Open ROADM MSA Device White paper

for release 2.2

V1.1 08/17/18

Please download the latest version on http://OpenROADM.org

1 TABLE OF CONTENTS

2 HIGH LEVEL DEVICE DESCRIPTION

2.1 PHYSICAL DESIGN

The physical design for Open ROADM devices is not specified in the MSA. The form factor (width, depth, shelf), power supply (AC/DC) or standards met (such as NEBS3) are up to the manufacturer or Network operator. The MSA only covers the functional aspects, as well as optical interoperability of the single- and multi-wave interface.

Since no function of a craft-interface terminal has been specified, a manual factory-reset mechanism (such as a physical button) is desired to get the network element back to the factory reset state (same state as before power was switched on the first time). Obviously, such reset mechanism needs to be hidden and clearly labeled that no accidental reset is triggered.

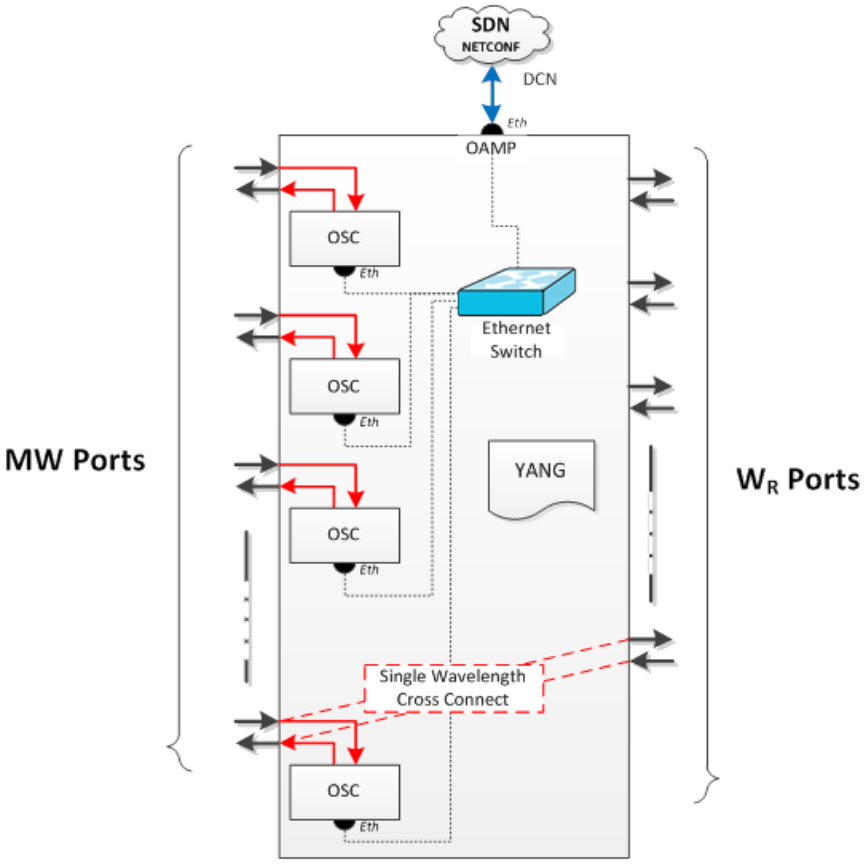
2.2 FUNCTIONAL DESIGN

2.2.1 ROADM

The Open ROADM MSA defines a ROADM device capable of providing colorless and directionless add/drop functionality. This means that a ROADM site can add/drop any wavelength at any port and connect that wavelength to any direction in a ROADM node[1] from the local transponder. The MSA does not define implementation details such as the number of degrees, form-factor, etc. The MSA defines:

- API using NETCONF interface with a YANG-based data model that abstracts the management, control and provisioning of multi-vendor ROADM devices
- Multi-wave (MW) interface which defines the optical specifications for the multi-wave DWDM interface between line degrees of the ROADM devices
    - The MW interface includes a GE OSC that provides LLDP-based topology information, support laser safety and provide MCN/DCN reach-through to remote ROADM nodes. Otherwise, there is no ROADM node to ROADM node communication and optical equalization is handled off-box by the controller.
- Single-wave (Wr) interface which defines the optical specifications for the add/drop ports of the ROADM devices

[1] The term “device”, “node” and “NE” are used interchangeably within this document.



Please refer to the Open ROADM MSA specification found at openroadm.org for the C/D ROADM specifications:[2]

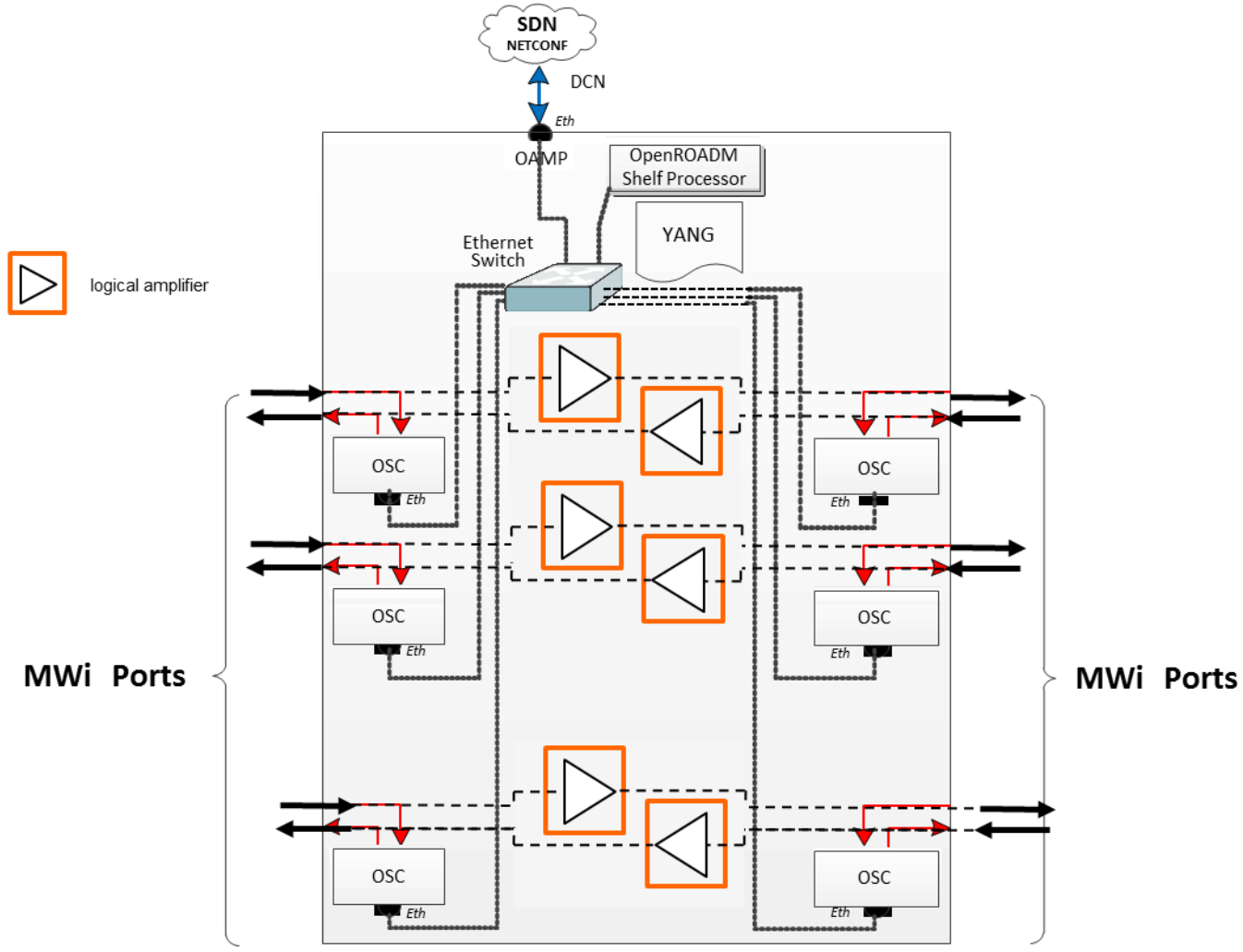
- General specs in the “Common” tab
- Physical specs defined in the “Physical spec” tab
- Line degree to line degree optical specs defined in the "MW-MW" tab
- Line degree to drop port and add port to line degree optical specs defined in the "MW-Wr" tab
- Local control specs defined in the "Local Control" tab
- Supported alarms defined in the "Alarms" tab
- Supported PMs defined in the "PMs" tab
- Ethernet OSC optical specs defined in the "OSC-Optical Line Port" tab
- Ethernet OSC functionality defined in the "OSC Overview" tab
- Automatic line shutoff functionality defined in the "Laser Safety" tab
- OAMP functions including protocol stack and specs in the “OAMP Port” tab
- OpenDaylight mounting requirements in the “Device mounting” tab
- OTDR functionality defined in the "OTDR" tab

