# IEC 62769-109-1

Edition 2.0 2020-04
REDLINE VERSION

# INTERNATIONAL STANDARD

colour inside

## Field device integration (FDI) – Part 109-1: Profiles – HART® and WirelessHART®



IEC 62769-109-1:2020-04 RLV(en)

IEC Central Office
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 It also gives information on projects, replaced and withdrawn publications.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.

**Electropedia - www.electropedia.org**
The world's leading online dictionary on electrotechnology, containing more than 22 000 terminological entries in English and French, with equivalent terms in 16 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

**IEC Glossary - std.iec.ch/glossary**
67 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

# IEC 62769-109-1

Edition 2.0 2020-04
REDLINE VERSION

# INTERNATIONAL STANDARD





## Field device integration (FDI) – Part 109-1: Profiles – HART® and WirelessHART®

INTERNATIONAL ELECTROTECHNICAL COMMISSION

ICS 25.040.40; 35.100.05

ISBN 978-2-8322-8291-5

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

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

# FIELD DEVICE INTEGRATION (FDI) –

# Part 109-1: Profiles – HART® and WirelessHART®

## 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. A vertical bar appears in the margin wherever a change has been made. Additions are in green text, deletions are in strikethrough red text.**

International Standard IEC 62769-109-1 has been prepared by subcommittee 65E: Devices and integration in enterprise systems, of IEC technical committee 65: Industrial-process measurement, control and automation.

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

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

a) support for generic protocol extension for faster adoption of other technologies;
b) support for Package Developers to build EDDs targeted for today's EDD bases system under a single development tool.

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

| CDV | Report on voting |
|---|---|
| 65E/624/CDV | 65E/687A/RVC |

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

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

A list of all parts in the 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.

## INTRODUCTION

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

a) method for the supplying and installation of device-specific functionalities, see Patent Family DE10357276;

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

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

d) extensible device object model, see Patent Family US12/893,680.

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

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

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

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

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

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

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

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

# FIELD DEVICE INTEGRATION (FDI) –

# Part 109-1: Profiles – HART® and WirelessHART®

## 1 Scope

This part of IEC 62769 specifies an FDI profile of IEC 62769 for IEC 61784-1_CP 9/1 (HART®)[1] and IEC 61784-1_CP 9/2 (WirelessHART®)[1].

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

IEC 62769-4:~~2015~~ [2], *Field device integration (FDI) – Part 4: FDI Packages*

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

IEC 62769-5, *Field device integration (FDI) – Part 5: FDI Information Model*

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

IEC 62769-7, *Field device integration (FDI) – Part 7: FDI Communication Devices*

~~NOTE IEC 62769-7 is technically identical to FDI-2027.~~

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

### 3.1 Terms and definitions

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

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

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



---

1 HART® and wirelessHART® are ~~the trade names of the non-profit consortium HART Communication Foundation, Austin,Texas, USA~~ the registered trademark of FieldComm Group. 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~~ the product named. ~~Compliance does not require use of the trade names. Use of the trade names requires permission of the trade name holder.~~ Equivalent products may be used if they can be shown to lead to the same results.

2 Under preparation. Stage at the time of publication: IEC/RFDIS 62769-4:2020.

### 3.2 Abbreviated terms and acronyms

For the purposes of this document, the following abbreviations apply:

| | |
|---|---|
| CP | Communication profile (see IEC 61784-1 or IEC 61784-2) |
| CPF | Communication profile family (see IEC 61784-1 or IEC 61784-2) |
| EDD | Electronic device description (see IEC 61804) |
| EDDL | Electronic device description language (see IEC 61804) |
| FDI | Field device integration |
| FSK | Frequency-Shift-Keying |
| HCF | HART Communication Foundation |
| ID | Identification |
| IM | Information Model |
| IP | Internet protocol |
| PDU | Protocol data unit |
| PSK | Phase-Shift-Keying |
| TCP | Transmission Control Protocol (see IETF RFC 793) |
| UDP | User Datagram Protocol (see IETF RFC 768) |
| XML | ~~Extended~~ Extensible markup language |

### 3.3 Conventions

#### 3.3.1 EDDL syntax

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

#### 3.3.2 XML syntax

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

EXAMPLE `<xs:simpleType name="ExampleT">`

#### 3.3.3 Capitalizations

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

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:

- FDI 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 like:

- PROFILE_ID or
- HART_Network.

# 4 Profile for CP 9/1 (HART®) or CP 9/2 (WirelessHART®)

## 4.1 General

This profile document to the FDI specification in IEC 62769 ~~specifies~~ selects the protocol specifics needed for FDI Packages describing FDI Communication Servers, gateways and devices.

## 4.2 Catalog profile

### 4.2.1 Protocol support file

~~No additional file is required for CP 9/1 or CP 9/2 FDI Device Packages.~~

Device information files provide metadata for the dynamic runtime data that is supplied by the device. This metadata is a subset of information that is contained in the EDD. The device information files may be extracted from the package by light-weight gateway or server implementations to exchange runtime device information with minimal implementation overhead. Device information files do not replace the need for the EDD part because device information files only contain a subset of the information from the EDD, and do not provide any user-interface elements.

The formats of the Device Information Files are described in Table 1.

**Table 1 – Device Information Files**

| Part | Content |
|---|---|
| Content Type | Not specified here |
| Root Namespace | Not specified here |
| Source Relationship | http://fdi-cooperation.com/2010/relationships/attachment-protocol |
| Filename | Not specified here |

The Device Information Files are specified in FCG AG21073.

### 4.2.2 CommunicationProfile definition

IEC 62769-4 defines a CommunicationProfileT ~~enumeration~~ string type for the Catalog XML schema. Table 2 defines the CP 9/1 specific values for this enumeration.

**Table 2 – CommunicationProfile definition**

| CommunicationProfile | Description |
|---|---|
| hart_fsk | CP 9/1 device type that supports an FSK physical layer (Frequency-Shift-Keying on a pair of wires) |
| hart_psk | CP 9/1 device type that supports a PSK physical layer (Phase-Shift-Keying on a pair of wires). Devices supporting PSK are required to also inherently support FSK, and therefore PSK will always be used only in combination with at least FSK. |

| CommunicationProfile | Description |
|---|---|
| hart_wirelesshart | CP 9/2 device type that supports a wireless physical layer (communication between device and gateway). |
| hart_ip | CP 9/1 device type that supports Internet Protocol (these devices support both TCP and UDP). |
| hart_rs485 | CP 9/1 device type that supports EIA-485 digital communication. |
| hart_ir | CP 9/1 device type that supports an Infrared physical layer (designed to be transparent to FSK masters – included only as information to indicate that the device supports IR connection). |
| NOTE It is possible for a single CP 9/1 device to support more than one CP. | |

### 4.2.3 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 3 focus on catalog content that is vendor independent.

**Table 3 – Catalog values for profile devices**

| Element | Attribute | Content |
|---|---|---|
| PackageType | — | Profile |
| DeviceModel | — | Empty |
| Manufacturer | — | Empty |

### 4.2.4 Protocol version information

IEC 62769-4 defines an element type named InterfaceT for the Catalog XML Schema. Element type InterfaceT contains an element named Version which is supposed to provide version information about the applied communication protocol profile. The value has to follow the IEC 62769-4 defined version information schema defined in element type VersionT. Subclause 4.2.4 describes how to apply the currently known protocol versions for CP 9/1 or CP 9/2 entries in the device catalog. The general rule is to use the Universal Revision of the protocol for the major version part of VersionT, and the value "0" for the minor version and build parts. Table 4 shows the Protocol Version Information.

**Table 4 – Protocol Version Information**

| Protocol Version | InterfaceT Version value |
|---|---|
| HART Universal Revision 5 | 5.0.0 |
| HART Universal Revision 6 | 6.0.0 |
| HART Universal Revision 7 | 7.0.0 |
| The Protocol Version defined in a package is provided for informational purposes only and shall not be used to determine the compatibility or applicability of a package to a device. | |

## 4.3 Associating a Package with a CP 9/1 device

### 4.3.1 Device type identification mapping

CP 9/1 device types are uniquely identified by parameters Manufacturer, Model and DeviceRevision. These parameters are used to associate a given device instance to an FDI Device Package. These parameters are mapped to the FDI Device Package Catalog according to Table 5.

**Table 5 – Device type catalog mapping**

| ~~Catalog element~~ | ~~CP mapping (see 5.4.2)~~ |
|---|---|
| ~~Manufacturer element of InterfaceT (IEC 62769-4:2015, Clause E.11)~~ | ~~Manufacturer~~<br>~~String format "0xdddd" where dddd is the Manufacturer number in hexadecimal format.~~ |
| ~~DeviceModel element of InterfaceT (IEC 62769-4:2015, Clause E.11)~~ | ~~Model~~<br>~~String format "0xdddd" where dddd is the Model number in hexadecimal format.~~ |
| ~~DeviceRevision element~~<br>~~ListOfSupportedDeviceRevisionsT (IEC 62769-4:2015, Clause E.21)~~ | ~~DeviceRevision~~<br>~~String format "x.0.0" where x is the DeviceRevision in decimal format (no leading zeros).~~ |

| Catalog element | CP mapping (SeeTable A.) |
|---|---|
| Manufacturer element of InterfaceT<br>(IEC 62769-4:–, Clause E.10) | Manufacturer<br>String format "0xdddd" where dddd is the MANUFACTURER_ID in hexadecimal format. |
| DeviceModel element of InterfaceT<br>(IEC 62769-4:–, Clause E.10) | Model<br>String format "0xdddd" where dddd is the DEVICE_TYPE in hexadecimal format. |
| DeviceRevision element<br>ListOfSupportedDeviceRevisionsT<br>(IEC 62769-4:–, Clause E.20) | DeviceRevision<br>String format "x.0.0" where x is the DEVICE_REVISION in decimal format (no leading zeros). |

### 4.3.2 Device type revision mapping

Each device type is identified as per 4.3.1. If a package with matching DeviceRevision is not available, any CP 9/1 FDI package for a corresponding manufacturer and model shall always be compatible with a field device as long as the device revision of the field device is equal to or greater than the device revision specified in the FDI package.

## 4.4 Information Model mapping

### 4.4.1 ProtocolType definition

Table 6 defines the ProtocolType used to identify CP 9/1 network communications.

**Table 6 – ProtocolType HART definition**

| Attribute | Value | | | | |
|---|---|---|---|---|---|
| BrowseName | HART | | | | |
| IsAbstract | False | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Inherits the properties of ProtocolType defined in IEC 62541-100. | | | | | |

### 4.4.2 DeviceType mapping

Each device type inherits the properties of DeviceType. The mapping of the inherited properties from DeviceType is defined in Table 7.

**Table 7 – Inherited DeviceType Property mapping**

| ~~Property~~ | ~~Foundation mapping~~ |
|---|---|
| ~~SerialNumber~~ | ~~3-byte unique ID of a device, returned in bytes 9 to 11 of Command 0 or Command 11 or Command 21~~ |
| ~~RevisionCounter~~ | ~~2-byte configuration change counter, returned in bytes 14 and 15 of Command 0 or Command 11 or Command 21.~~<br>~~-1 (not defined) for HART revision 5 devices~~ |
| ~~Manufacturer~~ | ~~For HART revision 7 or higher devices:~~<br>~~2-byte manufacturer code of a device, returned in bytes 17 and 18 of Command 0 or Command 11 or Command 21.~~<br>~~For HART revision 6 or lower devices:~~<br>~~Most significant byte shall be fixed to 0, and the least significant byte is returned in byte 1 of Command 0 or Command 11 or Command 21~~ |
| ~~Model~~ | ~~2-byte extended device type of a device, returned in bytes 1 and 2 of Command 0 or Command 11 or Command 21~~ |
| ~~DeviceManual~~ | ~~Entry text string (not supported)~~ ~~a~~ |
| ~~DeviceRevision~~ | ~~1-byte device revision level of a device, returned in byte 5 of Command 0 or Command 11 or Command 21~~ |
| ~~SoftwareRevision~~ | ~~1-byte software revision level of a device, returned in byte 6 of Command 0 or Command 11 or Command 21~~ |
| ~~HardwareRevision~~ | ~~1-byte hardware revision level of a device, returned in byte 7 (only 5 most significant bits) of Command 0 or Command 11 or Command 21~~ |
| ~~a~~ ~~Device manuals are exposed as attachments of the FDI Device Package.~~ | |

| Property | Foundation mapping |
|---|---|
| SerialNumber | Unique ID of a device, mapped to SERIAL_NUMBER of IdentificationT. |
| RevisionCounter | Configuration change counter, mapped to REV_COUNTER of IdentificationT |
| 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 | Entry text string (not supported) [a] |
| DeviceRevision | Device revision level of a device, mapped to DEVICE_REVISION of IdentificationT |
| SoftwareRevision | Software revision level of a device, mapped to SOFTWARE_REVISION of IdentificationT |
| HardwareRevision | Hardware revision level of a device, mapped to HARDWARE_REVISION of IdentificationT |
| [a] Device manuals are exposed as attachments of the FDI Device Package. | |



### 4.4.3 FunctionalGroup Identification definition

As defined in IEC 62541-100, each device representation in the FDI Server hosted Information Model shall contain a protocol specific FunctionalGroup called Identification. This FunctionalGroup organizes variables found in the device type instance. The FunctionalGroup Identification for CP 9/1 is defined in Table 8.

**Table 8 – Identification parameters**

| BrowseName | DataType | Optional/Mandatory |
|---|---|---|
| MANUFACTURER_ID | UInt16 | Mandatory |

| | | |
|---|---|---|
| DEVICE_TYPE | UInt16 | Mandatory |
| DEVICE_REVISION | UInt8 | Mandatory |
| UNIVERSAL_REVISION | UInt8 | Optional |
| SERIAL_NUMBER | UInt24 | Optional |
| HARDWARE_REVISION | UInt8 | Optional |
| SOFTWARE_REVISION | UInt8 | Optional |
| REVISION_COUNTER | UInt16 | Optional |

## 4.5 Topology elements

### 4.5.1 ConnectionPoint definition

#### 4.5.1.1 General

CP 9/1 devices can support up to five different ConnectionPoint types that are used for network communications.

#### 4.5.1.2 HART_TP5, HART_TP6, HART_TP7

The ConnectionPoint types HART_TP5, HART_TP6, and HART_TP7 shall be used to identify CP 9/1 token passing network communication and are defined in Table 9. HART_TP5, HART_TP6, and HART_TP7 all contain the same properties, but each provides different qualification information for some of the properties (described below). The Protocol Version (UNIVERSAL_REVISION) described in 4.2.4 can be used as an aid to determine which of the three token passing Connection Point types is the most appropriate. CP 9/1 token passing communications can be used on a variety of physical layers. FSK, PSK, RS485, and Infrared physical layer connections shall all use the HART_TP connection type. The ConnectionPoint types HART_TP5, HART_TP6, and HART_TP7 are subtypes of abstract type ConnectionPointType defined in IEC 62769-5.

The DevAddr property shall be the long address (5 bytes) for the device and is the only parameter necessary to communicate with the field device.

The DevMfg property shall be the 2-byte Manufacturer ID and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevType property shall be the 2-byte extended device type and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevRev property shall be the device revision and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevTag property shall be the long tag for HART® protocol version 6 or 7 devices. The DevTag property shall be the tag for protocol version 5 devices. The DevTag property can be used to help automate the process of assigning live devices in the scan list to offline placeholders. HART_TP5 Connection Points shall limit the DevTag to 8 characters in length. HART_TP6 and HART_TP7 Connection Points shall limit the DevTag to 32 characters in length.

The DevPollAddr property shall be the poll address and can be used to identify which device is located at a specific poll address. HART_TP5 Connection Points shall be limited to values between 0 and 15 for the DevPollAddr property. HART_TP6 Connection Points shall be limited to values between 0 and 31 for the DevPollAddr property. HART_TP7 Connection Points shall be limited to values between 0 and 63 for the DevPollAddr property.

For forward compatibility, a lower revision HART_TP Connection Point is compatible and can be used for a higher universal revision device connection. For example, if a future HART universal revision 8 device is encountered, and no HART_TP8 is available in the FDI server, HART_TP7 will be compatible and shall be used to connect to the device. If the Protocol Version (i.e. the Universal Revision) is unknown for any reason, the HART_TP5 Connection Point can be used, and will be forward compatible to later universal revisions.

