-09]

____

IEC 62769-100

Edition 2.0 2023-04

INTERNATIONAL STANDARD

NORME INTERNATIONALE

Field device integration (FDI®) –
Part 100: Profiles – Generic Protocols

Intégration des appareils de terrain (FDI®) –
Partie 100: Profils – Protocoles Génériques

IEC 62769-100:2023-04(en-fr)

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

____

FIELD DEVICE INTEGRATION (FDI®) –

Part 100: Profiles – Generic Protocols

FOREWORD

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

IEC 62769-100 has been prepared by subcommittee 65E: Devices and integration in enterprise systems, of IEC technical committee 65: Industrial-process measurement, control and automation. It is an International Standard.

This second edition cancels and replaces the first edition published in 2020. This edition constitutes a technical revision.

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

a) added ExtendedDeviceRevision implementing the FDI® Version scheme and the method ScanExtended.

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

Draft	Report on voting
65E/865/CDV	65E/922/RVC

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

The language used for the development of this International Standard is English.

This document was drafted in accordance with ISO/IEC Directives, Part 2, and developed in accordance with ISO/IEC Directives, Part 1 and ISO/IEC Directives, IEC Supplement, available at www.iec.ch/members_experts/refdocs. The main document types developed by IEC are described in greater detail at www.iec.ch/standardsdev/publications.

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

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

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

FIELD DEVICE INTEGRATION (FDI®) –

Part 100: Profiles – Generic Protocols

1 Scope

This part of IEC 62769 specifies an FDI®[1] profile of IEC 62769 for Generic Protocols. That means that all interfaces are defined and a host can add support for more protocols without changing its implementation. Nevertheless, there are some protocol specific definitions (PSD) that need to be specified per protocol using this profile. Annex C specifies what PSD need to be defined per protocol so that FDI® Device Packages, FDI® Communication Packages for Gateways and FDI® Communication Servers, FDI® Communication Server, Gateways and Devices supporting such a protocol can work together in a host not aware about this specific protocol.

NOTE A host not using FDI® Communication Server but a proprietary mechanism for communication needs to define its own means to deal with this profile to support several protocols without changing its implementation. This is specific to the proprietary way how the communication driver is bound to the host.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61804 (all parts), *Devices and integration in enterprise systems − Function blocks (FB) for process control and Electronic Device Description Language (EDDL)*

IEC 61804-3, *Devices and integration in enterprise systems − Function blocks (FB) for process control and electronic device description language (EDDL) − Part 3: EDDL syntax and semantics*

IEC 62541-100:2015, *OPC Unified Architecture – Part 100: Device Interface*

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

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

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

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

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

IEC 62769-7, *Field Device Integration (FDI®) – Part 7: Communication Devices*

[1] FDI® is a registered trademark of the non-profit organization Fieldbus Foundation, Inc. This information is given for the convenience of users of this document and does not constitute an endorsement by IEC of the trademark holder or any of its products. Compliance does not require use of the trade name. Use of the trade name requires permission of the trade name holder.

IEC 62769-151-1, *Field Device Integration (FDI®) – Part 151-1: Profiles – OPC UA*

3 Terms, definitions, abbreviated terms and acronyms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 61804 series, IEC 62541-100, IEC 62769-2, IEC 62769-3, IEC 62769-4, IEC 62769-5, and IEC 62769-7 apply.

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

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

3.2 Abbreviated terms and acronyms

EDD	Electronic Device Description
EDDL	Electronic Device Description Language (see IEC 61804 series)
FDI®	Field Device Integration
FCG	FieldComm Group
PSD	Protocol-specific definitions
XML	Extensible markup language (see REC-xml-20081126)

4 Conventions

4.1 EDDL syntax

This part of IEC 62769 specifies content for the EDD component that is part of FDI® Communication Packages. The specification content using EDDL syntax uses the font `Courier New`. The EDDL syntax is used for method signature, variable, data structure and component declarations.

4.2 XML syntax

XML syntax examples use font `Courier New`. The XML syntax is used to describe XML document schema.

Example: `<xs:simpleType name="ExampleType">`

4.3 Capitalizations

Capitalization of the first letter of words is used in the IEC 62769 series to emphasize an FDI® defined term.

EDD language elements are written with all letters in uppercase.

Some of these terms using an acronym as a prefix for example

- FDI® Client, or
- FDI® Server.

Some of these terms are compound terms such as:

- Communication Servers, or

- Profile Package.

Parameter names or attributes are concatenated to a single term, where the original terms start in this term with a capital letter such as:

- ProtocolSupportFile, or
- ProtocolType.

Parameter names or attributes can also be constructed by using an underscore character to concatenate two or more terms such as:

- DEVICE_REV, or
- DEVICE_MODEL.

5 Profile for Generic Protocols

5.1 General

This profile document to the FDI® specification in IEC 62769 specifies the protocol specifics needed for FDI® Packages describing Communication Servers, Gateways and Devices.

This document also defines protocol specifics for Communication Servers as these need to be considered in the Communication Servers hosted Information Model.

Annex B defines the XML schema for Direct Access Services.

5.2 Catalog profile

5.2.1 Protocol support file

5.2.1.1 FDI® Device Package

Protocol specific attachments are mentioned in the Package Catalog as defined in IEC 62769-5. As this document defines a profile generically suitable for many protocols, it does not define requirements for any protocol specific attachments. However, it does also not prevent the usage of protocol specific attachments. The PSD define the requirements on the need of ProtocolSupportFiles for a specific protocol. However, the configuration of a device using an FDI® Device Package shall not require the usage of a protocol specific attachment. Table 1 specifies the parameters of the ProtocolSupportFile in the FDI® Device Package in case one or many are provided.

Table 1 – ProtocolSupportFile for FDI® Device Packages

Parameter	Description
Content Type	text/plain
Root Namespace	empty
Source Relationship	http://FDI-cooperation.com/2010/relationship/attachment-protocol
Filename	Not defined

5.2.1.2 FDI® Communication Packages

The same rules as for FDI® Device Packages apply.

5.2.2 CommunicationProfile definition

IEC 62769-4 defines a CommunicationProfileT string for the Catalog XML schema. The string is protocol specific and defined as ProfileIdentifier in the PSD (see Annex C).

5.2.3 Profile device

A Profile Package shall provide the catalog values for profile devices, enabling the FDI® Server to leverage a generic device description, if a specific one is not available. The definitions in Table 2 focus on catalog content that is vendor independent.

Table 2 – Catalog values for profile devices

Element	Attribute	Content
PackageType	—	Profile
Manufacturer	—	Empty
DeviceModel	—	The format of the DeviceModel is protocol specific and details on the format are defined in the PSD (see Annex C). In order to assign a scan result with a Profile Package, the ProfileId of the scan result shall be mapped to the DeviceModel of the Profile Package.

5.2.4 Protocol version information

IEC 62769-4 defines an element type named InterfaceT for the Catalog XML schema. The element type InterfaceT contains an element named Version which is supposed to provide version information about the applied communication protocol profile. The value shall follow the IEC 62769-4 defined version information schema defined in the element type VersionT. The PSD (see Annex C) define the mapping of versions of a specific protocol to this field.

5.3 Associating a Package with a device

5.3.1 Device type identification mapping

The purpose of device type identification mapping is to enable FDI® Host systems to compare the scan result against the topology representation in the Information Model. FDI® Host systems shall also be enabled to determine the FDI® Device Package that fits for a device entry contained in the scan result. This will enable the user of an FDI® Host system to synchronize the Information Model with the actual installation.

The Communication Server implemented scan service (defined in 5.6.1.7) provides the scan result through an XML document (the schema is defined in Clause A.5).

The Gateway implemented scan service (defined in 5.6.2.7) provides the scan result by means of the Information Model that contains data structures created from EDD content as specified in 5.6.2.7.

Common for both ways of presenting the scan result is that scan results contain device type identification and device instance identification.

FDI® Host systems comparing the actual network topology configuration against the topology representation in the Information Model shall be enabled to handle the following situations:

a) The physical Device instance identified at a specific device address is not logically present in the Information Model (as Instance): Enable the FDI® Host system to find the appropriate FDI® Device Package according to the device catalog information.

b) The physical Device instance identified by the device address is logically present in the Information Model (as Instance): Enable the FDI® Host system to compare device type information presented in scan result (see the identification in Clause A.5) and the device type specific information of the Instance present in the Information Model.

The FDI® Device Package contains device type identification information that can be compared to scan result based on the Catalog Schema in IEC 62769-4 defining the XML (simple) element types "DeviceModel" and "Manufacturer". Both types are used in the (complex) element types "Protocol" and "RegDeviceType".

As a result of the FDI® Package deployment, the FDI® Package information is then present in the Information Model as the specified FunctionalGroup Identification containing SerialNumber and Tag (see 5.4.3).

The mapping between different device identification data sources is described in Table 3. Since scan results provided by the Communication Server or Gateway can convey data that is produced by the device (firmware), the device type identification mapping shall be supported by providing corresponding data in the FDI® Device Package contained Catalog and Information Model.

Table 3 – Device identification information mapping

FDI® Device Package	Information Model	Communication Server provided scan result	Gateway provided scan result
Catalog specified type Manufacturer	FunctionalGroup: Identification Browse Name: Manufacturer	Element (path): ConnectionPoint/Identification Attribute: Manufacturer	COLLECTION ConnectionPoint. Identification: Manufacturer
Catalog specified type DeviceModel	FunctionalGroup: Identification Browse Name: DeviceModel	Element (path): ConnectionPoint/Identification Attribute: DeviceModel	COLLECTION ConnectionPoint. Identification. DeviceModel

Since not all protocols that are intended to be used with this profile for Generic Protocols might support a mandatory discovery mechanism allowing to identify the type of device (Manufacturer and DeviceModel), the scan results provide the capability to exclude the identification of the device and only provide the address. In that case, some host-specific mechanisms might be used to assign the desired FDI® package to the device, e.g., by user interaction.

Since some protocols might not even have mandatory capabilities to identify if there is a device at all for a specific protocol address hosts should provide the capability that some users can add devices by manually specifying address information.

5.3.2 Device type revision mapping

IEC 62769-4 envisions a concept that allows to determine the compatibility between an FDI® Device Package and a Device. IEC 62769-4 specifies a life cycle management process bearing on a single version information provided for the entire device. This is captured in the DeviceRevision (see Table 4). The DeviceRevision might be available as single number or as a string. Mapping of version information is protocol-specific and needs to be defined in the PSD (see Annex C).