[2] The current version of the MSA specification as of the writing of this document is 20171121a-Open-ROADM-MSA-specification-ver-2-00.xlsx

2.2.2 In-Line Amplifier

The Open ROADM MSA defines an In-Line Amplifier (ILA) device capable of amplifying the different channels of the WDM multiplex. An ILA site amplifies different wavelengths of the multiplex in both directions between 2xN degrees. The MSA does not define implementation details such as the number of degrees, form-factor, etc. A logical amplifier entity can include one or several amplifiers that can be implemented through on one or several circuit-packs. The MSA defines:

- API using NETCONF interface with a YANG-based data model that abstracts the management, control and provisioning of multi-vendor ILA devices
- Multi-wave ILA (MWi) interface which defines the optical specifications for the multi-wave interface between ILAs or between ILA and line degrees of the ROADM devices.
    - The MWi interface includes a GE OSC that provides LLDP-based topology information, support laser safety and provide MCN/DCN reach-through to remote ILA nodes. Otherwise, there is no ILA node to ILA node or no ROADM node to ILA node communication. Optical equalization (in the limit of the specs) is handled off-box by the controller by setting target-tilt.



Please refer to the Open ROADM MSA specification found at openroadm.org for the ILA specifications:

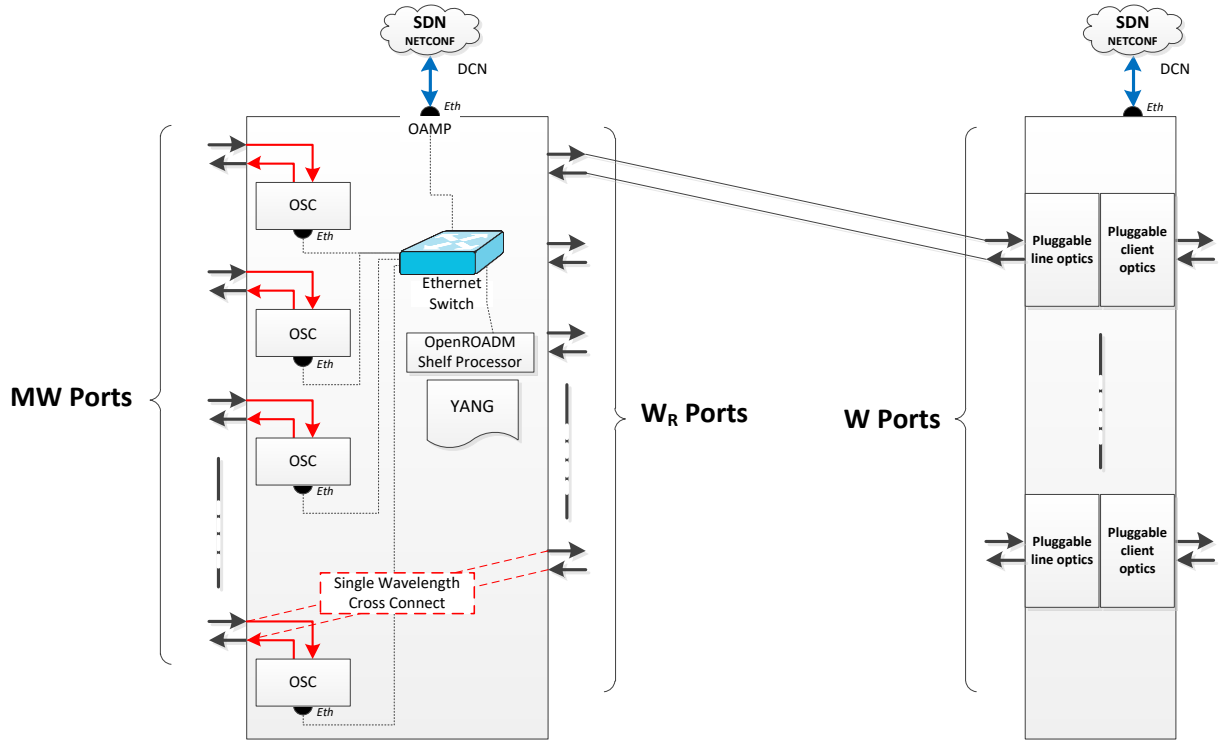
- General specs in the “Common” tab
- Optical specs defined in the "MWi(standard)" tab
- Local control specs defined in the "Local Control" tab
- Supported alarms defined in the "Alarms" tab
- Supported PMs defined in the "PMs" tab (MWi column)
- Ethernet OSC optical specs defined in the "OSC-Optical Line Port" tab
- Ethernet OSC functionality defined in the "OSC Overview" tab
- Automatic line shutoff functionality defined in the "Laser Safety" tab
- OAMP functions including protocol stack and specs in the “OAMP Port” tab
- OpenDaylight mounting requirements in the “Device mounting” tab

In MSA release 2.x, only one type of “standard” amplifier is defined. The specs of ILAs are in line with the specs of the ROADMs.