**Table 9 – ConnectionPointType HART_TP definition**

| Attribute | Value | | | | |
|---|---|---|---|---|---|
| BrowseName | ConnnectionPoint_HART_TP5 or ConnnectionPoint_HART_TP6 or ConnnectionPoint_HART_TP7 | | | | |
| IsAbstract | False | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Inherits the properties of ConnectionPointType defined in IEC 62769-5 | | | | | |
| HasProperty | Variable | DevAddr | UInt40 | PropertyType | Mandatory |
| HasProperty | Variable | DevMfg | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevType | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevRev | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevTag | String | PropertyType | Optional |
| HasProperty | Variable | DevPollAddr | UInt8 | PropertyType | Optional |

The ConnectionPoint type HART_TP5, HART_TP6, and HART_TP7 shall be described by an EDD element contained in a Communication Device related FDI Package that can drive a CP 9/1 network. Actual ConnectionPoint properties are declared by VARIABLE constructs grouped together in a COLLECTION named ConnectionPoint_HART_TP5, ConnectionPoint_HART_TP6, or ConnectionPoint_HART_TP7. The following EDDL source code is an example describing a TP5 Connection Point.

```
COMPONENT ConnectionPoint_HART_TP5
{
  LABEL "HART TP Connection Point";
  CLASSIFICATION NETWORK_CONNECTION_POINT;
  CAN_DELETE FALSE;
  PROTOCOL HART;
  CONNECTION_POINT ConnectionPoint_TP5;
}

VARIABLE DevAddr
{
  LABEL "Address";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER(5);
  HANDLING READ & WRITE;
}

VARIABLE DevMfg
{
  LABEL "Manufacturer";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER(2);
  HANDLING READ & WRITE;
}
```

```
VARIABLE DevType
{
  LABEL "Device Type";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER(2);
  HANDLING READ & WRITE;
}

VARIABLE DevRev
{
  LABEL "Device Revision";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER;
  HANDLING READ & WRITE;
}

VARIABLE DevTag
{
  LABEL "Tag";
  CLASS DEVICE;
  TYPE ASCII(32);
  HANDLING READ & WRITE;
}

VARIABLE DevPollAddr
{
  LABEL "Poll Address";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER
  {
    MAX_VALUE 15; //Define appropriate max value for various revisions
  }
  HANDLING READ & WRITE;
}

COLLECTION ConnectionPoint_TP5
{
  LABEL "Connection Point";
  MEMBERS
  {
    ADDRESS,    DevAddr,     "Device Address";
    MFG,        DevMfg,      "Manufacturer";
    DEV_TYPE,   DevType,     "Device Type";
    DEV_REV,    DevRev,      "Device Revision";
    TAG,        DevTag,      "Device Tag";
    POLL_ADDR,  DevPollAddr, "Poll Address";
  }
}
```



#### 4.5.1.3 HART_IP

The ConnectionPoint type HART_IP shall be used to identify CP 9/1 IP network communication and is defined in Table 10. HART_IP communications can be used on a variety of physical layers. Ethernet connections shall all use the HART_IP connection type. Additional physical layers developed in the future may also use the HART_IP connection type. The ConnectionPoint type HART_IP is a subtype of abstract type ConnectionPointType defined in IEC 62769-5.

The IPAddress property shall indicate the IP Address (up to 16 bytes) used for the connection.

The IPVersion property shall indicate the version of IP used for the connection (either 4 or 6).

The IPPort property shall be the IP port number for the connection. The default port number used for HART IP is 5 094.

The DevAddr property shall be the long address (5 bytes) for the device.

The DevMfg property shall be the 2-byte Manufacturer ID and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevType property shall be the 2-byte extended device type and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevRev property shall be the device revision and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevTag property shall be the long tag (with maximum 32 characters) and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

**Table 10 – ConnectionPointType HART_IP Definition**

| Attribute | Value | | | | |
|---|---|---|---|---|---|
| BrowseName | ConnnectionPoint_HART_IP | | | | |
| IsAbstract | False | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Inherits the properties of ConnectionPointType defined in IEC 62769-5 | | | | | |
| HasProperty | Variable | IPAddress | ByteString | PropertyType | Mandatory |
| HasProperty | Variable | IPVersion | UInt8 | PropertyType | Mandatory |
| HasProperty | Variable | IPPort | UInt16 | PropertyType | Mandatory |
| HasProperty | Variable | DevAddr | UInt40 | PropertyType | Mandatory |
| HasProperty | Variable | DevMfg | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevType | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevRev | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevTag | String | PropertyType | Optional |



The ConnectionPoint type HART_IP shall be described by an EDD element contained in a Communication Device related FDI Package that can drive a CP 9/1 network. Actual ConnectionPoint properties are declared by VARIABLE constructs grouped together in a COLLECTION named ConnectionPoint_HART_IP. The following EDDL source code is an example describing an IP Connection Point.

```
COMPONENT ConnectionPoint_HART_IP
{
  LABEL "HART IP Connection Point";
  CLASSIFICATION NETWORK_CONNECTION_POINT;
  CAN_DELETE FALSE;
  PROTOCOL HART;
  CONNECTION_POINT ConnectionPoint_IP;
}
```

```
ARRAY IPAddress
{
  LABEL "IP Address";
  CLASS DEVICE;
  TYPE OCTET(16);
  HANDLING READ & WRITE;
}

VARIABLE IPVersion
{
  LABEL "IP Version";
  CLASS DEVICE;
  TYPE ENUMERATED
  {
    { 4, "IPv4" },
    { 6, "IPv6" }
  }
  HANDLING READ & WRITE;
}

VARIABLE IPPort
{
  LABEL "IP Port";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER (2);
  DEFAULT_VALUE 5 094;
  HANDLING READ & WRITE;
}

COLLECTION ConnectionPoint_IP
{
  LABEL "Connection Point";
  MEMBERS
  {
    IPADDRESS, IPAddress, "IP Address";
    IPVERSION, IPVersion, "IP Version";
    IPPORT,    IPPort,    "IP Port";
    ADDRESS,   DevAddr,   "Device Address";
    MFG,       DevMfg,    "Manufacturer";
    DEV_TYPE,  DevType,   "Device Type";
    DEV_REV,   DevRev,    "Device Revision";
    TAG,       DevTag,    "Device Tag";
  }
}
```



#### 4.5.1.4 HART_TDMA

The ConnectionPoint type HART_TDMA shall be used to identify CP 9/2 time division media access network communication and is defined in Table 11. HART_TDMA communications can be used on a variety of physical layers. The ConnectionPoint type HART_TDMA is a subtype of abstract type ConnectionPointType defined in IEC 62769-5. WirelessHART connections shall all use the HART_TDMA connection type. Additional physical layers developed in the future may also use the HART_TDMA connection type.

The Network property shall be the network ID for the network.

The DevAddr property shall be the long address (5 bytes) for the device.

The DevMfg property shall be the 2 byte Manufacturer ID, and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevType property shall be the 2 byte extended device type, and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevRev property shall be the device revision and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevTag property shall be the long tag (with maximum 32 characters) and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

**Table 11 – ConnectionPointType HART_TDMA Definition**

| Attribute | Value | | | | |
|---|---|---|---|---|---|
| BrowseName | ConnnectionPoint_HART_TDMA | | | | |
| IsAbstract | False | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Inherits the properties of ConnectionPointType defined in IEC 62769-5 | | | | | |
| HasProperty | Variable | Network | UInt16 | PropertyType | Mandatory |
| HasProperty | Variable | DevAddr | UInt40 | PropertyType | Mandatory |
| HasProperty | Variable | DevMfg | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevType | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevRev | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevTag | String | PropertyType | Optional |

The ConnectionPoint type HART_TDMA shall be described by an EDD element contained in a Communication Device related FDI Package that can drive a CP 9/2 network. Actual ConnectionPoint properties are declared by VARIABLE constructs grouped together in a COLLECTION named ConnectionPoint_HART_TDMA. The following EDDL source code is an example describing a TDMA Connection Point.

```
COMPONENT ConnectionPoint_HART_TDMA
{
  LABEL "HART TDMA Connection Point";
  CLASSIFICATION NETWORK_CONNECTION_POINT;
  CAN_DELETE FALSE;
  PROTOCOL HART;
  CONNECTION_POINT ConnectionPoint_TDMA;
}

VARIABLE Network
{
  LABEL "Network ID";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER (2);
  HANDLING READ & WRITE;
```

```
}

COLLECTION ConnectionPoint_TDMA
{
  LABEL "Connection Point";
  MEMBERS
  {
    NETWORK,  Network, "Network ID";
    ADDRESS,  DevAddr, "Device Address";
    MFG,      DevMfg,  "Manufacturer";
    DEV_TYPE, DevType, "Device Type";
    DEV_REV,  DevRev,  "Device Revision";
    TAG,      DevTag,  "Device Tag";
  }
}
```

### 4.5.2 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 FDI Communication Server.

```
COMPONENT CommunicationServer_HART
{
  LABEL "HART Communication Server";
  CAN_DELETE TRUE;
  CLASSIFICATION NETWORK_COMPONENT;
  COMPONENT_RELATIONS { Communication_Device_Setup_HART}
}

COMPONENT_RELATION Communication_Device_Setup_HART
{
  LABEL "Relation between Device and Communication Device";
  RELATION_TYPE CHILD_COMPONENT;
  COMPONENTS
  {
    CommunicationDevice_HART{AUTO_CREATE 1;}
  }
  MINIMUM_NUMBER 1;
  MAXIMUM_NUMBER 4;
}
```

Semantics of the EDDL constructs shown above are described in IEC 62769-7. The EDDL COMPONENT will be utilized by the FDI server and FDI Communication Server to create an instance of type CommunicationServerType as described in IEC 62769-7.

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 is an example for a CP 9/1 communication device.

```
COMPONENT CommunicationDevice_HART
{
  LABEL "HART Communication Device";
  CAN_DELETE TRUE;
  CLASSIFICATION NETWORK_COMPONENT;
  COMPONENT_RELATIONS { ServiceProviderRelation_HART}
  BYTE_ORDER BIG_ENDIAN;
}
```

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

Semantics of the EDDL constructs shown above are described in IEC 62769-7. The EDDL COMPONENT will be utilized by the FDI server and FDI Communication Server to create an instance of type CommunicationServerChannelType as described in IEC 62769-7.

### 4.5.3 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 is an example for a CP 9/1 communication service provider component.

The component reference (ConnectionPoint_HART_IP) corresponds to one of the related Connection Point definitions in 4.5.1. The attribute BYTE_ORDER value is to be set according to the protocol.

```
COMPONENT CommunicationServiceProvider_HART
{
  LABEL "HART Communication Service Provider";
  CAN_DELETE FALSE;
  CLASSIFICATION NETWORK_COMMUNICATION_SERVICE_PROVIDER;
  COMPONENT_RELATIONS { ServiceProviderConnectionPointRelation_HART }
  BYTE_ORDER BIG_ENDIAN;
}

COMPONENT_RELATION ServiceProviderConnectionPointRelation_HART
{
  LABEL "Relation between communication service and connection point";
  RELATION_TYPE CHILD_COMPONENT;
  ADDRESSING
  {
    IPAddress
  }
  COMPONENTS
  {
    ConnectionPoint_HART_IP { AUTO_CREATE 1; }
  }
  MINIMUM_NUMBER 1;
  MAXIMUM_NUMBER 1;
}
```

Semantics of the EDDL constructs shown above are described in IEC 62769-7. The EDDL COMPONENT will be utilized by the FDI server and FDI Communication Server to create an instance of type CommunicationServiceType as described in IEC 62769-7.

### 4.5.4 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. The following EDDL source code is an example describing a network.

```
COMPONENT Network_HART
{
  LABEL "HART Network";
  CAN_DELETE TRUE;
  CLASSIFICATION NETWORK;
  COMPONENT_RELATIONS { NetworkConnectionPointRelation_HART }
}

COMPONENT_RELATION NetworkConnectionPointRelation_HART
{
  LABEL "Relation between network and connection point";
  RELATION_TYPE CHILD_COMPONENT;
  ADDRESSING {DevPollAddr}
  COMPONENTS
  {
    ConnectionPoint_HART_TP5
    {
      MAXIMUM_NUMBER 16;
    },
    ConnectionPoint_HART_TP6
    {
      MAXIMUM_NUMBER 32;
    },
    ConnectionPoint_HART_TP7
    {
      MAXIMUM_NUMBER 64;
    }
  }
  MINIMUM_NUMBER 1;
  MAXIMUM_NUMBER 64;    //Defined to limitations of the comm. device
}
```

Semantics of the EDDL constructs shown above are described in IEC 62769-7. The EDDL COMPONENT will be utilized by the FDI server and FDI Communication Server to create an instance of type NetworkType as described in IEC 62769-5. Some communication devices may organize multiple IO cards as individual networks, while others may present a single network that comprises all Connection Points across all IO cards.

## 4.6 Methods

### 4.6.1 Methods for FDI Communication Servers

#### 4.6.1.1 General

The FDI Communication Server shall implement services according to the method signatures described in 4.6.1 and according to the Information Model.

#### 4.6.1.2 Connect

Table 12 shows the Method Connect arguments.

**Signature:**

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

**Table 12 – Method Connect arguments**

| Argument | Description |
|---|---|
| CommunicationRelationId | The argument value contains the nodeId of the Device ConnectionPoint representing the connection between a device and a physical network which is directly connected to the FDI Communication Server hardware. The nodeId allows finding the direct parent-child relation. |
| LongAddress | The argument name shall match with the corresponding attribute name defined for the ConnectionPoint which is described by a corresponding EDD element specified in 4.5.1. The argument holds the device's long (5-byte) address. The value can be obtained by the method Scan. |
| ServiceError | 0: OK/execution finished, connection established successfully<br>-1: Connect Failed/canceled by caller<br>-2: Call Failed/unknown service ID<br>-3: Connect Failed/device not found<br>-4: Connect Failed/invalid device node address<br>-5: Connect Failed/invalid device identification |
| NOTE 1 IEC 62769-7, defines the argument AddressData of the Connect Method as an array of Variant. The LongAddress arguments defined in the table are represented as entries of the Variant array in the order they are specified above.<br><br>NOTE 2 IEC 62769-7, defines the argument DeviceInformation as a protocol specific argument list in which the Connect method stores the resulting data. The DeviceInformation argument is defined as an array of Variant. The DeviceInformation argument is not used for CP 9/1 devices. | |

#### 4.6.1.3 Disconnect

Table 13 shows the Method Disconnect arguments.

**Signature:**

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

**Table 13 – Method Disconnect arguments**

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

#### 4.6.1.4 Transfer

Table 14 shows the Method Transfer arguments.

**Signature**

```
Transfer(
    [in]  ByteString CommunicationRelationId,
    [in]  UInt16     Command,
    [in]  ByteString Request,
    [out] ByteString Reply,
    [out] Int32      ServiceError);
```

**Table 14 – 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 which is directly connected to the FDI Communication Server hardware. The nodeId allows finding the direct parent-child relation. |
| Command | Command number of the CP 9/1 command to be sent. The FDI Communication Server is required to generate and send the PDU in the proper format according to CP 9/1 protocol specifications (e.g. taking care of command expansion). |
| Request | An array of bytes representing the data bytes in the CP 9/1 command request (e.g. starting after the byte count, and ending before the checksum). |
| Reply | An array of bytes representing the data bytes in the CP 9/1 command response (e.g. starting after the byte count, and ending before the checksum). |
| ServiceError | 0: OK/execution finished~~, Reply contains the result~~<br>-1: Transfer Failed/cancelled by caller<br>-2: Call Failed/unknown service ID<br>-3: Transfer Failed/no existing communication relation<br>-4: Transfer Failed/invalid communication relation identifier<br>-5: Transfer Failed/invalid Request content<br>-6: Transfer Failed/invalid Reply format |
| NOTE 1 IEC 62769-7 defines the argument SendData of the Transfer Method as an array of Variant. The arguments Command and Request defined in the table are represented as entries of the Variant array in the order they are specified above.<br><br>NOTE 2 IEC 62769-7 defines the argument ReceiveData of the Transfer Method as an array of Variant. The argument Reply defined in the table is represented as the entry of the Variant array. | |

#### 4.6.1.5 GetPublishedData

CP 9/1 burst and event notification messages represent unsolicited messages as defined in IEC 62769-7. Table 15 shows the Method GetPublishedData arguments.

**Signature:**

```
GetPublishedData(
    [in]  ByteString CommunicationRelationId,
    [out] UInt16     Command,
    [out] ByteString Reply,
    [out] DateTime   TimeStamp,
    [out] Int32      ServiceError);
```

**Table 15 – Method GetPublishedData 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. |
| Command | Command number of the CP 9/1 command published. The FDI Communication Server is required to parse the PDU in various formats according to CP 9/1 Protocol standards (e.g. taking care of command expansion). |
| Reply | An array of bytes representing the data bytes in the published CP 9/1 command response (e.g. starting after the byte count and ending before the checksum). |
| TimeStamp | Time at which the published data was captured. |
| ServiceError | 0: OK/execution finished~~, Reply contains the result~~<br>-1: GetPublishedData Failed/cancelled by caller<br>-2: Call Failed/unknown service ID<br>-3: GetPublishedData Failed/not supported<br>-4: GetPublishedData Failed/no existing communication relation<br>-5: GetPublishedData Failed/invalid communication relation identifier<br>-8: GetPublishedData Failed/no burst / event data published |
| NOTE IEC 62769-7 defines the argument ReceiveData of the GetPublishedData Method as an array of Variant. The arguments Command and Reply defined in the table are represented as entries of the Variant array in the order they are specified above. | |

#### 4.6.1.6 SetAddress

Table 16 shows the Method SetAddress arguments.

**Signature**

```
SetAddress(
   [in]  UInt8 OldPollAddress,
   [in]  UInt8 NewPollAddress,
   [out] Int32 ServiceError);
```

**Table 16 – Method SetAddress arguments**

| Argument | Description |
|---|---|
| OldPollAddress | The argument value holds the current address of a device. Allowed values are 0..63. |
| NewPollAddress | The argument value holds the new address for a device. Allowed values are 0..63. |
| ServiceError | 0: OK/execution finished successfully<br>-1: SetAddress Failed/cancelled by caller<br>-2: Call Failed/unknown service ID<br>-3: SetAddress Failed/not initialized<br>-4: SetAddress Failed/not connected to a network<br>-5: SetAddress Failed/no device found responding to oldAddress<br>-6: SetAddress Failed/duplicate address error<br>-7: SetAddress Failed/device did not accept new address<br>-8: SetAddress Failed/invalid oldAddress<br>(in terms of syntax, data type, data format, and so on)<br>-9: SetAddress Failed/invalid newAddress<br>(in terms of syntax, data type, data format, and so on)<br>-10: SetAddress Failed/not possible in status connected |
| NOTE 1 IEC 62769-7 defines the argument OldAddress of the SetAddress Method as an array of Variant. The argument OldPollAddress defined in the table is represented as the entry of the Variant array.<br>NOTE 2 IEC 62769-7 defines the argument NewAddress of the SetAddress Method as an array of Variant. The argument NewPollAddress defined in the table is represented as the entry of the Variant array. | |

#### 4.6.1.7 Scan

The method signature specified in IEC 62769-7 applies. The corresponding topologyScanResult schema is specified in Annex A.

