|  |  |
| --- | --- |
|  | OPC UA Specification |
| OPC 10000-22 |
| OPC Unified Architecture  Part 22: Base Network Model  Release Candidate 1.04.00  2020-10-07 |

|  |  |  |  |
| --- | --- | --- | --- |
| Specification Type: | Industry Standard Specification | Comments: |  |
|  |  |  |  |
| Document Number | OPC 10000-22 |  |  |
| Title: | OPC Unified Architecture   :Base Network Model | Date: |  |
|  |  |  |  |
| Version: |  | Software: | MS-Word |
|  |  | Source: | OPC 10000-22 - UA Specification Part 22 - Base Network Model RC 1.04.00.docx |
|  |  |  |  |
| Author: |  | Status: |  |
|  |  |  |  |

**CONTENTS**

Page

[1 Scope 1](#_Toc52983983)

[2 Normative references 1](#_Toc52983984)

[3 Terms, definitions, abbreviated terms, and conventions 2](#_Toc52983985)

[3.1 Terms and definitions 2](#_Toc52983986)

[3.2 Abbreviated terms 3](#_Toc52983987)

[4 Concepts 3](#_Toc52983988)

[4.1 Type and Naming Conventions 3](#_Toc52983989)

[4.2 Usage of OPC UA Interfaces 3](#_Toc52983990)

[4.3 References to or from Object Instances of the Base Network Model 3](#_Toc52983991)

[5 Model 3](#_Toc52983992)

[5.1 Overview 3](#_Toc52983993)

[5.2 OPC UA Interfaces 5](#_Toc52983994)

[5.2.1 IIetfNetworkInterfaceType Interface 5](#_Toc52983995)

[5.2.2 IEthernetPortType Interface 6](#_Toc52983996)

[5.2.3 IAutoNegotiationFeatureType Interface 6](#_Toc52983997)

[5.2.4 IEthernetCapabilitiesType Interface 7](#_Toc52983998)

[5.2.5 ISrClassType Interface 7](#_Toc52983999)

[5.2.6 ITsnStreamType Interface 7](#_Toc52984000)

[5.2.7 ITsnTrafficSpecificationType Interface 8](#_Toc52984001)

[5.2.8 ITsnStatusStreamType Interface 9](#_Toc52984002)

[5.2.9 ITsnInterfaceConfigurationType Interface 9](#_Toc52984003)

[5.2.10 ITsnInterfaceConfigurationTalkerType Interface 9](#_Toc52984004)

[5.2.11 ITsnInterfaceConfigurationListenerType Interface 10](#_Toc52984005)

[5.2.12 ITsnMacAddressType Interface 10](#_Toc52984006)

[5.2.13 ITsnVlanTagType Interface 10](#_Toc52984007)

[5.2.14 IPriorityMappingEntryType Interface 10](#_Toc52984008)

[5.3 ReferenceTypes 11](#_Toc52984009)

[5.3.1 CommLinkTo 11](#_Toc52984010)

[5.4 DataTypes 11](#_Toc52984011)

[5.4.1 Enumeration DataTypes 11](#_Toc52984012)

[5.4.2 Structure DataTypes 15](#_Toc52984013)

[5.5 Instance Entry Points 15](#_Toc52984014)

[5.5.1 Resources Folder 16](#_Toc52984015)

[5.5.2 Communication Folder 16](#_Toc52984016)

[5.5.3 MappingTables Folder 17](#_Toc52984017)

[5.5.4 NetworkInterfaces Folder 17](#_Toc52984018)

[5.5.5 Streams Folder 17](#_Toc52984019)

[5.5.6 TalkerStreams Folder 17](#_Toc52984020)

[5.5.7 ListenerStreams Folder 17](#_Toc52984021)

[Annex A Modelling Examples (informative) 19](#_Toc52984022)

[A.1 Modelling Examples for Network Interfaces 19](#_Toc52984023)

[A.1.1 Virtual Network Interfaces 19](#_Toc52984024)

[A.1.2 Link Aggregation 19](#_Toc52984025)

[A.2 Modelling Examples for PriorityMappingEntry Interfaces 20](#_Toc52984026)

[A.3 Usage of BNM in other UA Specifications 22](#_Toc52984027)

[A.3.1 Usage of BNM for PubSub over TSN 22](#_Toc52984028)

[A.3.2 Usage of BNM in PROFINET Companion Spec 22](#_Toc52984029)

**Figures**

Figure 1 – Scope of Base Network Model 1

Figure 2 – Overview of OPC UA Interfaces 4

Figure 3 –Instance Entry Points for Network Interfaces and Streams 16

Figure A-1 Modelling Example for virtual network interfaces 19

Figure A-2 Modelling example for link aggregation 20

Figure A-3 - Modelling Example for Priority Mappings with IPriorityMappingEntryType as described in 5.2.14 21

Figure A-4 – Possible Integration of BNM into PubSub 22

Figure A-5 – Recommended Integration of BNM into Companion Spec exemplified by PROFINET 23

Figure A-6 – Example of recommended Integration of BNM into PROFINET Companion Spec 23

**Tables**

[Table 1 – IIetfNetworkInterfaceType definition 5](#_Toc52984039)

[Table 2 – IIetfNetworkInterfaceType Attribute values for child Nodes 5](#_Toc52984040)

[Table 3 – IEthernetPortType definition 6](#_Toc52984041)

[Table 4 – IEthernetPortType Attribute values for child Nodes 6](#_Toc52984042)

[Table 5 – IAutoNegotiationFeatureType definition 6](#_Toc52984043)

[Table 6 – IEthernetCapabilitiesType definition 7](#_Toc52984044)

[Table 7 – ISrClassType definition 7](#_Toc52984045)

[Table 8 – ITsnStreamType definition 7](#_Toc52984046)

[Table 9 – ITsnTrafficSpecificationType definition 8](#_Toc52984047)

[Table 10 – ITsnStatusStreamType definition 9](#_Toc52984048)

[Table 11 – ITsnInterfaceConfigurationType definition 9](#_Toc52984049)

[Table 12 – ITsnInterfaceConfigurationTalkerType definition 9](#_Toc52984050)

[Table 13 – ITsnInterfaceConfigurationListenerType definition 10](#_Toc52984051)

[Table 14 – ITsnMacAddressType definition 10](#_Toc52984052)

[Table 15 – ITsnVlanTagType definition 10](#_Toc52984053)

[Table 16 – IPriorityMappingEntryType definition 11](#_Toc52984054)