Table 4 – Device revision information mapping

FDI® Device Package	Information Model	Communication Server provided scan result	Gateway provided scan result
Catalog specified type ListOfSupportedDeviceRevisions	FunctionalGroup: Identification Browse Name: DeviceRevision	Element (path): ConnectionPoint/Identification Attribute: DeviceRevision	COLLECTION ConnectionPoint. Identification. DeviceRevision

5.4 Information Model mapping

5.4.1 ProtocolType definition

In Table 5, a subtype of ProtocolType is defined to identify network communication using this profile.

Table 5 – Protocol type GenericProtocol

Attribute	Value				
BrowseName	GenericProtocol				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the ProtocolType defined in IEC 62541-100.					
HasProperty	Variable	ProtocolIdentifier	String	PropertyType	Mandatory

The mandatory Variable ProtocolIdentifier defines which concrete protocol is represented using the GenericProtocol type. It shall match the ProtocolIdentifier defined for the CommunicationProfile in 5.2.2. The string is protocol specific and defined as ProfileIdentifier in the PSD (see Annex C).

5.4.2 DeviceType mapping

Each device type inherits the properties of DeviceType. The mapping of the inherited properties from DeviceType is defined in Table 6. Note that only the attributes defined in Annex C and therefore expected by each generic protocol are used. Specific protocols might provide for example a SoftwareRevsion but since this is not accessible for the host, this profile does not make use of it.

Table 6 – Inherited DeviceType property mapping

Property	Generic Protocol Mapping
SerialNumber	SerialNumber (see Annex C)
RevisionCounter	-1 (not defined)
Manufacturer	String taken from FDI® package catalog (ManufacturerName from PackageT)
Model	String taken from FDI® package catalog (Name of DeviceTypeT, which is a localized name)
DeviceManual	empty text string (not supported) [a]
DeviceRevision	DeviceRevision (see Annex C)
SoftwareRevision	empty string (not defined)
HardwareRevision	empty string (not defined)
[a] Device manuals are exposed as attachments of the FDI® Device Package.	

5.4.3 FunctionalGroup identification definition

As defined in IEC 62541-100:2015, 5.3, each device representation in the FDI® Server hosted Information Model shall contain a protocol specific FunctionalGroup named Identification. The Parameters of this FunctionalGroup are defined for generic protocol device types as follows:

Table 7 – Generic Protocol Device Types identification attributes

BrowseName	DataType	Mandatory/Optional
Manufacturer	String	Mandatory
DeviceModel	String	Mandatory
SerialNumber	String	Optional
Tag	String	Optional
DeviceRevision	UInt16	Optional
ExtendedDeviceRevision	String	Optional
ProfileId	String	Optional

The BaseDataVariable instances shall be created from VARIABLE declarations with identifiers that correspond to the browse names listed in Table 7.

5.5 Topology elements

5.5.1 ConnectionPoint definition

The ConnectionPoint type GenericConnectionPoint shall be used to parameterize network access points using the Generic Protocols. The ConnectionPoint type GenericConnectionPoint is a sub type of the abstract type ConnectionPointType defined in IEC 62541-100. Table 8 specifies the representation of the GenericConnectionPoint in the AddressSpace.

Table 8 – ConnectionPoint type for Generic Protocols

Attribute	Value				
BrowseName	GenericConnectionPoint				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Sub type of the ConnectionPointType defined in IEC 62541-100.					
HasProperty	Variable	Address	String	PropertyType	Mandatory
HasProperty	Variable	ProtocolIdentifier	String	PropertyType	Mandatory

The ConnectionPoint type GenericConnectionPoint shall be described by an EDD element contained in a Communication Device related FDI® Package that can drive a generic protocol network. Actual ConnectionPoint properties are declared by VARIABLE constructs grouped together in a COLLECTION named ConnectionPoint. For this profile, it shall only contain the CONNECTION_POINT_ADDRESS, mapped to the OPC UA Property Address. In addition, the PROTOCOL specified by the COMPONENT shall be mapped to the ProtocolIdentifier Property. The following EDDL source code is an example describing a Connection Point for an ExampleProtocol. The ProtocolIdentifier defined by the PSD (see Annex C) shall be used as PROTOCOL name in the EDD.

```
COMPONENT ConnectionPoint_Generic
{
  LABEL "Generic Connection Point";
  CLASSIFICATION NETWORK_CONNECTION_POINT;
  CAN_DELETE FALSE;
  PROTOCOL ExampleProtocol;
  CONNECTION_POINT ConnectionPoint;
}

VARIABLE Address
{
  LABEL "Address";
  HELP "Address of the device";
  TYPE EUC(<protocol-specific>);
  CLASS LOCAL;
}


COLLECTION ConnectionPoint
{
  LABEL "Connection Point";
  MEMBERS
  {
    CONNECTION_POINT_ADDRESS, Address;
  }
}
```

5.5.2 Communication Device definition

According to IEC 62769-7, each FDI® Communication Package shall contain an EDD element describing the communication device. The following EDDL source code is an example describing a Communication Server.

```
COMPONENT Generic_Communication_Server
{
  LABEL "Generic communication server";
  PRODUCT_URI "urn:Company Name:Product Name";
  CAN_DELETE TRUE;
  CLASSIFICATION NETWORK_COMPONENT;
  COMPONENT_RELATIONS
  {
    Generic_Communication_Device_Setup
  }
}

COMPONENT_RELATION Generic_Communication_Device_Setup
{
   LABEL "Relation between Device and communication device";
   RELATION_TYPE CHILD_COMPONENT;
   COMPONENTS
   {
       Generic_Communication_Device{AUTO_CREATE 1;}
   }
   MINIMUM_NUMBER 1;
   MAXIMUM_NUMBER 4;
}
```

According to IEC 62769-7, each FDI® Communication Package shall contain at least one EDD element describing at least one communication device component. The following EDDL source code in is an example for a generic protocol communication device:

```
COMPONENT Generic_Communication_Device
{
  LABEL "Generic communication device";
  CAN_DELETE TRUE;
  CLASSIFICATION NETWORK_COMPONENT;
  COMPONENT_RELATIONS { Generic_Service_Provider_Relation }
}

COMPONENT_RELATION Generic_Service_Provider_Relation
{
   LABEL "Relation to communication service provider";
   RELATION_TYPE CHILD_COMPONENT;
   COMPONENTS
   {
       Generic_Service_Provider{AUTO_CREATE 1;}
   }
   MINIMUM_NUMBER 1;
   MAXIMUM_NUMBER 1;
}
```

In an actual communication device, the ConnectionPoint_Generic needs to be adapted according to the supported protocol and the related connection point definitions given in 5.5. The attribute BYTE_ORDER shall not be used for this profile as the byte order handling shall be done in the gateway business logic.

5.5.3 Communication service provider definition

According to IEC 62769-7, each FDI® Communication Package shall contain at least one EDD element describing at least one communication service provider component. The following EDDL source code below is an example for a generic protocol communication service provider component.

The component reference (ConnectionPoint_Generic) corresponds to the related connection point definition in 5.5. The attribute BYTE_ORDER shall not be used for this profile as the byte order handling shall be done in the gateway business logic.

```
COMPONENT Generic_Service_Provider
{
  LABEL "Generic Protocol communication service provider";
  CAN_DELETE TRUE;
  CLASSIFICATION NETWORK_COMMUNICATION_SERVICE_PROVIDER;
  COMPONENT_RELATIONS
  {
    Generic_Service_Provider_Connection_Point_Relation
  }
}


COMPONENT_RELATION Generic_Service_Provider_Connection_Point_Relation
{
   LABEL "Relation between communication service provider and Connection Point";
   RELATION_TYPE CHILD_COMPONENT;
   ADDRESSING {Address}
   COMPONENTS
   {
       ConnectionPoint_Generic{ AUTO_CREATE 1;}
   }
   MINIMUM_NUMBER 1;
   MAXIMUM_NUMBER 1;
}
```

5.5.4 Network definition

According to IEC 62769-7, each FDI® Communication Package shall contain at least one EDD element describing network configuration constraints using the component construct.

```
COMPONENT Network_Generic
{
    LABEL "Generic Network";
    CAN_DELETE TRUE;
    CLASSIFICATION NETWORK;
    COMPONENT_RELATIONS
    {
      Generic_Network_Connection_Point_Relation
    }
}

COMPONENT_RELATION Generic_Network_Connection_Point_Relation
{
   LABEL "Relation between network and Connection Point";
   RELATION_TYPE CHILD_COMPONENT;
   ADDRESSING {Address}
   COMPONENTS
   {
       ConnectionPoint_Generic
   }
   MINIMUM_NUMBER 1;
   MAXIMUM_NUMBER 32;
}
```

5.6 Methods

5.6.1 Methods for FDI® Communication Servers

5.6.1.1 General

The Communication Server contained Information Model shall implement services according to method signatures described in 5.6.1.

5.6.1.2 Connect

Signature:

```
Connect(
    [in]   ByteString        CommunicationRelationId,
    [in]   String            Address,
    [out]  Int32             ServiceError);
```

Table 9 provides the description of the arguments.

Table 9 – Method Connect arguments

Argument	Description
CommunicationRelationId	The argument value contains the nodeId of the ConnectionPoint representing the connection between a device and a physical network which is directly connected to the Communication Server hardware. The nodeId allows finding the direct parent-child relation.
Address	The argument name shall match with the corresponding attribute name defined for the ConnectionPoint which is described by a corresponding EDD element specified in 5.5. The argument value holds the protocol-specific device's network address that is unique within the network segment.
ServiceError	0: OK / execution finished, connection established successfully -1: Connect Failed / cancelled by caller -3: Connect Failed / device not found -4: Connect Failed / invalid device address -5: Connect Failed / invalid device identification

5.6.1.3 Disconnect

Signature:

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

Table 10 provides the description of the arguments.

Table 10 – Method Disconnect arguments

Argument	Description
CommunicationRelationId	The argument value contains the nodeId of the ConnectionPoint representing the connection between a device and a physical network which is directly connected to the Communication Server hardware. The nodeId allows finding the direct parent-child relation.
ServiceError	0: OK / disconnect finished successfully -1: Disconnect Failed / no existing communication relation -2: Disconnect Failed / invalid communication relation identifier

5.6.1.4 Transfer

Signature:

```
Transfer(
    [in]  ByteString        CommunicationRelationId,
    [in]  String            Header,
    [in]  ByteString        RequestData,
    [in]  EddDataTypeInfo[] RequestDataTypes,
    [in]  EddDataTypeInfo[] ResponseDataTypes,
    [out] ByteString        ResponseData,
    [out] ByteString        RESPONSE_CODES,
    [out] Int32             ServiceError);
```

Table 11 provides the description of the arguments.