2.2.3 Transponder

The Open ROADM MSA defines a Transponder device capable of mapping a single 100GE or OTU4 client signal into a 100G OTU4 DWDM signal for transport across an Open ROADM infrastructure. The MSA does not define implementation details such as the number of client/line ports, form-factor, etc. The MSA (refer to openroadm.org) defines:

- API using NETCONF interface with a YANG-based data model that abstracts the management, control and provisioning of multi-vendor Transponder devices
- Single-wave (W) interface which defines the optical specifications for the full C-band tunable DWDM optical line interface of the Transponder that connects to a Wr add/drop port on the ROADM device
    - Line-side pluggable type must be CFP-DCO, CFP2-ACO or CFP2-DCO with LC connectors
- Client interfaces/ports must be pluggable QSFP28 with LC connectors and support:
    - 100GBASE-R mapped into OPU4 using PCS codeword transparent Ethernet mapping
    - OTU4



Please refer to the Open ROADM MSA specification found at openroadm.org for the Transponder specifications:

- General specs in the “Common” tab
- Optical specs defined in the “W Optical Spec” tab
- Support alarms defined in the” W ALM Spec” tab
- Support PMs defined in the “W PM Spec” tab
- Client functional specs defined in the “W TRPN functional” tab
- Physical specs defined in the “Physical spec” tab
- OAMP functions including protocol stack and specs in the “OAMP Port” tab
- OpenDaylight mounting requirements in the “Device mounting” tab

2.2.4 Muxponder / Switch

The Open ROADM MSA defines a general OTN switching model which allows for the support of ODUk/ODUj level cross connects between interfaces. The MSA does not define implementation details such as the number of client/line ports, form-factor, etc. The MSA (refer to openroadm.org) defines:

- API using NETCONF interface with a YANG-based data model that abstracts the management, control and provisioning of multi-vendor switch devices
- The line side interfaces have the same specifications and requirements as those defined for a Transponder. Single-wave (W) interface which defines the optical specifications for the full C-band tunable DWDM optical line interface that connects to a Wr add/drop port on the ROADM device
    - Line-side pluggable type must be CFP-DCO, CFP2-ACO or CFP2-DCO with LC connectors
- Client interfaces/ports are SFP / SFP+ / QSFP28:

    - 100GBASE-R mapped into OPU4 using PCS codeword transparent Ethernet mapping, PT=07
    - 10GBASE-R mapped into OPU2 using GFP-F, ITU-T G.7041 Clause 7.1 Ethernet MAC Payload, PT=05
    - 10GBASE-R mapped into OPU2 using GFP-F, ITU-T G.7041 Clause 7.9 Transporting Ethernet 10GBASE-R payloads with preamble transparency and ordered set information, PT=09
    - 10GBASE-R mapped into OPU2e using Bit-synchronous mapping (CBR10G3), PT=03
    - 1000BASE-X mapped into ODU0 using TTT and GMP, ITU-T G.709 Clause 17.7.1.1 1000BASE-X Transcoding, PT=07
    - OTU4/3/2/2e/1
- ODU Trail Termination Points (TTP) to represent the HO-ODUk/j
- ODU Connection Termination Points (CTP) to represent LO-ODUk/j
- ODU Trail Termination – Connection Termination Points (TTP-CTP) to represent path terminating CTP used to map client signals
- Definition of cross connect restrictions such that a single model can define the restricted case of a muxponder to the general case of an any to any non-blocking switch through the definition of switching pools
- Definition of port and slot capabilities to define what interface types are supported by the device
- Definition of port group restrictions to define possible dependencies on how ports can be configured

2.2.4.1 Termination Points

Three types of termination points are defined by the model to allow for provisioning of multiplex hierarchy and to facilitate cross connects

- ODU-TTP – An ODU TTP is associated with the context of a HO ODU. It is a PM layer terminated ODU which faces the line side under an OTU interface. For example, an ODU must be terminated in order to multiplex ODUj into it. An ODU-TTP always has a payload type of PT-20/21/22 indicating its tributary slot size and its ability to be muxed into it. It essentially has an MSI table which originates here.
- ODU-CTP – An ODU CTP (connection termination point) which can be part of a cross connect is considered a LO ODU. It is a non-terminated ODU which faces the line side under an OTUk or ODU-TTP. An ODU-CTP defines the endpoints of a cross connect.
- ODU-TTP-CTP – An ODU-TTP-CTP is both terminated and cross connected. An ODU-TTP-CTP is utilized in the context of mapping client interfaces into ODU which may then be cross connected.

An example diagram is provided below in Figure 1 for context.

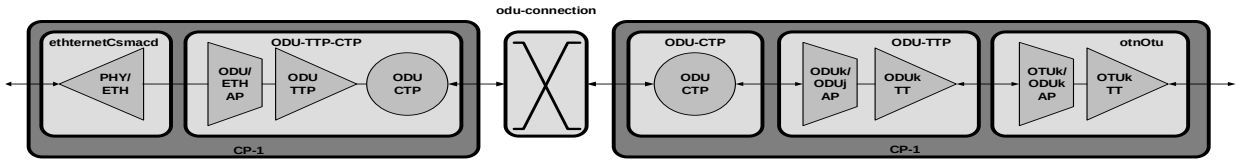


Figure 1 - ODU Termination Points

2.2.4.2 Cross Connects

The OTN cross connect is made by an odu-connection which has a src-if and dest-if. The source interface and destination interface are required to be an otnODU interface with a function of either ODU-CTP or ODU-TTP-CTP. Cross connects can either be bi-directional or uni-directional as defined by the direction attribute where bi-directional cross connects are the default. It is mandatory for implementations to support bi-directional cross connects. The support for uni-directional cross connects is optional.

2.2.4.3 Switching Pools

There are two types of switching pools defined by the MSA, non-blocking and blocking. A non-blocking switching-pool-types implies the presence of a single non-blocking-list and that any port-name in a defined port-list or any slot-name in a defined pluggable optics port list may be freely cross connected without restriction up to the interface bandwidth. An example of a non-blocking application is illustrated in Figure 2. In this example, the device supports the provisioning of cross connects between any of the ports illustrated. The device will indicate this through the odu-switching-pools where the defined pool will be of a non-blocking type and the non-blocking-list would have all ports, P1-Pn, in it.

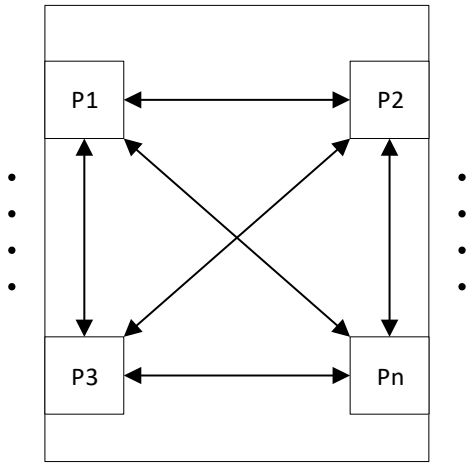


Figure 2 - Non Blocking Xponder

The definition of a blocking switching pool implies there is some restriction as to what ports may be cross connected. A blocking switching-pool-types has more than one non-blocking-list present in its definition. A simple example is that of a muxponder where the client ports may only be cross connected to the line port. For the example of a 10x10G muxponder with an OTU4 line side interface, the device model would indicate a blocking switching-pool-type comprised of 10 non-blocking-list where each such list had one unique client port and the common line port. For example (P1, P11), (P2, P11), (P3, P11), …, (P10, P11).

