OPC 10000-22

OPC Unified Architecture

Part 22: Base Network Model

Release 1.04

2021-10-01

OPC UA Specification

Specification Type:	Industry Standard Specification	Comments:	
Document Number	**OPC 10000-22**		
Title:	OPC Unified Architecture Part 22 :Base Network Model	Date:	2021-10-01
Version:	Release 1.04	Software:	MS-Word
		Source:	OPC 10000-22 - UA Specification Part 22 - Base Network Model 1.04.docx
Author:	OPC Foundation	Status:	Release

CONTENTS

Figures

Tables

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.

1 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.

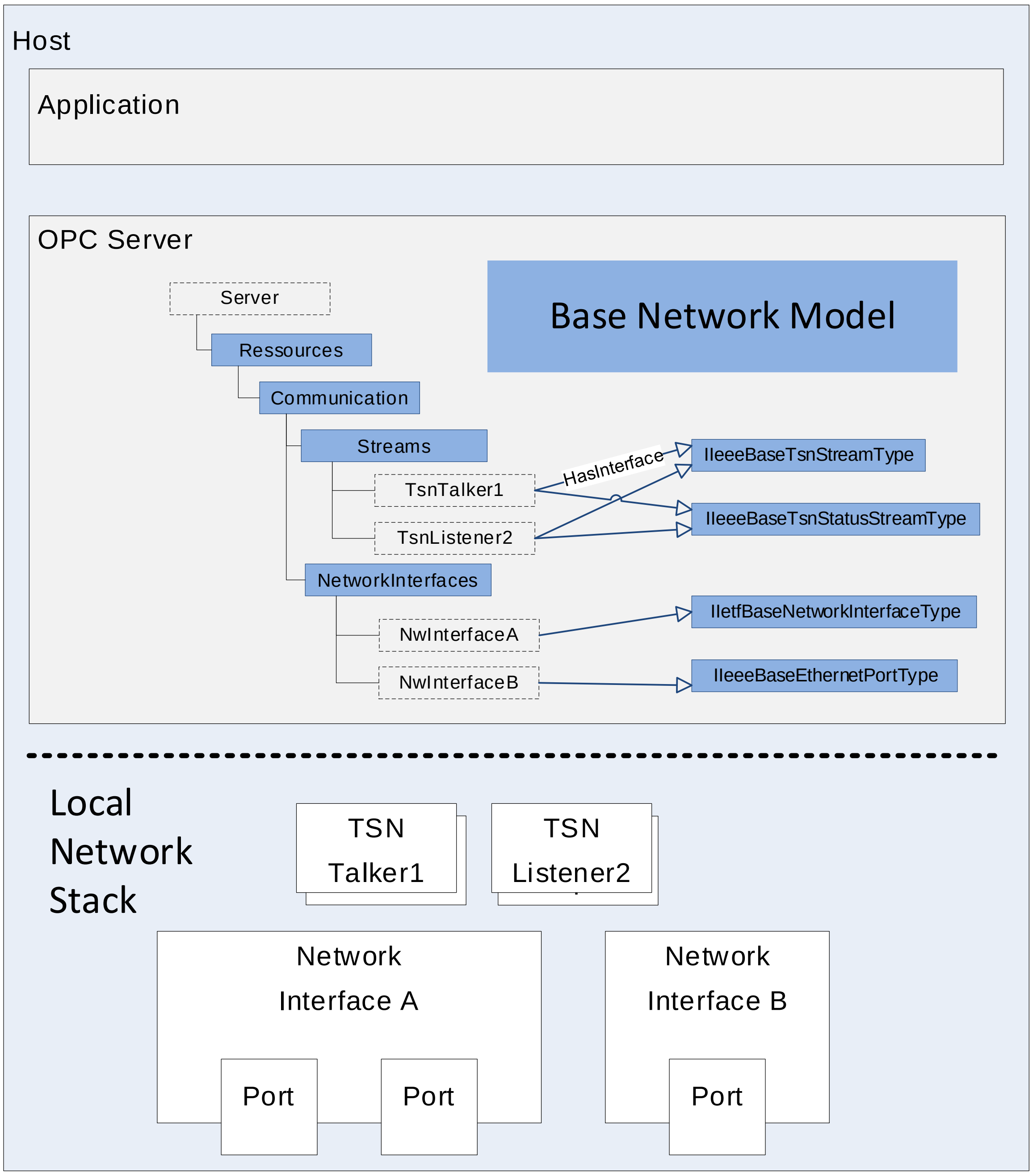

Figure 1 – Scope of Base Network Model

2 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/

IETF RFC 2863, *The Interfaces Group MIB*

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

3 Terms, definitions, abbreviated terms, and conventions

3.1 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.

3.2 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)

4 Concepts

4.1 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.

4.2 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*.