Table 11 – Method Transfer arguments

Argument	Description
CommunicationRelationId	The argument value contains the nodeId of the ConnectionPoint representing the connection between a device and a physical network within the Information Model.
Header	The protocol-specific information on the data to be transferred. The Header is used in the COMMANDs of the EDD and the PSD (see Annex C) defines the format of this string per protocol. The first communication service treating this string shall validate the string according to its format defined in the PSD.
RequestData	The argument name shall match with the corresponding COMMAND sub-element name REQUEST. The byte stream submitted trough the argument is created from definitions provided by the REQUEST element of the corresponding COMMAND that shall be processed.
RequestDataTypes	This array contains information on what data types are used to create the RequestData ByteString The byte order used in the RequestData is EDD specific and providing the information allows the gateway or communication server to change it to a protocol-specific byte order of the data type or applying any other protocol-specific characteristics (e.g. bit order). An empty array can be provided if the protocol's byte order matches the EDD specific byte order. If the protocol's byte order does not match the EDD specific byte order, and the provided data types match only part of the RequestData data, the remaining data is considered to require no adaption with respect to the byte order.
ResponseDataTypes	This array contains information on what data types are expected when receiving the ResponseData ByteString. The knowledge about the expected data types is in the EDD of the device and therefore the gateway or communication server needs to get this information in order to apply protocol-specific characteristics to the response data and knows how to separate the response data into an array of ByteString. An empty array can be provided if the protocol's byte order matches the EDD specific byte order. If the protocol's byte order does not match the EDD specific byte order, and the provided data types match only part of the ResponseData, the remaining data is considered to require no adaption with respect to the byte order.
ResponseData	The argument name shall match with the corresponding COMMAND sub-element name REPLY. The byte stream returned by this argument applies to definitions provided by the REPLY element of the corresponding COMMAND that shall be processed.
RESPONSE_CODES	The argument name shall match with the COMMAND sub-element name RESPONSE_CODES. The argument value conveys the specific communication service response bytes.
ServiceError	0: OK / execution finished -1: Transfer Failed / cancelled by caller -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

The EddDataTypeInfo DataType defines the data type information of an EDD data type used in a COMMAND. Its elements are defined in Table 12.

Table 12 – EddDataTypeInfo DataType Structure

Name	Type	Description
EddDataTypeInfo	structure	This structure specifies information about a data type used in an EDD COMMAND.
eddDataType	EddDataTypeEnum	The EddDataType used.
size	UInt32	The size of the eddDataType. It shall always be provided, even if not explicitly specified by the EDD. In that case the default value for that data type shall be provided. For data types where no size can be defined (e.g. BOOLEAN), the value shall be set to 0.

The EddDataTypeEnum DataType is an enumeration that defines the possible EDD data types. Its values are defined in Table 13.

Table 13 – EddDataTypeEnum Values

Value	Description
BOOLEAN	Data type as defined by IEC 61804-3
DOUBLE	Data type as defined by IEC 61804-3
FLOAT	Data type as defined by IEC 61804-3
INTEGER	Data type as defined by IEC 61804-3
UNSIGNED_INTEGER	Data type as defined by IEC 61804-3
DATE	Data type as defined by IEC 61804-3
DATE_AND_TIME	Data type as defined by IEC 61804-3
DURATION	Data type as defined by IEC 61804-3
TIME	Data type as defined by IEC 61804-3
TIME_VALUE	Data type as defined by IEC 61804-3
BIT_ENUMERATED	Data type as defined by IEC 61804-3
ENUMERATED	Data type as defined by IEC 61804-3
ASCII	Data type as defined by IEC 61804-3
BITSTRING	Data type as defined by IEC 61804-3
EUC	Data type as defined by IEC 61804-3
OCTET	Data type as defined by IEC 61804-3
PACKED_ASCII	Data type as defined by IEC 61804-3
PASSWORD	Data type as defined by IEC 61804-3
VISIBLE	Data type as defined by IEC 61804-3

5.6.1.5 GetPublishedData

This method is not supported by the Generic Protocols profile.

5.6.1.6 SetAddress

Signature:

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

Table 14 provides the description of the arguments.

Table 14 – Method SetAddress arguments

Argument	Description
OldAddress	The argument value holds the current address of a device. The same semantic applies to this field as the Address parameter in the Connect method.
NewAddress	The argument value holds the new address for a device. The same semantic applies to this field as the Address parameter in the Connect method.
ServiceError	0: OK / execution finished successfully -1: SetAddress Failed / cancelled by caller -3: SetAddress Failed / not initialized -4: SetAddress Failed / not connected to a network -5: SetAddress Failed / no device found responding to oldAddress -6: SetAddress Failed / duplicate address error -7: SetAddress Failed / device did not accept new address -8: SetAddress Failed / invalid oldAddress (in terms of syntax, data type, data format, and so on) -9: SetAddress Failed / invalid newAddress (in terms of syntax, data type, data format, and so on) -10: SetAddress Failed / not possible in status connected

5.6.1.7 Scan

The method signature specified in IEC 62769-7 applies. The corresponding topologyScanResult schema is specified in Annex A, whereas this method returns the identification data with the DeviceRevision as unsignedShort, see Annex C.

5.6.1.8 ScanExtended

The method signature specified in IEC 62769-7 applies. The corresponding topologyScanResult schema is specified in Annex A, whereas this method returns the identification data with the ExtendedDeviceRevision, see Annex C.

5.6.1.9 ResetScan

The method signature specified in IEC 62769-7 applies.

5.6.2 Methods for Gateways

5.6.2.1 General

The methods signatures defined in 5.6.2 apply. The methods shall be implemented in the EDD element (IEC 62769-4) contained in a Gateway related FDI® Package containing the communication device definitions.

5.6.2.2 Connect

Subclause 5.6.2.2 describes the generic protocol specific implementation of the service Connect specified in IEC 62769-7.

```
METHOD BeginConnect(
   DD_STRING                CommunicationRelationId,
   DD_STRING                Address,
   unsigned long            ServiceId,
   unsigned long            &DelayForNextCall,
   long                     &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}

METHOD EndConnect(
   DD_STRING                CommunicationRelationId,
   unsigned long            ServiceId,
   unsigned long            &DelayForNextCall,
   long                     &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}

METHOD CancelConnect(
   DD_STRING                CommunicationRelationId,
   unsigned long            ServiceId,
   long                     &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}
```

Table 15 provides the description of the arguments.

Table 15 – Connect service arguments

Argument	Description
CommunicationRelationId	The argument value contains the nodeId of the ConnectionPoint representing the connection between a device and a physical network which is directly connected to the Communication Server hardware. The nodeId allows finding the direct parent-child relation.
Address	The argument name shall match with the corresponding attribute name defined for the ConnectionPoint which is described by a corresponding EDD element specified in 5.5. The argument value holds the protocol-specific device’s network address that is unique within the network segment.
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 to limit the EndConnect invocation cycle that shall not be faster than specified in the argument value.
ServiceError	1: OK / function started asynchronously, result has to be polled with EndConnect 0: 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

5.6.2.3 Disconnect

Subclause 5.6.2.3 describes the generic protocol specific implementation of the service Disconnect specified in IEC 62769-7.

```
METHOD Disconnect(
   DD_STRING CommunicationRelationId,
   long   &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}
```

All the arguments of the Disconnect method are described in Table 10.

5.6.2.4 Transfer

Subclause 5.6.2.4 describes the generic protocol specific implementation of the service Transfer specified in IEC 62769-7. Due to limitations in the capabilities of EDD methods (it is not possible to transfer any collections of a dynamic size), this profile requires some additional handling when using the transfer via Gateways. In addition to the calls of the EDD method, a host first needs to provide the data type information of the request and the response data. The following order needs to be executed by the host:

1) Add the request and response data type information into a list defined by the gateway EDD (called DATA_TYPE_INFO_PER_SERVICE_ID). The entry shall include the ServiceId also used in the method calls.
2) Call BeginTransfer using the same ServiceId and follow the asynchronous call pattern as defined in IEC 62769-7.

3) When the host has received the response, it is the responsibility of the host to remove the entry from the list. Thus, only the host is manipulating the list and the gateway business logic is only accessing but not changing the list in order to get the information on the data types.

```
METHOD BeginTransfer(
   DD_STRING                CommunicationRelationId,
   DD_STRING                Header,
   DD_STRING                RequestData,
   unsigned long            ServiceId,
   DD_STRING                &ResponseData,
   DD_STRING                &RESPONSE_CODES,
   long                     &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}

METHOD EndTransfer(
   DD_STRING                CommunicationRelationId,
   unsigned long            ServiceId,
   DD_STRING                &ResponseData,
   DD_STRING                &RESPONSE_CODES,
   unsigned long            &DelayForNextCall,
   long                     &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}

METHOD CancelTransfer(
   DD_STRING                CommunicationRelationId,
   DD_STRING                &ResponseData,
   DD_STRING                &RESPONSE_CODES,
   unsigned long            ServiceId,
   long                     &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}
```

Table 16 provides the description of the arguments.