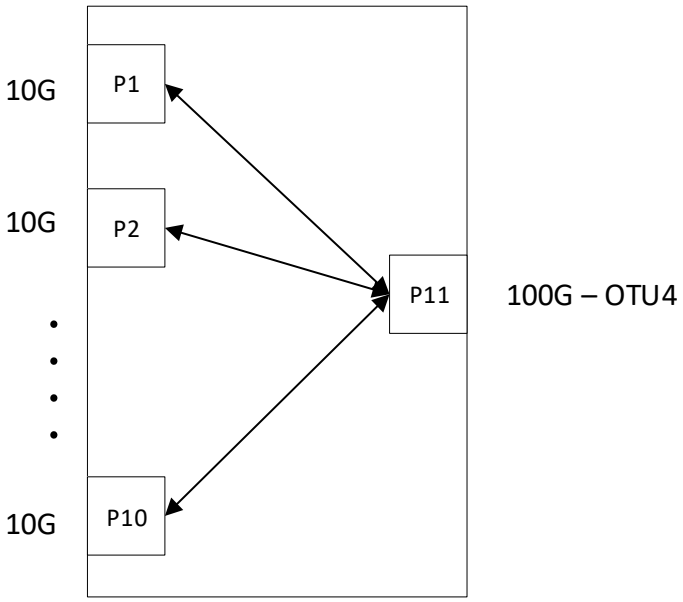


Figure 3 - Simple Muxponder

A further restriction of a muxponder may be to what trib port number and trib slot a given client port can be cross connected into the line interface. If such restrictions are present, they are indicated by the device model as a part of the port-capability-grp within the mpdr-client-restriction container.

Within a given switching pool, the device may allow for cross connects to be made between two or more non-blocking-list. If such a capability is present, it may be constrained by the amount of bandwidth which can be connected between such non-blocking-list. Let's consider the example of a two-card set which provides a line side OTU4 interface and 10x10G client interfaces on each card. However, as opposed to the simple muxponder, each card is capable of non-blocking switching between all of its ports to support hairpin connections. In addition, traffic can be cross connected between line and client-side ports on each card, but up to a limit of 100G of bi-directional traffic. This example is illustrated below in Figure 4.

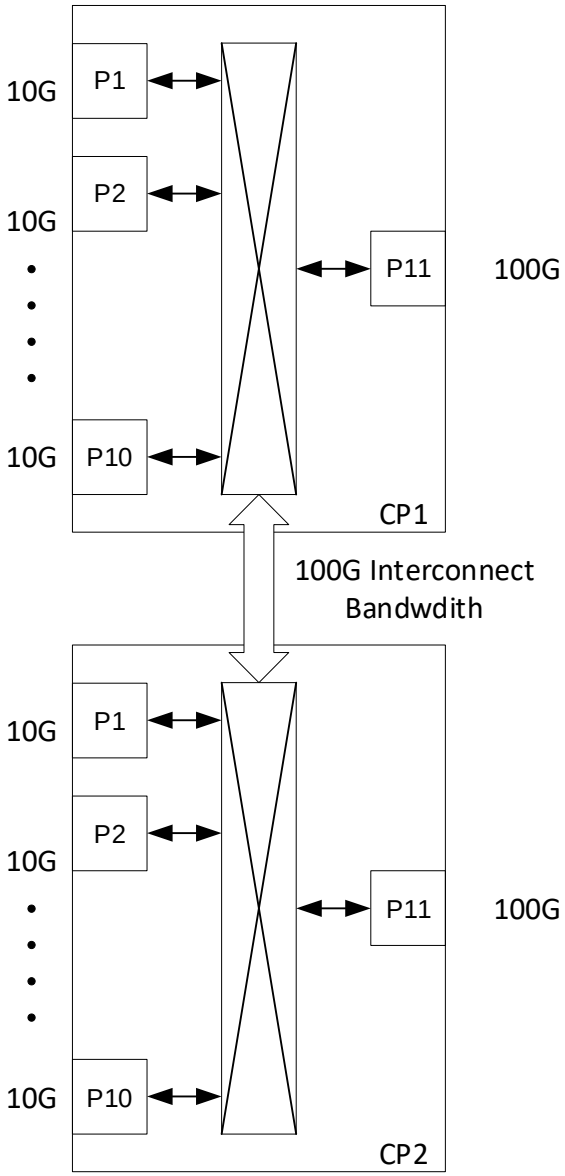


Figure 4 - Switching Pool with Interconnect BW

In this case, the device may present a blocking switching-pool-type comprised of two non-blocking-list where each list contains all the ports on a given card. The bandwidth constraint between the two non-blocking-list is modeled by the interconnect-bandwidth and interconnect-bandwidth-unit. For example, this interconnect could represent an internal ODU4 link comprised of 80 OPU4 based tributary slots. In such a case, the total interconnect-bandwidth would be represented by 80 X 1,301,683,217 bps interconnect-bandwidth-units. The interconnect-bandwidth-unit in this case is the minimum ODTU4.ts rate.

In a different case, the interconnect-bandwidth may not be quantized as in the example above. In such a case it can be appropriate to present the interconnect-bw as a single unit of say 105,000,000,000 Gbps.

Switching pool BW is not modified by the device due to the provisioning of cross connects. It is the responsibility of the controller to maintain the usage of any interconnect bandwidth. Note that the usage of interconnect bandwidth can be derived from cross connect provisioning information.

The device is responsible for updating switching pool information as cards are added/deleted from the system or card provisioning updates the port information. Switching pools reflect the capabilities of the device to cross connect services between the currently configured port mode of operation. For example, the ports available may be altered through port group configuration. Switching pools reflect the capability to perform cross connection between both fixed ports and pluggable optics holders. For the case of the pluggable optics holder, the pluggable optics themselves do not need to be installed for the switching pool to identify the capability.

2.2.4.4 Port Capabilities

Before introducing port capabilities, a quick word about how ports are referenced, as this can be confusing. In the model, a port can be in reference to a circuit pack where the circuit pack is represented by a pluggable optics module itself. Such a circuit pack nominally has a single port and all such circuit packs can have a common port-name, such as 'P1', given there is a unique circuit-pack-name. As such there are defined both fixed ports and pluggable ports in the model. A fixed port is in reference to an interface which is a part of a circuit pack such as represented by a line module. A pluggable port is reference to an optical module/circuit pack which might be plugged into an SFP, QSFP, … cage on the line module. Such a circuit pack should have the Boolean 'is-pluggable-optics' set as True.

In addition to ports, the model defines cp-slots. These are in reference to the SFP, QSFP, CFP… cages themselves. An enumeration within a cp-slot defines it to be a 'pluggable-optics-holder'. Such slots are often considered ports on the device and as such the port capabilities and switching pools can be in reference to the cp-slots.

Port capabilities allows the device to advertise to the controller the functionality offered by the device. The MSA allows flexibility in the offered set of services by the device subject to the qualification of such device by the end user.

The fundamental set of capabilities announced is provided by the list of supported-interface-capability. This list identifies port configuration options which may be used to determine a set of service types offered by the interface. For example, a cp-slot representing an SFP/SFP+ cage might indicate the ability to offer service types ranging from 10GbE, OTU2, OTU2e, 1GE, etc… When a pluggable optics module is installed into such a slot, that port may too have a set of capabilities that are subset of those offered by the cp-slot. For example, in the example above the pluggable optics may only indicate a capability to provide a 1GbE service type. In such a case, the cp-slot capabilities remain constant while the installed pluggable optics may filter the supported list. It is the responsibility of the controller to understand the intersection of the capabilities to ensure the desired service type can be offered.