5 Model

5.1 Overview

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

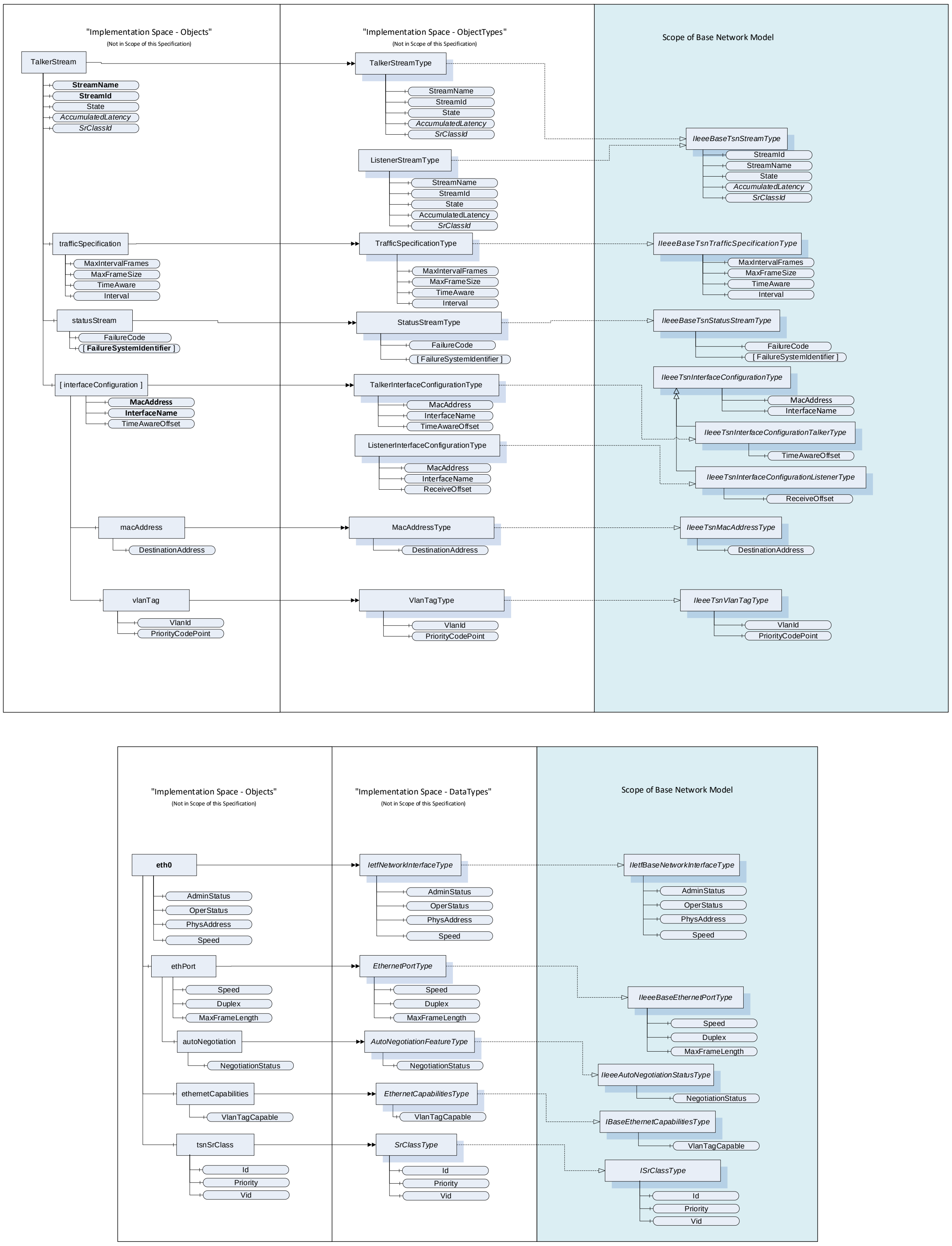

Figure 2 – Overview of OPC UA Interfaces

5.2 OPC UA Interfaces

5.2.1 IIetfBaseNetworkInterfaceType Interface

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

Table 1 – IIetfBaseNetworkInterfaceType definition

Attribute	**Value**				
BrowseName	IIetfBaseNetworkInterfaceType				
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 IETF RFC 2863: The Interfaces Group MIB - ifAdminStatus). The *InterfaceAdminStatus Enumeration* is defined in 5.3.1.2.

OperStatus of *DataType InterfaceOperStatus* specifies the current operational state of the network interface. This *Variable* has the same semantics as ifOperStatus (see IETF RFC 2863: The Interfaces Group MIB - ifOperStatus). The *InterfaceOperStatus Enumeration* is defined in 5.3.1.3.

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 IETF 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 IETF RFC 2863: ifSpeed, ifHighSpeed).

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

Table 2 – IIetfBaseNetworkInterfaceType 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	-

5.2.2 IIeeeBaseEthernetPortType Interface

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

Table 3 – IIeeeBaseEthernetPortType definition

Attribute	**Value**				
BrowseName	IIeeeBaseEthernetPortType				
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, clause 30.3.1.1.32, “aDuplexStatus”). The *Duplex DataType* is defined in 5.3.1.1.

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, clause 30.3.1.1.37, “aMaxFrameLength”).