[Table 17 – CommLinkTo ReferenceType 11](#_Toc52984055)

[Table 18 –Duplex Values 11](#_Toc52984056)

[Table 19 –Duplex Definition 12](#_Toc52984057)

[Table 20 – InterfaceAdminStatus Values 12](#_Toc52984058)

[Table 21 – InterfaceAdminStatus Definition 12](#_Toc52984059)

[Table 22 – InterfaceOperStatus Values 12](#_Toc52984060)

[Table 23 – InterfaceOperStatus Definition 12](#_Toc52984061)

[Table 24 – NegotiationStatus Values 13](#_Toc52984062)

[Table 25 – NegotiationStatus Definition 13](#_Toc52984063)

[Table 26 – TsnFailureCode values 14](#_Toc52984064)

[Table 27 – TsnFailureCode Definition 14](#_Toc52984065)

[Table 28 – TsnStreamState Values 15](#_Toc52984066)

[Table 29 – TsnStreamState Definition 15](#_Toc52984067)

[Table 30 – FractionDataType structure 15](#_Toc52984068)

[Table 31 – FractionDataType Definition 15](#_Toc52984069)

[Table 32 – Resources definition 16](#_Toc52984070)

[Table 33 – Communication definition 17](#_Toc52984071)

[Table 34 – MappingTables definition 17](#_Toc52984072)

[Table 35 – NetworkInterfaces definition 17](#_Toc52984073)

[Table 36 – Streams definition 17](#_Toc52984074)

[Table 37 – TalkerStreams definition 17](#_Toc52984075)

[Table 38 – ListenerStreams definition 18](#_Toc52984076)

OPC Foundation

\_\_\_\_\_\_\_\_\_\_\_\_

UNIFIED ARCHITECTURE

FOREWORD

This specification is the specification for developers of OPC UA applications. The specification is a result of an analysis and design process to develop a standard interface to facilitate the development of applications by multiple vendors that shall inter-operate seamlessly together.

**Copyright © 2006-2020, OPC Foundation, Inc.**

AGREEMENT OF USE

COPYRIGHT RESTRICTIONS

Any unauthorized use of this specification may violate copyright laws, trademark laws, and communications regulations and statutes. This document contains information which is protected by copyright. All Rights Reserved. No part of this work covered by copyright herein may be reproduced or used in any form or by any means--graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems--without permission of the copyright owner.

OPC Foundation members and non-members are prohibited from copying and redistributing this specification. All copies must be obtained on an individual basis, directly from the OPC Foundation Web site  
H[TUhttp://www.opcfoundation.orgUT](http://www.opcfoundation.org)H.

PATENTS

The attention of adopters is directed to the possibility that compliance with or adoption of OPC specifications may require use of an invention covered by patent rights. OPC shall not be responsible for identifying patents for which a license may be required by any OPC specification, or for conducting legal inquiries into the legal validity or scope of those patents that are brought to its attention. OPC specifications are prospective and advisory only. Prospective users are responsible for protecting themselves against liability for infringement of patents.

WARRANTY AND LIABILITY DISCLAIMERS

WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PROVIDED "AS IS" AND MAY CONTAIN ERRORS OR MISPRINTS. THE OPC FOUDATION MAKES NO WARRANTY OF ANY KIND, EXPRESSED OR IMPLIED, WITH REGARD TO THIS PUBLICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR USE. IN NO EVENT SHALL THE OPC FOUNDATION BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER DAMAGES, INCLUDING LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY ANY USER OR ANY THIRD PARTY IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The entire risk as to the quality and performance of software developed using this specification is borne by you.

RESTRICTED RIGHTS LEGEND

This Specification is provided with Restricted Rights. Use, duplication or disclosure by the U.S. government is subject to restrictions as set forth in (a) this Agreement pursuant to DFARs 227.7202-3(a); (b) subparagraph (c)(1)(i) of the Rights in Technical Data and Computer Software clause at DFARs 252.227-7013; or (c) the Commercial Computer Software Restricted Rights clause at FAR 52.227-19 subdivision (c)(1) and (2), as applicable. Contractor / manufacturer are the OPC Foundation,. 16101 N. 82nd Street, Suite 3B, Scottsdale, AZ, 85260-1830

COMPLIANCE

The OPC Foundation shall at all times be the sole entity that may authorize developers, suppliers and sellers of hardware and software to use certification marks, trademarks or other special designations to indicate compliance with these materials. Products developed using this specification may claim compliance or conformance with this specification if and only if the software satisfactorily meets the certification requirements set by the OPC Foundation. Products that do not meet these requirements may claim only that the product was based on this specification and must not claim compliance or conformance with this specification.

Trademarks

Most computer and software brand names have trademarks or registered trademarks. The individual trademarks have not been listed here.

GENERAL PROVISIONS

Should any provision of this Agreement be held to be void, invalid, unenforceable or illegal by a court, the validity and enforceability of the other provisions shall not be affected thereby.

This Agreement shall be governed by and construed under the laws of the State of Minnesota, excluding its choice or law rules.

This Agreement embodies the entire understanding between the parties with respect to, and supersedes any prior understanding or agreement (oral or written) relating to, this specification.

ISSUE REPORTING

The OPC Foundation strives to maintain the highest quality standards for its published specifications; hence they undergo constant review and refinement. Readers are encouraged to report any issues and view any existing errata here: H[TUhttp://www.opcfoundation.org/errataUT](http://www.opcfoundation.org/errata)H

# Scope

The Base Network Model (BNM) specifies an OPC UA *Information Model* for a basic set of network related components to be used in other OPC specifications.

The initial version defines parameter sets for TSN Talkers and Listeners as well as network interfaces and ports as shown in Figure 1. A future version of this specification is expected to have a broader scope of other network technologies than Ethernet only.



Figure 1 – Scope of Base Network Model

# Normative references

The following referenced documents are indispensable for the application of this OPC UA part. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments and errata) applies.