Within the port-capability-grp, there can further exist information associated with OTN interfaces defined by the otn-capability-grp, odu-mux-hierarhcy-grp, and mpdr-client-restriction.

The otn-capability-grp defines a set of functionalities associated with the OTN interface. Capability information covers

- List of supported protection types as defined in prot-otn-linear-aps

- Proactive delay measurement support at PM and TCM layers
- TCM and TCM direction support
- ODU payload support in terms of PT20/21 indication

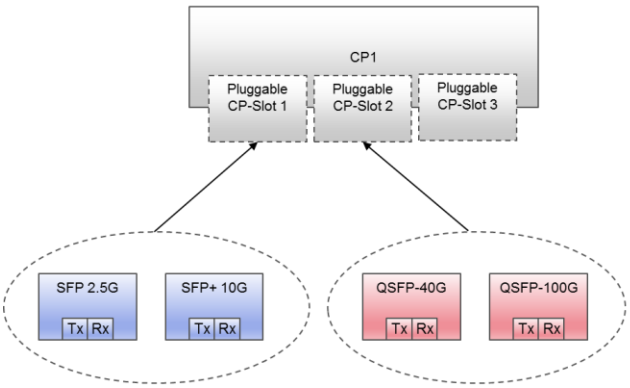
The odu-mux-hierarchy-grp defines support for

- Single vs multi-stage multiplexing
- Identity of LO ODU rates supported
- Proactive delay measurement support at PM and TCM layers for LO ODU
- TCM and TCM direction support for LO ODU

The mpdr-client-restrictions identify restrictions to the general cross connect model that may be present on tributary port number and tributary slot usage for client services being cross connected into what would nominally be considered a network side interface. The mpdr-client-restrictions would be present on the client ports being mapped into LO ODU and then cross connected to a network side interface. Such restrictions are common to many muxponder implementations. For example, a 10x10G muxponder may have the restriction that client port 1 must be mapped into trib port 1, trib slots 1-8, client port 2 must be into trib port 2, trib slots 9-16, etc… The mpdr-client-restrictions allow for a common provisioning model for xponder, muxponder, and transponder applications. It is understood that an existing transponder provisioning model exists from R1.x and that remains supported. The mpdr-client-restrictions include…

- Definition of the network side interface to which this LO ODU is associated
- The tributary port number associated with the network side interface as this LO ODU is multiplexed into the network side HO ODUk
- The tributary slots required to be used on the network HO ODUk

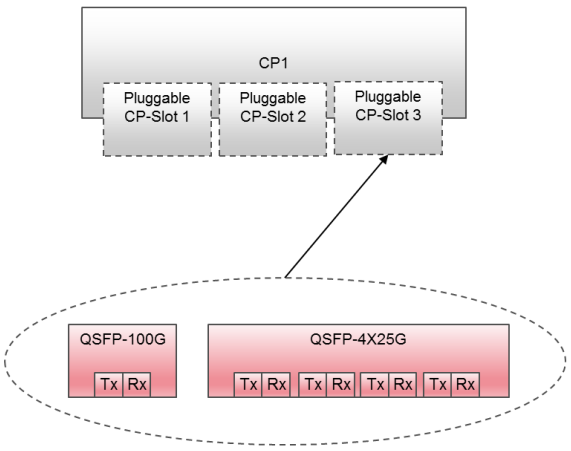
In Figure 5, an example is illustrated of a circuit pack with three pluggable slots. The first slot accepts SFP/SFP+ type pluggable optics while the second supports QSFP / QSFP28 pluggable optics. In each case, the supported-interface-capabilities provides a list of service types supported along with the further capabilities of that interface type.



circuit-pack-name	cp-slot-name	circuit-pack-type	port-name	port-capabilities
1	1	SFP	1	supported-interface-capability* = {if-och-otu1, if-1gbe} **For if-och-otu1** {if-protection-capability, tcm-direction, ho-odu-index, … } **For if-1gbe** {client-mapping-odu-type, odtu-type, network-ho-odu-trib-port-number, … }
		SFP-Plus	1	supported-interface-capability* = {if-och-otu1, if-1gbe, if-och-otu2, if-10gbe} **For if-och-otu1** {if-protection-capability, tcm-direction, ho-odu-index, … } **For if-1gbe** {client-mapping-odu-type, odtu-type, network-ho-odu-trib-port-number, … } **For if-och-otu2** {if-protection-capability, tcm-direction, ho-odu-index, … } **For if-10gbe** {client-mapping-odu-type, odtu-type, network-ho-odu-trib-port-number, … }
	2	QSFP-40G	2	supported-interface-capability* = {if-och-otu3, if-40gbe} **For if-och-otu3** {if-protection-capability, tcm-direction, ho-odu-index, … } **For if-40gbe** {client-mapping-odu-type, odtu-type, network-ho-odu-trib-port-number, … }
		QSFP-100G	2	supported-interface-capability* = {if-och-otu3, if-och-otu4, if-100gbe, } **For if-och-otu3** {if-protection-capability, tcm-direction, ho-odu-index, … } **For if-och-otu4** {if-protection-capability, tcm-direction, ho-odu-index, … } **For if-100gbe** {client-mapping-odu-type, odtu-type, network-ho-odu-trib-port-number, … }

Figure 5 - Port Capabilities

A further example is illustrated in Figure 6. This example is provided to illustrate the intended extensibility of the model to support multiple interfaces from a single optical module as might happen in pig-tail application.



circuit-pack-name	cp-slot-name	circuit-pack-type	port-name	port-capabilities
1	3	QSFP-100G	3	supported-interface-capability* = {if-och-otu4, if-100gbe} **For if-och-otu4** {if-protection-capability, tcm-direction, ho-odu-index, … } **For if-100gbe** {client-mapping-odu-type, odtu-type, network-ho-odu-trib-port-number, … }
		QSFP-4x25G	3-1	supported-interface-capability* = {if-och-otu2, if-10gbe, if-25gbe} **For if-och-otu2** {if-protection-capability, tcm-direction, ho-odu-index, … } **For if-10gbe** {client-mapping-odu-type, odtu-type, network-ho-odu-trib-port-number, … } **For if-25gbe** {client-mapping-odu-type, odtu-type, network-ho-odu-trib-port-number, … }
			3-2	supported-interface-capability* = {if-och-otu2, if-10gbe, if-25gbe} " … "
			3-3	supported-interface-capability* = {if-och-otu2, if-10gbe, if-25gbe} "…"
			3-4	supported-interface-capability* = {if-och-otu2, if-10gbe, if-25gbe} " … "

Figure 6 - Port Capabilities - Multi Phy

2.2.4.5 Port Group Restrictions

The concept of port group restrictions is based on the ability of a device to offer a set of software programmable service types on a group of interfaces. However, within that group of interfaces there may be restrictions on what can simultaneously be provisioned. For example, consider the case of a 10G muxponder which provides 8 x SFP based ports. It may be that these ports are software configurable to support a range of services. For example, 8 GbE or 4 x OC-48/STM16. The device, however, may present restrictions on what can be configured on which interfaces at one time.