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

Table 4 – IIeeeBaseEthernetPortType 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	-

5.2.3 IIeeeAutoNegotiationStatusType Interface

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

Table 5 – IIeeeAutoNegotiationStatusType definition

Attribute	**Value**				
BrowseName	IIeeeAutoNegotiationStatusType				
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 Std 802.3, clause 30.6.1.1.4, “aAutoNegAutoConfig”). The *NegotiationStatus DataType* is defined in 5.3.1.4.

5.2.4 IBaseEthernetCapabilitiesType Interface

This OPC UA Interface defines if an Ethernet-based port is VLAN Tag capable. The *IBaseEthernetCapabilitiesType* is formally defined in Table 6.

Table 6 – IBaseEthernetCapabilitiesType definition

Attribute	Value				
BrowseName	IBaseEthernetCapabilitiesType				
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 clause 46.2.3.7.1.).

5.2.5 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, clause 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-2018 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, clause 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-2018 clause 35.2.1.4) that the associated streams will be tagged with by the Talker (see IEEE 802.1Q-2018, clause 35.2.2.9.4, SRclassVID).

5.2.6 IIeeeBaseTsnStreamType Interface

The *IIeeeBaseTsnStreamType* 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 *IIeeeBaseTsnStreamType* is formally defined Table 8.

Table 8 – IIeeeBaseTsnStreamType definition

Attribute	Value				
BrowseName	IIeeeBaseTsnStreamType				
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 5.3.1.6.

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).

5.2.7 IIeeeBaseTsnTrafficSpecificationType Interface

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

Table 9 – IIeeeBaseTsnTrafficSpecificationType definition

Attribute	**Value**				
BrowseName	IIeeeBaseTsnTrafficSpecificationType				
IsAbstract	True				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
Subtype of the *BaseInterfaceType* defined in OPC 10000-5					
HasComponent	Variable	MaxIntervalFrames	UInt16	BaseDataVariableType	Mandatory
HasComponent	Variable	MaxFrameSize	UInt32	BaseDataVariableType	Mandatory
HasComponent	Variable	Interval	UnsignedRationalNumber	BaseDataVariableType	Mandatory

The *MaxFrameSize* of *DataType* UInt16 specifies the maximum size frame that will be sent by a Talker for this Stream (see IEEE 802.1Q-2018 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 IEEE 802.1Q-2018, clause 35.2.2.8.4b, “MaxIntervalFrames” or IEEE 802.1Qcc-2018, clause 35.2.2.10.6, “MaxFramesPerInterval”)

Interval of *DataType UnsignedRationalNumber* 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 *UnsignedRationalNumber* is defined in 5.3.2.1.

5.2.8 IIeeeBaseTsnStatusStreamType Interface

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

Table 10 – IIeeeBaseTsnStatusStreamType definition

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

TalkerStatus of *DataType TsnTalkerStatus* contains the Reservation Failure Code as defined in the “FailureInformation” in IEEE 802.1Qcc-2018 clause 46.2.5.1.1.

ListenerStatus of *DataType TsnListenerStatus* contains the Reservation Failure Code as defined in the “FailureInformation” in IEEE 802.1Qcc-2018 clause 46.2.5.1.2.

FailureCode of *DataType TsnFailureCode* contains the Reservation Failure Code as defined in the “FailureInformation” in IEEE 802.1Qcc-2018 clause 46.2.5.1.3.

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.

5.2.9 IIeeeTsnInterfaceConfigurationType Interface

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

Table 11 – IIeeeTsnInterfaceConfigurationType definition

Attribute	**Value**				
BrowseName	IIeeeTsnInterfaceConfigurationType				
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 IEEE 802.1Qcc-2018, clause 46.2.5.3.

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

5.2.10 IIeeeTsnInterfaceConfigurationTalkerType Interface

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

Table 12 – IIeeeTsnInterfaceConfigurationTalkerType definition

Attribute	**Value**				
BrowseName	IIeeeTsnInterfaceConfigurationTalkerType				
IsAbstract	True				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *IIeeeTsnInterfaceConfigurationType* 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).

5.2.11 IIeeeTsnInterfaceConfigurationListenerType Interface

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

Table 13 – IIeeeTsnInterfaceConfigurationListenerType definition

Attribute	**Value**				
BrowseName	IIeeeTsnInterfaceConfigurationListenerType				
IsAbstract	True				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *IIeeeTsnInterfaceConfigurationType* 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.

5.2.12 IIeeeTsnMacAddressType Interface

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

Table 14 – IIeeeTsnMacAddressType definition

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

DestinationAddress is defined according to the destination_mac_address in IEEE 802.1Qcc-2018 clause 46.2.3.4.1, 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_mac_address.

SourceAddress is defined according to the source_mac_address in IEEE 802.1Qcc-2018 clause 46.2.3.4.1, which represents the source MAC address in the Ethernet header of the streamed data packets. Entry[n] of *SourceAddress* is mapped to octet[n] of source_mac_address.

5.2.13 IIeeeTsnVlanTagType Interface

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

Table 15 – IIeeeTsnVlanTagType definition

Attribute	**Value**				
BrowseName	IIeeeTsnVlanTagType				
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, clause 35.2.2.8.5a, Data Frame Priority.