#### 4.6.1.8 ResetScan

The method signature specified in IEC 62769-7 applies. The corresponding topologyScanResult schema is specified in Annex A.

### 4.6.2 Methods for Gateways

#### 4.6.2.1 General

The method signatures described in 4.6.2 shall be implemented in the EDD element IEC 62769-4 contained in a Gateway related FDI Package containing the communication device definitions.

#### 4.6.2.2 Connect

Table 17 shows the Method Connect arguments.

**Signature:**

```
METHOD BeginConnect(
   DD_String      CommunicationRelationId,
   DD_String      LongAddress,
   unsigned long  ServiceID,
   unsigned long &DelayForNextCall,
   long          &ServiceError)
{
   DEFINITION {<Gateway specific implementation>}
}

METHOD EndConnect(
   DD_String      CommunicationRelationId,
   unsigned long  ServiceID,
   unsigned long &DelayForNextCall,
   long          &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}

METHOD CancelConnect(
   DD_String      CommunicationRelationId,
   unsigned long  ServiceID,
   long          &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}
```

**Table 17 – 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 FDI Communication Server hardware. The nodeId allows finding the direct parent-child relation. |
| LongAddress | The argument name shall match with the corresponding attribute name defined for the ConnectionPoint which is described by a corresponding EDD element specified in 4.5.1. The argument holds the devices long (5-byte) address. The value can be obtained by the method Scan. |
| 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<br>0: OK/execution finished, connection established successfully<br>-1: Connect Failed/cancelled by caller<br>-2: Call Failed/unknown service ID<br>-3: Connect Failed/device not found<br>-4: Connect Failed/invalid device address<br>-5: Connect Failed/invalid device identification |

#### 4.6.2.3 Disconnect

Table 18 shows the Method Disconnect arguments.

**Signature:**

```
METHOD Disconnect(
   DD_String  communicationRelationId,
   Long       &serviceError)
{
   DEFINITION {<Gateway specific implementation>}
}
```

**Table 18 – 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 FDI Communication Server hardware. The nodeId allows finding the direct parent-child relation. |
| ServiceError | 0: OK/disconnect finished successfully<br>-1: Disconnect Failed/no existing communication relation<br>-2: Disconnect Failed/invalid communication relation identifier |

#### 4.6.2.4 Transfer

Table 19 shows the Method Transfer arguments.

**Signature:**

```
METHOD BeginTransfer(
   DD_String       CommunicationRelationId,
   unsigned short  Command,
   DD_String       Request,
   DD_String       &Reply,
   unsigned long   ServiceId,
   unsigned long   &DelayForNextCall,
   long            &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}

METHOD EndTransfer(
   DD_String       CommunicationRelationId,
   DD_String      &Reply,
   unsigned long   ServiceId,
   unsigned long  &DelayForNextCall,
   long           &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}

METHOD CancelTransfer(
   DD_String       CommunicationRelationId,
   unsigned long   ServiceId,
   long           &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}
```

**Table 19 – 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 which is directly connected to the FDI Communication Server hardware. The nodeId allows finding the direct parent-child relation. |
| Command | Command number of the CP 9/1 command to be sent. The FDI Communication Server is required to generate and send the PDU in the proper format according to CP 9/1 protocol specifications (e.g. taking care of command expansion). |
| Request | An array of bytes representing the data bytes in the CP 9/1 command request (e.g. starting after the byte count, and ending before the checksum). |
| Reply | An array of bytes representing the data bytes in the CP 9/1 command response (e.g. starting after the byte count, and ending before the checksum). |
| 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<br>0: OK/execution finished~~, REPLY contains the result~~<br>-1: Transfer Failed/cancelled by caller<br>-2: Call Failed/unknown service ID<br>-3: Transfer Failed/no existing communication relation<br>-4: Transfer Failed/invalid communication relation identifier<br>-5: Transfer Failed/invalid REQUEST content<br>-6: Transfer Failed invalid REPLY format |

#### 4.6.2.5 GetPublishedData

CP 9/1 burst and event notification messages represent unsolicited messages as defined with IEC 62769-7. Table 20 shows the Method GetPublishedData arguments.

**Signature**

```
METHOD BeginGetPublishedData(
   DD_String          CommunicationRelationId,
   unsigned short &Command,
   DD_String          &Reply,
   TIME               &TimeStamp
   unsigned long       ServiceId,
   unsigned long  &DelayForNextCall,
   long               &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}

METHOD EndGetPublishedData(
   DD_String          CommunicationRelationId,
   unsigned short &Command,
   DD_STRING          &Reply,
   TIME               &TimeStamp
   unsigned long       ServiceId,
   unsigned long  &DelayForNextCall,
   long               &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
```

```
}

METHOD CancelGetPublishedData(
   DD_String       CommunicationRelationId,
   unsigned long   ServiceId,
   long           &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}
```

**Table 20 – Method GetPublishedData 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. |
| Command | Command number of the CP 9/1 command published. The FDI Communication Server is required to parse the PDU in various formats according to CP 9/1 Protocol standards (e.g. taking care of command expansion). |
| Reply | An array of bytes representing the data bytes in the published CP 9/1 command response (e.g. starting after the byte count and ending before the checksum). |
| TimeStamp | Time at which the published data was captured. |
| 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 EndGetPublishedData 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 EndGetPublishedData<br>0: OK/execution finished~~, ReceivedData and TimeStamp contains the result~~<br>-1: GetPublishedData Failed/cancelled by caller<br>-2: Call Failed/unknown service ID<br>-3: GetPublishedData Failed/not supported<br>-4: GetPublishedData Failed/no existing communication relation<br>-5: GetPublishedData Failed/invalid communication relation identifier<br>-8: GetPublishedData Failed/no burst information published |



#### 4.6.2.6 SetAddress

Table 21 shows the Method SetAddress arguments.

**Signature**

```
METHOD BeginSetAddress(
   unsigned char   OldPollAddress,
   unsigned char   NewPollAddress,
   unsigned long   ServiceId,
   unsigned long  &DelayForNextCall,
   long           &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}

METHOD EndSetAddress(
   unsigned long   ServiceId,
   unsigned long  &DelayForNextCall,
```

```
    long           &ServiceError);
{
    DEFINITION {<Gateway specific implementation>}
}

METHOD BeginSetAddress(
    unsigned long  ServiceId,
    long           &ServiceError);
{
    DEFINITION {<Gateway specific implementation>}
}
```

**Table 21 – Method SetAddress arguments**

| Argument | Description |
|---|---|
| OldPollAddress | The argument value holds the current address of a device. The allowed values are 0..63. |
| NewPollAddress | The argument value holds the new address for a device. The allowed values are 0..63. |
| 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<br>0: OK/execution finished successfully<br>-1: SetAddress Failed/cancelled by caller<br>-2: Call Failed/unknown service ID<br>-3: SetAddress Failed/not initialized<br>-4: SetAddress Failed/not connected to a network<br>-5: SetAddress Failed/no device found responding to oldAddress<br>-6: SetAddress Failed/duplicate address error<br>-7: SetAddress Failed/device did not accept new address<br>-8: SetAddress Failed/invalid oldAddress<br>(in terms of syntax, data type, data format, and so on)<br>-9: SetAddress Failed/invalid newAddress<br>(in terms of syntax, data type, data format, and so on)<br>-10: SetAddress Failed/not possible in status connected |



#### 4.6.2.7 Scan

The method signature specified in IEC 62769-7 applies. The HART Gateway business logic shall create the scan result using the following EDD definitions, which shall be included in the communication device COMPONENT definition. The EDD LIST also held in the COMPONENT holds the list of devices corresponding to the data structure defined in the XML schema. This LIST shall be referred to by the SCAN_LIST attribute of the communication device component. The corresponding topologyScanResult schema is specified in Annex A.

```
VARIABLE DevAddr
{
  LABEL "Address";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER(5);
  HANDLING READ & WRITE;
}
```

```
VARIABLE DevMfg
{
  LABEL "Manufacturer";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER(2);
  HANDLING READ & WRITE;
}

VARIABLE DevType
{
  LABEL "Device Type";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER(2);
  HANDLING READ & WRITE;
}

VARIABLE DevRev
{
  LABEL "Device Revision";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER;
  HANDLING READ & WRITE;
}

VARIABLE DevTag
{
  LABEL "Tag";
  CLASS DEVICE;
  TYPE ASCII(32);
  HANDLING READ & WRITE;
}

COLLECTION ScanItemType_TP
{
  MEMBERS
  {
    ADDRESS, DevAddr;
  }
}

COLLECTION DeviceIdentificationType
{
  MEMBERS
  {
    MFG,      DevMfg;
    DEV_TYPE, DevType;
    DEV_REV,  DevRev;
    TAG,      DevTag;
  }
}

COLLECTION ScanResultType
{
  MEMBERS
  {
    ScanItem,             ScanItemType_TP;
    DeviceIdentification, DeviceIdentificationType;
  }
}

LIST TopologyScanResult
{
  TYPE ScanResultType;
```

```
  CAPACITY 64;    //Defined to limitation of the comm. device
}
```

#### 4.6.2.8 ScanNext

The method signature specified in IEC 62769-7 applies. The HART Gateway business logic shall create the scan result using the same EDD definitions defined in 4.6.2.7. The corresponding topologyScanResult schema is specified in Annex A.

# Annex A
(normative)

# Topology scan schema

## A.1 General

The Topology scan result schema specified in Annex A describes the HART 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 IdentificationType

The element content corresponds to the "FunctionalGroup Identification".

The XML schema for IdentificationType is:

```
<xs:complexType name="IdentificationT">
  <xs:attribute name="MANUFACTURER_ID" type="xs:unsignedShort"
use="required">
  </xs:attribute>
  <xs:attribute name="DEVICE_TYPE" type="xs:unsignedShort"
use="required">
  </xs:attribute>
  <xs:attribute name="UNIVERSAL_REVISION" type="xs:unsignedByte"
use="required">
  </xs:attribute>
  <xs:attribute name="DEVICE_REVISION" type="xs:unsignedByte"
use="required">
  </xs:attribute>
  <xs:attribute name="SERIAL_NUMBER" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:unsignedInt">
        <xs:minInclusive value="0"/>
        <xs:maxInclusive value="16777216"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="HARDWARE_REVISION" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:unsignedByte">
        <xs:minInclusive value="0"/>
        <xs:maxInclusive value="3231"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="SOFTWARE_REVISION" type="xs:unsignedByte"
use="required">
  </xs:attribute>
  <xs:attribute name="REV_COUNTER" type="xs:unsignedShort"
default="0">
  </xs:attribute>
  <xs:attribute name="TAG" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:maxLength value="32"/>
      </xs:restriction>
```

```
    </xs:simpleType>
  </xs:attribute>
</xs:complexType>
```

The attributes of IdentificationType are described in Table A.1.

**Table A.1 – Attributes of IdentificationT**

| Attribute | Description |
|---|---|
| MANUFACTURER_ID | Manufacturer identification number.<br>For HART universal revision 7 or higher devices:<br>2-byte manufacturer code of a device, returned in bytes 17 and 18 of Command 0 or Command 11 or Command 21.<br>For HART revision 6 or lower devices:<br>Most significant byte shall be fixed to 0, and the least significant byte is returned in byte 1 of Command 0 or Command 11 or Command 21. |
| DEVICE_TYPE | Manufacturer's model number associated with the resource.<br>For HART universal revision 7 or higher devices:<br>2 byte device type returned in bytes 1 and 2 of Command 0 or Command 11 or Command 21.<br>For HART revision 6 or lower devices:<br>Most significant byte shall be fixed to 0, and the least significant byte is returned in byte 2 of Command 0 or Command 11 or Command 21. |
| UNIVERSAL_REVISION | Universal HART revision associated with the device. Returned in byte 4 of Command 0 or Command 11 or Command 21. |
| DEVICE_REVISION | Manufacturer's device revision number associated with the device. Returned in byte 5 of Command 0 or Command 11 or Command 21. |
| SERIAL_NUMBER | Unique serial number of the device. 3-byte unique ID of a device, returned in bytes 9 to 11 of Command 0 or Command 11 or Command 21. |
| HARDWARE_REVISION | Manufacturer hardware revision associated with the device. Returned in byte 7 (only 5 most significant bits) of Command 0 or Command 11 or Command 21. |
| SOFTWARE_REVISION | Manufacturer firmware revision associated with the device. Returned in byte 6 of Command 0 or Command 11 or Command 21. |
| REV_COUNTER | The configuration change counter associated with the device.<br>For HART universal revision 6 or higher devices:<br>Returned in bytes 14 and 15 of Command 0 or Command 11 or Command 21.<br>For HART revision 5 or lower devices:<br>-1 (not defined). |
| TAG | The configured Identification tag associated with the device.<br>For HART universal revision 6 or higher devices:<br>32 character long tag returned in Command 20 or Command 21.<br>For HART revision 5 or lower devices:<br>8 character tag returned in Command 11 or Command 13. |

## A.3 AddressTypeTP

The XML schema for AddressTypeTP is:

```
<xs:complexType name="AddressTypeTP">
  <xs:complexContent>
    <xs:extension base="hart:LongAddressT">
      <xs:sequence>
        <xs:element name="DevPollAddr" minOccurs="0">
          <xs:simpleType>
            <xs:restriction base="xs:unsignedByte">
              <xs:minInclusive value="0"/>
              <xs:maxInclusive value="63"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The elements of AddressTypeTP are described in Table A.2.

**Table A.2 – Elements of AddressTypeTP**

| Element | Description |
|---|---|
| DevAddr | Holds the 5-byte HART address of the associated device. |
| DevPollAddr | Holds the HART short address used for polling the associated device. |

## A.4 AddressTypeIP

The XML schema for AddressTypeIP is:

```
<xs:simpleType name="IPv4T">
  <xs:restriction base="xs:string">
    <xs:pattern value="(((25[0-5]|2[0-4][0-9]|[01]?[0-9][0-
     9]?)\.){3}(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?))"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="IPv6T">
  <xs:restriction base="xs:string">
    <xs:pattern value="(([A-Fa-f0-9]{1,4}:){7}[A-Fa-f0-9]{1,4})"/>
  </xs:restriction>
</xs:simpleType>

<xs:complexType name="AddressTypeIP">
  <xs:complexContent>
    <xs:extension base="hart:LongAddressT">
      <xs:sequence>
        <xs:choice>
          <xs:element name="IPv4Address" type="hart:IPv4T">
          </xs:element>
          <xs:element name="IPv6Address" type="hart:IPv6T">
          </xs:element>
        </xs:choice>
        <xs:element name="IPPort" type="xs:unsignedShort">
        </xs:element>
```

```
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The elements of AddressTypeIP are described in Table A.3.

**Table A.3 – Elements of AddressTypeIP**

| Element | Description |
|---|---|
| DevAddr | Holds the 5-byte HART address of the associated device. |
| IPAddress | Holds the network IP address of the associated device. |
| IPPort | Holds the IP Port number used for the associated device. |

## A.5 AddressTypeTDMA

The XML schema for AddressTypeTDMA is:

```
<xs:complexType name="AddressTypeTDMA">
  <xs:complexContent>
    <xs:extension base="hart:LongAddressT">
      <xs:sequence>
        <xs:element name="NetworkID" type="xs:unsignedShort">
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The elements of AddressTypeTDMA are described in Table A.4.

**Table A.4 – Elements of AddressTypeTDMA**

| Element | Description |
|---|---|
| DevAddr | Holds the 5-byte HART address of the associated device. |
| IPAddress | Holds the network IP address of the associated device. |
| IPVersion | Holds the IP version number used for the associated device (either version 4 or 6). |
| IPPort | Holds the IP Port number used for the associated device. |

## A.6 AddressType

The XML schema for AddressType is:

```
<xs:complexType name="AddressT">
  <xs:choice>
    <xs:element name="AddressTP" type="hart:AddressTypeTP">
    </xs:element>
    <xs:element name="AddressIP" type="hart:AddressTypeIP">
    </xs:element>
    <xs:element name="AddressTDMA" type="hart:AddressTypeTDMA">
    </xs:element>
  </xs:choice>
```

```
</xs:complexType>

<xs:complexType name="LongAddressT">
  <xs:sequence>
    <xs:element name="DevAddr">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:pattern value="([A-Fa-f0-9]{10})"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

The elements of AddressType are described in Table A.5.

**Table A.5 – Elements of AddressT**

| Element | Description |
|---|---|
| AddressTP | Token passing addresses hold the address information used for ConnectionPointType_HART_TP5, 6, or 7 described in 4.5.1.2. |
| AddressIP | IP addresses hold the address information described in ConnectionPointType_HART_IP described in 4.5.1.3. |
| AddressTDMA | TDMA addresses hold the address information described in ConnectionPointType_HART_TDMA described in 4.5.1.3. |

## A.7 ConnectionPointType

The XML schema for ConnectionPointType is:

```
<xs:complexType name="ConnectionPointT">
  <xs:sequence>
    <xs:element name="Identification" type="hart:IdentificationT">
    </xs:element>
    <xs:element name="Address" type="hart:AddressT">
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

The elements of a ConnectionPointType are described in Table A.6.

**Table A.6 – Elements of ConnectionPointT**

| Element | Description |
|---|---|
| Identification | The element data holds the device type identification data. Compared to the Information Model (IEC 62769-5) 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. |
| Address | The address element holds the specific information used to uniquely address the device in the network. The address will be one of three possible types described by AddressType. |

## A.8 NetworkType

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 a 1:1 relation to a network instance.

The XML schema for a NetworkType is:

```
<xs:complexType name="NetworkT">
  <xs:sequence>
    <xs:element name="ConnectionPoint" type="hart:ConnectionPointT"
     maxOccurs="unbounded">
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

The elements of a NetworkType are described in Table A.7.