The model provides the ability of the device to document such restrictions through a port-group-restriction-grp grouping. Nominally the group models a table with a set of ports which can include a port-list for fixed ports and a pluggable-optics-holder-list for pluggables. This group is referenced by a port-sharing-id to identify ports which have dependency on configuration and/or bandwidth usage. Within such a group, there is modeled a list of possible-port-config. Each entry in this list is identified by a config-id and identifies the possible service types that can be configured as indicated by the supported-if-capability.

An example of port group restrictions is introduced in Figure 7 below. In this example the group is comprised of two sets of four interfaces where bandwidth and configuration is restricted between interfaces 1-4 and 5-8. Interfaces 1-7 are pluggable-optics-holder, while port 8 is fixed port residing on a common circuit pack, CP1.

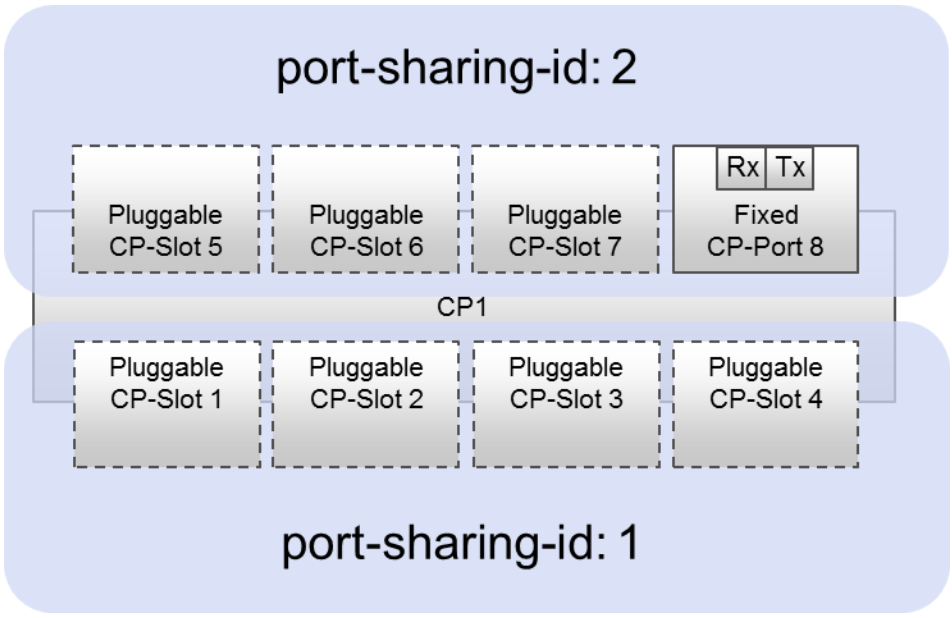


Figure 7 - Port Group Restriction Interfaces

Within ports 1-4 and 5-8 there is a restriction the total BW of the configured ports should not exceed 5Gbps. Within each group, interface types can be configured for GE, OC3/STM1, OC12/STM14, OC48/STM16, or OTU1. However, all such types cannot be satisfied on any port at any time in any combination. The possible-port-config identifies the supported combinations as illustrated in Figure 8.

In the table of Figure 8, one can identify that config-id 1 restricts the interface type of FC-400 to only be supported on port 1, config-id 2 supports OTU1 and/or OC48/STM16 on ports 1 and 3, config-id 3 allows port 1 to be either OTU1 or OC48/STM16 while ports 3 and 4 can be any combination of GE, OC3/STM1, OC12/STM4, etc…

port-sharing-id	circuit-pack-name	port-name	cp-slot-name	shared-bandwidth	config-id	circuit-pack-name	port-name	cp-slot-name	port-if-type*
1	CP1		1	5G	1	CP1	1	1	FC-400
	CP1		2		2	CP1	1	1	OTU1
	CP1		3						OC48
	CP1		4			CP1	3	3	OTU1
									OC48
					3	CP1	1	1	OTU1
									OC48
						CP1	3	3	GE
									OC3
									OC12
						CP1	4	4	GE
									OC3
									OC12
					4	CP1	1	1	GE
									OC3
									OC12
						CP1	2	2	GE
									OC3
									OC12
						CP1	3	3	OTU1
									OC48
					5	CP1	1	1	GE
									OC3
									OC12
						CP1	2	2	GE
									OC3
									OC12
						CP1	3	3	GE
									OC3
									OC12
						CP1	4	4	GE
									OC3
									OC12

Figure 8 - Port Group Restrictions Definition

2.2.4.6 Mupxonder specifics

The OTN muxponder is a special type of OTN switch with the additional constraint that the mapping between the client side and network side is fixed. There is no flexibility in the hardware to support a variable mapping, so a routing function in the controller would need to understand the fixed mapping constraint.

The muxponder is modeled as an OTN switch with additional constraints that allow only certain mappings between the client and the network side with regards to tributary slots and tributary port number. In addition, only certain client mapping into ODU payloads are permitted.

Open ROADM supports a limited set of muxponder clients and network mappings to enable interoperability:

- 10 x 10GE/OTU2 clients mapping into a 100G network (see Table 1). Mapping of the 10GE into an ODU2 and ODU2E are both supported.
- 8 x 1GE clients mapping into a 10G network (see Table 2)

Table 1: 10 x 10GE/OTU2 to 100G Mapping

Client Port	Network ODU mapping
1	TPN 1, TS 1-8
2	TPN 2, TS 9-16
3	TPN 3, TS 17-24
4	TPN 4, TS 25-32
5	TPN 5, TS 33-40
6	TPN 6, TS 41-48
7	TPN 7, TS 49-56
8	TPN 8, TS 57-64
9	TPN 9, TS 65-72
10	TPN 10, TS 73-80

Table 2: 8 x 1GE to 10G Mapping

Client Port	Network ODU mapping
1	TPN 1, TS 1
2	TPN 2, TS 2
3	TPN 3, TS 3
4	TPN 4, TS 4
5	TPN 5, TS 5
6	TPN 6, TS 6
7	TPN 7, TS 7
8	TPN 8, TS 8

2.2.4.6.1 Muxponder Modeling

The OTN muxponder would be modeled as an OTN switch with additional constraints to account for the fixed mapping between the client and network side of the muxponder. This includes advertising muxponder connectivity via the switching pool, supporting the provisioning of explicit cross connects, allowing for the optional advertisement of "fake" ODU interfaces if the hardware only supports one ODU interface, and the capabilities advertisement specific to muxponders.

2.2.4.6.2 Switching pool advertisement

The muxponder will advertise potential connectivity between the client and network side based on switching pools similar to OTN switches. Open ROADM supports muxponders that can map between the client and network ports, but not between the client ports themselves. Figure 9 shows an example

muxponder that supports ten (10) 10GE pluggable client ports and one (1) 100G OTU4 network port. The switching pool advertisement for this muxponder would be as follows:

```
odu-switching-pools
        switching-pool-number           1
        switching-pool-type             blocking
        non-blocking-list
                {nbl-number             1
                interconnect-bandwidth-unit     0
                interconnect-bandwidth          0
                port-list [{CP1, C1}, {CP-NWK, N1}]
                },
                {nbl-number             2
                interconnect-bandwidth-unit     0
                interconnect-bandwidth          0
                port-list [{CP2, C2}, {CP-NWK, N1}]
                },
                …,
                {nbl-number             10
                interconnect-bandwidth-unit     0
                interconnect-bandwidth          0
                port-list [{CP10, C10}, {CP-NWK, N1}]
                }
```

The non-blocking list elements, specifically the port list, indicate that each client port and network port are not blocked. But the interconnect bandwidth of 0 indicates that the client ports cannot be connected to each other.