OPC 10000-1, *OPC Unified Architecture - Part 1: Overview and Concepts*

<http://www.opcfoundation.org/UA/Part1/>

OPC 10000-3, *OPC Unified Architecture - Part 3: Address Space Model*

<http://www.opcfoundation.org/UA/Part3/>

OPC 10000-5, *OPC Unified Architecture - Part 5: Information Model*

<http://www.opcfoundation.org/UA/Part5/>

OPC 10000-8, *OPC Unified Architecture - Part 8: Data Access*

<http://www.opcfoundation.org/UA/Part8/>

IEEE Std 802.3, *ETHERNET*

<http://www.ieee802.org/3/>

IEEE 802.1Q-2018, *IEEE Standard for Local and Metropolitan Area Networks Bridges and Bridged Networks*

<http://www.ieee802.org/1/>

IEEE 802.1Qcc-2018, *Bridges and Bridged Networks, Amendment: Stream Reservation Protocol (SRP) Enhancements and Performance Improvements*

<http://www.ieee802.org/1/>

IEC/IEEE 60802, *TSN Profile for Industrial Automation*

<https://1.ieee802.org/tsn/iec-ieee-60802>

IETF RFC 8343, *A YANG Data Model for Interface Management*

<https://tools.ietf.org/html/rfc8343>

IETF RFC 2863, *The Interfaces Group MIB*

<https://tools.ietf.org/html/rfc2863>

IETF RFC 4836, *Definitions of Managed Objects for IEEE 802.3 Medium Attachment Units (MAUs)*

<https://tools.ietf.org/html/rfc4836>

# Terms, definitions, abbreviated terms, and conventions

## Terms and definitions

For the purposes of this document, the terms and definitions given in OPC 10000-1, OPC 10000-3, OPC 10000-5 and OPC 10000-8 apply.

All used terms are *italicized* in this document.

## Abbreviated terms

AVB Audio Video Bridging

BNM Base Network Model

CUC Centralized User Configuration

CNC Centralized Network Configuration

IEEE Institute of Electrical and Electronics Engineers

IETF Internet Engineering Task Force

MAU Medium Attachment Units

MIB Management Information Base

TSN Time Sensitive Networks

VLAN Virtual Local Area Network

YANG Yet Another Next Generation (Data modelling language for network management)

# Concepts

## Type and Naming Conventions

The BNM shall align its parameters to existing standards defined by IETF and the IEEE to allow an effortless mapping against existing network technologies. Therefore, selected *DataTypes* shall fit to the types used by the related managed objects of IEEE and IETF. *BrowseNames* of *Variables* and parameter sets (UA interface) are preferably derived from standardized IETF / IEEE YANG models. If no standardized YANG representation is available, MIB definitions are chosen.

## Usage of OPC UA Interfaces

The parameters of the BNM are grouped in the form of OPC UA *Interfaces*. *Interfaces* have been chosen instead of concrete *ObjectTypes* to define parameter sets independent of the implementation in future *ObjectType* hierarchies.

It is expected that a future version of the BNM will define a collection of network related *ObjectTypes*.

## References to or from Object Instances of the Base Network Model

Logical dependencies and hierarchies between *Object* instances of the BNM are represented by dedicated *References* e.g. *CommLinkTo* defined in 5.3.1.

# Model

## Overview

The OPC UA *Interfaces* defined in this document are shown in Figure 2.



Figure 2 – Overview of OPC UA Interfaces

## OPC UA Interfaces

### IIetfNetworkInterfaceType Interface

This OPC UA Interface defines the basis of an IETF network interface. The *IIetfNetworkInterfaceType* is formally defined in Table 1.

Table 1 – IIetfNetworkInterfaceType definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | IIetfNetworkInterfaceType | | | | |
| IsAbstract | True | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Subtype of the *BaseInterfaceType* defined in OPC 10000-5 | | | | | |
| HasComponent | Variable | AdminStatus | InterfaceAdminStatus | BaseDataVariableType | Mandatory |
| HasComponent | Variable | OperStatus | InterfaceOperStatus | BaseDataVariableType | Mandatory |
| HasComponent | Variable | PhysAddress | String | BaseDataVariableType | Optional |
| HasComponent | Variable | Speed | UInt64 | AnalogUnitType | Mandatory |

*AdminStatus* of *DataType* *InterfaceAdminStatus* specifies the desired state of the network interface. This *Variable* has the same read semantics as ifAdminStatus (see RFC 2863: The Interfaces Group MIB - ifAdminStatus). The *InterfaceAdminStatus* *Enumeration* is defined in 5.4.1.2.

*OperStatus* of *DataType* *InterfaceOperStatus* specifies the current operational state of the network interface. This *Variable* has the same semantics as ifOperStatus (see RFC 2863: The Interfaces Group MIB - ifOperStatus). The *InterfaceOperStatus* *Enumeration* is defined in 5.4.1.3.

*PhysAddress* of *DataType* *String* specifies the network interface's address at its protocol sub-layer. For example, for an 802.x network interface, this parameter normally contains a Media Access Control (MAC) address. The network interface's media-specific modules must define the bit and byte ordering and the format of the value of this object. For network interfaces that do not have such an address (e.g., a serial line), this node is not present (see RFC 2863: The Interfaces Group MIB - ifPhysAddress).

*Speed* of *DataType* *UInt64* specifies an estimate of the network interface's current bandwidth in bits per second. For network interfaces that do not vary in bandwidth or for those where no accurate estimation can be made, this Value should contain the nominal bandwidth (see RFC2863: ifSpeed, ifHighSpeed).

The component *Variables* of the *IIetfNetworkInterfaceType* have the *Attribute* values defined in Table 2.

Table 2 – IIetfNetworkInterfaceType Attribute values for child Nodes

|  |  |  |
| --- | --- | --- |
| **Source Path** | **Value Attribute** | **Description Attribute** |
| |  | | --- | | Speed | | 0:EngineeringUnits | | NamespaceUri: http://www.opcfoundation.org/UA/units/un/cefact  UnitId: 4337968 DisplayName: bit/s Description: bit per second | - |

### IEthernetPortType Interface

This OPC UA Interface defines capabilities of an Ethernet-based port. The *IEthernetPortType* is formally defined in Table 3.

Table 3 – IEthernetPortType definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | IEthernetPortType | | | | |
| IsAbstract | True | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **Modelling Rule** |
| Subtype of the *BaseInterfaceType* defined in OPC 10000-5 | | | | | |
| HasComponent | Variable | Speed | UInt64 | AnalogUnitType | Mandatory |
| HasComponent | Variable | Duplex | Duplex | BaseDataVariableType | Mandatory |
| HasComponent | Variable | MaxFrameLength | UInt16 | BaseDataVariableType | Mandatory |

*Speed* of *DataType* *UInt64* specifies the configured, negotiated, or actual speed of an Ethernet port in entities of 1 Mb/s (data rate). The default value is implementation-dependent (see IEEE Std 802.3).