**Table A.7 – Elements of NetworkT**

| 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.9 Network

The subsequent element hierarchy is used to return the scan result according to the Information Model (IEC 62769-5) representation of a topology.

The XML schema for a Network element is:

```
<xs:element name="Network" type="hart:NetworkT">
</xs:element>
```

# 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 be reflected in the Information Model. The IEC 62769-6 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 following XML schema documents the protocol specifics.

## B.2 receiveData

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

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

## B.3 sendData

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

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

## B.4 TransferResultDataT

The XML element type defines the service parameter data format that shall be applied to the EndTransfer and GetPublishedData defined return value. The attribute names defined in TransferResultDataType correspond to same name of Transfer method arguments specified in 4.6.1.4 and 4.6.2.4.

```
<xs:complexType name="TransferResultDataT">
      <xs:attribute name="COMMAND" type="xs:unsignedInt"
      use="required">
      </xs:attribute>
      <xs:attribute name="REPLY" type="xs:hexBinary" use="required">
      </xs:attribute>
</xs:complexType>
```

The attributes of TransferResultDataType are described in Table B.1.

**Table B.1 – Attributes of TransferResultDataT**

| Attribute | Description |
|---|---|
| COMMAND | The attribute corresponds to the CP9/1 command number. |
| REPLY | The attribute corresponds to the REPLY data bytes in the CP9/1 command data field. |

## B.5 TransferSendDataT

The XML element type defines the service parameter data format that shall be applied to BeginTransfer defined argument sendData. The attribute names defined in TransferSendDataType correspond to the same name of Transfer method arguments specified in 4.6.1.4 and 4.6.2.4.

```
<xs:complexType name="TransferSendDataT">
      <xs:attribute name="COMMAND" type="xs:unsignedInt"
      use="required">
      </xs:attribute>
      <xs:attribute name="REQUEST" type="xs:hexBinary" use="required">
      </xs:attribute>
</xs:complexType>
```

The attributes of TransferSendDataType are described in Table B.2.

**Table B.2 – Attributes of TransferSendDataT**

| Attribute | Description |
|---|---|
| COMMAND | The attribute corresponds to the CP9/1 command number. |
| REQUEST | The attribute corresponds to the REQUEST data bytes in the CP9/1 command data field. |

# Bibliography

IEC 61784-1, *Industrial communication networks - Profiles - Part 1: Fieldbus profiles*

IEC 61784-2, *Industrial communication networks - Profiles - Part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC/IEEE 8802-3*

IEC 61804 (all parts), *Function blocks (FB) for process control and electronic device description language (EDDL)*

IEC 62769-1, *Field device integration (FDI) - Part 1: Overview*

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

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

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

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

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

IEC 62769-6, *Field Device Integration (FDI) - Part 6: FDI Technology Mapping*

~~NOTE 4 IEC 62769-6 is technically identical to FDI-2026.~~

HCF_SPEC-13, *HART® Field Communication Protocol Specification*, available at ~~<www.hartcomm.org>~~ <http://www.fieldcommgroup.org> [viewed 2018-11-30]

HCF_SPEC-99, *Command Summary Specification,* available at ~~<www.hartcomm.org>~~ <http://www.fieldcommgroup.org> [viewed 2018-11-30]

HCF_SPEC-127, *Universal Command Specification*, available at ~~<www.hartcomm.org>~~ <http://www.fieldcommgroup.org> [viewed 2018-11-30]

HCF_SPEC-151, *Common Practice Command Specification,* available at ~~<www.hartcomm.org>~~ <http://www.fieldcommgroup.org> [viewed 2018-11-30]

IETF RFC 768, *User Datagram Protocol*, available at <http://www.ietf.org> [viewed 2018-11-30]

IETF RFC 793, *Transmission Control Protocol*, available at <http://www.ietf.org> [viewed 2018-11-30]

IEC 62769-109-1

Edition 2.0 2020-04

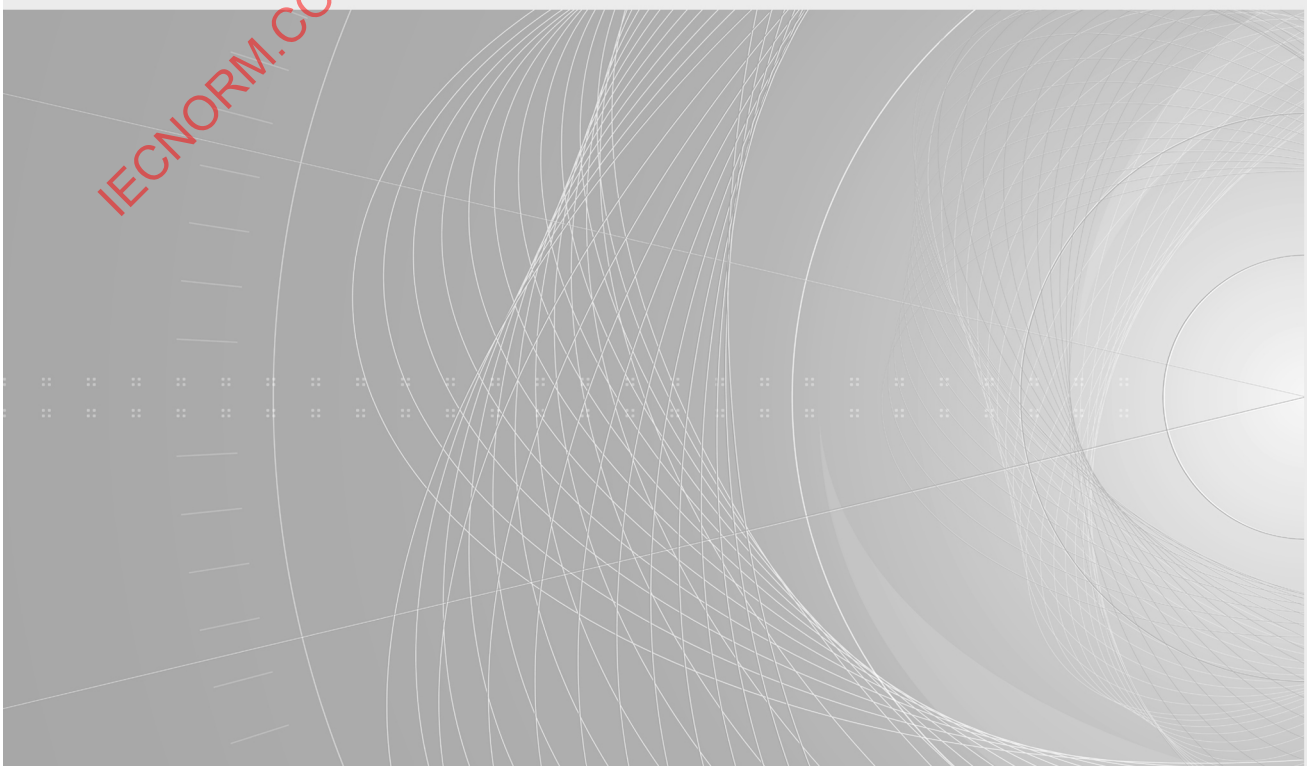
# INTERNATIONAL STANDARD

# NORME INTERNATIONALE



## Field device integration (FDI) – Part 109-1: Profiles – HART® and WirelessHART®

## Intégration des appareils de terrain (FDI) – Partie 109-1: Profils – HART® et WirelessHART®



IEC 62769-109-1:2020-04(en-fr)

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

# FIELD DEVICE INTEGRATION (FDI) –

# Part 109-1: Profiles – HART® and WirelessHART®

## FOREWORD

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

International Standard IEC 62769-109-1 has been prepared by subcommittee 65E: Devices and integration in enterprise systems, of IEC technical committee 65: Industrial-process measurement, control and automation.

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

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

a) support for generic protocol extension for faster adoption of other technologies;
b) support for Package Developers to build EDDs targeted for today's EDD bases system under a single development tool.

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

| CDV | Report on voting |
|---|---|
| 65E/624/CDV | 65E/687A/RVC |

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

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

A list of all parts in the 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 109-1: Profiles – HART® and WirelessHART®

## 1 Scope

This part of IEC 62769 specifies an FDI profile of IEC 62769 for IEC 61784-1_CP 9/1 (HART®)[1] and IEC 61784-1_CP 9/2 (WirelessHART®)[1].

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

IEC 62769-4:–[2], *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: FDI Communication Devices*

## 3 Terms, definitions, abbreviated terms and conventions

### 3.1 Terms and definitions

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

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

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

### 3.2 Abbreviated terms and acronyms

For the purposes of this document, the following abbreviations apply:

| | |
|---|---|
| CP | Communication profile (see IEC 61784-1 or IEC 61784-2) |
| CPF | Communication profile family (see IEC 61784-1 or IEC 61784-2) |
| EDD | Electronic device description (see IEC 61804) |
| EDDL | Electronic device description language (see IEC 61804) |

[1] HART® and wirelessHART® are the registered trademark of 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.

[2] Under preparation. Stage at the time of publication: IEC/RFDIS 62769-4:2020.

| | |
|---|---|
| FDI | Field device integration |
| FSK | Frequency-Shift-Keying |
| HCF | HART Communication Foundation |
| ID | Identification |
| IM | Information Model |
| IP | Internet protocol |
| PDU | Protocol data unit |
| PSK | Phase-Shift-Keying |
| TCP | Transmission Control Protocol (see IETF RFC 793) |
| UDP | User Datagram Protocol (see IETF RFC 768) |
| XML | Extensible markup language |

## 3.3 Conventions

### 3.3.1 EDDL syntax

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

### 3.3.2 XML syntax

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

EXAMPLE `<xs:simpleType name="ExampleT">`

### 3.3.3 Capitalizations

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

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:

- FDI 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 like:

- PROFILE_ID or
- HART_Network.

# 4 Profile for CP 9/1 (HART®) or CP 9/2 (WirelessHART®)

## 4.1 General

This profile document to the FDI specification in IEC 62769 selects the protocol specifics needed for FDI Packages describing FDI Communication Servers, gateways and devices.

## 4.2 Catalog profile

### 4.2.1 Protocol support file

Device information files provide metadata for the dynamic runtime data that is supplied by the device. This metadata is a subset of information that is contained in the EDD. The device information files may be extracted from the package by light-weight gateway or server implementations to exchange runtime device information with minimal implementation overhead. Device information files do not replace the need for the EDD part because device information files only contain a subset of the information from the EDD, and do not provide any user-interface elements.

The formats of the Device Information Files are described in Table 1.

**Table 1 – Device Information Files**

| Part | Content |
|---|---|
| Content Type | Not specified here |
| Root Namespace | Not specified here |
| Source Relationship | http://fdi-cooperation.com/2010/relationships/attachment-protocol |
| Filename | Not specified here |

The Device Information Files are specified in FCG AG21073.

### 4.2.2 CommunicationProfile definition

IEC 62769-4 defines a CommunicationProfileT string type for the Catalog XML schema. Table 2 defines the CP 9/1 specific values for this enumeration.

**Table 2 – CommunicationProfile definition**

| CommunicationProfile | Description |
|---|---|
| hart_fsk | CP 9/1 device type that supports an FSK physical layer (Frequency-Shift-Keying on a pair of wires) |
| hart_psk | CP 9/1 device type that supports a PSK physical layer (Phase-Shift-Keying on a pair of wires). Devices supporting PSK are required to also inherently support FSK, and therefore PSK will always be used only in combination with at least FSK. |
| hart_wirelesshart | CP 9/2 device type that supports a wireless physical layer (communication between device and gateway). |
| hart_ip | CP 9/1 device type that supports Internet Protocol (these devices support both TCP and UDP). |
| hart_rs485 | CP 9/1 device type that supports EIA-485 digital communication. |
| hart_ir | CP 9/1 device type that supports an Infrared physical layer (designed to be transparent to FSK masters – included only as information to indicate that the device supports IR connection). |
| NOTE It is possible for a single CP 9/1 device to support more than one CP. | |

### 4.2.3 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 3 focus on catalog content that is vendor independent.

**Table 3 – Catalog values for profile devices**

| Element | Attribute | Content |
|---|---|---|
| PackageType | — | Profile |
| DeviceModel | — | Empty |
| Manufacturer | — | Empty |

### 4.2.4 Protocol version information

IEC 62769-4 defines an element type named InterfaceT for the Catalog XML Schema. Element type InterfaceT contains an element named Version which is supposed to provide version information about the applied communication protocol profile. The value has to follow the IEC 62769-4 defined version information schema defined in element type VersionT. Subclause 4.2.4 describes how to apply the currently known protocol versions for CP 9/1 or CP 9/2 entries in the device catalog. The general rule is to use the Universal Revision of the protocol for the major version part of VersionT, and the value "0" for the minor version and build parts. Table 4 shows the Protocol Version Information.

**Table 4 – Protocol Version Information**

| Protocol Version | InterfaceT Version value |
|---|---|
| HART Universal Revision 5 | 5.0.0 |
| HART Universal Revision 6 | 6.0.0 |
| HART Universal Revision 7 | 7.0.0 |
| The Protocol Version defined in a package is provided for informational purposes only and shall not be used to determine the compatibility or applicability of a package to a device. | |

## 4.3 Associating a Package with a CP 9/1 device

### 4.3.1 Device type identification mapping

CP 9/1 device types are uniquely identified by parameters Manufacturer, Model and DeviceRevision. These parameters are used to associate a given device instance to an FDI Device Package. These parameters are mapped to the FDI Device Package Catalog according to Table 5.

**Table 5 – Device type catalog mapping**

| Catalog element | CP mapping (SeeTable A.) |
|---|---|
| Manufacturer element of InterfaceT<br>(IEC 62769-4:–, Clause E.10) | Manufacturer<br><br>String format "0xdddd" where dddd is the MANUFACTURER_ID in hexadecimal format. |
| DeviceModel element of InterfaceT<br>(IEC 62769-4:–, Clause E.10) | Model<br><br>String format "0xdddd" where dddd is the DEVICE_TYPE in hexadecimal format. |

| Catalog element | CP mapping (SeeTable A.) |
|---|---|
| DeviceRevision element<br><br>ListOfSupportedDeviceRevisionsT<br>(IEC 62769-4:–, Clause E.20) | DeviceRevision<br><br>String format "x.0.0" where x is the DEVICE_REVISION in decimal format (no leading zeros). |

### 4.3.2 Device type revision mapping

Each device type is identified as per 4.3.1. If a package with matching DeviceRevision is not available, any CP 9/1 FDI package for a corresponding manufacturer and model shall always be compatible with a field device as long as the device revision of the field device is equal to or greater than the device revision specified in the FDI package.

## 4.4 Information Model mapping

### 4.4.1 ProtocolType definition

Table 6 defines the ProtocolType used to identify CP 9/1 network communications.

**Table 6 – ProtocolType HART definition**

| Attribute | Value | | | | |
|---|---|---|---|---|---|
| BrowseName | HART | | | | |
| IsAbstract | False | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Inherits the properties of ProtocolType defined in IEC 62541-100. | | | | | |

### 4.4.2 DeviceType mapping

Each device type inherits the properties of DeviceType. The mapping of the inherited properties from DeviceType is defined in Table 7.

**Table 7 – Inherited DeviceType Property mapping**

| Property | Foundation mapping |
|---|---|
| SerialNumber | Unique ID of a device, mapped to SERIAL_NUMBER of IdentificationT. |
| RevisionCounter | Configuration change counter, mapped to REV_COUNTER of IdentificationT |
| 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 | Entry text string (not supported) [a] |
| DeviceRevision | Device revision level of a device, mapped to DEVICE_REVISION of IdentificationT |
| SoftwareRevision | Software revision level of a device, mapped to SOFTWARE_REVISION of IdentificationT |
| HardwareRevision | Hardware revision level of a device, mapped to HARDWARE_REVISION of IdentificationT |
| [a] Device manuals are exposed as attachments of the FDI Device Package. | |

### 4.4.3 FunctionalGroup Identification definition

As defined in IEC 62541-100, each device representation in the FDI Server hosted Information Model shall contain a protocol specific FunctionalGroup called Identification. This FunctionalGroup organizes variables found in the device type instance. The FunctionalGroup Identification for CP 9/1 is defined in Table 8.

**Table 8 – Identification parameters**

| BrowseName | DataType | Optional/Mandatory |
|---|---|---|
| MANUFACTURER_ID | UInt16 | Mandatory |
| DEVICE_TYPE | UInt16 | Mandatory |
| DEVICE_REVISION | UInt8 | Mandatory |
| UNIVERSAL_REVISION | UInt8 | Optional |
| SERIAL_NUMBER | UInt24 | Optional |
| HARDWARE_REVISION | UInt8 | Optional |
| SOFTWARE_REVISION | UInt8 | Optional |
| REVISION_COUNTER | UInt16 | Optional |

## 4.5 Topology elements

### 4.5.1 ConnectionPoint definition

#### 4.5.1.1 General

CP 9/1 devices can support up to five different ConnectionPoint types that are used for network communications.

#### 4.5.1.2 HART_TP5, HART_TP6, HART_TP7

The ConnectionPoint types HART_TP5, HART_TP6, and HART_TP7 shall be used to identify CP 9/1 token passing network communication and are defined in Table 9. HART_TP5, HART_TP6, and HART_TP7 all contain the same properties, but each provides different qualification information for some of the properties (described below). The Protocol Version (UNIVERSAL_REVISION) described in 4.2.4 can be used as an aid to determine which of the three token passing Connection Point types is the most appropriate. CP 9/1 token passing communications can be used on a variety of physical layers. FSK, PSK, RS485, and Infrared physical layer connections shall all use the HART_TP connection type. The ConnectionPoint types HART_TP5, HART_TP6, and HART_TP7 are subtypes of abstract type ConnectionPointType defined in IEC 62769-5.