5.2.14 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_PCP	Byte	BaseDataVariableType	Optional
HasComponent	Variable	PriorityValue_DSCP	UInt32	BaseDataVariableType	Optional

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_PCP of *DataType Byte* is the VLAN Tag based 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 and if the value is set, this specific PCP shall overwrite the PCP field inside a packet.

PriorityValue_DSCP of *DataType UInt32* is the IP 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 and if this value is set, this specific DSCP shall overwrite the DSCP field inside a packet.

5.3 DataTypes

5.3.1 Enumeration DataTypes

5.3.1.1 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 17.

Table 17 – 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 18.

Table 18 – 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	

5.3.1.2 InterfaceAdminStatus enumeration

The *InterfaceAdminStatus* is an enumeration for the possible desired states of the network interface (see IETF RFC 2863: The Interfaces Group MIB - ifAdminStatus). The values of the *InterfaceAdminStatus Enumeration* are defined in Table 19.

Table 19 – 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 20.

Table 20 – 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	

5.3.1.3 InterfaceOperStatus enumeration

The *InterfaceOperStatus* is an enumeration for the possible operational states of the network interface (see IETF RFC 2863: The Interfaces Group MIB - ifOperStatus). The values of the *InterfaceOperStatus Enumeration* are defined in Table 21.

Table 21 – 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.
NotPresent	5	Some component (typically hardware) is missing.
LowerLayerDown	6	Down due to state of lower-layer interface(s).

Its representation in the *AddressSpace* is defined in Table 22.

Table 22 – 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	

5.3.1.4 NegotiationStatus enumeration

The *NegotiationStatus* is an enumeration representing the status of the auto-negotiation protocol (see IEEE Std 802.3, clause 30.6.1.1.4, aAutoNegAutoConfig). The values of the *NegotiationStatus Enumeration* are defined in Table 23.

Table 23 – NegotiationStatus Values

Name	Value	Description
InProgress	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).
NoNegotiation	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 24.

Table 24 – 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	

5.3.1.5 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 25.

Table 25 – 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
PriorityIsNotAnSrcClass	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 26.

Table 26 – 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	

5.3.1.6 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 27.

Table 27 – TsnStreamState Values

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

Its representation in the *AddressSpace* is defined in Table 28.

Table 28 – 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	

5.3.1.7 TsnTalkerStatus Enumeration

The *TsnTalkerStatus* is an enumeration representing the state of the TSN Talker configuration.

The default value is *None*. The *TsnTalkerStatus Enumeration* is defined in Table 27.

Table 29 – TsnTalkerStatus Values

Name	Value	Description
None	0	No Talker detected.
Ready	1	Talker ready (configured).
Failed	2	Talker failed.

Its representation in the *AddressSpace* is defined in Table 28.

Table 30 – TsnTalkerStatus Definition

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

5.3.1.8 TsnListenerStatus Enumeration

The *TsnListenerStatus* is an enumeration representing the state of the TSN Listener configuration.

The default value is *None*. The *TsnListenerStatus Enumeration* is defined in Table 27.

Table 31 – TsnListenerStatus Values

Name	Value	Description
None	0	No Listener detected.
Ready	1	Listener ready (configured).
PartialFailed	2	One or more Listeners ready, and one or more Listeners failed.
Failed	3	Listener failed.

Its representation in the *AddressSpace* is defined in Table 28.

Table 32 – TsnListenerStatus Definition

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

5.3.2 Structure DataTypes

5.3.2.1 UnsignedRationalNumber

This *Structure DataType* is used to represent an unsigned rational number as a fraction of two integral numbers. The *UnsignedRationalNumber* is formally defined in Table 33.

Table 33 – UnsignedRationalNumber structure

Name	Type	Description
UnsignedRationalNumber	Structure	
Numerator	UInt32	Numerator of the rational number.
Denominator	UInt32	Denominator of the rational number.

Its representation in the *AddressSpace* is defined in Table 34.