*Duplex* of *DataType* *Duplex* represents the configured, negotiated, or actual duplex mode of an Ethernet port (see IEEE Std 802.3, 30.3.1.1.32, “aDuplexStatus”). The *Duplex* *DataType* is defined in 5.4.1.1.

*MaxFrameLength* of *DataType* *UInt16* indicates the MAC frame length (including FCS bytes) at which frames are dropped for being too long (see IEEE Std 802.3, 30.3.1.1.37 “aMaxFrameLength”).

The component *Variables* of the *IEthernetPortType* have the *Attribute* values defined in Table 4.

Table 4 – IEthernetPortType Attribute values for child Nodes

|  |  |  |
| --- | --- | --- |
| **Source Path** | **Value Attribute** | **Description Attribute** |
| |  | | --- | | Speed | | 0:EngineeringUnits | | NamespaceUri: http://www.opcfoundation.org/UA/units/un/cefact  UnitId: 4534832 DisplayName: Mbit/s Description: megabit per second | - |

### IAutoNegotiationFeatureType Interface

This OPC UA Interface defines the auto negotiation feature of an Ethernet-based port. The *IAutoNegotiationFeatureType* is formally defined in Table 5.

Table 5 – IAutoNegotiationFeatureType definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | IAutoNegotiationFeatureType | | | | |
| IsAbstract | True | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **Modelling Rule** |
| Subtype of the *BaseInterfaceType* defined in OPC 10000-5 | | | | | |
| HasComponent | Variable | NegotiationStatus | NegotiationStatus | BaseDataVariableType | Mandatory |

*NegotiationStatus* of *DataType* *NegotiationStatus* specifies the status of the auto-negotiation protocol (see IEEE 802.3, 30.6.1.1.4, “aAutoNegAutoConfig”). The *NegotiationStatus* *DataType* is defined in 5.4.1.4.

### IEthernetCapabilitiesType Interface

This OPC UA Interface defines capabilities of an Ethernet-based port. The *IEthernetCapabilitiesType* is formally defined in Table 6.

Table 6 – IEthernetCapabilitiesType definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | IEthernetCapabilitiesType | | | | |
| IsAbstract | True | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **Modelling Rule** |
| Subtype of the *BaseInterfaceType* defined in OPC 10000-5 | | | | | |
| HasComponent | Variable | VlanTagCapable | Boolean | BaseDataVariableType | Mandatory |

When *VlanTagCapable* is true, the network interface supports the ability to tag/untag frames using a Customer VLAN Tag (C-TAG of Clause 9) provided by the network. (See IEEE 802.1Qcc-2018 46.2.3.7.1.)

### ISrClassType Interface

This OPC UA Interface defines the content of an SrClass. The *ISrClassType* is formally defined in Table 7.

Table 7 – ISrClassType definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | ISrClassType | | | | |
| IsAbstract | True | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **Modelling Rule** |
| Subtype of the *BaseInterfaceType* defined in OPC 10000-5 | | | | | |
| HasComponent | Variable | Id | Byte | BaseDataVariableType | Mandatory |
| HasComponent | Variable | Priority | Byte | BaseDataVariableType | Mandatory |
| HasComponent | Variable | Vid | UInt16 | BaseDataVariableType | Mandatory |

*Id* is a Byte and specifies the SRclassID in a numeric representation of the SR classes which is supported by a particular Bridge Port (see IEEE 802.1Q-2018 35.2.2.9.2 SRclassID). Only Values between 0 and 7 shall be used.

*Priority* is a *Byte* and holds the Data Frame Priority (item a) in IEEE 802.1Q clause 35.2.2.8.5) value that will be used for streams that belong to the associated SR class. (see IEEE 802.1Q-2018 35.2.2.9.3 SRclassPriority). Only Values between 0 and 7 shall be used.

*Vid* is an *UInt16* and contains the SR\_PVID [item i) in IEEE 802.1Q clause 35.2.1.4) that the associated streams will be tagged with by the Talker (see IEEE 802.1Q-2018 35.2.2.9.4 SRclassVID).

### ITsnStreamType Interface

The *ITsnStreamType* **contains *Variables* which are common for both TSN talkers and TSN listeners. They represent the configuration properties and diagnostic values like reservation status and failure codes of a TSN stream.** The *ITsnStreamType* is formally defined Table 8.

Table 8 – ITsnStreamType definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | ITsnStreamType | | | | |
| IsAbstract | True | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Subtype of the *BaseInterfaceType* defined in OPC 10000-5 | | | | | |
| HasComponent | Variable | StreamId | Byte [8] | BaseDataVariableType | Mandatory |
| HasComponent | Variable | StreamName | String | BaseDataVariableType | Mandatory |
| HasComponent | Variable | State | TsnStreamState | BaseDataVariableType | Mandatory |
| HasComponent | Variable | AccumulatedLatency | UInt32 | BaseDataVariableType | Optional |
| HasComponent | Variable | SrClassId | Byte | BaseDataVariableType | Optional |

*StreamId* is an array of 8 *Bytes* defined according to the StreamID in IEEE 802.1Qcc-2018 clause 35.2.2.8.2. The *StreamId* shall be unique in the scope of the related TSN Network. The mapping between the *StreamId* *Byte* array and the IEEE octet string StreamID is as follows: Entry[n] of *StreamId* is mapped to octet[n] of StreamID. The *StreamId* shall be provided in the TSN stream *Objects* for diagnostic reasons.

Note: In the distributed configuration model the StreamId is typically generated by the TSN control stack of the endstation. In the fully centralized configuration model the StreamId is typically generated by the CUC.

*StreamName* is a *String* identifying the related stream in the network. The format of the *String* is application specific. The uniqueness of the *StreamName* inside the network segment shall be guaranteed by the application. If multiple applications use the network segment, they need to agree on a naming scheme.

*State* represents the current state of the TSN configuration process of a TSN stream. The *TsnStreamState* Enumeration is defined in 5.4.1.6.

*AccumulatedLatency* of *DataType* *UInt32* is the maximum worst case propagation delay in nanoseconds calculated and guaranteed by the TSN Control Layer for this Listener. Once the stream reservation has succeeded the *AccumulatedLatency* is not expected to increase during the lifecycle of the TSN Stream (see IEEE 802.1Q-2018 clause 35.2.2.8.6).

*SrClassId* of *DataType* *Byte* contains the Stream Reservation Class that is used for this stream (see IEEE 802.1Qcc-2018 clause 35.2.2.9.2).