The DevAddr property shall be the long address (5 bytes) for the device and is the only parameter necessary to communicate with the field device.

The DevMfg property shall be the 2-byte Manufacturer ID and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevType property shall be the 2-byte extended device type and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevRev property shall be the device revision and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevTag property shall be the long tag for HART® protocol version 6 or 7 devices. The DevTag property shall be the tag for protocol version 5 devices. The DevTag property can be used to help automate the process of assigning live devices in the scan list to offline

placeholders. HART_TP5 Connection Points shall limit the DevTag to 8 characters in length. HART_TP6 and HART_TP7 Connection Points shall limit the DevTag to 32 characters in length.

The DevPollAddr property shall be the poll address and can be used to identify which device is located at a specific poll address. HART_TP5 Connection Points shall be limited to values between 0 and 15 for the DevPollAddr property. HART_TP6 Connection Points shall be limited to values between 0 and 31 for the DevPollAddr property. HART_TP7 Connection Points shall be limited to values between 0 and 63 for the DevPollAddr property.

For forward compatibility, a lower revision HART_TP Connection Point is compatible and can be used for a higher universal revision device connection. For example, if a future HART universal revision 8 device is encountered, and no HART_TP8 is available in the FDI server, HART_TP7 will be compatible and shall be used to connect to the device. If the Protocol Version (i.e. the Universal Revision) is unknown for any reason, the HART_TP5 Connection Point can be used, and will be forward compatible to later universal revisions.

**Table 9 – ConnectionPointType HART_TP definition**

| Attribute | Value | | | | |
|---|---|---|---|---|---|
| BrowseName | ConnnectionPoint_HART_TP5 or ConnnectionPoint_HART_TP6 or ConnnectionPoint_HART_TP7 | | | | |
| IsAbstract | False | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Inherits the properties of ConnectionPointType defined in IEC 62769-5 | | | | | |
| HasProperty | Variable | DevAddr | UInt40 | PropertyType | Mandatory |
| HasProperty | Variable | DevMfg | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevType | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevRev | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevTag | String | PropertyType | Optional |
| HasProperty | Variable | DevPollAddr | UInt8 | PropertyType | Optional |

The ConnectionPoint type HART_TP5, HART_TP6, and HART_TP7 shall be described by an EDD element contained in a Communication Device related FDI Package that can drive a CP 9/1 network. Actual ConnectionPoint properties are declared by VARIABLE constructs grouped together in a COLLECTION named ConnectionPoint_HART_TP5, ConnectionPoint_HART_TP6, or ConnectionPoint_HART_TP7. The following EDDL source code is an example describing a TP5 Connection Point.

```
COMPONENT ConnectionPoint_HART_TP5
{
  LABEL "HART TP Connection Point";
  CLASSIFICATION NETWORK_CONNECTION_POINT;
  CAN_DELETE FALSE;
  PROTOCOL HART;
  CONNECTION_POINT ConnectionPoint_TP5;
}

VARIABLE DevAddr
{
  LABEL "Address";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER(5);
  HANDLING READ & WRITE;
}
```

```
VARIABLE DevMfg
{
  LABEL "Manufacturer";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER(2);
  HANDLING READ & WRITE;
}

VARIABLE DevType
{
  LABEL "Device Type";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER(2);
  HANDLING READ & WRITE;
}

VARIABLE DevRev
{
  LABEL "Device Revision";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER;
  HANDLING READ & WRITE;
}

VARIABLE DevTag
{
  LABEL "Tag";
  CLASS DEVICE;
  TYPE ASCII(32);
  HANDLING READ & WRITE;
}

VARIABLE DevPollAddr
{
  LABEL "Poll Address";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER
  {
    MAX_VALUE 15; //Define appropriate max value for various revisions
  }
  HANDLING READ & WRITE;
}

COLLECTION ConnectionPoint_TP5
{
  LABEL "Connection Point";
  MEMBERS
  {
    ADDRESS,    DevAddr,     "Device Address";
    MFG,        DevMfg,      "Manufacturer";
    DEV_TYPE,   DevType,     "Device Type";
    DEV_REV,    DevRev,      "Device Revision";
    TAG,        DevTag,      "Device Tag";
    POLL_ADDR, DevPollAddr, "Poll Address";
  }
}
```

#### 4.5.1.3 HART_IP

The ConnectionPoint type HART_IP shall be used to identify CP 9/1 IP network communication and is defined in Table 10. HART_IP communications can be used on a variety of physical layers. Ethernet connections shall all use the HART_IP connection type. Additional physical layers developed in the future may also use the HART_IP connection type. The ConnectionPoint type HART_IP is a subtype of abstract type ConnectionPointType defined in IEC 62769-5.

The IPAddress property shall indicate the IP Address (up to 16 bytes) used for the connection.

The IPVersion property shall indicate the version of IP used for the connection (either 4 or 6).

The IPPort property shall be the IP port number for the connection. The default port number used for HART IP is 5 094.

The DevAddr property shall be the long address (5 bytes) for the device.

The DevMfg property shall be the 2-byte Manufacturer ID and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevType property shall be the 2-byte extended device type and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevRev property shall be the device revision and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevTag property shall be the long tag (with maximum 32 characters) and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

**Table 10 – ConnectionPointType HART_IP Definition**

| Attribute | Value | | | | |
|---|---|---|---|---|---|
| BrowseName | ConnnectionPoint_HART_IP | | | | |
| IsAbstract | False | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Inherits the properties of ConnectionPointType defined in IEC 62769-5 | | | | | |
| HasProperty | Variable | IPAddress | ByteString | PropertyType | Mandatory |
| HasProperty | Variable | IPVersion | UInt8 | PropertyType | Mandatory |
| HasProperty | Variable | IPPort | UInt16 | PropertyType | Mandatory |
| HasProperty | Variable | DevAddr | UInt40 | PropertyType | Mandatory |
| HasProperty | Variable | DevMfg | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevType | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevRev | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevTag | String | PropertyType | Optional |

The ConnectionPoint type HART_IP shall be described by an EDD element contained in a Communication Device related FDI Package that can drive a CP 9/1 network. Actual ConnectionPoint properties are declared by VARIABLE constructs grouped together in a COLLECTION named ConnectionPoint_HART_IP. The following EDDL source code is an example describing an IP Connection Point.

```
COMPONENT ConnectionPoint_HART_IP
{
  LABEL "HART IP Connection Point";
  CLASSIFICATION NETWORK_CONNECTION_POINT;
  CAN_DELETE FALSE;
  PROTOCOL HART;
  CONNECTION_POINT ConnectionPoint_IP;
}

ARRAY IPAddress
{
  LABEL "IP Address";
  CLASS DEVICE;
  TYPE OCTET(16);
  HANDLING READ & WRITE;
}

VARIABLE IPVersion
{
  LABEL "IP Version";
  CLASS DEVICE;
  TYPE ENUMERATED
  {
    { 4, "IPv4" },
    { 6, "IPv6" }
  }
  HANDLING READ & WRITE;
}

VARIABLE IPPort
{
  LABEL "IP Port";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER (2);
  DEFAULT_VALUE 5 094;
  HANDLING READ & WRITE;
}

COLLECTION ConnectionPoint_IP
{
  LABEL "Connection Point";
  MEMBERS
  {
    IPADDRESS, IPAddress, "IP Address";
    IPVERSION, IPVersion, "IP Version";
    IPPORT,    IPPort,    "IP Port";
    ADDRESS,   DevAddr,   "Device Address";
    MFG,       DevMfg,    "Manufacturer";
    DEV_TYPE,  DevType,   "Device Type";
    DEV_REV,   DevRev,    "Device Revision";
    TAG,       DevTag,    "Device Tag";
  }
}
```



#### 4.5.1.4 HART_TDMA

The ConnectionPoint type HART_TDMA shall be used to identify CP 9/2 time division media access network communication and is defined in Table 11. HART_TDMA communications can be used on a variety of physical layers. The ConnectionPoint type HART_TDMA is a subtype of abstract type ConnectionPointType defined in IEC 62769-5. WirelessHART connections shall all use the HART_TDMA connection type. Additional physical layers developed in the future may also use the HART_TDMA connection type.

The Network property shall be the network ID for the network.

The DevAddr property shall be the long address (5 bytes) for the device.

The DevMfg property shall be the 2 byte Manufacturer ID, and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevType property shall be the 2 byte extended device type, and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevRev property shall be the device revision and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

The DevTag property shall be the long tag (with maximum 32 characters) and can be used to help automate the process of assigning live devices in the scan list to offline placeholders.

**Table 11 – ConnectionPointType HART_TDMA Definition**

| Attribute | Value | | | | |
|---|---|---|---|---|---|
| BrowseName | ConnnectionPoint_HART_TDMA | | | | |
| IsAbstract | False | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Inherits the properties of ConnectionPointType defined in IEC 62769-5 | | | | | |
| HasProperty | Variable | Network | UInt16 | PropertyType | Mandatory |
| HasProperty | Variable | DevAddr | UInt40 | PropertyType | Mandatory |
| HasProperty | Variable | DevMfg | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevType | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevRev | UInt16 | PropertyType | Optional |
| HasProperty | Variable | DevTag | String | PropertyType | Optional |

The ConnectionPoint type HART_TDMA shall be described by an EDD element contained in a Communication Device related FDI Package that can drive a CP 9/2 network. Actual ConnectionPoint properties are declared by VARIABLE constructs grouped together in a COLLECTION named ConnectionPoint_HART_TDMA. The following EDDL source code is an example describing a TDMA Connection Point.

```
COMPONENT ConnectionPoint_HART_TDMA
{
  LABEL "HART TDMA Connection Point";
  CLASSIFICATION NETWORK_CONNECTION_POINT;
  CAN_DELETE FALSE;
  PROTOCOL HART;
  CONNECTION_POINT ConnectionPoint_TDMA;
}

VARIABLE Network
{
  LABEL "Network ID";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER (2);
  HANDLING READ & WRITE;
```

```
}

COLLECTION ConnectionPoint_TDMA
{
  LABEL "Connection Point";
  MEMBERS
  {
    NETWORK,  Network, "Network ID";
    ADDRESS,  DevAddr, "Device Address";
    MFG,      DevMfg,  "Manufacturer";
    DEV_TYPE, DevType, "Device Type";
    DEV_REV,  DevRev,  "Device Revision";
    TAG,      DevTag,  "Device Tag";
  }
}
```

### 4.5.2 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 FDI Communication Server.

```
COMPONENT CommunicationServer_HART
{
  LABEL "HART Communication Server";
  CAN_DELETE TRUE;
  CLASSIFICATION NETWORK_COMPONENT;
  COMPONENT_RELATIONS { Communication_Device_Setup_HART}
}

COMPONENT_RELATION Communication_Device_Setup_HART
{
  LABEL "Relation between Device and Communication Device";
  RELATION_TYPE CHILD_COMPONENT;
  COMPONENTS
  {
    CommunicationDevice_HART{AUTO_CREATE 1;}
  }
  MINIMUM_NUMBER 1;
  MAXIMUM_NUMBER 4;
}
```

Semantics of the EDDL constructs shown above are described in IEC 62769-7. The EDDL COMPONENT will be utilized by the FDI server and FDI Communication Server to create an instance of type CommunicationServerType as described in IEC 62769-7.

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 is an example for a CP 9/1 communication device.

```
COMPONENT CommunicationDevice_HART
{
  LABEL "HART Communication Device";
  CAN_DELETE TRUE;
  CLASSIFICATION NETWORK_COMPONENT;
  COMPONENT_RELATIONS { ServiceProviderRelation_HART}
  BYTE_ORDER BIG_ENDIAN;
}
```

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

Semantics of the EDDL constructs shown above are described in IEC 62769-7. The EDDL COMPONENT will be utilized by the FDI server and FDI Communication Server to create an instance of type CommunicationServerChannelType as described in IEC 62769-7.

### 4.5.3 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 is an example for a CP 9/1 communication service provider component.

The component reference (ConnectionPoint_HART_IP) corresponds to one of the related Connection Point definitions in 4.5.1. The attribute BYTE_ORDER value is to be set according to the protocol.

```
COMPONENT CommunicationServiceProvider_HART
{
  LABEL "HART Communication Service Provider";
  CAN_DELETE FALSE;
  CLASSIFICATION NETWORK_COMMUNICATION_SERVICE_PROVIDER;
  COMPONENT_RELATIONS { ServiceProviderConnectionPointRelation_HART }
  BYTE_ORDER BIG_ENDIAN;
}

COMPONENT_RELATION ServiceProviderConnectionPointRelation_HART
{
  LABEL "Relation between communication service and connection point";
  RELATION_TYPE CHILD_COMPONENT;
  ADDRESSING
  {
    IPAddress
  }
  COMPONENTS
  {
    ConnectionPoint_HART_IP { AUTO_CREATE 1; }
  }
  MINIMUM_NUMBER 1;
  MAXIMUM_NUMBER 1;
}
```

Semantics of the EDDL constructs shown above are described in IEC 62769-7. The EDDL COMPONENT will be utilized by the FDI server and FDI Communication Server to create an instance of type CommunicationServiceType as described in IEC 62769-7.

### 4.5.4 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. The following EDDL source code is an example describing a network.

```
COMPONENT Network_HART
{
  LABEL "HART Network";
  CAN_DELETE TRUE;
  CLASSIFICATION NETWORK;
  COMPONENT_RELATIONS { NetworkConnectionPointRelation_HART }
}

COMPONENT_RELATION NetworkConnectionPointRelation_HART
{
  LABEL "Relation between network and connection point";
  RELATION_TYPE CHILD_COMPONENT;
  ADDRESSING {DevPollAddr}
  COMPONENTS
  {
    ConnectionPoint_HART_TP5
    {
      MAXIMUM_NUMBER 16;
    },
    ConnectionPoint_HART_TP6
    {
      MAXIMUM_NUMBER 32;
    },
    ConnectionPoint_HART_TP7
    {
      MAXIMUM_NUMBER 64;
    }
  }
  MINIMUM_NUMBER 1;
  MAXIMUM_NUMBER 64;    //Defined to limitations of the comm. device
}
```

Semantics of the EDDL constructs shown above are described in IEC 62769-7. The EDDL COMPONENT will be utilized by the FDI server and FDI Communication Server to create an instance of type NetworkType as described in IEC 62769-5. Some communication devices may organize multiple IO cards as individual networks, while others may present a single network that comprises all Connection Points across all IO cards.

## 4.6 Methods

### 4.6.1 Methods for FDI Communication Servers

#### 4.6.1.1 General

The FDI Communication Server shall implement services according to the method signatures described in 4.6.1 and according to the Information Model.

#### 4.6.1.2 Connect

Table 12 shows the Method Connect arguments.

**Signature:**

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

**Table 12 – Method Connect arguments**

| Argument | Description |
|---|---|
| CommunicationRelationId | The argument value contains the nodeId of the Device ConnectionPoint representing the connection between a device and a physical network which is directly connected to the FDI Communication Server hardware. The nodeId allows finding the direct parent-child relation. |
| LongAddress | The argument name shall match with the corresponding attribute name defined for the ConnectionPoint which is described by a corresponding EDD element specified in 4.5.1. The argument holds the device's long (5-byte) address. The value can be obtained by the method Scan. |
| ServiceError | 0: OK/execution finished, connection established successfully<br>-1: Connect Failed/canceled by caller<br>-2: Call Failed/unknown service ID<br>-3: Connect Failed/device not found<br>-4: Connect Failed/invalid device node address<br>-5: Connect Failed/invalid device identification |
| NOTE 1 IEC 62769-7, defines the argument AddressData of the Connect Method as an array of Variant. The LongAddress arguments defined in the table are represented as entries of the Variant array in the order they are specified above.<br><br>NOTE 2 IEC 62769-7, defines the argument DeviceInformation as a protocol specific argument list in which the Connect method stores the resulting data. The DeviceInformation argument is defined as an array of Variant. The DeviceInformation argument is not used for CP 9/1 devices. | |

#### 4.6.1.3 Disconnect

Table 13 shows the Method Disconnect arguments.

**Signature:**

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

**Table 13 – Method Disconnect arguments**

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

#### 4.6.1.4 Transfer

Table 14 shows the Method Transfer arguments.

**Signature**

```
Transfer(
    [in]  ByteString CommunicationRelationId,
    [in]  UInt16     Command,
    [in]  ByteString Request,
    [out] ByteString Reply,
    [out] Int32      ServiceError);
```

**Table 14 – 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 which is directly connected to the FDI Communication Server hardware. The nodeId allows finding the direct parent-child relation. |
| Command | Command number of the CP 9/1 command to be sent. The FDI Communication Server is required to generate and send the PDU in the proper format according to CP 9/1 protocol specifications (e.g. taking care of command expansion). |
| Request | An array of bytes representing the data bytes in the CP 9/1 command request (e.g. starting after the byte count, and ending before the checksum). |
| Reply | An array of bytes representing the data bytes in the CP 9/1 command response (e.g. starting after the byte count, and ending before the checksum). |
| ServiceError | 0: OK/execution finished<br>-1: Transfer Failed/cancelled by caller<br>-2: Call Failed/unknown service ID<br>-3: Transfer Failed/no existing communication relation<br>-4: Transfer Failed/invalid communication relation identifier<br>-5: Transfer Failed/invalid Request content<br>-6: Transfer Failed/invalid Reply format |
| NOTE 1 IEC 62769-7 defines the argument SendData of the Transfer Method as an array of Variant. The arguments Command and Request defined in the table are represented as entries of the Variant array in the order they are specified above.<br><br>NOTE 2 IEC 62769-7 defines the argument ReceiveData of the Transfer Method as an array of Variant. The argument Reply defined in the table is represented as the entry of the Variant array. | |

#### 4.6.1.5 GetPublishedData

CP 9/1 burst and event notification messages represent unsolicited messages as defined in IEC 62769-7. Table 15 shows the Method GetPublishedData arguments.