Table 34 – UnsignedRationalNumber Definition

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

5.4 Instance Entry Points

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

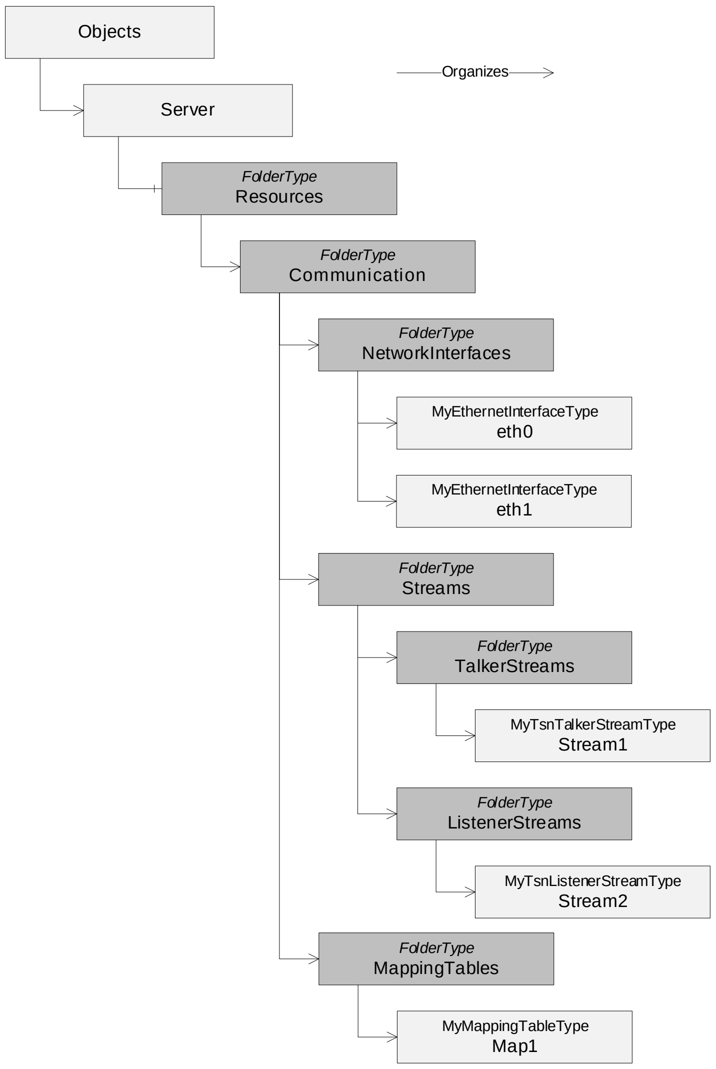

Figure 3 – Instance Entry Points for Network Interfaces and Streams

5.4.1 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 35.

Table 35 – 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 5.4.2

5.4.2 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 36.

The *Communication Object* is referenced by an *Organizes Reference* from the *Resources Object* defined in 5.4.1.

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 36 – Communication definition

Attribute	Value		
BrowseName	Communication		
References	**NodeClass**	**BrowseName**	**Comment**
HasTypeDefinition	ObjectType	FolderType	
Organizes	Object	MappingTables	Defined in 5.4.3
Organizes	Object	NetworkInterfaces	Defined in 5.4.4
Organizes	Object	Streams	Defined in 5.4.5

5.4.3 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 38.

Table 37 – MappingTables definition

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

5.4.4 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 38.

Table 38 – NetworkInterfaces definition

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

5.4.5 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 39.

Table 39 – Streams definition

Attribute	Value		
BrowseName	Streams		
References	**NodeClass**	**BrowseName**	**Comment**
HasTypeDefinition	ObjectType	FolderType	
Organizes	Object	TalkerStreams	Defined in 5.4.6
Organizes	Object	ListenerStreams	Defined in 5.4.7

5.4.6 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 40.

Table 40 – TalkerStreams definition

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

5.4.7 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 41.

Table 41 – ListenerStreams definition

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

Annex A Modelling Examples (informative)

A.1 Modelling Examples for Network Interfaces

A.1.1 Virtual Network Interfaces

A virtual interfaces configuration may be represented by *Objects* representing the virtual and the physical network interface instances. *References* point from the *Object(s)* representing the virtual interface(s) to the *Object* representing the physical interface. All *Objects* implement the *IIetfBaseNetworkInterfaceType*. An example is shown in Figure A-1.

Note: The usage of the *References* 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”).