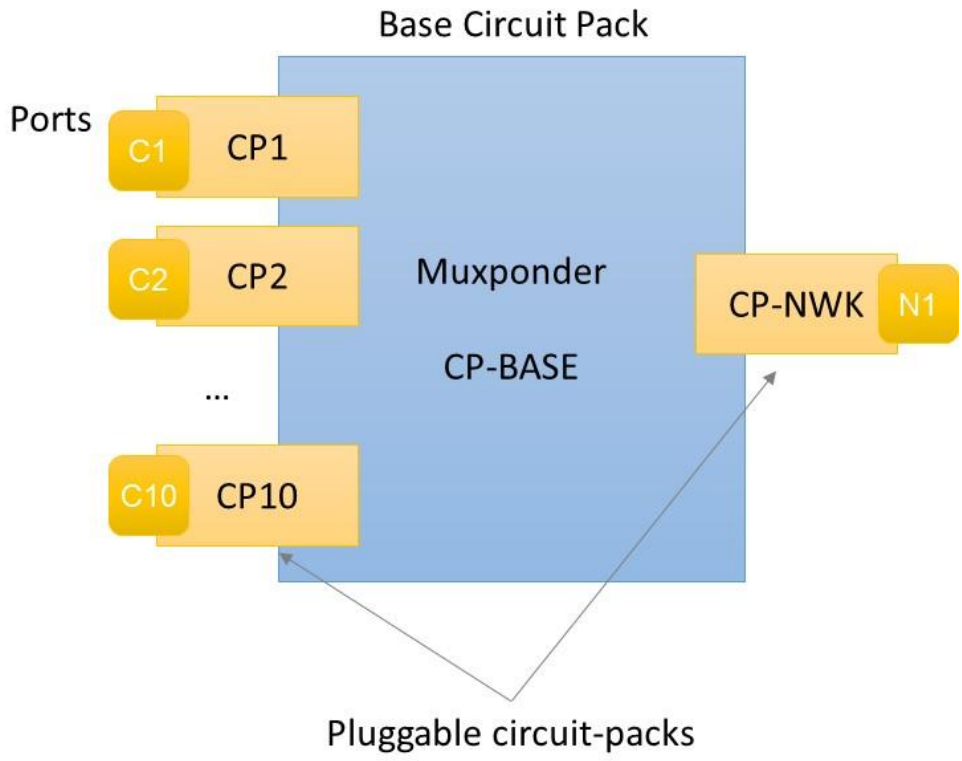


Figure 9: Example 10x10GE to 100G Muxponder

2.2.4.6.3 Explicit ODU cross connects

ODU services that traverse over an OTN muxponder are established with an explicit ODU cross connect (odu-connection), even if the hardware has no ODU grooming flexibility. To support ODU cross connects, there will be ODU interfaces created on the network side and an ODU interface created on the client side with an odu-connection(s) connecting the two interfaces.

In Open ROADM at the ODU layer, implementations are required to support bidirection odu-connection entities. Implementations may optionally also support unidirectional odu-connections.

For muxponders, implementations must support one bi-directional cross connect from each client port to the network port, and may additionally support the ability to provision this using two uni-directional cross connects.

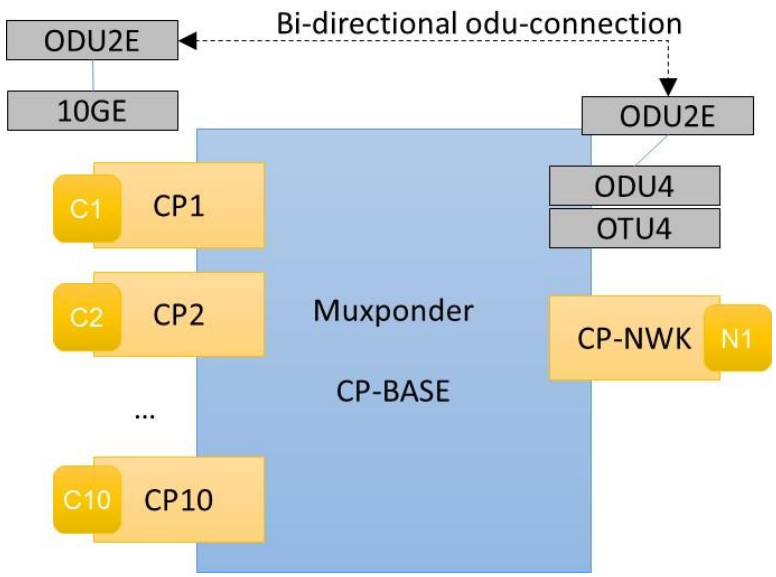


Figure 10: ODU connection

2.2.4.6.4 Advertisement of ODU capabilities based on hardware limitations

Some implementations of muxponders may only support one ODU interface at the physical hardware level. But to support the muxponder's odu-connections, the model will have two ODU interfaces (client and network side). Thus, one of the two ODU facilities will be "virtual" such that there is no functionality in the hardware.

To indicate this case to the controller, the no-oam-function attribute was added to the ODU interface. This indicates that the ODU facility cannot support OAM functions such as alarming, PM, threshold and TCAs, provisioning of TCM, delay measurement, etc.

2.2.4.6.5 Muxponder specific capabilities announcements

The port capabilities will provide the mapping restriction between the client and network side. This capability is advertised against the client port and provides information about how it maps to the network side:

- network-ho-odu-circuit-pack-name and network-ho-odu-port-name: identifies the port that would hosts the HO-ODU interface (note: the capability advertisement may exist before the HO-ODU is created)
- odtu-type, network-ho-odu-trib-port-number, and network-ho-odu-trib-slots: identifies the type of HO-ODU as well as the mapping of the LO-ODU.

2.2.5 Pluggable Optics

The Open ROADM MSA defines the use of standards-based pluggable optics for the Transponder device. Client interfaces on the Transponder must use standard QSFP28 pluggable optics. Line interfaces on the Transponder must use standard CFP-DCO, CFP2-ACO or CFP2-DCO pluggable optics that conform to the W optical specification.

Otherwise, the Open ROADM MSA does not dictate nor preclude the use of pluggable optics to perform other functions within the ROADM or Transponder device.

2.2.6 YANG Data Model

The YANG data model consists of:

- Config / Operational data defines database objects which can be queried by the controller. Some database objects can be read/write (config), while others are read-only (operational). Examples include shelf commissioning data, wavelength connections, etc.
- Notifications for the purposes of reporting autonomous events to the controller. Examples include alarms, inventory changes, re-start, etc.
- Remote Procedure Calls (RPC) that do not effect a change in the device configuration data, e.g., get-connection-port-trail, file transfers, database backup, etc.