**Signature:**

```
GetPublishedData(
    [in]  ByteString CommunicationRelationId,
    [out] UInt16     Command,
    [out] ByteString Reply,
    [out] DateTime   TimeStamp,
    [out] Int32      ServiceError);
```

**Table 15 – Method GetPublishedData 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. |
| Command | Command number of the CP 9/1 command published. The FDI Communication Server is required to parse the PDU in various formats according to CP 9/1 Protocol standards (e.g. taking care of command expansion). |
| Reply | An array of bytes representing the data bytes in the published CP 9/1 command response (e.g. starting after the byte count and ending before the checksum). |
| TimeStamp | Time at which the published data was captured. |
| ServiceError | 0: OK/execution finished<br>-1: GetPublishedData Failed/cancelled by caller<br>-2: Call Failed/unknown service ID<br>-3: GetPublishedData Failed/not supported<br>-4: GetPublishedData Failed/no existing communication relation<br>-5: GetPublishedData Failed/invalid communication relation identifier<br>-8: GetPublishedData Failed/no burst / event data published |
| NOTE IEC 62769-7 defines the argument ReceiveData of the GetPublishedData Method as an array of Variant. The arguments Command and Reply defined in the table are represented as entries of the Variant array in the order they are specified above. | |

#### 4.6.1.6 SetAddress

Table 16 shows the Method SetAddress arguments.

**Signature**

```
SetAddress(
   [in]  UInt8 OldPollAddress,
   [in]  UInt8 NewPollAddress,
   [out] Int32 ServiceError);
```

**Table 16 – Method SetAddress arguments**

| Argument | Description |
|---|---|
| OldPollAddress | The argument value holds the current address of a device. Allowed values are 0..63. |
| NewPollAddress | The argument value holds the new address for a device. Allowed values are 0..63. |
| ServiceError | 0: OK/execution finished successfully<br>-1: SetAddress Failed/cancelled by caller<br>-2: Call Failed/unknown service ID<br>-3: SetAddress Failed/not initialized<br>-4: SetAddress Failed/not connected to a network<br>-5: SetAddress Failed/no device found responding to oldAddress<br>-6: SetAddress Failed/duplicate address error<br>-7: SetAddress Failed/device did not accept new address<br>-8: SetAddress Failed/invalid oldAddress<br>(in terms of syntax, data type, data format, and so on)<br>-9: SetAddress Failed/invalid newAddress<br>(in terms of syntax, data type, data format, and so on)<br>-10: SetAddress Failed/not possible in status connected |
| NOTE 1 IEC 62769-7 defines the argument OldAddress of the SetAddress Method as an array of Variant. The argument OldPollAddress defined in the table is represented as the entry of the Variant array.<br>NOTE 2 IEC 62769-7 defines the argument NewAddress of the SetAddress Method as an array of Variant. The argument NewPollAddress defined in the table is represented as the entry of the Variant array. | |

#### 4.6.1.7 Scan

The method signature specified in IEC 62769-7 applies. The corresponding topologyScanResult schema is specified in Annex A.

#### 4.6.1.8 ResetScan

The method signature specified in IEC 62769-7 applies. The corresponding topologyScanResult schema is specified in Annex A.

### 4.6.2 Methods for Gateways

#### 4.6.2.1 General

The method signatures described in 4.6.2 shall be implemented in the EDD element IEC 62769-4 contained in a Gateway related FDI Package containing the communication device definitions.

#### 4.6.2.2 Connect

Table 17 shows the Method Connect arguments.

**Signature:**

```
METHOD BeginConnect(
   DD_String      CommunicationRelationId,
   DD_String      LongAddress,
   unsigned long  ServiceID,
   unsigned long &DelayForNextCall,
   long          &ServiceError)
{
   DEFINITION {<Gateway specific implementation>}
}

METHOD EndConnect(
   DD_String      CommunicationRelationId,
   unsigned long  ServiceID,
   unsigned long &DelayForNextCall,
   long          &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}

METHOD CancelConnect(
   DD_String      CommunicationRelationId,
   unsigned long  ServiceID,
   long          &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}
```

**Table 17 – 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 FDI Communication Server hardware. The nodeId allows finding the direct parent-child relation. |
| LongAddress | The argument name shall match with the corresponding attribute name defined for the ConnectionPoint which is described by a corresponding EDD element specified in 4.5.1. The argument holds the devices long (5-byte) address. The value can be obtained by the method Scan. |
| 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<br>0: OK/execution finished, connection established successfully<br>-1: Connect Failed/cancelled by caller<br>-2: Call Failed/unknown service ID<br>-3: Connect Failed/device not found<br>-4: Connect Failed/invalid device address<br>-5: Connect Failed/invalid device identification |

#### 4.6.2.3 Disconnect

Table 18 shows the Method Disconnect arguments.

**Signature:**

```
METHOD Disconnect(
   DD_String  communicationRelationId,
   Long       &serviceError)
{
   DEFINITION {<Gateway specific implementation>}
}
```

**Table 18 – 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 FDI Communication Server hardware. The nodeId allows finding the direct parent-child relation. |
| ServiceError | 0: OK/disconnect finished successfully<br>-1: Disconnect Failed/no existing communication relation<br>-2: Disconnect Failed/invalid communication relation identifier |

#### 4.6.2.4 Transfer

Table 19 shows the Method Transfer arguments.

**Signature:**

```
METHOD BeginTransfer(
   DD_String       CommunicationRelationId,
   unsigned short  Command,
   DD_String       Request,
   DD_String       &Reply,
   unsigned long   ServiceId,
   unsigned long   &DelayForNextCall,
   long            &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}

METHOD EndTransfer(
   DD_String       CommunicationRelationId,
   DD_String      &Reply,
   unsigned long   ServiceId,
   unsigned long  &DelayForNextCall,
   long           &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}

METHOD CancelTransfer(
   DD_String       CommunicationRelationId,
   unsigned long   ServiceId,
   long           &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}
```

**Table 19 – 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 which is directly connected to the FDI Communication Server hardware. The nodeId allows finding the direct parent-child relation. |
| Command | Command number of the CP 9/1 command to be sent. The FDI Communication Server is required to generate and send the PDU in the proper format according to CP 9/1 protocol specifications (e.g. taking care of command expansion). |
| Request | An array of bytes representing the data bytes in the CP 9/1 command request (e.g. starting after the byte count, and ending before the checksum). |
| Reply | An array of bytes representing the data bytes in the CP 9/1 command response (e.g. starting after the byte count, and ending before the checksum). |
| 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<br>0: OK/execution finished<br>-1: Transfer Failed/cancelled by caller<br>-2: Call Failed/unknown service ID<br>-3: Transfer Failed/no existing communication relation<br>-4: Transfer Failed/invalid communication relation identifier<br>-5: Transfer Failed/invalid REQUEST content<br>-6: Transfer Failed invalid REPLY format |

#### 4.6.2.5 GetPublishedData

CP 9/1 burst and event notification messages represent unsolicited messages as defined with IEC 62769-7. Table 20 shows the Method GetPublishedData arguments.

**Signature**

```
METHOD BeginGetPublishedData(
   DD_String          CommunicationRelationId,
   unsigned short &Command,
   DD_String          &Reply,
   TIME               &TimeStamp
   unsigned long       ServiceId,
   unsigned long  &DelayForNextCall,
   long               &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}

METHOD EndGetPublishedData(
   DD_String          CommunicationRelationId,
   unsigned short &Command,
   DD_STRING          &Reply,
   TIME               &TimeStamp
   unsigned long       ServiceId,
   unsigned long  &DelayForNextCall,
   long               &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
```

```
}

METHOD CancelGetPublishedData(
   DD_String      CommunicationRelationId,
   unsigned long  ServiceId,
   long           &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}
```

**Table 20 – Method GetPublishedData 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. |
| Command | Command number of the CP 9/1 command published. The FDI Communication Server is required to parse the PDU in various formats according to CP 9/1 Protocol standards (e.g. taking care of command expansion). |
| Reply | An array of bytes representing the data bytes in the published CP 9/1 command response (e.g. starting after the byte count and ending before the checksum). |
| TimeStamp | Time at which the published data was captured. |
| 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 EndGetPublishedData 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 EndGetPublishedData<br>0: OK/execution finished<br>-1: GetPublishedData Failed/cancelled by caller<br>-2: Call Failed/unknown service ID<br>-3: GetPublishedData Failed/not supported<br>-4: GetPublishedData Failed/no existing communication relation<br>-5: GetPublishedData Failed/invalid communication relation identifier<br>-8: GetPublishedData Failed/no burst information published |

#### 4.6.2.6 SetAddress

Table 21 shows the Method SetAddress arguments.

**Signature**

```
METHOD BeginSetAddress(
   unsigned char  OldPollAddress,
   unsigned char  NewPollAddress,
   unsigned long  ServiceId,
   unsigned long &DelayForNextCall,
   long           &ServiceError);
{
   DEFINITION {<Gateway specific implementation>}
}

METHOD EndSetAddress(
   unsigned long  ServiceId,
   unsigned long &DelayForNextCall,
```

```
    long           &ServiceError);
{
    DEFINITION {<Gateway specific implementation>}
}

METHOD BeginSetAddress(
    unsigned long  ServiceId,
    long           &ServiceError);
{
    DEFINITION {<Gateway specific implementation>}
}
```

**Table 21 – Method SetAddress arguments**

| Argument | Description |
|---|---|
| OldPollAddress | The argument value holds the current address of a device. The allowed values are 0..63. |
| NewPollAddress | The argument value holds the new address for a device. The allowed values are 0..63. |
| 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<br>0: OK/execution finished successfully<br>-1: SetAddress Failed/cancelled by caller<br>-2: Call Failed/unknown service ID<br>-3: SetAddress Failed/not initialized<br>-4: SetAddress Failed/not connected to a network<br>-5: SetAddress Failed/no device found responding to oldAddress<br>-6: SetAddress Failed/duplicate address error<br>-7: SetAddress Failed/device did not accept new address<br>-8: SetAddress Failed/invalid oldAddress<br>(in terms of syntax, data type, data format, and so on)<br>-9: SetAddress Failed/invalid newAddress<br>(in terms of syntax, data type, data format, and so on)<br>-10: SetAddress Failed/not possible in status connected |



#### 4.6.2.7 Scan

The method signature specified in IEC 62769-7 applies. The HART Gateway business logic shall create the scan result using the following EDD definitions, which shall be included in the communication device COMPONENT definition. The EDD LIST also held in the COMPONENT holds the list of devices corresponding to the data structure defined in the XML schema. This LIST shall be referred to by the SCAN_LIST attribute of the communication device component. The corresponding topologyScanResult schema is specified in Annex A.

```
VARIABLE DevAddr
{
  LABEL "Address";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER(5);
  HANDLING READ & WRITE;
}
```

```
VARIABLE DevMfg
{
  LABEL "Manufacturer";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER(2);
  HANDLING READ & WRITE;
}

VARIABLE DevType
{
  LABEL "Device Type";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER(2);
  HANDLING READ & WRITE;
}

VARIABLE DevRev
{
  LABEL "Device Revision";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER;
  HANDLING READ & WRITE;
}

VARIABLE DevTag
{
  LABEL "Tag";
  CLASS DEVICE;
  TYPE ASCII(32);
  HANDLING READ & WRITE;
}

COLLECTION ScanItemType_TP
{
  MEMBERS
  {
    ADDRESS, DevAddr;
  }
}

COLLECTION DeviceIdentificationType
{
  MEMBERS
  {
    MFG,      DevMfg;
    DEV_TYPE, DevType;
    DEV_REV,  DevRev;
    TAG,      DevTag;
  }
}

COLLECTION ScanResultType
{
  MEMBERS
  {
    ScanItem,             ScanItemType_TP;
    DeviceIdentification, DeviceIdentificationType;
  }
}

LIST TopologyScanResult
{
  TYPE ScanResultType;
```

```
  CAPACITY 64;   //Defined to limitation of the comm. device
}
```

#### 4.6.2.8 ScanNext

The method signature specified in IEC 62769-7 applies. The HART Gateway business logic shall create the scan result using the same EDD definitions defined in 4.6.2.7. The corresponding topologyScanResult schema is specified in Annex A.

# Annex A
(normative)

# Topology scan schema

## A.1 General

The Topology scan result schema specified in Annex A describes the HART 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 IdentificationType

The element content corresponds to the "FunctionalGroup Identification".

The XML schema for IdentificationType is:

```
<xs:complexType name="IdentificationT">
  <xs:attribute name="MANUFACTURER_ID" type="xs:unsignedShort"
use="required">
  </xs:attribute>
  <xs:attribute name="DEVICE_TYPE" type="xs:unsignedShort"
use="required">
  </xs:attribute>
  <xs:attribute name="UNIVERSAL_REVISION" type="xs:unsignedByte"
use="required">
  </xs:attribute>
  <xs:attribute name="DEVICE_REVISION" type="xs:unsignedByte"
use="required">
  </xs:attribute>
  <xs:attribute name="SERIAL_NUMBER" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:unsignedInt">
        <xs:minInclusive value="0"/>
        <xs:maxInclusive value="16777216"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="HARDWARE_REVISION" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:unsignedByte">
        <xs:minInclusive value="0"/>
        <xs:maxInclusive value="31"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="SOFTWARE_REVISION" type="xs:unsignedByte"
use="required">
  </xs:attribute>
  <xs:attribute name="REV_COUNTER" type="xs:unsignedShort"
default="0">
  </xs:attribute>
  <xs:attribute name="TAG" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:maxLength value="32"/>
      </xs:restriction>
```

```
    </xs:simpleType>
  </xs:attribute>
</xs:complexType>
```

The attributes of IdentificationType are described in Table A.1.

**Table A.1 – Attributes of IdentificationT**

| Attribute | Description |
|---|---|
| MANUFACTURER_ID | Manufacturer identification number.<br>For HART universal revision 7 or higher devices:<br>2-byte manufacturer code of a device, returned in bytes 17 and 18 of Command 0 or Command 11 or Command 21.<br>For HART revision 6 or lower devices:<br>Most significant byte shall be fixed to 0, and the least significant byte is returned in byte 1 of Command 0 or Command 11 or Command 21. |
| DEVICE_TYPE | Manufacturer's model number associated with the resource.<br>For HART universal revision 7 or higher devices:<br>2 byte device type returned in bytes 1 and 2 of Command 0 or Command 11 or Command 21.<br>For HART revision 6 or lower devices:<br>Most significant byte shall be fixed to 0, and the least significant byte is returned in byte 2 of Command 0 or Command 11 or Command 21. |
| UNIVERSAL_REVISION | Universal HART revision associated with the device. Returned in byte 4 of Command 0 or Command 11 or Command 21. |
| DEVICE_REVISION | Manufacturer's device revision number associated with the device. Returned in byte 5 of Command 0 or Command 11 or Command 21. |
| SERIAL_NUMBER | Unique serial number of the device. 3-byte unique ID of a device, returned in bytes 9 to 11 of Command 0 or Command 11 or Command 21. |
| HARDWARE_REVISION | Manufacturer hardware revision associated with the device. Returned in byte 7 (only 5 most significant bits) of Command 0 or Command 11 or Command 21. |
| SOFTWARE_REVISION | Manufacturer firmware revision associated with the device. Returned in byte 6 of Command 0 or Command 11 or Command 21. |
| REV_COUNTER | The configuration change counter associated with the device.<br>For HART universal revision 6 or higher devices:<br>Returned in bytes 14 and 15 of Command 0 or Command 11 or Command 21.<br>For HART revision 5 or lower devices:<br>-1 (not defined). |
| TAG | The configured Identification tag associated with the device.<br>For HART universal revision 6 or higher devices:<br>32 character long tag returned in Command 20 or Command 21.<br>For HART revision 5 or lower devices:<br>8 character tag returned in Command 11 or Command 13. |

## A.3 AddressTypeTP

The XML schema for AddressTypeTP is:

```
<xs:complexType name="AddressTypeTP">
  <xs:complexContent>
    <xs:extension base="hart:LongAddressT">
      <xs:sequence>
        <xs:element name="DevPollAddr" minOccurs="0">
          <xs:simpleType>
            <xs:restriction base="xs:unsignedByte">
              <xs:minInclusive value="0"/>
              <xs:maxInclusive value="63"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The elements of AddressTypeTP are described in Table A.2.

**Table A.2 – Elements of AddressTypeTP**

| Element | Description |
|---|---|
| DevAddr | Holds the 5-byte HART address of the associated device. |
| DevPollAddr | Holds the HART short address used for polling the associated device. |

## A.4 AddressTypeIP

The XML schema for AddressTypeIP is:

```
<xs:simpleType name="IPv4T">
  <xs:restriction base="xs:string">
    <xs:pattern value="(((25[0-5]|2[0-4][0-9]|[01]?[0-9][0-
     9]?)\.){3}(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?))"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="IPv6T">
  <xs:restriction base="xs:string">
    <xs:pattern value="(([A-Fa-f0-9]{1,4}:){7}[A-Fa-f0-9]{1,4})"/>
  </xs:restriction>
</xs:simpleType>

<xs:complexType name="AddressTypeIP">
  <xs:complexContent>
    <xs:extension base="hart:LongAddressT">
      <xs:sequence>
        <xs:choice>
          <xs:element name="IPv4Address" type="hart:IPv4T">
          </xs:element>
          <xs:element name="IPv6Address" type="hart:IPv6T">
          </xs:element>
        </xs:choice>
        <xs:element name="IPPort" type="xs:unsignedShort">
        </xs:element>
```

```
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The elements of AddressTypeIP are described in Table A.3.