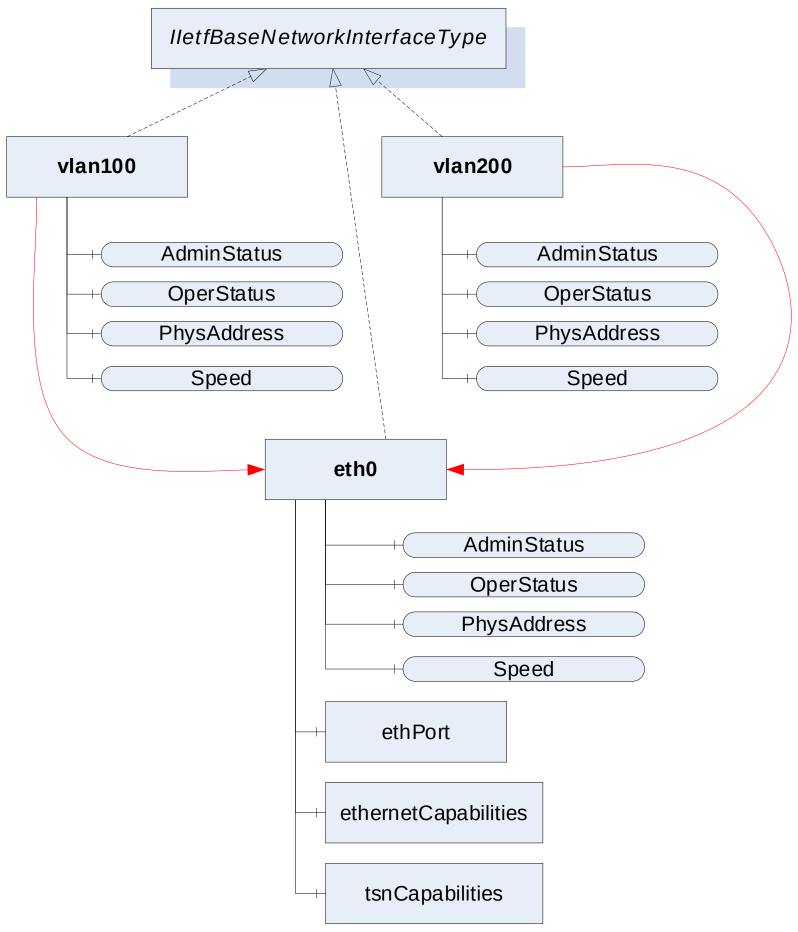

Figure A-1 – Modelling Example for virtual network interfaces

A.1.2 Link Aggregation

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

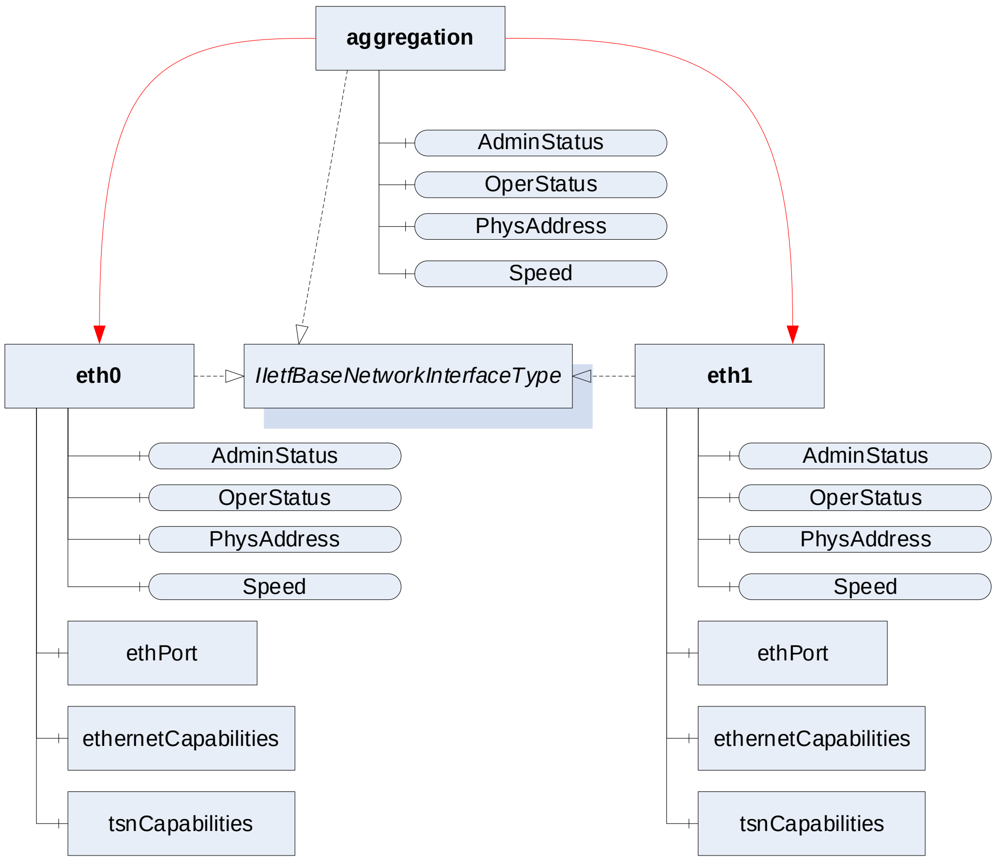