### ITsnTrafficSpecificationType Interface

This OPC UA Interface is used to represent the traffic specification of a TSN stream. The *ITsnTrafficSpecificationType* is formally defined in Table 9.

Table 9 – ITsnTrafficSpecificationType definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | ITsnTrafficSpecificationType | | | | |
| IsAbstract | True | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Subtype of the *BaseInterfaceType* defined in OPC 10000-5 | | | | | |
| HasComponent | Variable | MaxIntervalFrames | UInt16 | BaseDataVariableType | Mandatory |
| HasComponent | Variable | MaxFrameSize | UInt32 | BaseDataVariableType | Mandatory |
| HasComponent | Variable | Interval | FractionDataType | BaseDataVariableType | Mandatory |

The *MaxFrameSize* of *DataType* UInt16 specifies the maximum size frame that will be sent by a Talker for this Stream. (see 802.1Q clause 35.2.2.8.4a).

Note: According to 802.1Q *MaxFrameSize* only counts the number of bytes of the Ethernet payload without the media specific framing bytes. (i.e. without 8-byte preamble, 14-byte IEEE 802.3 header, 4-byte IEEE 802.1Q priority/VID Tag, 4-byte CRC, 12-byte inter frame gap). Same rules apply for counting *MaxBytesPerInterval.*

The *MaxIntervalFrames* of *DataType* UInt16 Variable specifies the maximum number of frames that will be sent during an *Interval.* (see 802.1Q clause 35.2.2.8.4b “MaxIntervalFrames” or 802.1Qcc clause 35.2.2.10.6 “MaxFramesPerInterval”)

*Interval* of *DataType* *FractionDataType* defines the time period of the TSN Stream in nanoseconds. In that interval a specified number of frames (MaxIntervalFrames) with a maximum payload size per frame (MaxFrameSize) and a maximum total number of bytes (MaxBytesPerInterval) will be transmitted. The *Interval* therefore shall either represent the “class measurement interval” as used for AVB based Streams (see IEEE 802.1Q-2018 clause 35 or the “Interval” parameter used in the TrafficSpecification group in IEEE 802.1Qcc-2018 clause 46.2.3.5.1. The *FractionDataType* is defined in 5.4.2.1.

### ITsnStatusStreamType Interface

This OPC UA Interface is used to represent the status of a TSN stream. The *ITsnStatusStreamType* is formally defined in Table 10.

Table 10 – ITsnStatusStreamType definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | ITsnStatusStreamType | | | | |
| IsAbstract | True | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Subtype of the *BaseInterfaceType* defined in OPC 10000-5 | | | | | |
| HasComponent | Variable | FailureCode | TsnFailureCode | BaseDataVariableType | Mandatory |
| HasComponent | Variable | FailureSystemIdentifier | Byte [][8] | BaseDataVariableType | Mandatory |

*FailureCode* of *DataType* *TsnFailureCode* contains the Reservation Failure Code as defined in the “FailureInformation” in IEEE 802.1Q-2018 clause 35.2.2.8.7.

*FailureSystemIdentifier* is an Array of Arrays of 8 Bytes and contains the System Identifiers representing the network nodes where the failure occurred. See “System Identifier” in “FailureInformation” in IEEE 802.1Q-2018 clause 35.2.2.8.7.

### ITsnInterfaceConfigurationType Interface

This OPC UA Interface is used to represent an interface configuration which is part of a TSN stream (on the end-device). The *ITsnInterfaceConfigurationType* is formally defined in Table 11.

Table 11 – ITsnInterfaceConfigurationType definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | ITsnInterfaceConfigurationType | | | | |
| IsAbstract | True | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Subtype of the *BaseInterfaceType* defined in OPC 10000-5 | | | | | |
| HasComponent | Variable | MacAddress | String | BaseDataVariableType | Mandatory |
| HasComponent | Variable | InterfaceName | String | BaseDataVariableType | Optional |

*MacAddress* of *DataType* *String* contains the MAC Address of the Interface the configuration will be applied to, see 46.2.5.3 of IEEE 802.1Qcc-2018.

*InterfaceName* of *DataType* *String* is optional and supports the identification of the Interface to be configured, see 46.2.5.3 of IEEE 802.1Qcc-2018.

### ITsnInterfaceConfigurationTalkerType Interface

This OPC UA Interface is used to represent a talker (sender) interface configuration of a TSN stream. The *ITsnInterfaceConfigurationTalkerType* is formally defined in Table 12.

Table 12 – ITsnInterfaceConfigurationTalkerType definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | ITsnInterfaceConfigurationTalkerType | | | | |
| IsAbstract | True | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Subtype of the *ITsnInterfaceConfigurationType* defined in 5.2.9 | | | | | |
| HasComponent | Variable | TimeAwareOffset | UInt32 | BaseDataVariableType | Optional |

*TimeAwareOffset* of *DataType* *UInt32* specifies the time offset in nanoseconds relative to the start of the Interval that the Talker shall use for transmission (see IEEE 802.1Qcc-2018 clause 46.2.5.3.5).

### ITsnInterfaceConfigurationListenerType Interface

This OPC UA Interface is used to represent a listener (receiver) interface configuration of a TSN stream. The *ITsnInterfaceConfigurationListenerType* is formally defined in Table 13.

Table 13 – ITsnInterfaceConfigurationListenerType definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | ITsnInterfaceConfigurationListenerType | | | | |
| IsAbstract | True | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Subtype of the *ITsnInterfaceConfigurationType* defined in 5.2.9 | | | | | |
| HasComponent | Variable | ReceiveOffset | UInt32 | BaseDataVariableType | Optional |

*ReceiveOffset* of *DataType* *UInt32* specifies the offset in nanoseconds within the Interval at which the Listener will receive the first frame of the TSN Stream.

### ITsnMacAddressType Interface

This OPC UA Interface is used to represent a MAC address based stream identification of a TSN stream. The *ITsnMacAddressType* is formally defined in Table 14.

Table 14 – ITsnMacAddressType definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | ITsnMacAddressType | | | | |
| IsAbstract | True | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Subtype of the *BaseInterfaceType* defined in OPC 10000-5 | | | | | |
| HasComponent | Variable | DestinationAddress | Byte [6] | BaseDataVariableType | Mandatory |

*DestinationAddress* is defined according to the destination\_address in IEEE 802.1Q-2018 clause 35.2.2.8.3a) which represents the destination MAC address in the Ethernet header of the streamed data packets. Entry[n] of *DestinationAddress* is mapped to octet[n] of destination\_address.