**Table A.3 – Elements of AddressTypeIP**

| Element | Description |
|---|---|
| DevAddr | Holds the 5-byte HART address of the associated device. |
| IPAddress | Holds the network IP address of the associated device. |
| IPPort | Holds the IP Port number used for the associated device. |

## A.5 AddressTypeTDMA

The XML schema for AddressTypeTDMA is:

```
<xs:complexType name="AddressTypeTDMA">
  <xs:complexContent>
    <xs:extension base="hart:LongAddressT">
      <xs:sequence>
        <xs:element name="NetworkID" type="xs:unsignedShort">
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The elements of AddressTypeTDMA are described in Table A.4.

**Table A.4 – Elements of AddressTypeTDMA**

| Element | Description |
|---|---|
| DevAddr | Holds the 5-byte HART address of the associated device. |
| IPAddress | Holds the network IP address of the associated device. |
| IPVersion | Holds the IP version number used for the associated device (either version 4 or 6). |
| IPPort | Holds the IP Port number used for the associated device. |

## A.6 AddressType

The XML schema for AddressType is:

```
<xs:complexType name="AddressT">
  <xs:choice>
    <xs:element name="AddressTP" type="hart:AddressTypeTP">
    </xs:element>
    <xs:element name="AddressIP" type="hart:AddressTypeIP">
    </xs:element>
    <xs:element name="AddressTDMA" type="hart:AddressTypeTDMA">
    </xs:element>
  </xs:choice>
```

```
</xs:complexType>

<xs:complexType name="LongAddressT">
  <xs:sequence>
    <xs:element name="DevAddr">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:pattern value="([A-Fa-f0-9]{10})"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

The elements of AddressType are described in Table A.5.

**Table A.5 – Elements of AddressT**

| Element | Description |
|---|---|
| AddressTP | Token passing addresses hold the address information used for ConnectionPointType_HART_TP5, 6, or 7 described in 4.5.1.2. |
| AddressIP | IP addresses hold the address information described in ConnectionPointType_HART_IP described in 4.5.1.3. |
| AddressTDMA | TDMA addresses hold the address information described in ConnectionPointType_HART_TDMA described in 4.5.1.3. |

## A.7 ConnectionPointType

The XML schema for ConnectionPointType is:

```
<xs:complexType name="ConnectionPointT">
  <xs:sequence>
    <xs:element name="Identification" type="hart:IdentificationT">
    </xs:element>
    <xs:element name="Address" type="hart:AddressT">
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

The elements of a ConnectionPointType are described in Table A.6.

**Table A.6 – Elements of ConnectionPointT**

| Element | Description |
|---|---|
| Identification | The element data holds the device type identification data. Compared to the Information Model (IEC 62769-5) 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. |
| Address | The address element holds the specific information used to uniquely address the device in the network. The address will be one of three possible types described by AddressType. |

## A.8 NetworkType

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 a 1:1 relation to a network instance.

The XML schema for a NetworkType is:

```
<xs:complexType name="NetworkT">
  <xs:sequence>
    <xs:element name="ConnectionPoint" type="hart:ConnectionPointT"
     maxOccurs="unbounded">
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

The elements of a NetworkType are described in Table A.7.

**Table A.7 – Elements of NetworkT**

| 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.9 Network

The subsequent element hierarchy is used to return the scan result according to the Information Model (IEC 62769-5) representation of a topology.

The XML schema for a Network element is:

```
<xs:element name="Network" type="hart:NetworkT">
</xs:element>
```

# 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 be reflected in the Information Model. The IEC 62769-6 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 following XML schema documents the protocol specifics.

## B.2 receiveData

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

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

## B.3 sendData

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

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

## B.4 TransferResultDataT

The XML element type defines the service parameter data format that shall be applied to the EndTransfer and GetPublishedData defined return value. The attribute names defined in TransferResultDataType correspond to same name of Transfer method arguments specified in 4.6.1.4 and 4.6.2.4.

```
<xs:complexType name="TransferResultDataT">
      <xs:attribute name="COMMAND" type="xs:unsignedInt"
      use="required">
      </xs:attribute>
      <xs:attribute name="REPLY" type="xs:hexBinary" use="required">
      </xs:attribute>
</xs:complexType>
```

The attributes of TransferResultDataType are described in Table B.1.

**Table B.1 – Attributes of TransferResultDataT**

| Attribute | Description |
|---|---|
| COMMAND | The attribute corresponds to the CP9/1 command number. |
| REPLY | The attribute corresponds to the REPLY data bytes in the CP9/1 command data field. |

## B.5 TransferSendDataT

The XML element type defines the service parameter data format that shall be applied to BeginTransfer defined argument sendData. The attribute names defined in TransferSendDataType correspond to the same name of Transfer method arguments specified in 4.6.1.4 and 4.6.2.4.

```
<xs:complexType name="TransferSendDataT">
      <xs:attribute name="COMMAND" type="xs:unsignedInt"
      use="required">
      </xs:attribute>
      <xs:attribute name="REQUEST" type="xs:hexBinary" use="required">
      </xs:attribute>
</xs:complexType>
```

The attributes of TransferSendDataType are described in Table B.2.

**Table B.2 – Attributes of TransferSendDataT**

| Attribute | Description |
|---|---|
| COMMAND | The attribute corresponds to the CP9/1 command number. |
| REQUEST | The attribute corresponds to the REQUEST data bytes in the CP9/1 command data field. |

# Bibliography

IEC 61784-1, *Industrial communication networks - Profiles - Part 1: Fieldbus profiles*

IEC 61784-2, *Industrial communication networks - Profiles - Part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC/IEEE 8802-3*

IEC 61804 (all parts), *Function blocks (FB) for process control and electronic device description language (EDDL)*

IEC 62769-1, *Field device integration (FDI) - Part 1: Overview*

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

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

IEC 62769-6, *Field Device Integration (FDI) - Part 6: FDI Technology Mapping*

HCF_SPEC-13, *HART® Field Communication Protocol Specification*, available at <http://www.fieldcommgroup.org> [viewed 2018-11-30]

HCF_SPEC-99, *Command Summary Specification,* available at <http://www.fieldcommgroup.org > [viewed 2018-11-30]

HCF_SPEC-127, *Universal Command Specification*, available at <http://www.fieldcommgroup.org > [viewed 2018-11-30]

HCF_SPEC-151, *Common Practice Command Specification,* available at <http://www.fieldcommgroup.org > [viewed 2018-11-30]

IETF RFC 768, *User Datagram Protocol*, available at <http://www.ietf.org> [viewed 2018-11-30]

IETF RFC 793, *Transmission Control Protocol*, available at <http://www.ietf.org> [viewed 2018-11-30]

# SOMMAIRE

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

_____

# INTÉGRATION DES APPAREILS DE TERRAIN (FDI) –

# Partie 109-1: Profils – HART® et WirelessHART®

## 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 et de ne pas avoir signalé leur existence.



La Norme internationale IEC 62769-109-1 a été établie par le sous-comité 65E: Les dispositifs et leur intégration dans les systèmes de l'entreprise, du comité d'études 65 de l'IEC: Mesure, commande et automation dans les processus industriels.

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

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

a) prise en charge d'extensions de protocoles génériques, pour une adoption plus rapide d'autres technologies;
b) capacité offerte aux Développeurs de Paquetages d'élaborer des EDD ciblant les systèmes actuels de bases EDD, en exploitant un seul outil de développement.

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

| CDV | Rapport de vote |
|---|---|
| 65E/624/CDV | 65E/687A/RVC |

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

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

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

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

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

# INTÉGRATION DES APPAREILS DE TERRAIN (FDI) –

# Partie 109-1: Profils – HART® et WirelessHART®

## 1 Domaine d'application

La présente partie de l'IEC 62769 spécifie un profil FDI de l'IEC 62769 pour l'IEC 61784-1_CP 9/1 (HART®)[1] et l'IEC 61784-1_CP 9/2 (WirelessHART®)[1].

## 2 Références normatives

Les documents suivants cités dans le texte 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 62541-100:2015, *Architecture unifiée OPC – Partie 100: Interface d'appareils*

IEC 62769-4:–[2], *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 FDI*

## 3 Termes, définitions, abréviations et conventions

### 3.1 Termes et définitions

Pour les besoins du présent document, les termes et définitions donnés dans l'IEC 62541-100, l'IEC 62769-4, l'IEC 62769-5 et 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 http://www.electropedia.org/;
- ISO Online browsing platform: disponible à l'adresse http://www.iso.org/obp.

### 3.2 Termes abégés et acronymes

Pour les besoins du présent document, les abréviations suivantes s'appliquent:

CP Profil de communication (Communication Profile, voir l'IEC 61784-1 ou l'IEC 61784-2)



---

[1] HART® et WirelessHART® sont des marques déposées de FieldComm Group. Cette information est donnée à l'intention des utilisateurs du présent document et ne signifie nullement que l'IEC approuve ou recommande l'emploi exclusif du produit ainsi désigné. Des produits équivalents peuvent être utilisés, s'il peut être attesté qu'ils donnent les mêmes résultats.

[2] En cours de préparation. Stade au moment de la publication: IEC/RFDIS 62769-4:2020.

CPF Famille de profils de communication (Communication Profile Family, voir l'IEC 61784-1 ou l'IEC 61784-2)

EDD Description d'appareil électronique (Electronic Device Description, voir l'IEC 61804)

EDDL Langage de description d'appareil électronique (Electronic Device Description Language, voir l'IEC 61804)

FDI Intégration des appareils de terrain (Field Device Integration)

FSK Modulation par déplacement de fréquence (Frequency-Shift-Keying)

HCF HART Communication Foundation

ID Identification

IM Modèle d'Information (Information Model)

IP Protocole Internet (Internet Protocol)

PDU Unité de données de protocole (Protocol Data Unit)

PSK Modulation par déplacement de phase (Phase-Shift-Keying)

TCP Protocole de commande de transmission (Transmission Control Protocol, voir le RFC 793 de l'IETF)

UDP Protocole datagramme d'utilisateur (User Datagram Protocol, voir le RFC 768 de l'IETF)

XML Langage de balisage extensible (eXtensible Markup Language)

## 3.3 Conventions

### 3.3.1 Syntaxe EDDL

Le présent document spécifie le contenu du composant EDD qui fait partie des Paquetages de Communication FDI. La syntaxe EDDL utilise la police `Courier New`. La syntaxe EDDL est utilisée pour les déclarations des signatures de méthodes, des variables, des structures de données et des composants.