Table 16 – Method Transfer arguments

Argument	Description
CommunicationRelationId	The argument value contains the nodeId of the ConnectionPoint representing the connection between a device and a physical network within the Information Model.
Header	The protocol-specific information on the data to be transferred. The Header is used in the COMMANDs of the EDD and the PSD (see Annex C) defines the format of this string per protocol. The first communication service treating this string shall validate the string according to its format defined in the PSD.
RequestData	The argument name shall match with the corresponding COMMAND sub-element name REQUEST. The byte stream submitted trough the argument is created from definitions provided by the REQUEST element of the corresponding COMMAND that shall be processed. The byte stream is separated in several byte streams for the different requested data types. This allows to consider a protocol-specific byte order of specific data types in the gateway or communication server.
ResponseData	The argument name shall match with the corresponding COMMAND sub-element name REPLY. The byte stream returned by this argument applies to definitions provided by the REPLY element of the corresponding COMMAND that shall be processed. The data are separated into several byte strings based on the ResponseDataTypes field. This allows the gateway or communication server to consider protocol-specific byte order of data types.
RESPONSE_CODES	The argument name shall match with the COMMAND sub-element name RESPONSE_CODES. The argument value conveys the specific communication service response bytes.
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 to limit the EndTransfer invocation cycle that shall not be faster than specified in the argument value.
ServiceError	1: OK / function started asynchronously, result has to be polled with EndTransfer 0: OK / execution finished -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

The List that is used to add the data type information so the business logic of the gateway can do the needed data type transfers shall be called DATA_TYPE_INFO_LIST. It is defined in the following:

```
VARIABLE EddDataTypeEnum
{
  LABEL "Edd Data Type";
  CLASS LOCAL;
    TYPE ENUMERATED
    {
                { 0x00, "BOOLEAN" }
                { 0x01, "DOUBLE" }
                { 0x02, "FLOAT" }
                { 0x03, "INTEGER" }
                { 0x04, "UNSIGNED_INTEGER" }
                { 0x05, "DATE" }
                { 0x06, "DATE_AND_TIME" }
                { 0x07, "DURATION" }
                { 0x08, "TIME" }
                { 0x09, "TIME_VALUE" }
                { 0x0a, "BIT_ENUMERATED" }
                { 0x0b, "ENUMERATED" }
                { 0x0c, "ASCII" }
                { 0x0d, "BITSTRING" }
                { 0x0e, "EUC" }
                { 0x10, "OCTET" }
                { 0x11, "PACKED_ASCII" }
                { 0x12, "PASSWORD" }
                { 0x13, "VISIBLE" }
     }
}

VARIABLE SizeV
{
  LABEL "Size of data type";
  TYPE UNSIGNED_INTEGER(4);
  CLASS LOCAL;
}

COLLECTION DataTypeInfo
{
  MEMBERS
  {
    Size, SizeV;
    EddDataType, EddDataTypeEnum;
  }
}

LIST DataTypeInfoList
{
  TYPE DataTypeInfo;
}

VARIABLE ServiceIdentification
{
  LABEL "Identification of service call";
  TYPE UNSIGNED_INTEGER(4);
  CLASS LOCAL;
}

COLLECTION DataTypeInfoPerServiceId
{
  MEMBERS
  {
    ServiceId, ServiceIdentification;
    RequestDataTypes, DataTypeInfoList;
    ResponseDataTypes, DataTypeInfoList;
  }
}
LIST DATA_TYPE_INFO_PER_SERVICE_ID
{
  TYPE DataTypeInfoPerServiceId;
}
```

5.6.2.5 GetPublishedData

This method is not supported in the generic protocol profile.

5.6.2.6 SetAddress

Subclause 5.6.2.6 describes the generic protocol specific implementation of the service SetAddress specified in IEC 62769-7.

```
BeginSetAddress(
   DD_STRING                   OldAddress,
   DD_STRING                   NewAddress,
   unsigned long               ServiceId,
   unsigned long               &DelayForNextCall,
   long                        &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}

EndSetAddress(
   unsigned long               ServiceId,
   unsigned long               &DelayForNextCall,
   long                        &ServiceError)
{
  ACCESS ONLINE;
  DEFINITION{<Gateway specific implementation>}
}
```

Table 17 provides the description of the arguments.

Table 17 – Method SetAddress arguments

Argument	Description
OldAddress	The argument value holds the current address of a device. The same semantic applies to this field as the Address parameter in the Connect method.
NewAddress	The argument value holds the new address for a device. The same semantic applies to this field as the Address parameter in the Connect method.
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 to limit the EndTransfer invocation cycle that shall not be faster than specified in the argument value.
ServiceError	1: OK / function started asynchronously, result has to be polled with EndSetAddress 0: OK / execution finished successfully -1: SetAddress Failed / cancelled by caller -2: Call Failed / unknown service ID -3: SetAddress Failed / not initialized -4: SetAddress Failed / not connected to a network -5: SetAddress Failed / no device found responding to oldAddress -6: SetAddress Failed / duplicate address error -7: SetAddress Failed / device did not accept new address -8: SetAddress Failed / invalid oldAddress (in terms of syntax, data type, data format, and so on) -9: SetAddress Failed / invalid newAddress (in terms of syntax, data type, data format, and so on) -10: SetAddress Failed / not possible in status connected

5.6.2.7 Scan

The method signature specified in IEC 62769-7 applies. The generic protocol gateway business logic shall create the scan result following IEC 62769-7. The following definitions shall be present in the COMPONENT declaration that holds the definitions for a communication device. The data structure corresponds to the data structure defined in the XML schema in Annex A. The SCAN_LIST attribute inside the COMPONENT declaration shall refer to LIST TopologyScanResult.

```
VARIABLE GenericAddress
{
  LABEL "Station address";
  TYPE EUC(<protocol-specific>);
  CLASS LOCAL;
}

VARIABLE Manufacturer
{
  LABEL "Manufacturer ID";
  TYPE EUC(<protocol-specific>);
  CLASS LOCAL;
}

VARIABLE DeviceModel
{
  LABEL "Device ID";
  TYPE EUC(<protocol-specific>);
  CLASS LOCAL;
}
```

```
VARIABLE DeviceRevision
{
  LABEL "Device revision";
  TYPE UNSIGNED_INTEGER(2);
  CLASS LOCAL;
}

VARIABLE SerialNumber
{
  LABEL "Serial number";
  TYPE EUC(<protocol-specific>);
  CLASS LOCAL;
}

VARIABLE Tag
{
  LABEL "Tag";
  TYPE EUC(<protocol-specific>);
  CLASS LOCAL;
}

VARIABLE ProfileId
{
  LABEL "ProfileId";
  TYPE EUC(<protocol-specific>);
  CLASS LOCAL;
}


COLLECTION ScanItemType
{
  MEMBERS
  {
    ADDRESS, GenericAddress;
    MANUFACTURER_ID, Manufacturer;
    DEVICE_MODEL, DeviceModel;
    DEVICE_REV, DeviceRevision;
    SERIAL_NUMBER, SerialNumber;
    TAG, Tag;
    PROFILEID, ProfileId;
  }
}
LIST TopologyScanResult
{
  TYPE ScanItemType;
}
```

5.6.2.8 ScanNext

The method signature specified in IEC 62769-7 applies. The generic gateway business logic shall create the scan result following IEC 62769-7. The method ScanNext stores the result into data structures described for the method Scan (5.6.2.7).

Annex A
(normative)

Topology scan result schema

A.1 General

The topology scan result schema specified in Annex A describes the Generic Protocols specific format Method Scan argument `topologyScanResult`. The XML document content and structure shall correspond to the Information Model designed concept to describe a topology in order to enable generic matching between physical devices connected to the network and the FDI® Server hosted Information Model.

A.2 Network

The subsequent element is used to return the scan result corresponding to the Information Model described in IEC 62769-5.

The XML schema for a Network element is:

```
<xs:element name="Network" type="GP:GenericNetworkT"/>
```

A.3 GenericNetworkT

The element type describes the complete scan result for a single network because of the scan method that is provided per instance of a "Communication Device" which exists in 1:1 relation to a network instance.

The XML schema for a GenericNetworkT type is:

```
<xs:complexType name="GenericNetworkT">
   <xs:sequence>
      <xs:element name="ConnectionPoint" type="GP:GenericConnectionPointT"
         maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

The elements of a GenericNetworkT type are described in Table A.1.

Table A.1 – Elements of GenericNetworkT

Element	Description
ConnectionPoint	The ConnectionPoint element holds the address and identification of the network connected device that has been found during bus scan operations.

A.4 GenericConnectionPointT

The XML schema for a GenericConnectionPointT type is:

```
<xs:complexType name="GenericConnectionPointT">
   <xs:sequence>
      <xs:choice minOccurs=”0”>
         <xs:element name="Identification" type="GP:GenericIdentificationT"/>
         <xs:element name="IdentificationExt"
         type="GP:GenericIdentificationExtendedT"/>
      </xs:choice>
   </xs:sequence>
   <xs:attribute name="Address" type="GP:GenericAddressT" use="required"/>
</xs:complexType>
```

The attributes of a GenericConnectionPointT type are described in Table A.2.

Table A.2 – Attributes of GenericConnectionPointT

Attribute	Description
Address	The attribute value holds the address of the network connected device used to uniquely address the device in the network.

The elements of a GenericConnectionPointT type are described in Table A.3.

Table A.3 – Elements of GenericConnectionPointT

Element	Description
Identification	The element data holds the device type identification data. Compared to the Information Model (IEC 62541-100), the ConnectionPoint does not contain or refer to the device type identification data. But in order to support the FDI® Host system in finding the package that matches the connected device, this schema associates the device type identification with the ConnectionPoint. This identification may contain the DeviceRevision as unsignedShort or as String in the minor.major.revision format. Note that this element is optional since a protocol might not support identification of a device.

A.5 GenericIdentificationT

The element content corresponds to the "FunctionalGroup Identification".

The XML schema for a GenericIdentificationT type is:

```
<xs:complexType name="GenericIdentificationT">
   <xs:attribute name="Manufacturer" type="xs:string" use="required"/>
   <xs:attribute name="DeviceModel" type="xs:string" use="required"/>
   <xs:attribute name="SerialNumber" type="xs:string" use="optional"/>
   <xs:attribute name="Tag" type="xs:string" use="optional"/>
   <xs:attribute name="DeviceRevision" type="xs:unsignedShort"
      use="optional"/>
   <xs:attribute name="ProfileId" type="xs:string" use="optional"/>
</xs:complexType>
```

The attributes of a GenericIdentificationT type are described in Table A.4.

Table A.4 – Attributes of GenericIdentificationT

Attribute	Description
Manufacturer	See Table 7
DeviceModel	See Table 7
SerialNumber	See Table 7
Tag	See Table 7
DeviceRevision	See Table 7
ProfileId	See Table 7

A.6 GenericAddressT

The protocol-specific address used to uniquely identify a device in a network segment.

The XML schema for a GenericAddressT type is:

```
<xs:simpleType name="GenericAddressT">
   <xs:restriction base="xs:string"/>
</xs:simpleType>
```

A.7 GenericIdentificationExtendedT

The element content corresponds to the "FunctionalGroup Identification". This type includes the extended DeviceRevision information in the major.minor.revision format as defined in IEC 62769-151-1.

The XML schema for a GenericIdentificationExtendedT type is:

```
<xs:complexType name="GenericIdentificationT">
   <xs:attribute name="Manufacturer" type="xs:string" use="required"/>
   <xs:attribute name="DeviceModel" type="xs:string" use="required"/>
   <xs:attribute name="SerialNumber" type="xs:string" use="optional"/>
   <xs:attribute name="Tag" type="xs:string" use="optional"/>
   <xs:attribute name="ExtendedDeviceRevision" type="FDI:VersionT"
      use="optional"/>
   <xs:attribute name="ProfileId" type="xs:string" use="optional"/>
</xs:complexType>
```

The attributes of a GenericIdentificationExtendedT type are described in Table A.5.

Table A.5 – Attributes of GenericIdentificationExtendedT

Attribute	Description
Manufacturer	See Table 7
DeviceModel	See Table 7
SerialNumber	See Table 7
Tag	See Table 7
ExtendedDeviceRevision	See Table 7, ExtendedDeviceRevision in major.minor.revision format as defined in IEC 62769-151-1
ProfileId	See Table 7

Annex B
(normative)

Transfer service parameters

B.1 General

Direct Access Services specified in IEC 62769-2 enable the User Interface Plug-in (UIP) to directly exchange data with the device. Direct data exchange means that data exchanged between a device and a UIP cannot be reflected in the Information Model. The IEC 62769-5 defined interface IDirectAccess corresponds to the IEC 62769-2 specified Direct Access Services. Interface IDirectAccess defined functions BeginTransfer and EndTransfer need to convey protocol specific information. The protocol specifics shall be encapsulated in the XML document defined by this schema.

B.2 sendData

The element described in the following contains data to be submitted through the IDirectAccess function BeginTransfer defined argument sendData.

The XML schema for a sendData element is:

```
<xs:element name="sendData" type="GP:TransferSendDataT"/>
```

B.3 receiveData

The element described in the following contains data that is returned through the IDirectAccess function EndTransfer defined return value.

The XML schema for a receiveData element is:

```
<xs:element name="receiveData" type="GP:TransferResultDataT"/>
```

B.4 TransferSendDataT

A complex type that defines the service parameter data format that shall be applied to Transfer defined argument sendData.

The XML schema for a TransferSendDataT type is:

```
<xs:complexType name="TransferSendDataT">
   <xs:sequence>
      <xs:element name="RequestDataTypes" type="GP:EddDataTypeInfoListT"/>
      <xs:element name="ResponseDataTypes" type="GP:EddDataTypeInfoListT"/>
   </xs:sequence>
   <xs:attribute name="Header" type="xs:string" use="required"/>
   <xs:attribute name="RequestData" type="xs:hexBinary" use="required"/>
</xs:complexType>
```

The attributes of a TransferSendDataT type are described in Table B.1.

Table B.1 – Attributes of TransferSendDataT

Attribute	Description
Header	The attribute corresponds to the Transfer method argument Header.
RequestData	The attribute corresponds to the Transfer method argument RequestData.

The elements of a TransferSendDataT type are described in Table B.2.

Table B.2 – Elements of TransferSendDataT

Element	Description
RequestDataTypes	The element corresponds to the Transfer method argument RequestDataTypes
ResponseDataTypes	The element corresponds to the Transfer method argument ResponseDataTypes

B.5 EddDataTypeInfoListT

List of EDD data type information.

The XML schema for a EddDataTypeInfoListT type is:

```
<xs:complexType name="EddDataTypeInfoListT">
  <xs:sequence>
    <xs:element name="EddDataTypeInfo" type="GP:EddDataTypeInfoT"
      maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

The elements of a EddDataTypeInfoListT type are described in Table B.3.

Table B.3 – Elements of EddDataTypeInfoListT

Element	Description
EddDataTypeInfo	Describes an individual EDD data type and its size.

B.6 EddDataTypeInfoT

Describes an individual EDD data type and its size.

The XML schema for a EddDataTypeInfoT type is:

```
<xs:complexType name="EddDataTypeInfoT">
  <xs:attribute name="EddDataType" type="GP:EddDataTypeT" use="required"/>
  <xs:attribute name="Size" type="xs:nonNegativeInteger" use="required"/>
</xs:complexType>
```

The attributes of a EddDataTypeInfoT type are described in Table B.4.

Table B.4 – Attributes of EddDataTypeInfoT

Attribute	Description
EddDataType	The EddDataType used.
Size	The size of the eddDataType. It shall always be provided, even if not explicitly specified by the EDD. In that case the default value for that data type shall be provided. For data types where no size can be defined (e.g. BOOLEAN), the value shall be set to 0.

B.7 EddDataTypeT

A simple type that defines the possible EDD data types.

The XML schema for a EddDataTypeT enumeration type is:

```
<xs:simpleType name="EddDataTypeT">
   <xs:restriction base="xs:string">
      <xs:enumeration value="BOOLEAN"/>
      <xs:enumeration value="DOUBLE"/>
      <xs:enumeration value="FLOAT"/>
      <xs:enumeration value="INTEGER"/>
      <xs:enumeration value="UNSIGNED_INTEGER"/>
      <xs:enumeration value="DATE"/>
      <xs:enumeration value="DATE_AND_TIME"/>
      <xs:enumeration value="DURATION"/>
      <xs:enumeration value="TIME"/>
      <xs:enumeration value="TIME_VALUE"/>
      <xs:enumeration value="BIT_ENUMERATED"/>
      <xs:enumeration value="ENUMERATED"/>
      <xs:enumeration value="ASCII"/>
      <xs:enumeration value="BITSTRING"/>
      <xs:enumeration value="EUC"/>
      <xs:enumeration value="OCTET"/>
      <xs:enumeration value="PACKED_ASCII"/>
      <xs:enumeration value="PASSWORD"/>
      <xs:enumeration value="VISIBLE"/>
   </xs:restriction>
</xs:simpleType>
```

The enumeration values of a EddDataTypeT enumeration type are described in Table B.5.

Table B.5 – Enumerations of EddDataTypeT

Enumeration	Description
BOOLEAN	Data type as defined in IEC 61804-3
DOUBLE	Data type as defined in IEC 61804-3
FLOAT	Data type as defined in IEC 61804-3
INTEGER	Data type as defined in IEC 61804-3
UNSIGNED_INTEGER	Data type as defined in IEC 61804-3
DATE	Data type as defined in IEC 61804-3
DATE_AND_TIME	Data type as defined in IEC 61804-3
DURATION	Data type as defined in IEC 61804-3
TIME	Data type as defined in IEC 61804-3
TIME_VALUE	Data type as defined in IEC 61804-3
BIT_ENUMERATED	Data type as defined in IEC 61804-3
ENUMERATED	Data type as defined in IEC 61804-3
ASCII	Data type as defined in IEC 61804-3
BITSTRING	Data type as defined in IEC 61804-3
EUC	Data type as defined in IEC 61804-3
OCTET	Data type as defined in IEC 61804-3
PACKED_ASCII	Data type as defined in IEC 61804-3
PASSWORD	Data type as defined in IEC 61804-3
VISIBLE	Data type as defined in IEC 61804-3

B.8 TransferResultDataT

A complex type that defines the service parameter data format that shall be applied to Transfer defined receivedData return value.

```
<xs:complexType name="TransferResultDataT">
   <xs:attribute name="ResponseData" type="xs:hexBinary" use="required"/>
   <xs:attribute name="RESPONSE_CODES" type="xs:hexBinary" use="required"/>
</xs:complexType>
```

The attributes of a TransferResultDataT type are described in Table B.6.

Table B.6 – Attributes of TransferResultDataT

Attribute	Description
ResponseData	The attribute corresponds to the Transfer method argument ResponseData.
RESPONSE_CODES	The attribute corresponds to the Transfer method argument RESPONSE_CODES.

Annex C
(normative)

Protocol specific definitions

C.1 General

This annex defines what the PSD need to define per protocol.

C.2 Header

The HEADER attribute is used in an EDD to define the addressing of a COMMAND. When using this annex, each EDD shall comply with the profile for generic protocol extensions defined in IEC 61804-3 and define COMMANDs in the EDD using the HEADER as addressing in order to access the corresponding VARIABLEs in the device. The HEADER consists of a string. Per protocol it needs to be defined how the protocol-specific addressing is mapped to the HEADER attribute. Examples of protocol-specific addressing is the COMMAND number in HART®[2] or SLOT and INDEX in PROFIBUS®[3].

The first communication service treating this string shall validate the string according to its format defined in the PSD.

C.3 ProtocolIdentifier

The ProtocolIdentifier is used in the Package Catalog defined in IEC 62769-4 (called CommunicationProfile) as well as in the EDD for Gateways defined in IEC 61804-3 (called PROTOCOL) to identify the protocol. It is represented as a string in the EDD and in the Package Catalog. In order to have a unique identifier per protocol, the ProtocolIdentifier needs to be defined once per protocol and a URI syntax should be used in order to guarantee the uniqueness of the string.

C.4 Address

The address uniquely identifies a device in a network segment and is provided in the scan result as well as connect and SetAddress methods of an FDI® Communication Server or a Gateway. This profile uses an OPC UA String, XML String or an EDD EUC or DD_STRING in EDD methods as data structure in order to allow high flexibility on the actual used protocol-specific address as well as an easy way to edit an address by the user. For each protocol, the size of the EUC needs to be defined, the details on how to structure the data in the data type as well as what corresponding protocol-specific addressing is used and whether the SetAddress method can be supported. If SetAddress is not supported, Gateways and Communication Server shall not provide this method at all.

[2] HART® is a registered trademark of the non-profit organization FieldComm Group, Inc. This information is given for convenience of users of this document and does not constitute an endorsement by IEC of the trademark holder or any of its products. Compliance does not require use of the trademark. Use of the trademark requires permission of the trademark holder.

[3] PROFIBUS® is the registered trademark of the non-profit organization PROFIBUS Nutzerorganisation e.V. (PNO). This information is given for the convenience of users of this document and does not constitute an endorsement by IEC of the trademark holder or any of its products. Compliance does not require use of the trademark. Use of the trademark requires permission of the trademark holder.

C.5 Manufacturer

The Manufacturer identifies the manufacturer of a device and is provided in the scan result and in the OPC UA information model and in the Package Catalog. This profile uses an XML String, an EDD EUC, or an OPC UA String data type as data structure. For each protocol, the mapping to a protocol-specific representation needs to be defined as well as how to get the information when scanning a network. The Manufacturer is used to identify corresponding FDI® Device Packages which also contain the Manufacturer information.

An EDD also contains manufacturer information, in MANUFACTURER. Unless defined differently in the PSD, MANUFACTURER shall be set to all bits in the integer value to one, and the MANUFACTURER_EXT shall contain the manufacturer string as defined in the PSD.