### ITsnVlanTagType Interface

This OPC UA Interface is used to represent the VLAN configuration of a TSN stream. The *ITsnVlanTagType* is formally defined in Table 15.

Table 15 – ITsnVlanTagType definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | ITsnVlanTagType | | | | |
| IsAbstract | True | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Subtype of the *BaseInterfaceType* defined in OPC 10000-5 | | | | | |
| HasComponent | Variable | VlanId | UInt16 | BaseDataVariableType | Mandatory |
| HasComponent | Variable | PriorityCodePoint | Byte | BaseDataVariableType | Mandatory |

*VlanId* of *DataType* *UInt16* defines the 12-Bit VLAN-Identifier of the VLAN tag in the Ethernet header for the related stream. Only values between 0 and 4095 shall be used according to IEEE 802.1Q-2018 Table 9-2.

*PriorityCodePoint* of *DataType* Byte defines the 3 Bit priority code point inside the VLAN tag of the Ethernet header of the related stream. Only values between 0 and 7 are supported as defined by IEEE 802.1Q-2018 35.2.2.8.5a Data Frame Priority.

### IPriorityMappingEntryType Interface

This OPC UA Interface is used to translate a priority label like PriorityLabel (defined in Part 14) to a concrete network priority value (e.g. DSCP or PCP).

Note: This UA Interface is typically used to form a mapping rule table containing a set of object entities implementing the *IPriorityMappingEntryType*. Each network interface supporting priority mapping may reference such a mapping table *Object*.

Table 16 – IPriorityMappingEntryType definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | IPriorityMappingEntryType | | | | |
| IsAbstract | True | | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** | **ModellingRule** |
| Subtype of the *BaseInterfaceType* defined in OPC 10000-5 | | | | | |
| HasComponent | Variable | MappingUri | String | BaseDataVariableType | Mandatory |
| HasComponent | Variable | PriorityLabel | String | BaseDataVariableType | Mandatory |
| HasComponent | Variable | PriorityValue | UInt32 | BaseDataVariableType | Mandatory |

*MappingUri* of *DataType* *String* specifies a named identifier of a well-known predefined set of priority labels.

*PriorityLabel* of *DataType* *String* is a textual representation of the desired transport priority configured within the QoS settings of a communication relation, e.g. WriterGroup defined in Part14.

*PriorityValue* of *DataType* *UInt32* is the transport priority attached to a priority label. The devices shall translate the defined priority label for each packet according to the communication relation (e.g. WriterGroup) and linked interface. Depending on the transport protocol mapping the value defines e.g. a PCP value in the Ethernet frame or DSCP value in the UDP packet.

## ReferenceTypes

### CommLinkTo

The *CommLinkTo ReferenceType* is a concrete *ReferenceType* that can be used directly. It is a subtype of the *HierarchicalReferences* *ReferenceType*.

The References of this type shall be used between *Objects* to describe their communication dependencies and relations.

Note: E.g. a TsnStream (Talker or Listener) has a *CommLinkTo* *Reference* to a particular *IIetfNetworkInterfaceType* or *IEthernetPortType* because it needs access to the network. The direction of the reference shall represent the hierarchical dependency of the linked Objects and not necessarily the direction of the communication data flow.

*Servers* shall provide the inverse *Reference*.

The representation of the *CommLinkTo ReferenceType* in the *AddressSpace* is specified in Table 17.

Table 17 – CommLinkTo ReferenceType

|  |  |  |  |
| --- | --- | --- | --- |
| **Attributes** | **Value** | | |
| BrowseName | CommLinkTo | | |
| InverseName | CommLinkFrom | | |
| Symmetric | False | | |
| IsAbstract | False | | |
| **References** | **NodeClass** | **BrowseName** | **Comment** |
| Subtype of HierarchicalReferences defined in OPC 10000-5. | | | |

## DataTypes

### Enumeration DataTypes

#### Duplex enumeration

The *Duplex* is an enumeration representing the configured, negotiated, or actual duplex mode of an Ethernet interface (see IEEE Std 802.3, 30.3.1.1.32, aDuplexStatus). The values of the *Duplex* *Enumeration* are defined in Table 18.

Table 18 –Duplex Values

|  |  |  |
| --- | --- | --- |
| Name | Value | Description |
| full | 0 | Full duplex. |
| half | 1 | Half duplex. |
| unknown | 2 | Link is currently disconnected or initializing. |

Its representation in the *AddressSpace* is defined in Table 19.

Table 19 –Duplex Definition

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Attribute** | | **Value** | | | | |
| BrowseName | | Duplex | | | | |
| IsAbstract | | False | | | | |
| **References** | **NodeClass** | | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the Enumeration type defined in OPC 10000-5 | | | | | | |
| HasProperty | Variable | | EnumValues | EnumValueType[] | PropertyType |  |

#### InterfaceAdminStatus enumeration

The *InterfaceAdminStatus* is an enumeration for the possible desired states of the network interface (see RFC 2863: The Interfaces Group MIB - ifAdminStatus). The values of the *InterfaceAdminStatus* *Enumeration* are defined in Table 20.

Table 20 – InterfaceAdminStatus Values

|  |  |  |
| --- | --- | --- |
| Name | Value | Description |
| up | 0 | Ready to pass packets. |
| down | 1 | Not ready to pass packets and not in some test mode. |
| testing | 2 | In some test mode. |

Its representation in the *AddressSpace* is defined in Table 21.

Table 21 – InterfaceAdminStatus Definition

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Attribute** | | **Value** | | | | |
| BrowseName | | InterfaceAdminStatus | | | | |
| IsAbstract | | False | | | | |
| **References** | **NodeClass** | | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the Enumeration type defined in OPC 10000-5 | | | | | | |
| HasProperty | Variable | | EnumValues | EnumValueType[] | PropertyType |  |

#### InterfaceOperStatus enumeration

The *InterfaceOperStatus* is an enumeration for the possible operational states of the network interface (see RFC 2863: The Interfaces Group MIB - ifOperStatus). The values of the *InterfaceOperStatus* *Enumeration* are defined in Table 22.