### 3.3.2 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="ExampleT">
```

### 3.3.3 Majuscules

La série IEC 62769 utilise des termes en majuscules pour souligner que ces termes ont une signification spécifique à la FDI.

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:

- PROFILE_ID, ou
- HART_Network.

# 4 Profil pour CP 9/1 (HART®) ou CP 9/2 (WirelessHART®)

## 4.1 Généralités

Le présent document de profil de la spécification FDI donnée dans l'IEC 62769 décrit les spécificités du protocole dont ont besoin les Paquetages FDI décrivant des Serveurs de Communication FDI, des passerelles et des appareils.

## 4.2 Profil de catalogue

### 4.2.1 Fichier de prise en charge de protocole

Les fichiers d'information appareil fournissent des métadonnées relatives aux données dynamiques d'exécution fournies par l'appareil. Ces métadonnées sont un sous-ensemble d'informations contenues dans l'EDD. Les fichiers d'informations appareil peuvent être extraits du paquetage par des mises en œuvre de serveur léger ou de passerelle légère, dans le but d'échanger des informations d'exécution appareil avec un traitement minimal des mises en œuvre. Les fichiers d'informations appareil ne sauraient se substituer à la nécessité d'une partie EDD, car ils peuvent uniquement contenir un sous-ensemble d'informations de l'EDD et ne fournissent pas d'éléments sur l'interface utilisateur.

Les formats des Fichiers d'informations appareil sont décrits dans le Tableau 1.

**Tableau 1 – Fichiers d'informations appareil**

| Partie | Contenu |
|---|---|
| Content Type | Non spécifié ici |
| Root Namespace | Non spécifié ici |
| Source Relationship | http://fdi-cooperation.com/2010/relationships/attachment-protocol |
| Filename | Non spécifié ici |

Les Fichiers d'information appareil sont spécifiés dans la spécification FCG AG21073.

### 4.2.2 Définition de CommunicationProfile

L'IEC 62769-4 définit un type de chaîne CommunicationProfileT pour le schéma XML Catalog. Le Tableau 2 définit les valeurs spécifiques au CP 9/1 pour cette énumération.

**Tableau 2 – Définition de CommunicationProfile**

| CommunicationProfile | Description |
|---|---|
| hart_fsk | Type d'appareil CP 9/1 qui prend en charge une couche physique FSK (modulation par déplacement de fréquence sur une paire de fils). |
| hart_psk | Type d'appareil CP 9/1 qui prend en charge une couche physique PSK (modulation par déplacement de phase sur une paire de fils). Il est exigé que les appareils prenant en charge la modulation PSK prennent également intrinsèquement en charge la modulation FSK. Par conséquent, la PSK n'est toujours utilisée qu'en combinaison avec au moins la FSK. |
| hart_wirelesshart | Type d'appareil CP 9/2 qui prend en charge une couche physique sans fil (communication entre appareil et passerelle). |
| hart_ip | Type d'appareil CP 9/1 qui prend en charge le Protocole Internet (ces appareils prennent en charge les protocoles TCP et UDP). |
| hart_rs485 | Type d'appareil CP 9/1 qui prend en charge la communication numérique EIA-485. |
| hart_ir | Type d'appareil CP 9/1 qui prend en charge une Couche physique infrarouge (conçue pour être transparente aux maîtres FSK – incluse uniquement à titre d'information pour indiquer que l'appareil prend en charge la connexion infrarouge). |
| NOTE Un même appareil CP 9/1 peut prendre en charge plus d'un CP. | |

### 4.2.3 Appareil de profil

Un Paquetage de Profil doit fournir les valeurs de catalogue pour les appareils de profil, permettant au Serveur FDI d'exploiter une description d'appareil générique, si une description spécifique n'est pas disponible. Les définitions du Tableau 3 sont axées sur le contenu du catalogue qui est indépendant du fournisseur.

**Tableau 3 – Valeurs de catalogue pour les appareils de profil**

| Elément | Attribut | Contenu |
|---|---|---|
| PackageType | — | Profil |
| DeviceModel | — | Vide |
| Manufacturer | — | Vide |

### 4.2.4 Informations relatives à la version de 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 nommé Version qui est censé fournir des informations de version relatives au profil de protocole de communication appliqué. La valeur suit le schéma d'informations de version défini par l'IEC 62769-4, lequel schéma est défini dans le type d'élément VersionT. Le Paragraphe 4.2.4 décrit comment appliquer les versions de protocole actuellement connues pour les entrées CP 9/1 ou CP 9/2 du catalogue d'appareils. La règle générale est d'utiliser la Révision Universelle du protocole pour la partie version majeure de VersionT, et la valeur "0" pour les parties version mineure et numéro de build (mouture). Le Tableau 4 donne les informations relatives à la Version de Protocole.

**Tableau 4 – Informations relatives à la Version de Protocole**

| Version de Protocole | Valeur de l'élément InterfaceT Version |
|---|---|
| Révision Universelle HART 5 | 5.0.0 |
| Révision Universelle HART 6 | 6.0.0 |
| Révision Universelle HART 7 | 7.0.0 |
| La Version de Protocole définie dans un paquetage est fournie à des fins d'information seulement et ne doit pas être utilisée pour déterminer la compatibilité ou l'applicabilité d'un paquetage vis-à-vis d'un appareil. | |

## 4.3 Association d'un Paquetage avec un appareil CP 9/1

### 4.3.1 Mapping d'identification du type d'appareil

Les types d'appareils CP 9/1 sont identifiés de façon univoque par les paramètres Manufacturer, Model et DeviceRevision. Ces paramètres sont utilisés pour associer une instance d'appareil donnée avec un Paquetage d'Appareil FDI. Ces paramètres sont mappés au Catalogue de Paquetage d'Appareil FDI conformément au Tableau 5.

**Tableau 5 – Mapping dans le catalogue des types d'appareils**

| Elément de Catalog | Mapping des types d'appareils CP (voir Tableau A.) |
|---|---|
| Elément Manufacturer d'InterfaceT<br>(IEC 62769-4, E.10) | Manufacturer<br><br>Format de chaîne "0xdd", où dddd est le numéro MANUFACTURER_ID au format hexadécimal. |
| Elément DeviceModel d'InterfaceT<br>(IEC 62769-4, E.10) | Model<br><br>Format de chaîne "0xdd", où dddd est le numéro DEVICE_TYPE au format hexadécimal. |
| Elément DeviceRevision<br><br>ListOfSupportedDeviceRevisionsT<br>(IEC 62769-4, E.20) | DeviceRevision<br><br>Format de chaîne "x.0.0", où x est le numéro DeviceRevision au format décimal (pas de zéro de tête). |

### 4.3.2 Mapping des révisions de type d'appareil

Chaque type d'appareil est identifié conformément à 4.3.1. Si un paquetage avec une DeviceRevision concordante n'est pas disponible, un quelconque paquetage FDI CP 9/1 pour un fabricant et un modèle qui correspondent doit toujours être compatible avec un appareil de terrain tant que la révision d'appareil relative à l'appareil de terrain est supérieure ou égale à la révision d'appareil spécifiée dans le paquetage FDI.

## 4.4 Mapping du Modèle d'Information

### 4.4.1 Définition de ProtocolType

Le Tableau 6 définit le ProtocolType utilisé pour identifier des communications réseau CP 9/1.

**Tableau 6 – Définition du ProtocolType HART**

| Attribut | Valeur | | | | |
|---|---|---|---|---|---|
| BrowseName | HART | | | | |
| IsAbstract | False | | | | |
| **Références** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Hérite des propriétés du ProtocolType définies dans l'IEC 62541-100. | | | | | |

### 4.4.2 Mapping de DeviceType

Chaque type d'appareil hérite des propriétés du DeviceType. Le mapping des propriétés héritées du DeviceType est défini dans le Tableau 7.

**Tableau 7 – Mapping des propriétés héritées du DeviceType**

| Propriété | Mapping Foundation |
|---|---|
| SerialNumber | ID unique d'un appareil, mappé sur l'attribut SERIAL_NUMBER de IdentificationT |
| RevisionCounter | Compteur de changement de configuration, mappé sur l'attribut REV_COUNTER de IdentificationT |
| Manufacturer | Chaîne obtenue à partir du catalogue de paquetages FDI (élément ManufacturerName issu de PackageT) |
| Model | Chaîne obtenue à partir du catalogue de paquetages FDI (élément Name de DeviceTypeT, qui est un nom localisé) |
| DeviceManual | Chaîne textuelle d'entrée (non prise en charge) [a] |
| DeviceRevision | Niveau de révision d'appareil d'un appareil, mappé sur l'attribut DEVICE_REVISION de IdentificationT |
| SoftwareRevision | Niveau de révision de logiciel d'un appareil, mappé sur l'attribut SOFTWARE_REVISION de IdentificationT |
| HardwareRevision | Niveau de révision de matériel d'un appareil, mappé sur l'attribut HARDWARE_REVISION de IdentificationT |
| [a] Les manuels d'appareils sont exposés comme pièces jointes du Paquetage d'Appareil FDI. | |

### 4.4.3 Définition du FunctionalGroup "Identification"

Comme défini dans l'IEC 62541-100, chaque représentation d'appareil dans le Modèle d'Information hébergé sur le Serveur FDI doit contenir un FunctionalGroup spécifique à un protocole, appelé Identification. Ce FunctionalGroup organise les variables détectées dans l'instance de type d'appareil. Le FunctionalGroup "Identification" pour CP 9/1 est défini dans le Tableau 8.

**Tableau 8 – Paramètres du FunctionalGroup Identification**

| BrowseName | DataType | Facultatif/Obligatoire |
|---|---|---|
| MANUFACTURER_ID | UInt16 | Obligatoire |
| DEVICE_TYPE | UInt16 | Obligatoire |
| DEVICE_REVISION | UInt8 | Obligatoire |
| UNIVERSAL_REVISION | UInt8 | Facultatif |
| SERIAL_NUMBER | UInt24 | Facultatif |
| HARDWARE_REVISION | UInt8 | Facultatif |
| SOFTWARE_REVISION | UInt8 | Facultatif |
| REVISION_COUNTER | UInt16 | Facultatif |



## 4.5 Eléments de topologie

### 4.5.1 Définition de ConnectionPoint

#### 4.5.1.1 Généralités

Les appareils CP 9/1 peuvent prendre en charge jusqu'à cinq types différents de ConnectionPoint qui sont utilisés pour les communications réseau.

#### 4.5.1.2 HART_TP5, HART_TP6, HART_TP7

Les types de ConnectionPoint HART_TP5, HART_TP6 et HART_TP7 doivent être utilisés pour identifier la communication de réseau avec passage de jeton CP 9/1 et sont définis dans le Tableau 9. Les types HART_TP5, HART_TP6 et HART_TP7 contiennent tous les mêmes

propriétés, mais chacun d'eux fournit des informations de qualification différentes pour certaines des propriétés (décrites ci-dessous). La Version de Protocole (UNIVERSAL_REVISION) décrite en 4.2.4 peut être utilisée comme aide pour déterminer lequel des trois types de Points de Connexion avec passage de jeton est le plus approprié. Les communications avec passage de jeton CP 9/1 peuvent être utilisées sur une diversité de couches physiques. Les connexions de couche physique FSK, PSK, RS485 et infrarouges doivent toutes utiliser le type de connexion HART_TP. Les types de ConnectionPoint HART_TP5, HART_TP6 et HART_TP7 sont des sous-types du type abstrait ConnectionPointType défini dans l'IEC 62769-5.

La propriété DevAddr doit être l'adresse longue (5 octets) pour l'appareil. En outre, elle est le seul paramètre nécessaire pour communiquer avec l'appareil de terrain.

La propriété DevMfg doit être le Manufacturer ID de 2 octets et elle peut être utilisée pour aider à automatiser le processus consistant à assigner des appareils actifs de la liste de balayage à des réceptacles hors ligne.

La propriété DevType doit être le type étendu d'appareil de 2 octets et elle peut être utilisée pour aider à automatiser le processus consistant à assigner des appareils actifs de la liste de balayage à des réceptacles hors ligne.

La propriété DevRev doit être la révision d'appareil et elle peut être utilisée pour aider à automatiser le processus consistant à assigner des appareils actifs de la liste de balayage à des réceptacles hors ligne.

La propriété DevTag doit être le marqueur long pour les appareils de la version de protocole HART® 6 ou 7. La propriété DevTag doit être le marqueur pour les appareils de la version de protocole 5. La propriété DevTag peut être utilisée pour aider à automatiser le processus consistant à assigner des appareils actifs de la liste de balayage à des réceptacles hors ligne. Les Points de Connexion HART_TP5 doivent limiter à 8 caractères la longueur de DevTag. Les Points de Connexion HART_TP6 et HART_TP7 doivent limiter à 32 caractères la longueur de DevTag.

La propriété DevPollAddr doit être l'adresse d'interrogation. Par ailleurs, elle peut être utilisée pour identifier quel appareil est localisé à une adresse d'interrogation spécifique. Les Points de Connexion HART_TP5 doivent être limités à des valeurs comprises entre 0 et 15 pour la propriété DevPollAddr. Les Points de Connexion HART_TP6 doivent être limités à des valeurs comprises entre 0 et 31 pour la propriété DevPollAddr. Les Points de Connexion HART_TP7 doivent être limités à des valeurs comprises entre 0 et 63 pour la propriété DevPollAddr.

Pour la post-compatibilité, un Point de Connexion HART_TP de révision inférieure est compatible et peut être utilisé pour une connexion d'appareil d'une révision universelle supérieure. Par exemple, si un appareil d'une future révision universelle HART 8 est rencontré et aucun point de connexion HART_TP8 n'est disponible dans le Serveur FDI, le point de connexion HART_TP7 est compatible et doit être utilisé pour se connecter à l'appareil. Si la Version de Protocole (c'est-à-dire la Révision Universelle) est inconnue pour une raison quelconque, le Point de Connexion HART_TP5 peut être utilisé et est post-compatible avec les révisions universelles ultérieures.



**Tableau 9 – Définition du ConnectionPointType HART_TP**

| Attribut | Valeur | | | | |
|---|---|---|---|---|---|
| BrowseName | ConnnectionPoint_HART_TP5 ou ConnnectionPoint_HART_TP6 ou ConnnectionPoint_HART_TP7 | | | | |
| IsAbstract | False | | | | |
| **Références** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Hérite des propriétés de ConnectionPointType définies dans l'IEC 62769-5 | | | | | |

| Attribut | Valeur | | | | |
|---|---|---|---|---|---|
| HasProperty | Variable | DevAddr | UInt40 | PropertyType | Obligatoire |
| HasProperty | Variable | DevMfg | UInt16 | PropertyType | Facultatif |
| HasProperty | Variable | DevType | UInt16 | PropertyType | Facultatif |
| HasProperty | Variable | DevRev | UInt16 | PropertyType | Facultatif |
| HasProperty | Variable | DevTag | String | PropertyType | Facultatif |
| HasProperty | Variable | DevPollAddr | UInt8 | PropertyType | Facultatif |

Les types de ConnectionPoint HART_TP5, HART_TP6, et HART_TP7 doivent être décrits par un élément EDD contenu dans un Paquetage FDI associé à l'Appareil de Communication, en mesure de piloter un réseau CP 9/1. Les propriétés réelles de ConnectionPoint sont déclarées par les constructions VARIABLE regroupées dans une COLLECTION nommée ConnectionPoint_HART_TP5, ConnectionPoint_HART_TP6 ou ConnectionPoint_HART_TP7. Le code source EDDL ci-après est un exemple décrivant un Point de Connexion TP5.

```
COMPONENT ConnectionPoint_HART_TP5
{
  LABEL "HART TP Connection Point";
  CLASSIFICATION NETWORK_CONNECTION_POINT;
  CAN_DELETE FALSE;
  PROTOCOL HART;
  CONNECTION_POINT ConnectionPoint_TP5;
}

VARIABLE DevAddr
{
  LABEL "Address";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER(5);
  HANDLING READ & WRITE;
}

VARIABLE DevMfg
{
  LABEL "Manufacturer";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER(2);
  HANDLING READ & WRITE;
}

VARIABLE DevType
{
  LABEL "Device Type";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER(2);
  HANDLING READ & WRITE;
}

VARIABLE DevRev
{
  LABEL "Device Revision";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER;
  HANDLING READ & WRITE;
}
```

```
VARIABLE DevTag
{
  LABEL "Tag";
  CLASS DEVICE;
  TYPE ASCII(32);
  HANDLING READ & WRITE;
}

VARIABLE DevPollAddr
{
  LABEL "Poll Address";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER
  {
    MAX_VALUE 15; //Define appropriate max value for various revisions
  }
  HANDLING READ & WRITE;
}

COLLECTION ConnectionPoint_TP5
{
  LABEL "Connection Point";
  MEMBERS
  {
    ADDRESS,    DevAddr,     "Device Address";
    MFG,        DevMfg,      "Manufacturer";
    DEV_TYPE,   DevType,     "Device Type";
    DEV_REV,    DevRev,      "Device Revision";
    TAG,        DevTag,      "Device Tag";
    POLL_ADDR, DevPollAddr, "Poll Address";
  }
}
```

#### 4.5.1.3 HART_IP

Le type de ConnectionPoint HART_IP doit être utilisé pour identifier la communication de réseau IP CP 9/1 et est défini dans le Tableau 10. Les communications HART_IP peuvent être utilisées sur une diversité de couches physiques. Toutes les connexions Ethernet doivent utiliser le type de connexion HART_IP. Les couches physiques supplémentaires qui seront développées dans le futur peuvent aussi utiliser le type de connexion HART_IP. Le type de ConnectionPoint HART_IP est un sous-type du type abstrait ConnectionPointType défini dans l'IEC 62769-5.

La propriété IPAddress doit indiquer l'adresse IP (jusqu'à 16 octets) utilisée pour la connexion.

La propriété IPVersion doit indiquer la version IP utilisée pour la connexion (version 4 ou 6).

La propriété IPPort doit être le numéro de port IP pour la connexion. Le numéro de port par défaut utilisé pour HART IP est 5 094.

La propriété DevAddr doit être l'adresse longue (5 octets) pour l'appareil.

La propriété DevMfg doit être le Manufacturer ID de 2 octets et elle peut être utilisée pour aider à automatiser le processus consistant à assigner des appareils actifs de la liste de balayage à des réceptacles hors ligne.

La propriété DevType doit être le type étendu d'appareil de 2 octets et elle peut être utilisée pour aider à automatiser le processus consistant à assigner des appareils actifs de la liste de balayage à des réceptacles hors ligne.

La propriété DevRev doit être la révision d'appareil et elle peut être utilisée pour aider à automatiser le processus consistant à assigner des appareils actifs de la liste de balayage à des réceptacles hors ligne.

La propriété DevTag doit être le marqueur long (avec 32 caractères au maximum) et elle peut être utilisée pour aider à automatiser le processus consistant à assigner des appareils actifs de la liste de balayage à des réceptacles hors ligne.

**Tableau 10 – Définition du ConnectionPointType HART_IP**

| **Attribut** | **Valeur** | | | | |
|---|---|---|---|---|---|
| BrowseName | ConnnectionPoint_HART_IP | | | | |
| IsAbstract | False | | | | |
| **Références** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Hérite des propriétés de ConnectionPointType définies dans l'IEC 62769-5 | | | | | |
| HasProperty | Variable | IPAddress | ByteString | PropertyType | Obligatoire |
| HasProperty | Variable | IPVersion | UInt8 | PropertyType | Obligatoire |
| HasProperty | Variable | IPPort | UInt16 | PropertyType | Obligatoire |
| HasProperty | Variable | DevAddr | UInt40 | PropertyType | Obligatoire |
| HasProperty | Variable | DevMfg | UInt16 | PropertyType | Facultatif |
| HasProperty | Variable | DevType | UInt16 | PropertyType | Facultatif |
| HasProperty | Variable | DevRev | UInt16 | PropertyType | Facultatif |
| HasProperty | Variable | DevTag | String | PropertyType | Facultatif |

Le type de ConnectionPoint HART_IP doit être décrit par un élément EDD contenu dans un Paquetage FDI associé à l'Appareil de Communication, en mesure de piloter un réseau CP 9/1. Les propriétés réelles de ConnectionPoint sont déclarées par les constructions VARIABLE regroupées dans une COLLECTION nommée ConnectionPoint_HART_IP. Le code source EDDL ci-après est un exemple décrivant un Point de Connexion IP.

```
COMPONENT ConnectionPoint_HART_IP
{
  LABEL "HART IP Connection Point";
  CLASSIFICATION NETWORK_CONNECTION_POINT;
  CAN_DELETE FALSE;
  PROTOCOL HART;
  CONNECTION_POINT ConnectionPoint_IP;
}

ARRAY IPAddress
{
  LABEL "IP Address";
  CLASS DEVICE;
  TYPE OCTET(16);
  HANDLING READ & WRITE;
}
```

```
VARIABLE IPVersion
{
  LABEL "IP Version";
  CLASS DEVICE;
  TYPE ENUMERATED
  {
    { 4, "IPv4" },
    { 6, "IPv6" }
  }
  HANDLING READ & WRITE;
}

VARIABLE IPPort
{
  LABEL "IP Port";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER (2);
  DEFAULT_VALUE 5 094;
  HANDLING READ & WRITE;
}

COLLECTION ConnectionPoint_IP
{
  LABEL "Connection Point";
  MEMBERS
  {
    IPADDRESS, IPAddress, "IP Address";
    IPVERSION, IPVersion, "IP Version";
    IPPORT,    IPPort,    "IP Port";
    ADDRESS,   DevAddr,   "Device Address";
    MFG,       DevMfg,    "Manufacturer";
    DEV_TYPE,  DevType,   "Device Type";
    DEV_REV,   DevRev,    "Device Revision";
    TAG,       DevTag,    "Device Tag";
  }
}
```

#### 4.5.1.4 HART_TDMA

Le type de ConnectionPoint HART_TDMA doit être utilisé pour identifier une communication réseau à accès multiple par répartition dans le temps CP 9/2 et il est défini dans le Tableau 11. Les communications HART_TDMA peuvent être utilisées sur une diversité de couches physiques. Le type de ConnectionPoint HART_TDMA est un sous-type du type abstrait ConnectionPointType défini dans l'IEC 62769-5. Toutes les connexions WirelessHART doivent utiliser le type de connexion HART_TDMA. Les couches physiques supplémentaires qui seront développées dans le futur peuvent aussi utiliser le type de connexion HART_TDMA.

La propriété Network doit être l'ID de réseau pour le réseau.

La propriété DevAddr doit être l'adresse longue (5 octets) pour l'appareil.

La propriété DevMfg doit être le Manufacturer ID de 2 octets et elle peut être utilisée pour aider à automatiser le processus consistant à assigner des appareils actifs de la liste de balayage à des réceptacles hors ligne.

La propriété DevType doit être le type étendu d'appareil de 2 octets et elle peut être utilisée pour aider à automatiser le processus consistant à assigner des appareils actifs de la liste de balayage à des réceptacles hors ligne.

La propriété DevRev doit être la révision d'appareil et elle peut être utilisée pour aider à automatiser le processus consistant à assigner des appareils actifs de la liste de balayage à des réceptacles hors ligne.

La propriété DevTag doit être le marqueur long (avec 32 caractères au maximum) et elle peut être utilisée pour aider à automatiser le processus consistant à assigner des appareils actifs de la liste de balayage à des réceptacles hors ligne.

**Tableau 11 – Définition du ConnectionPointType HART_TDMA**

| Attribut | Valeur | | | | |
|---|---|---|---|---|---|
| BrowseName | ConnnectionPoint_HART_TDMA | | | | |
| IsAbstract | False | | | | |
| **Références** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Hérite des propriétés de ConnectionPointType définies dans l'IEC 62769-5 | | | | | |
| HasProperty | Variable | Network | UInt16 | PropertyType | Obligatoire |
| HasProperty | Variable | DevAddr | UInt40 | PropertyType | Obligatoire |
| HasProperty | Variable | DevMfg | UInt16 | PropertyType | Facultatif |
| HasProperty | Variable | DevType | UInt16 | PropertyType | Facultatif |
| HasProperty | Variable | DevRev | UInt16 | PropertyType | Facultatif |
| HasProperty | Variable | DevTag | String | PropertyType | Facultatif |

Le type de ConnectionPoint HART_TDMA doit être décrit par un élément EDD contenu dans un Paquetage FDI associé à l'Appareil de Communication, en mesure de piloter un réseau CP 9/2. Les propriétés réelles de ConnectionPoint sont déclarées par les constructions VARIABLE regroupées dans une COLLECTION nommée ConnectionPoint_HART_TDMA. Le code source EDDL ci-après est un exemple décrivant un Point de Connexion TDMA.

```
COMPONENT ConnectionPoint_HART_TDMA
{
  LABEL "HART TDMA Connection Point";
  CLASSIFICATION NETWORK_CONNECTION_POINT;
  CAN_DELETE FALSE;
  PROTOCOL HART;
  CONNECTION_POINT ConnectionPoint_TDMA;
}

VARIABLE Network
{
  LABEL "Network ID";
  CLASS DEVICE;
  TYPE UNSIGNED_INTEGER (2);
  HANDLING READ & WRITE;
```