C.6 DeviceModel

The DeviceModel identifies the type of device and is provided in the scan result and in the OPC UA information model and in the Package Catalog. This profile uses an XML String, an EDD EUC, or an OPC UA String data type as data structure. For each protocol, the mapping to a protocol-specific representation needs to be defined as well as how to get the information when scanning a network. The DeviceModel is used to identify corresponding FDI® Device Packages which also contain the DeviceModel information.

An EDD also contains information about the type of device, in DEVICE_TYPE. Unless defined differently in the PSD, the DEVICE_TYPE shall be set to all bits in the integer value to one.

C.7 DeviceRevision

The DeviceRevision allows to determine the compatibility between an FDI® Device Package and a Device and is optionally provided in the scan result and in the OPC UA information model. This profile uses an XML unsignedShort, an EDD UNSIGNED_INTEGER (2), or an OPC UA UInt16 data type as data structure. For each protocol, it needs to be specified whether the DeviceRevision is supported and if it is supported, the mapping to a protocol-specific representation needs to be defined as well as how to get the information when scanning a network.

C.8 SerialNumber

The SerialNumber uniquely identifies a device. It is optionally provided in the scan result and in the OPC UA information model. This profile uses an XML string, an EDD EUC or an OPC UA String data type as data structure. For each protocol, it needs to be specified whether the SerialNumber is supported and if it is supported, the mapping to a protocol-specific representation needs to be defined as well as how to get the information when scanning a network and the size of the EUC. The SerialNumber can be used by Host applications to identify a device even if it has been moved in the network or is accessible via different network paths.

C.9 Tag

The Tag provides a unique name for a device and is optionally provided in the scan result and in the OPC UA information model. This profile uses an XML string, an EDD EUC, or an OPC UA String data type as data structure. For each protocol, it needs to be specified whether the Tag is supported and if it is supported, the mapping to a protocol-specific representation needs to be defined as well as how to get the information when scanning a network and the size of the EUC.

C.10 ProfileId

The ProfileId identifies the supported profile of a device. It is optionally provided in the scan result and in the OPC UA information model. This profile uses an XML string, an EDD EUC, or an OPC UA String data type as data structure. For each protocol, it needs to be specified whether the ProfileId is supported and if it is supported, the mapping to a protocol-specific representation needs to be defined as well as how to get the information when scanning a network and the size of the EUC.

C.11 Version

The protocol version is used in the FDI® Package Catalog to define which version of a protocol is supported by the FDI® Package. It contains a major, minor and revision number. The mapping of protocol versions to this schema needs to be defined per protocol.

C.12 ProtocolSupportFile

The ProtocolSupportFiles can be provided by an FDI® Package. Whether such files are required needs to be defined per protocol, including what type of files need to be supported.

C.13 ExtendedDeviceRevision

The ExtendedDeviceRevision allows to determine the compatibility between an FDI® Device Package and a Device and is optionally provided in the scan result and in the OPC UA information model. ExtendedDeviceRevision uses the FDI® Versioning Format major.minor.revision. For each protocol, it needs to be specified whether the ExtendedDeviceRevision is supported and if it is supported, the mapping to a protocol-specific representation needs to be defined as well as how to get the information when scanning a network. If the protocol supports only a mapping for DeviceRevision, the ExtendedDeviceRevision will contain the DeviceRevision as major. Minor and revision is "00" in this case.

Bibliography

REC-xml-20081126, Extensible Markup Language (XML) 1.0 (Fifth Edition) – W3C Recommendation 26 November 2008, available at http://www.w3.org/TR/xml/ [viewed 2023-01-09]

SOMMAIRE

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

____

INTÉGRATION DES APPAREILS DE TERRAIN (FDI®) –

Partie 100: Profils – Protocoles Génériques

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.
9) L'attention est attirée sur le fait que certains des éléments de la présente Publication de l'IEC peuvent faire l'objet de droits de brevet. L'IEC ne saurait être tenue pour responsable de ne pas avoir identifié de tels droits de brevets.

L'IEC 62769-100 a été établie par le sous-comité 65E: Les dispositifs et leur intégration dans les systèmes de l'entreprise, du comité d'études 65 de l'IEC: Mesure, commande et automation dans les processus industriels. Il s'agit d'une Norme internationale.

Cette deuxième édition annule et remplace la première édition parue en 2020. Cette édition constitue une révision technique.

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

a) ajout d'ExtendedDeviceRevision qui met en œuvre le schéma de version FDI® et la méthode ScanExtended.

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

Projet	Rapport de vote
65E/865/CDV	65E/922/RVC

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

La langue employée pour l'élaboration de cette Norme internationale est l'anglais.

Ce document a été rédigé selon les Directives ISO/IEC, Partie 2, il a été développé selon les Directives ISO/IEC, Partie 1 et les Directives ISO/IEC, Supplément IEC, disponibles sous www.iec.ch/members_experts/refdocs. Les principaux types de documents développés par l'IEC sont décrits plus en détail sous www.iec.ch/standardsdev/publications.

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

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

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

INTÉGRATION DES APPAREILS DE TERRAIN (FDI®) –

Partie 100: Profils – Protocoles Génériques

1 Domaine d'application

La présente partie de l'IEC 62769 spécifie un profil FDI®[1] de l'IEC 62769 pour les Protocoles Génériques. Cela signifie que toutes les interfaces sont définies, et qu'un hôte peut ajouter une prise en charge pour davantage de protocoles sans modifier la mise en œuvre. Néanmoins, certaines définitions spécifiques au protocole (PSD) nécessitent d'être spécifiées à l'aide de ce profil. L'Annexe C spécifie les PSD qu'il est nécessaire de définir par protocole, de sorte que les Paquetages d'Appareils FDI®, les Paquetages de Communication FDI® pour les Passerelles et les Serveurs de Communication FDI®, le Serveur de Communication FDI®, les Passerelles et les Appareils qui prennent en charge ce type de protocole puissent fonctionner ensemble dans un hôte, quel que soit ce protocole spécifique.

NOTE Un hôte qui n'utilise pas de Serveur de Communication FDI® mais un mécanisme propriétaire pour sa communication nécessite de définir ses propres moyens de gérer ce profil de manière à prendre en charge différents protocoles sans modifier sa mise en œuvre.

2 Références normatives

Les documents suivants sont cités dans le texte de sorte qu'ils constituent, pour tout ou partie de leur contenu, des exigences du présent document. Pour les références datées, seule l'édition citée s'applique. Pour les références non datées, la dernière édition du document de référence s'applique (y compris les éventuels amendements).

IEC 61804 (toutes les parties), *Les dispositifs et leur intégration dans les systèmes de l'entreprise – Blocs fonctionnels (FB) pour les procédés industriels et le langage de description électronique de produit (EDDL)*

IEC 61804-3, *Les dispositifs et leur intégration dans les systèmes de l'entreprise – Blocs fonctionnels (FB) pour les procédés industriels et le langage de description électronique de produit (EDDL) – Partie 3: Sémantique et syntaxe EDDL*

IEC 62541-100:2015, *Architecture unifiée OPC – Partie 100: Interface d'appareils*

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

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

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

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

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

IEC 62769-7, *Intégration des appareils de terrain (FDI®) – Partie 7: Appareils de communication*

[1] FDI® est une marque déposée de l'organisation à but non lucratif Fieldbus Foundation, Inc. Cette information est donnée à l'intention des utilisateurs du présent document et ne signifie nullement que l'IEC approuve le détenteur de la marque ou l'emploi de ses produits. La conformité n'exige pas l'utilisation de la marque. L'utilisation de la marque exige l'autorisation du détenteur de la marque.

IEC 62769-151-1, *Intégration des appareils de terrain (FDI®) – Partie 151-1: Profils – OPC UA*

3 Termes, définitions, abréviations et acronymes

3.1 Termes et définitions

Pour les besoins du présent document, les termes et définitions de la série IEC 61804, de l'IEC 62541-100, de l'IEC 62769-2, de l'IEC 62769-3, de l'IEC 62769-4, de l'IEC 62769-5 et de l'IEC 62769-7 s'appliquent.

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

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

3.2 Abréviations et acronymes

EDD (Electronic Device Description)	Description électronique de produit
EDDL (Electronic Device Description Language)	Langage de description électronique de produit (voir série IEC 61804)
FDI® (Field Device Integration)	Intégration des appareils de terrain
FCG	FieldComm Group
PSD (Protocol-specific definitions)	Définitions spécifiques au protocole
XML (Extensible Markup Language)	Langage de balisage extensible (voir REC-xml-20081126)

4 Conventions

4.1 Syntaxe EDDL

La présente partie de l'IEC 62769 spécifie le contenu du composant EDD qui fait partie des Paquetages de Communication FDI®. Le contenu de la spécification qui utilise la syntaxe EDDL est rédigé avec la police `Courier New`. La syntaxe EDDL est utilisée pour les déclarations de signature de méthode, de variable, de structure de données et de composant.

4.2 Syntaxe XML

Les exemples de syntaxe XML utilisent la police `Courier New`. La syntaxe XML est utilisée pour décrire le schéma des documents XML.

Exemple: `<xs:simpleType name="ExampleType">`

4.3 Utilisation de majuscules

La mise en majuscules de la première lettre des mots est utilisée dans la série IEC 62769 pour souligner un terme défini spécifique à la FDI®.

Les éléments de langage EDD sont écrits en majuscules.

Certains de ces termes utilisent un acronyme comme suffixe, par exemple:

- Client FDI®; ou
- Serveur FDI®.

Certains de ces termes sont des termes composés, par exemple:

- Serveurs de Communication; ou
- Paquetage de Profil.

Les noms de paramètres ou attributs sont concaténés en un seul terme, dans lequel les différents termes d'origine commencent par une lettre majuscule, par exemple:

- ProtocolSupportFile; ou
- ProtocolType.

Les noms de paramètres ou attributs peuvent aussi être construits en utilisant le caractère de soulignement pour concaténer au moins deux termes, par exemple:

- DEVICE_REV; ou
- DEVICE_MODEL.

5 Profil de Protocoles Génériques

5.1 Généralités

Le présent document de profil lié à la spécification FDI® de l'IEC 62769 spécifie les particularités du protocole nécessaires aux Paquetages FDI® décrivant les Serveurs de Communication, les Passerelles et les Appareils.

Le présent document définit également les éléments spécifiques du protocole pour les Serveurs de Communication car ils nécessitent d'être pris en compte dans le Modèle d'Information hébergé sur les Serveurs de Communication.

