



IEC 62769-109-1

Edition 3.0 2023-04
REDLINE VERSION

INTERNATIONAL STANDARD



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





THIS PUBLICATION IS COPYRIGHT PROTECTED
Copyright © 2023 IEC, Geneva, Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either IEC or IEC's member National Committee in the country of the requester. If you have any questions about IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

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

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

About the IEC

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

About IEC publications

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

IEC publications search - webstore.iec.ch/advsearchform

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 It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished

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

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

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

IEC Products & Services Portal - products.iec.ch

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

Electropedia - www.electropedia.org

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



IEC 62769-109-1

Edition 3.0 2023-04
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 978-2-8322-6888-9

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

© Registered trademark of the International Electrotechnical Commission

AVA Documents
Document Sharing Hub

CONTENTS

FOREWORD	4
1 Scope	6
2 Normative references	6
3 Terms, definitions, abbreviated terms and conventions acronyms	6
3.1 Terms and definitions	6
3.2 Abbreviated terms and acronyms	7
4 Conventions	7
4.1 EDDL syntax	7
4.2 XML syntax	7
4.3 Capitalizations	7
5 Profile for CP 9/1 (HART®) or CP 9/2 (WirelessHART®)	8
5.1 General	8
5.2 Catalog profile	8
5.2.1 Protocol support file	8
5.2.2 CommunicationProfile definition	8
5.2.3 Profile device	9
5.2.4 Protocol version information	9
5.3 Associating a Package with a CP 9/1 device	10
5.3.1 Device type identification mapping	10
5.3.2 Device type revision mapping	10
5.4 Information Model mapping	10
5.4.1 ProtocolType definition	10
5.4.2 DeviceType mapping	10
5.4.3 FunctionalGroup Identification definition	11
5.5 Topology elements	11
5.5.1 ConnectionPoint definition	11
5.5.2 Communication Device definition	17
5.5.3 Communication service provider definition	18
5.5.4 Network definition	19
5.6 Methods	20
5.6.1 Methods for FDI® Communication Servers	20
5.6.2 Methods for Gateways	24
Annex A (normative) Topology scan schema	33
A.1 General	33
A.2 IdentificationType	33
A.3 AddressTypeTP	36
A.4 AddressTypeIP	37
A.5 AddressTypeTDMA	37
A.6 AddressType	38
A.7 ConnectionPointType	39
A.8 NetworkType	39
A.9 Network	40
Annex B (normative) Transfer service parameters	41
B.1 General	41
B.2 receiveData	41
B.3 sendData	41

B.4	TransferResultDataT	41
B.5	TransferSendDataT	42
Annex C (informative)	Mapping to PA DIM	43
C.1	General	43
C.2	Mapping Table	43
Bibliography		44
Table 1	– Device Information Files	8
Table 2	– CommunicationProfile definition	9
Table 3	– Catalog values for profile devices	9
Table 4	– Protocol Version Information	9
Table 5	– Device type catalog mapping	10
Table 6	– ProtocolType HART definition	10
Table 7	– Inherited DeviceType Property mapping	11
Table 8	– Identification parameters	11
Table 9	– ConnectionPointType HART_TP definition	12
Table 10	– ConnectionPointType HART_IP Definition	14
Table 11	– ConnectionPointType HART_TDMA Definition	17
Table 12	– Method Connect arguments	21
Table 13	– Method Disconnect arguments	21
Table 14	– Method Transfer arguments	22
Table 15	– Method GetPublishedData arguments	23
Table 16	– Method SetAddress arguments	24
Table 17	– Method Connect arguments	25
Table 18	– Method Disconnect arguments	26
Table 19	– Method Transfer arguments	27
Table 20	– Method GetPublishedData arguments	29
Table 21	– Method SetAddress arguments	30
Table A.1	– Attributes of IdentificationT	35
Table A.2	– Elements of AddressTypeTP	36
Table A.3	– Elements of AddressTypeIP	37
Table A.4	– Elements of AddressTypeTDMA	38
Table A.5	– Elements of AddressT	38
Table A.6	– Elements of ConnectionPointT	39
Table A.7	– Elements of NetworkT	39
Table B.1	– Attributes of TransferResultDataT	41
Table B.2	– Attributes of TransferSendDataT	42
Table C.1	– Mapping from HART standard parameters to PA DIM	43

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 IEC 62769-109-1:2020. A vertical bar appears in the margin wherever a change has been made. Additions are in green text, deletions are in strikethrough red text.