Table 22 – InterfaceOperStatus Values

|  |  |  |
| --- | --- | --- |
| Name | Value | Description |
| up | 0 | Ready to pass packets. |
| down | 1 | The interface does not pass any packets. |
| testing | 2 | In some test mode. No operational packets can be passed. |
| unknown | 3 | Status cannot be determined for some reason. |
| dormant | 4 | Waiting for some external event. |
| not\_present | 5 | Some component (typically hardware) is missing. |
| lower\_layer\_down | 6 | Down due to state of lower-layer interface(s). |

Its representation in the *AddressSpace* is defined in Table 23.

Table 23 – InterfaceOperStatus Definition

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Attribute** | | **Value** | | | | |
| BrowseName | | InterfaceOperStatus | | | | |
| IsAbstract | | False | | | | |
| **References** | **NodeClass** | | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the Enumeration type defined in OPC 10000-5 | | | | | | |
| HasProperty | Variable | | EnumValues | EnumValueType[] | PropertyType |  |

#### NegotiationStatus enumeration

The *NegotiationStatus* is an enumeration representing the status of the auto-negotiation protocol (see IEEE 802.3, 30.6.1.1.4, aAutoNegAutoConfig). The values of the *NegotiationStatus Enumeration* are defined in Table 24.

Table 24 – NegotiationStatus Values

|  |  |  |
| --- | --- | --- |
| Name | Value | Description |
| in\_progress | 0 | The auto-negotiation protocol is running and negotiation is currently in-progress. |
| complete | 1 | The auto-negotiation protocol has completed successfully. |
| failed | 2 | The auto-negotiation protocol has failed. |
| unknown | 3 | The auto-negotiation status is not currently known, this could be because it is still negotiating or the protocol cannot run (e.g., if no medium is present). |
| no\_negotiation | 4 | No auto-negotiation is executed. The auto-negotiation function is either not supported on this interface or has not been enabled. |

Its representation in the *AddressSpace* is defined in Table 25.

Table 25 – NegotiationStatus Definition

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Attribute** | | **Value** | | | | |
| BrowseName | | NegotiationStatus | | | | |
| IsAbstract | | False | | | | |
| **References** | **NodeClass** | | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the Enumeration type defined in OPC 10000-5 | | | | | | |
| HasProperty | Variable | | EnumValues | EnumValueType[] | PropertyType |  |

#### TsnFailureCode Enumeration

The *TsnFailureCode* is an *Enumeration* to provide detailed error information for failures occurring during TSN stream establishment (see IEEE 802.1Qcc-2018 Table 46-15, TSN Failure Codes). The *TsnFailureCode Enumeration* is defined in Table 26.

Table 26 – TsnFailureCode values

|  |  |  |
| --- | --- | --- |
| Name | Value | Description |
| noFailure | 0 | No failure |
| insufficientBandwidth | 1 | Insufficient bandwidth |
| insufficientResources | 2 | Insufficient bridge resources |
| insufficientTrafficClassBandwidth | 3 | Insufficient bandwidth for Traffic Class |
| streamIDInUse | 4 | StreamID in use by another Talker |
| streamDestinationAddressInUse | 5 | Stream destination address already in use |
| streamPreemptedByHigherRank | 6 | Stream pre-empted by higher rank |
| latencyHasChanged | 7 | Reported latency has changed |
| egressPortNotAVBCapable | 8 | Egress port is not AVBCapable |
| useDifferentDestinationAddress | 9 | Use a different destination address |
| outOfMSRPResources | 10 | Out of MSRP resources |
| outOfMMRPResources | 11 | Out of MMRP resources |
| cannotStoreDestinationAddress | 12 | Cannot store destination address |
| priorityIsNoAnSRCLass | 13 | Requested priority is not an SR Class priority |
| maxFrameSizeTooLarge | 14 | MaxFrameSize is too large for media |
| maxFanInPortsLimitReached | 15 | maxFanInPorts limit has been reached |
| firstValueChangedForStreamID | 16 | Changes in FirstValue for a registered StreamID |
| vlanBlockedOnEgress | 17 | VLAN is blocked on this egress port (Registration Forbidden) |
| vlanTaggingDisabledOnEgress | 18 | VLAN tagging is disabled on this egress port (untagged set) |
| srClassPriorityMismatch | 19 | SR class priority mismatch |
| featureNotPropagated | 20 | Enhanced feature cannot be propagated to original Port |
| maxLatencyExceeded | 21 | MaxLatency exceeded |
| bridgeDoesNotProvideNetworkID | 22 | Nearest Bridge cannot provide network identification for stream transformation |
| streamTransformNotSupported | 23 | Stream transformation not supported |
| streamIDTypeNotSupported | 24 | Stream identification type not supported for stream transformation |
| featureNotSupported | 25 | Enhanced feature cannot be supported without a CNC |

Its representation in the *AddressSpace* is defined in Table 27.

Table 27 – TsnFailureCode Definition

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Attribute** | | **Value** | | | | |
| BrowseName | | TsnFailureCode | | | | |
| IsAbstract | | False | | | | |
| **References** | **NodeClass** | | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the Enumeration type defined in OPC 10000-5 | | | | | | |
| HasProperty | Variable | | EnumValues | EnumValueType[] | PropertyType |  |

#### TsnStreamState Enumeration

The *TsnStreamState* is an enumeration representing the state of the configuration process of a TSN Talker or Listener.

The default value is *Disabled*. The *TsnStreamState* *Enumeration* is defined in Table 28.

Table 28 – TsnStreamState Values

|  |  |  |
| --- | --- | --- |
| Name | Value | Description |
| Disabled | 0 | The related TSN Streamis currently disabled. |
| Configuring | 1 | The related TSN Streamis in the process of receiving configuration parameters from the TSN Control Layer. |
| Ready | 2 | The related TSN Streamhas successfully received and applied the configuration from the TSN Control Layer. The related TSN Streamis not fully operational as long as local preconditions (e.g. synchronization state) are not valid. |
| Operational | 3 | The related TSN Streamobject is configured and all other required preconditions (e.g. synchronization state) for sending / receiving data are valid. |
| Error | 4 | The related TSN Streamobject is in an error state. |

Its representation in the *AddressSpace* is defined in Table 29.

Table 29 – TsnStreamState Definition

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Attribute** | | **Value** | | | | |
| BrowseName | | TsnStreamState | | | | |
| IsAbstract | | False | | | | |
| **References** | **NodeClass** | | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the Enumeration type defined in OPC 10000-5 | | | | | | |
| HasProperty | Variable | | EnumValues | EnumValueType[] | PropertyType |  |

### Structure DataTypes

#### FractionDataType

This *Structure DataType* is used to represent a fraction. The *FractionDataType* is formally defined in Table 30.