L'Annexe B définit le schéma XML des Services d'Accès Direct.

5.2 Profil du catalogue

5.2.1 Fichier de prise en charge du protocole

5.2.1.1 Paquetage d'Appareil FDI®

Les pièces jointes spécifiques à un protocole sont mentionnées dans le Catalogue de Paquetage, comme cela est défini dans l'IEC 62769-5. Même si le présent document définit un profil en principe adapté à un grand nombre de protocoles, elle ne définit aucune exigence relative pour les pièces jointes spécifiques du protocole. Cependant, il n'empêche pas d'utiliser des pièces jointes spécifiques au protocole. Les PSD définissent les exigences relatives au besoin de ProtocolSupportFiles pour un protocole donné. Toutefois, la configuration d'un appareil qui utilise un Paquetage d'Appareil FDI® ne doit pas exiger l'utilisation d'une pièce jointe spécifique du protocole. Le Tableau 1 spécifie les paramètres de ProtocolSupportFile dans le Paquetage d'Appareil FDI® si un ou plusieurs paramètres sont fournis.

Tableau 1 – ProtocolSupportFile pour les Paquetages d'Appareils FDI®

Paramètre	Description
Type de contenu	Texte en clair
Espace de noms racine	Vide
Relation source	http://FDI-cooperation.com/2010/relationship/attachment-protocol
Nom de fichier	Non défini

5.2.1.2 Paquetages de Communication FDI®

Les mêmes règles que pour les Paquetages d'Appareils FDI® s'appliquent.

5.2.2 Définition du CommunicationProfile

L'IEC 62769-4 définit une chaîne CommunicationProfileT pour le schéma XML Catalog. La chaîne est spécifique au protocole et définie en tant que ProfileIdentifier dans la PSD (voir Annexe C).

5.2.3 Appareil de profil

Un Paquetage de Profil doit fournir les valeurs de catalogue des appareils de profil, en permettant au Serveur FDI® d'optimiser la description d'appareil générique, si une description spécifique n'est pas disponible. Les définitions du Tableau 2 se concentrent sur le contenu de catalogue indépendant du fournisseur.

Tableau 2 – Valeurs de catalogue pour les appareils de profil

Elément	Attribut	Contenu
PackageType	—	Profil
Manufacturer	—	Vide
DeviceModel	—	Le format de DeviceModel est spécifique au protocole et les détails relatifs au format sont définis dans les PSD (voir Annexe C). Pour attribuer un résultat d'analyse avec un Paquetage de Profil, le ProfileId de ce résultat doit être mappé au DeviceModel du Paquetage de Profil.

5.2.4 Informations relatives à la version du protocole

L'IEC 62769-4 définit un type d'élément nommé InterfaceT pour le schéma XML Catalog. Le type d'élément InterfaceT contient un élément appelé Version, qui a pour objet de fournir des informations de version concernant le profil du protocole de communication appliqué. La valeur doit respecter le schéma d'informations de version de l'IEC 62769-4 défini dans le type d'élément VersionT. Les PSD (voir Annexe C) définissent le mapping de versions d'un protocole spécifique à cette zone.

5.3 Association d'un Paquetage à un appareil

5.3.1 Mapping d'identification de type d'appareil

L'objet du mapping d'identification de type d'appareil est de configurer les systèmes Hôtes FDI® afin qu'ils comparent le résultat de balayage à la représentation topologique dans le Modèle d'Information. Les systèmes Hôtes FDI® doivent également pouvoir déterminer le Paquetage d'Appareils FDI® qui correspond à une entrée d'appareil contenue dans le résultat d'analyse. Cela permet à l'utilisateur d'un système Hôte FDI® de synchroniser le Modèle d'Information avec l'installation réelle.

Le service d'analyse mis en œuvre par le Serveur de Communication (défini en 5.6.1.7) donne le résultat de l'analyse par l'intermédiaire d'un document XML (le schéma est défini à l'Article A.5).

Le service d'analyse mis en œuvre par la Passerelle (défini en 5.6.2.7) donne le résultat de l'analyse au moyen du Modèle d'Information contenant les structures de données issues du contenu EDD comme cela est spécifié en 5.6.2.7.

Les deux manières de présenter le résultat de balayage ont en commun le fait que les résultats de balayage contiennent une identification du type d'appareil et une identification d'instance d'appareil.

Les systèmes Hôtes FDI® qui comparent la configuration topologique actuelle du réseau à la représentation topologique dans le Modèle d'Information doivent être en mesure de gérer les situations suivantes:

a) L'instance d'Appareil physique identifiée à une adresse d'appareil spécifique n'est pas logiquement présente dans le Modèle d'Information (en tant qu'Instance): configurer le système Hôte FDI® afin qu'il trouve le Paquetage d'Appareil FDI® approprié en fonction des informations du catalogue d'appareils.

b) L'instance d'Appareil physique identifiée par l'adresse de l'appareil est logiquement présente dans le Modèle d'Information (en tant qu'Instance): configurer le système Hôte FDI® afin qu'il compare les informations de type d'appareil fournies dans le résultat de balayage (voir l'identification à l'Article A.5) aux informations spécifiques du type d'appareil de l'Instance fournies dans le Modèle d'Information.

Le Paquetage d'Appareil FDI® contient des informations d'identification de type d'appareil qui peuvent être comparées au résultat de balayage issu du Schéma Catalog de la IEC 62769-4, qui définit les types d'éléments XML (simples) "DeviceModel" et "Manufacturer". Ces deux types sont utilisés dans les types d'éléments (complexes) "Protocol" et "RegDeviceType".

Après le déploiement du Paquetage FDI®, les informations relatives au Paquetage FDI® sont alors présentes dans le Modèle d'Information comme l'identification FunctionalGroup spécifiée qui contient les éléments SerialNumber et Tag (voir 5.4.3).

Le mapping entre les différentes sources de données d'identification d'appareil est décrit dans le Tableau 3. Etant donné que les résultats de balayage fournis par le Serveur de Communication ou la Passerelle peuvent comporter des données produites par l'appareil (micrologiciel), le mapping d'identification de type d'appareil doit être assuré en fournissant les données correspondantes qui figurent dans le Catalogue et le Modèle d'Information du Paquetage d'Appareil FDI®.

Tableau 3 – Mapping des informations d'identification de type d'appareil

Paquetage d'Appareil FDI®	Modèle d'Information	Résultat d'analyse fourni par le Serveur de Communication	Résultat d'analyse fourni par la Passerelle
Type spécifié par le catalogue Manufacturer	FunctionalGroup: Identification Nom d'exploration: Manufacturer	Elément (chemin): ConnectionPoint/Identification Attribut: Manufacturer	COLLECTION ConnectionPoint. Identification: Manufacturer
Type spécifié par le catalogue DeviceModel	FunctionalGroup: Identification Nom d'exploration: DeviceModel	Elément (chemin): ConnectionPoint/Identification Attribut: DeviceModel	COLLECTION ConnectionPoint. Identification. DeviceModel

Etant donné que tous les protocoles destinés à être utilisés avec ce profil pour les Protocoles Génériques ne peuvent pas prendre en charge un mécanisme de reconnaissance obligatoire permettant d'identifier le type d'appareil (Manufacturer et DeviceModel), les résultats d'analyse offrent la possibilité d'exclure l'identification de l'appareil et de ne donner que l'adresse. Dans ce cas, certains mécanismes propres à l'hôte peuvent être utilisés pour attribuer le paquetage FDI® souhaité à l'appareil, par exemple au moyen d'une action de l'utilisateur.

Dans la mesure où certains protocoles n'ont même pas les capacités obligatoires pour identifier s'il existe un appareil pour une adresse spécifique au protocole, il convient que les hôtes proposent les fonctions nécessaires pour que certains utilisateurs puissent ajouter des appareils en spécifiant manuellement l'adresse.

5.3.2 Mapping de révision de type d'appareil

L'IEC 62769-4 prévoit un concept qui permet de déterminer la compatibilité entre un Paquetage d'Appareil FDI® et un Appareil. L'IEC 62769-4 spécifie un processus de gestion du cycle de vie qui s'appuie sur une information de version unique fournie pour l'ensemble de l'appareil. Ce point est reflété dans la DeviceRevision (voir Tableau 4). La DeviceRevision peut être disponible sous forme de numéro unique ou de chaîne. Le mapping des informations de version est spécifique au protocole, et nécessite d'être défini dans les PSD (voir Annexe C).

Tableau 4 – Mapping des informations de révision d'appareil

Paquetage d'Appareil FDI®	Modèle d'Information	Résultat d'analyse fourni par le Serveur de Communication	Résultat d'analyse fourni par la Passerelle
Type spécifié par le catalogue ListOfSupportedDeviceRevisions	FunctionalGroup: Identification Nom d'exploration: DeviceRevision	Elément (chemin): ConnectionPoint/Identification Attribut: DeviceRevision	COLLECTION ConnectionPoint. Identification. DeviceRevision

5.4 Mapping du Modèle d'Information

5.4.1 Définition de ProtocolType

Dans le Tableau 5, un sous-type du ProtocolType est défini pour identifier la communication réseau à l'aide de ce profil.

Tableau 5 – Type de protocole GenericProtocol

Attribut	Valeur				
BrowseName	GenericProtocol				
IsAbstract	False				
Références	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Sous-type du ProtocolType défini dans l'IEC 62541-100.					
HasProperty	Variable	ProtocolIdentifier	String	PropertyType	Obligatoire

La Variable obligatoire ProtocolIdentifier définit le protocole concret représenté à l'aide du type GenericProtocol. Elle doit correspondre au ProtocolIdentifier défini pour le CommunicationProfile en 5.2.2. La chaîne est spécifique au protocole et définie en tant que ProfileIdentifier dans la PSD (voir Annexe C).

5.4.2 Mapping de DeviceType

Chaque type d'appareil hérite des propriétés du DeviceType. Le mapping des propriétés héritées du DeviceType est défini dans le Tableau 6. Noter que seuls les attributs définis dans l'Annexe C, et par conséquent attendus par chaque protocole générique, sont utilisés. Les protocoles spécifiques peuvent fournir, par exemple, une SoftwareRevision, mais comme elle n'est pas accessible par l'hôte, ce profil ne l'utilise pas.