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

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

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

- a) added content type for DeviceInfo files;
- b) added mapping from HART standard parameters to PA DIM;

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

Draft	Report on voting
65E/864/CDV	65E/921/RVC

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

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

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

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

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

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

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

FIELD DEVICE INTEGRATION (FDI®) –

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

1 Scope

This part of IEC 62769 specifies an FDI®¹ profile of IEC 62769 for IEC 61784-1_CP 9/1 (HART®)² and IEC 61784-1_CP 9/2 (WirelessHART®)³.

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: OPC Device Interface*

IEC 62769-4:~~4~~, *Field device integration (FDI®) – Part 4: FDI® Packages*

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

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

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

3.1 Terms and definitions

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

² ~~HART® and wirelessHART® are~~ is the ~~registered trademark~~ trade name of the non-profit consortium FieldComm Group. This information is given for the convenience of users of this technical report and does not constitute an endorsement by IEC of the ~~product named~~ trademark holder or any of its products. ~~Equivalent products may be used if they can be shown to lead to the same results.~~ Compliance does not require use of the trade name. Use of the trade name requires permission of the trade name holder.

³ WirelessHART is the trade name of the non-profit consortium FieldComm Group. This information is given for the convenience of users of this technical report and does not constitute an endorsement by IEC of the trademark holder or any of its products. Compliance does not require use of the trade name. Use of the trade name requires permission of the trade name holder.

⁴ ~~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 abbreviated terms and acronyms 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 the IEC 61804 series)
EDDL	Electronic device description language (see the IEC 61804 series)
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 Extended markup language

4 Conventions

4.1 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.

4.2 XML syntax

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

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

4.3 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

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

5.1 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.

Annex B defines the XML schema for Direct Access Services. Annex C provides an overview of mapping PROFIBUS standard parameters to PA DIM.

5.2 Catalog profile

5.2.1 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 application/vnd.hart.json
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.

5.2.2 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.	

5.2.3 Profile device

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

5.2.4 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 5.2.4 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.	

5.3 Associating a Package with a CP 9/1 device

5.3.1 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.1)
Manufacturer element of InterfaceT (IEC 62769-4:2023, Clause E.11)	Manufacturer String format "0xddd" where dddd is the MANUFACTURER_ID in hexadecimal format.
DeviceModel element of InterfaceT (IEC 62769-4:2023, Clause E.11)	Model String format "0xddd" where dddd is the DEVICE_TYPE in hexadecimal format.
DeviceRevision element ListOfSupportedDeviceRevisionsT (IEC 62769-4:2023, Clause E.20)	DeviceRevision String format "x.0.0" where x is the DEVICE_REVISION in decimal format (no leading zeros).

5.3.2 Device type revision mapping

Each device type is identified as per 5.3.1. 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.

5.4 Information Model mapping

5.4.1 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	Data Type	TypeDefinition	ModellingRule
Inherits the properties of ProtocolType defined in IEC 62541-100					

5.4.2 DeviceType mapping

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

5.4.3 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

5.5 Topology elements

5.5.1 ConnectionPoint definition

5.5.1.1 General

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

5.5.1.2 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 5.2.3 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	ConnectionPoint_HART_TP5 or ConnectionPoint_HART_TP6 or ConnectionPoint_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";
    }
}
```

5.5.1.3 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	ConnectionPoint_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 5-0945094;
    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";
    }
}