Figure A-2 – Modelling example for link aggregation

A.2 Modelling Examples for PriorityMappingEntry Interfaces

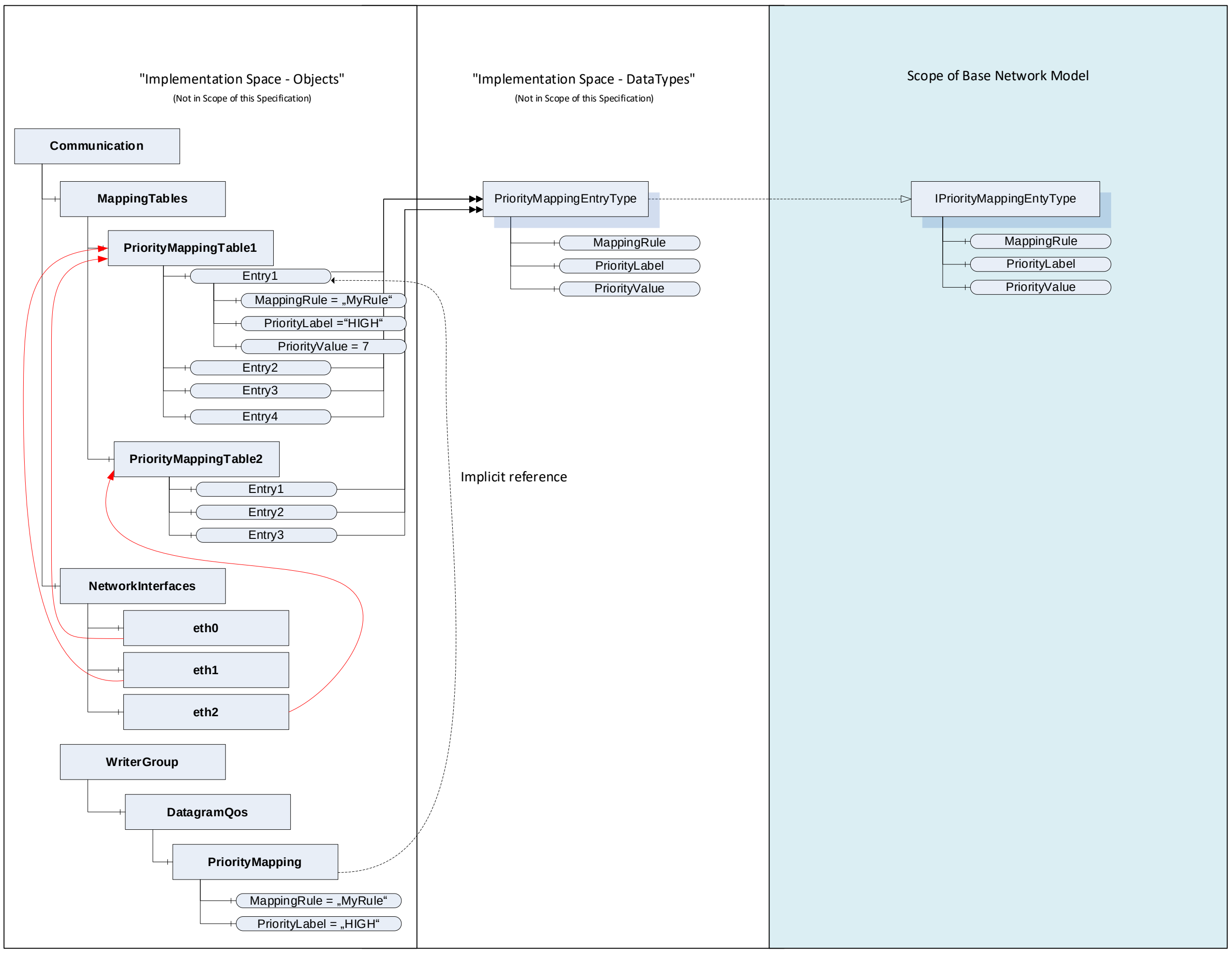

Figure A-3 – Modelling Example for Priority Mappings with IPriorityMappingEntryType as described in 5.2.14