Tableau 6 – Mapping des propriétés héritées du DeviceType

Propriété	Mapping de Protocole Générique
SerialNumber	SerialNumber (voir Annexe C)
RevisionCounter	-1 (non défini)
Manufacturer	Chaîne issue du catalogue de paquetages FDI® (ManufacturerName de PackageT)
Model	Chaîne issue du catalogue de paquetages FDI® (Nom de DeviceTypeT, qui est un nom localisé)
DeviceManual	Chaîne de texte vide (non prise en charge) [a]
DeviceRevision	DeviceRevision (voir Annexe C)
SoftwareRevision	chaîne vide (non définie)
HardwareRevision	chaîne vide (non définie)

[a] Les manuels d'appareils sont exposés en tant que pièces jointes du Paquetage d'Appareil FDI®.

5.4.3 Définition du FunctionalGroup "Identification"

Comme cela est défini en dans l'IEC 62541-100:2015, 5.3, chaque représentation d'appareil dans le Modèle d'Information hébergé sur le Serveur FDI® doit contenir un FunctionalGroup spécifique d'un protocole, appelé Identification. Les paramètres de ce FunctionalGroup sont définis pour les types d'appareils de protocole générique, comme suit:

Tableau 7 – Attributs d'identification des Types d'Appareils de Protocole Générique

BrowseName	DataType	Obligatoire/Facultatif
Manufacturer	String	Obligatoire
DeviceModel	String	Obligatoire
SerialNumber	String	Facultatif
Tag	String	Facultatif
DeviceRevision	UInt16	Facultatif
ExtendedDeviceRevision	String	Facultatif
ProfileId	String	Facultatif

Les instances de BaseDataVariable doivent être créées à partir des déclarations de VARIABLE avec des identifiants qui correspondent aux noms d'exploration énumérés dans le Tableau 7.

5.5 Eléments de topologie

5.5.1 Définition de ConnectionPoint

Le GenericConnectionPoint de type ConnectionPoint doit être utilisé pour paramétrer les points d'accès au réseau utilisant ces Protocoles Génériques. Le ConnectionPoint de type GenericConnectionPoint est un sous-type du type abstrait ConnectionPointType défini dans l'IEC 62541-100. Le Tableau 8 spécifie la représentation du GenericConnectionPoint dans l'AddressSpace.

Tableau 8 – Type de ConnectionPoint pour les Protocoles Génériques

Attribut	**Valeur**				
BrowseName	GenericConnectionPoint				
IsAbstract	False				
Références	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Sous-type du ConnectionPointType défini dans l'IEC 62541-100.					
HasProperty	Variable	Address	String	PropertyType	Obligatoire
HasProperty	Variable	ProtocolIdentifier	String	PropertyType	Obligatoire

Le GenericConnectionPoint de type ConnectionPoint doit être décrit par un élément EDD contenu dans un Paquetage FDI® lié à l'Appareil de Communication qui peut gérer un réseau de protocole générique. Les propriétés réelles de ConnectionPoint sont déclarées par des constructions de VARIABLE regroupées dans une COLLECTION nommée ConnectionPoint. Pour ce profil, il ne doit contenir que CONNECTION_POINT_ADDRESS, mappé à l'Adresse de Propriété OPC UA. De plus, le PROTOCOL spécifié par le COMPONENT doit être mappé à la propriété ProtocolIdentifier. Le code source EDDL suivant est un exemple représentatif du Connection Point d'un ExampleProtocol. Le ProtocolIdentifier défini par les PSD (voir Annexe C) doit être utilisé comme nom de PROTOCOL dans l'EDD.

```
COMPONENT ConnectionPoint_Generic
{
  LABEL "Generic Connection Point";
  CLASSIFICATION NETWORK_CONNECTION_POINT;
  CAN_DELETE FALSE;
  PROTOCOL ExampleProtocol;
  CONNECTION_POINT ConnectionPoint;
}

VARIABLE Address
{
  LABEL "Address";
  HELP "Address of the device";
  TYPE EUC(<protocol-specific>);
  CLASS LOCAL;
}


COLLECTION ConnectionPoint
{
  LABEL "Connection Point";
  MEMBERS
  {
    CONNECTION_POINT_ADDRESS, Address;
  }
}
```

5.5.2 Définition d'Appareil de Communication

Conformément à l'IEC 62769-7, chaque Paquetage de Communication FDI® doit contenir un élément EDD qui décrit l'appareil de communication. Le code source EDDL suivant est un exemple représentatif d'un Serveur de Communication.

```
COMPONENT Generic_Communication_Server
{
  LABEL "Generic communication server";
  PRODUCT_URI "urn:Company Name:Product Name";
  CAN_DELETE TRUE;
  CLASSIFICATION NETWORK_COMPONENT;
  COMPONENT_RELATIONS
  {
    Generic_Communication_Device_Setup
  }
}

COMPONENT_RELATION Generic_Communication_Device_Setup
{
   LABEL "Relation between Device and communication device";
   RELATION_TYPE CHILD_COMPONENT;
   COMPONENTS
   {
       Generic_Communication_Device{AUTO_CREATE 1;}
   }
   MINIMUM_NUMBER 1;
   MAXIMUM_NUMBER 4;
}
```

Conformément à l'IEC 62769-7, chaque Paquetage de Communication FDI® doit contenir au moins un élément EDD qui décrit au moins un composant Appareil de communication. Le code source EDDL suivant est un exemple pour un appareil de communication de protocole générique:

```
COMPONENT Generic_Communication_Device
{
  LABEL "Generic communication device";
  CAN_DELETE TRUE;
  CLASSIFICATION NETWORK_COMPONENT;
  COMPONENT_RELATIONS { Generic_Service_Provider_Relation }
}

COMPONENT_RELATION Generic_Service_Provider_Relation
{
   LABEL "Relation to communication service provider";
   RELATION_TYPE CHILD_COMPONENT;
   COMPONENTS
   {
       Generic_Service_Provider{AUTO_CREATE 1;}
   }
   MINIMUM_NUMBER 1;
   MAXIMUM_NUMBER 1;
}
```

Dans un appareil de communication réel, il est nécessaire d'adapter le ConnectionPoint_Generic en fonction du protocole pris en charge et des définitions du point de connexion associé données en 5.5. L'attribut BYTE_ORDER ne doit pas être utilisé pour ce profil, car le traitement de l'ordre des octets doit être assuré dans la logique applicative de la passerelle.

5.5.3 Définition du fournisseur de service de communication

Conformément à l'IEC 62769-7, chaque Paquetage de Communication FDI® doit contenir au moins un élément EDD qui décrit au moins un composant fournisseur de service de communication. Le code source EDDL suivant est un exemple de composant de fournisseur de services de communication de protocole générique

La référence du composant (ConnectionPoint_Generic) correspond à la définition du point de connexion associé du 5.5). L'attribut BYTE_ORDER ne doit pas être utilisé pour ce profil, car le traitement de l'ordre des octets doit être assuré dans la logique applicative de la passerelle.

```
COMPONENT Generic_Service_Provider
{
  LABEL "Generic Protocol communication service provider";
  CAN_DELETE TRUE;
  CLASSIFICATION NETWORK_COMMUNICATION_SERVICE_PROVIDER;
  COMPONENT_RELATIONS
  {
    Generic_Service_Provider_Connection_Point_Relation
  }
}


COMPONENT_RELATION Generic_Service_Provider_Connection_Point_Relation
{
   LABEL "Relation between communication service provider and Connection Point";
   RELATION_TYPE CHILD_COMPONENT;
   ADDRESSING {Address}
   COMPONENTS
   {
       ConnectionPoint_Generic{ AUTO_CREATE 1;}
   }
   MINIMUM_NUMBER 1;
   MAXIMUM_NUMBER 1;
}
```

5.5.4 Définition du Réseau

Conformément à l'IEC 62769-7, chaque Paquetage de Communication FDI® doit contenir au moins un élément EDD qui décrit les contraintes de configuration réseau à l'aide de la construction COMPONENT.

```
COMPONENT Network_Generic
{
    LABEL "Generic Network";
    CAN_DELETE TRUE;
    CLASSIFICATION NETWORK;
    COMPONENT_RELATIONS
    {
      Generic_Network_Connection_Point_Relation
    }
}

COMPONENT_RELATION Generic_Network_Connection_Point_Relation
{
   LABEL "Relation between network and Connection Point";
   RELATION_TYPE CHILD_COMPONENT;
   ADDRESSING {Address}
   COMPONENTS
   {
       ConnectionPoint_Generic
   }
   MINIMUM_NUMBER 1;
   MAXIMUM_NUMBER 32;
}
```

5.6 Méthodes

5.6.1 Méthodes pour les Serveurs de Communication FDI®

5.6.1.1 Généralités

Le Serveur de Communication contenu dans le Modèle d'Information doit mettre en œuvre les services selon les signatures de méthode décrites en 5.6.1.

5.6.1.2 Connect

Signature:

```
Connect(
    [in]   ByteString        CommunicationRelationId,
    [in]   String            Address,
    [out]  Int32             ServiceError);
```

Le Tableau 9 fournit la description des arguments.

Tableau 9 – Arguments de la méthode Connect

Argument	Description
CommunicationRelationId	La valeur de l'argument contient le nodeId du ConnectionPoint qui représente la connexion entre un appareil et un réseau physique directement connecté au matériel du Serveur de Communication. Le nodeId permet de trouver la relation parent-enfant directe.
Address	Le nom d'argument doit être le nom d'attribut correspondant défini pour le ConnectionPoint qui est décrit par un élément EDD correspondant spécifié en 5.5. La valeur d'argument contient l'adresse réseau de l'appareil spécifique au protocole, qui est unique au sein du segment de réseau.
ServiceError	0: OK/exécution terminée, connexion établie avec succès -1: Échec de la connexion/annulée par l'appelant -3: Échec de la connexion/appareil introuvable -4: Échec de la connexion/adresse de l'appareil non valide -5: Échec de la connexion/identification de l'appareil non valide

5.6.1.3 Disconnect

Signature:

```
Disconnect(

    [in]  ByteString  CommunicationRelationId,

    [out] Int32       ServiceError);
```

Le Tableau 10 fournit la description des arguments.

Tableau 10 – Arguments de la méthode Disconnect

Argument	Description
CommunicationRelationId	La valeur de l'argument contient le nodeId du ConnectionPoint qui représente la connexion entre un appareil et un réseau physique directement connecté au matériel du Serveur de Communication. Le nodeId permet de trouver la relation parent-enfant directe.
ServiceError	0: OK/la déconnexion a abouti -1: Échec de la déconnexion/il n'existe aucune relation de communication -2: Échec de la déconnexion/identifiant de relation de communication non valide