```

5.5.1.4 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	ConnectionPoint_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";
    }
}

```

5.5.2 Communication Device definition

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

5.5.3 Communication service provider definition

According to IEC 62769-7 each FDI® Communication Package shall contain at least one EDD element describing at least one communication service provider component. The following EDDL source code 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 5.5.1. 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.

5.5.4 Network definition

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

5.6 Methods

5.6.1 Methods for FDI[®] Communication Servers

5.6.1.1 General

The FDI[®] Communication Server shall implement services according to the method signatures described in this section and according to the FDI[®] Information Model.

5.6.1.2 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 5.5.1. 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 -1: Connect Failed / canceled by caller -2: Call Failed / unknown service ID -3: Connect Failed / device not found -4: Connect Failed / invalid device node address -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.	
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.	

5.6.1.3 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 -1: Disconnect Failed / no existing communication relation -2: Disconnect Failed / invalid communication relation identifier

5.6.1.4 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 -1: Transfer Failed / cancelled by caller -2: Call Failed / unknown service ID -3: Transfer Failed / no existing communication relation. -4: Transfer Failed / invalid communication relation identifier -5: Transfer Failed / invalid Request content -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.	
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.	

5.6.1.5 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 -1: GetPublishedData Failed / cancelled by caller -2: Call Failed / unknown service ID -3: GetPublishedData Failed / not supported -4: GetPublishedData Failed / no existing communication relation. -5: GetPublishedData Failed / invalid communication relation identifier -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.	

5.6.1.6 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 -1: SetAddress Failed / cancelled by caller -2: Call Failed / unknown service ID -3: SetAddress Failed / not initialized -4: SetAddress Failed / not connected to a network -5: SetAddress Failed / no device found responding to oldAddress -6: SetAddress Failed / duplicate address error -7: SetAddress Failed / device did not accept new address -8: SetAddress Failed / invalid oldAddress (in terms of syntax, data type, data format, and so on) -9: SetAddress Failed / invalid newAddress (in terms of syntax, data type, data format, and so on) -10: SetAddress Failed / not possible in status connected
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.	
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.	

5.6.1.7 Scan

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

5.6.1.8 ResetScan

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

5.6.2 Methods for Gateways

5.6.2.1 General

The method signatures described in 5.6.2 shall be implemented in the EDD element IEC 62769-4 contained in a Gateway related FDI[®] Package containing the communication device definitions.

5.6.2.2 Connect

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 5.5.1. 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 0: OK / execution finished, connection established successfully -1: Connect Failed / cancelled by caller -2: Call Failed / unknown service ID -3: Connect Failed / device not found -4: Connect Failed / invalid device address -5: Connect Failed / invalid device identification

5.6.2.3 Disconnect

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

5.6.2.4 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 0: OK / execution finished -1: Transfer Failed / cancelled by caller -2: Call Failed / unknown service ID -3: Transfer Failed / no existing communication relation. -4: Transfer Failed / invalid communication relation identifier -5: Transfer Failed / invalid REQUEST content -6: Transfer Failed / invalid REPLY format

5.6.2.5 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 noId 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 0: OK / execution finished -1: GetPublishedData Failed / cancelled by caller -2: Call Failed / unknown service ID -3: GetPublishedData Failed / not supported -4: GetPublishedData Failed / no existing communication relation. -5: GetPublishedData Failed / invalid communication relation identifier -8: GetPublishedData Failed / no burst information published

5.6.2.6 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 0: OK / execution finished successfully -1: SetAddress Failed / cancelled by caller -2: Call Failed / unknown service ID -3: SetAddress Failed / not initialized -4: SetAddress Failed / not connected to a network -5: SetAddress Failed / no device found responding to oldAddress -6: SetAddress Failed / duplicate address error -7: SetAddress Failed / device did not accept new address -8: SetAddress Failed / invalid oldAddress (in terms of syntax, data type, data format, and so on) -9: SetAddress Failed / invalid newAddress (in terms of syntax, data type, data format, and so on) -10: SetAddress Failed /not possible in status connected

5.6.2.7 Scan

The method signature specified in IEC 62769-7 applies. The 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
}

```

5.6.2.8 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 5.6.2.7. 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="1677721616777215"/>
      </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	<p>Manufacturer identification number.</p> <p>For HART universal revision 7 or higher devices:</p> <p>2-byte manufacturer code of a device, returned in bytes 17 and 18 of Command 0 or Command 11 or Command 21.</p> <p>For HART revision 6 or lower devices:</p> <p>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</p>
DEVICE_TYPE	<p>Manufacturer's model number associated with the resource.</p> <p>For HART universal revision 7 or higher devices:</p> <p>2 byte device type returned in bytes 1 and 2 of Command 0 or Command 11 or Command 21</p> <p>For HART revision 6 or lower devices:</p> <p>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</p>
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	<p>The configuration change counter associated with the device.</p> <p>For HART universal revision 6 or higher devices:</p> <p>Returned in bytes 14 and 15 of Command 0 or Command 11 or Command 21.</p> <p>For HART revision 5 or lower devices:</p> <p>-1 (not defined).</p>
TAG	<p>The configured Identification tag associated with the device.</p> <p>For HART universal revision 6 or higher devices:</p> <p>32 character long tag returned in Command 20 or Command 21.</p> <p>For HART revision 5 or lower devices:</p> <p>8 character tag returned in Command 11 or Command 13.</p>

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 5.5.1.2.
AddressIP	IP addresses hold the address information described in ConnectionPointType_HART_IP described in 5.5.1.2.
AddressTDMA	TDMA addresses hold the address information described in ConnectionPointType_HART_TDMA described in 5.5.1.2.

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>
```