A.3 Usage of BNM in other UA Specifications

A.3.1 Usage of BNM for PubSub over TSN

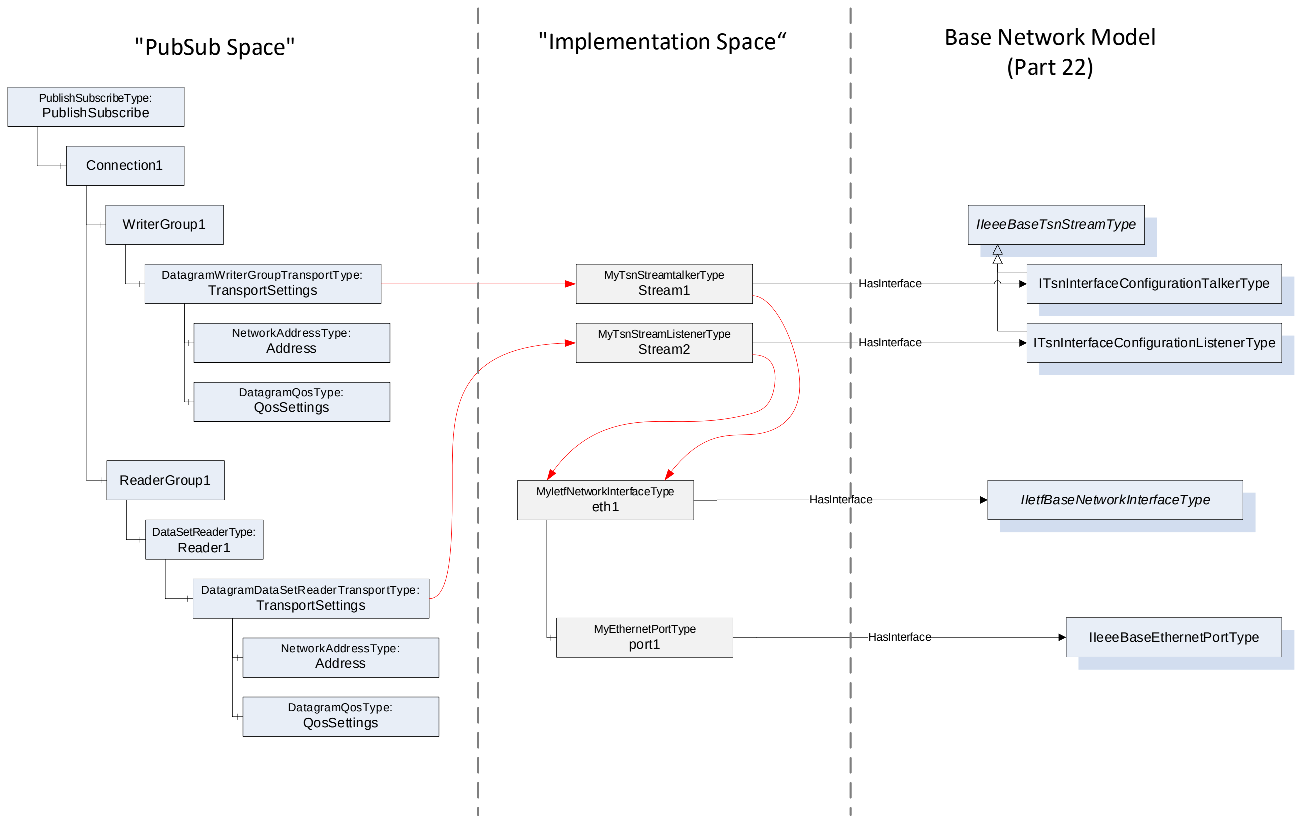
"PubSub Space"
"Implementation Space"
Base Network Model (Part 22)
PublishSubscribeType: PublishSubscribe
Connection1
WriterGroup1
DatagramWriterGroupTransportType: TransportSettings
NetworkAddressType: Address
DatagramQosType: QosSettings
ReaderGroup1
DataSetReaderType: Reader1
DatagramDataSetReaderTransportType: TransportSettings
NetworkAddressType: Address
DatagramQosType: QosSettings
MyTsnStreamtalkerType Stream1
MyTsnStreamListenerType Stream2
MyIetfNetworkInterfaceType eth1
MyEthernetPortType port1
HasInterface
IIeeeBaseTsnStreamType
ITsnInterfaceConfigurationTalkerType
ITsnInterfaceConfigurationListenerType
IIetfBaseNetworkInterfaceType
IIeeeBaseEthernetPortType

Figure A-4 – Possible Integration of BNM into PubSub

A.3.2 Usage of BNM in PROFINET Companion Spec

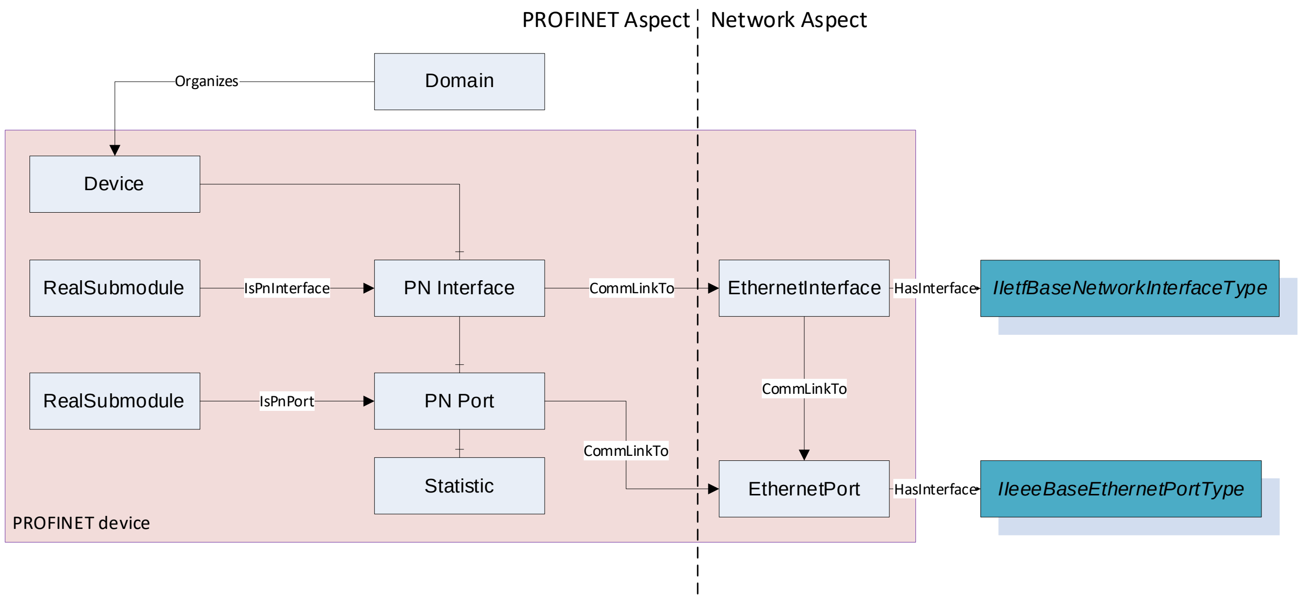

Figure A-5 – Recommended Integration of BNM into Companion Spec exemplified by PROFINET

____