Table 30 – FractionDataType structure

| **Name** | **Type** | **Description** |
| --- | --- | --- |
| FractionDataType | Structure |  |
| Numerator | UInt32 | The top number of the fraction. |
| Denominator | UInt32 | The bottom number of the fraction. |

Its representation in the *AddressSpace* is defined in Table 31.

Table 31 – FractionDataType Definition

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Attribute** | | **Value** | | | | |
| BrowseName | | FractionDataType | | | | |
| IsAbstract | | False | | | | |
| **References** | **NodeClass** | | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the Structure DataType defined in OPC 10000-5 | | | | | | |

## Instance Entry Points

Overview and location of the instance entry points are shown in Figure 3.



Figure 3 –Instance Entry Points for Network Interfaces and Streams

### Resources Folder

The *Resources* *Object* shall be used as the browse entry point for physical and logical resources of the device the *Server* is running on. It shall reside in the *Server* *Object* defined in OPC 10000-5. It may contain a set of *Organizes* References that point to other *Objects* representing specific resources. It is formally defined in Table 32.

Table 32 – Resources definition

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Value** | | |
| BrowseName | Resources | | |
| **References** | **NodeClass** | **BrowseName** | **Comment** |
| ComponentOf of the Server Object defined in Part 5. | | | |
| HasTypeDefinition | ObjectType | FolderType |  |
| Organizes | Object | Communication | Defined in 5.5.2 |

### Communication Folder

The *Communication* *Object* shall be used as the browse entry point for communication related resources of the physical device the *Server* is running on. It is formally defined in Table 33.

The *Communication* *Object* is referenced by an *Organizes* *Reference* from the *Recourses* *Object* defined in 5.5.1.

The *Communication* *Object* may include the following subfolders:

* MappingTables
* NetworkInterfaces
* Streams

Note: It is recommended to keep TSN-Streams and possible future (DetNet-)Flows separated in specific folders.

Table 33 – Communication definition

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Value** | | |
| BrowseName | Communication | | |
| **References** | **NodeClass** | **BrowseName** | **Comment** |
| HasTypeDefinition | ObjectType | FolderType |  |
| Organizes | Object | MappingTables | Defined in 5.5.3 |
| Organizes | Object | NetworkInterfaces | Defined in 5.5.4 |
| Organizes | Object | Streams | Defined in 5.5.5 |

### MappingTables Folder

The *MappingTables* *Object* shall be used as the browse entry point for mapping tables of priority values and their application labels. It is formally defined in Table 35.

Table 34 – MappingTables definition

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Value** | | |
| BrowseName | MappingTables | | |
| **References** | **NodeClass** | **BrowseName** | **Comment** |
| HasTypeDefinition | ObjectType | FolderType |  |

### NetworkInterfaces Folder

The *NetworkInterfaces* *Object* shall be used as the browse entry point for network interfaces of the device the *Server* is running on. It is formally defined in Table 35.

Table 35 – NetworkInterfaces definition

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Value** | | |
| BrowseName | NetworkInterfaces | | |
| **References** | **NodeClass** | **BrowseName** | **Comment** |
| HasTypeDefinition | ObjectType | FolderType |  |

### Streams Folder

The *Streams* *Object* shall be used as the browse entry point for network streams of the device the *Server* is running on. It is formally defined in Table 36.

Table 36 – Streams definition

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Value** | | |
| BrowseName | Streams | | |
| **References** | **NodeClass** | **BrowseName** | **Comment** |
| HasTypeDefinition | ObjectType | FolderType |  |
| Organizes | Object | TalkerStreams | Defined in 5.5.6 |
| Organizes | Object | ListenerStreams | Defined in 5.5.7 |

### TalkerStreams Folder

The *TalkerStreams* *Object* shall be used as the browse entry point for sending network streams of the device the *Server* is running on. It is formally defined in Table 37.

Table 37 – TalkerStreams definition

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Value** | | |
| BrowseName | TalkerStreams | | |
| **References** | **NodeClass** | **BrowseName** | **Comment** |
| HasTypeDefinition | ObjectType | FolderType |  |

### ListenerStreams Folder

The *ListenerStreams* *Object* shall be used as the browse entry point for receiving network streams of the device the *Server* is running on. It is formally defined in Table 38.

Table 38 – ListenerStreams definition

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Value** | | |
| BrowseName | ListenerStreams | | |
| **References** | **NodeClass** | **BrowseName** | **Comment** |
| HasTypeDefinition | ObjectType | FolderType |  |

1. Modelling Examples (informative)
   1. Modelling Examples for Network Interfaces
      1. Virtual Network Interfaces

A virtual interfaces configuration may be represented by *Objects* representing the virtual and the physical network interface instances. CommLinkTo *References* point from the *Object(s)* representing the virtual interface(s) to the *Object* representing the physical interface. All *Objects* implement the *IIetfNetworkInterfaceType*. An example is shown in Figure A-.

Note: The usage of the CommLinkTo *Reference* is following the YANG modelling approach which is using reference pointers in the YANG-interface nodes to point to other interface nodes in lower or higher layers (“higher-layer-if”, “lower-layer-if”).



Figure A-1 Modelling Example for virtual network interfaces

* + 1. Link Aggregation

A link aggregation configuration may be represented by *Objects* representing the logical aggregation and the particular physical interface instances. CommLinkTo *References* shall point from the *Object(s)* representing the aggregation interface to the *Objects* representing the physical interfaces. All *Objects* implement the *IIetfNetworkInterfaceType*. An example is shown in Figure A-2.



Figure A-2 Modelling example for link aggregation

* 1. Modelling Examples for PriorityMappingEntry Interfaces



Figure A-3 - Modelling Example for Priority Mappings with IPriorityMappingEntryType as described in 5.2.14

* 1. Usage of BNM in other UA Specifications
     1. Usage of BNM for PubSub over TSN



Figure A-4 – Possible Integration of BNM into PubSub

* + 1. Usage of BNM in PROFINET Companion Spec

Editor Node: This section still needs to be updated and is outdated.



Figure A-5 – Recommended Integration of BNM into Companion Spec exemplified by PROFINET

In Figure A-5, the orange marked Nodes of the PROFINET device (upper grey box of the figure) might implement Interfaces defined in this specification. In addition, they might directly reference network equipment as shown in the lower grey box of the figure.



Figure A-6 – Example of recommended Integration of BNM into PROFINET Companion Spec

\_\_\_\_\_\_\_\_\_\_\_