The Open ROADM device YANG data model defines the following objects and RPCs to abstract the implementation of the ROADM and Transponder device:

- Info provide general node information including node name, IP address, etc.
- Shelves provide shelf information. A node can consist of one or more shelves.
- Circuit Packs represents a physical piece of equipment which contains a group of hardware functional blocks such as common equipment, cards, plug-in-units and/or pluggable optics.
    - The Ports container defines the ports associated with a circuit pack or pluggable optics and the associated port attributes
    - Capabilities are defined for the slots (cp-slots) and ports to describe additional metadata capabilities such as what interfaces can be created on the slot/port
- Internal Links reflect the connectivity within each circuit pack. These objects are read only and report attributes of the circuit pack themselves.
- Physical Links reflect the connectivity between ports across different circuit packs. The controller pushes this data to the device and reflects the actual inter-card fibering/cabling.
- External Link objects are placeholders for data about the far end device. Data for these objects is pushed from the controller.
- Degrees define the grouping of circuit packs that form a line degree
- Shared Risk Groups define the grouping of circuit packs that form a colorless/ directionless add/drop bank
- Line amplifier describes the directions of an in-line amplifier node including gain and tilt control
- Xponder defines the logical grouping of ports that make up a transponder, muxponder, regenerator or switchponder
- Connection Map is wavelength agnostic and reflects any connectivity restrictions / blocking in the device (not wavelength contention). Note that Connection Map is used for transponder, regenerators and ROADM devices
- ODU Switching Pools describe the connectivity associated with the ODU layer. Non-blocking as well as blocking configurations are supported. ODU switching pools are applicable to muxponders, regenerators and switchponders.
- Interfaces defines supported interface types and are associated with Port objects. The following interfaces are supported: Ethernet, MC, NMC, OCH, OTS, OMS, OTU, ODU.
- Cross connects are supported in the form of ROADM connections (OCH/NMC layer) and ODU connections (ODU layer).
- Protection groups are supported for the ODU SNCP layer.

- LLDP and RSTP protocols
- Additional bandwidth restrictions across ports using the Port Group Restriction container
- Software download and database operations
- Firmware version reporting and firmware update procedures. Firmware is automatically applied when non-service affecting. Firmware is applied following a cold reboot when service affecting. A manual RPC is also supported to manually apply a firmware.
- Syslog support
- Other RPC operations including: restarts, LED control, debug information retrieval (tech info), get connection port trail to determine the route of a service through a device, disable automatic laser shutdown, start OTDR scan, and set current date and time.
- Wavelength map has been obsoleted in v2 with the change to a flexgrid modeling approach for optical channels.

A device will also support the common models that would provide the following objects and RPCs:

- Active alarm list including issuing alarm notifications (push model)
- Performance monitoring including both a current list and a historical list. The data tree view of the historical PM is optional for vendors to implement. Vendors must implement the retrieval of historical PM data via a file using the collect-historical-pm-file RPC
- Potential TCA list provides a means to view the potential TCAs on a node and to set the threshold levels. TCA notifications are also supported
- Software manifest YANG module. Note that this is not intended to be implemented on the device but provides the format of data that would be provided out-of-band to a controller to guide the controller on software download and database operations.
- Other RPC operations including: clear PM register and change password

2.2.7 Open ROADM controller

The Open ROADM architecture assumes the existence of an Open ROADM controller that controls the Open ROADM devices and provides device (inventory), network and service APIs to northbound OSS systems. The Open ROADM MSA does not specific the requirements, implementation or operation of an Open ROADM controller. The exception is the specification of the Open ROADM device, network and service models that would be used at the controller level.

Some controller functions and/or procedures may be provided within this document. These items should be taken as guidance or examples of a possible controller implementation to guide the development of an Open ROADM devices.

2.3 Open Interfaces

2.3.1 W

The Single-Wavelength interface “W” specifies the interoperability of two transponders. The Optical specification outlines a minimum number of parameters for framing and bit ordering to enable the interoperability between different hardware manufacturers.

2.3.2 Wr

The Single-Wavelength port on the ROADM (also known as add/drop port) is called “Wr”. Here, a single-wavelength output “W” from an Xponder is plugged into the ROADM.

2.3.3 MW

The interoperability of different manufacturers ROADMs is guaranteed by defining the Multi-Wavelength interface “MW”. The Optical Specification sheet from OpenROADM.org contains the minimum interoperability specifications with some performance metrics of the ROADM (see MW-MW and MW-Wr tabs).

2.3.4 MWi

The interoperability of different manufacturers ILAs and ILAs to ROADMs is guaranteed by defining the Multi-Wavelength interface “MWi”. The Optical Specification sheet from OpenROADM.org contains the minimum interoperability specifications with some performance metrics of the ROADM (see MWi).

2.3.5 OSC

The optical supervisory channel is part of the MW interoperability. The MSA has defined 1000BASE-LX interoperability and a simple Ethernet wayside channel (see OSC tabs in spreadsheet). Aside from the mandatory safety automatic power shutdown control loop (see Laser safety tab), no control loops are running between two adjacent ROADMs. All control loops are abstracted into the centralized controller. There are, however, a few local link control loops running on the device, such as transient control. See Local Control tab for further details.

3 NODE DESCRIPTION, INSTALLATION AND DISCOVERY

3.1 FILE OPERATIONS AND STRUCTURE

3.1.1 Local device file structure

The file system on the Open ROADM device follows a flat structure with no subdirectories. When an operation generates multiple files, these would be tar’d then zip’d in a single file which is stored locally on the flat structure on the device.

The allocated space on the device shall be large enough to accommodate at least one copy of the files required to support operations requiring files exchange with the Open ROADM controller (ex: debug, syslogs, database for backup and restore, software loads for upgrade, OTDR trace, PM historical file for 15m, 24h and NA granularities, etc.)

3.1.2 SFTP specification

SFTP is used to exchange files between the Open ROADM controller and the devices; in such case the device is the SFTP client and the Open ROADM controller is the SFTP server, in such scenario the device would initiate the SFTP connection to the server, a file transfer to the device would translate in a “get” operation triggered by the device and a file transfer from the device would translate in a “put”

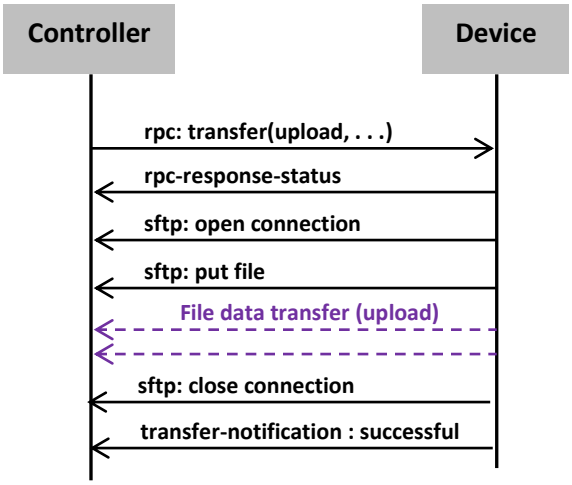
operation triggered by the device. The SFTP server port can be specified as an optional parameter in the URL of the transfer RPC, if a port is not specified the device shall use default port 22.

3.1.3 File transfer (upload)

rpc: transfer

- action: upload (from device to Open ROADM Controller)
- local-file-path: file on the device
- remote-file-path: URI of file on the Open ROADM Controller including credentials

It was agreed that the operation be asynchronous with rpc-response-status is sent when the request is accepted

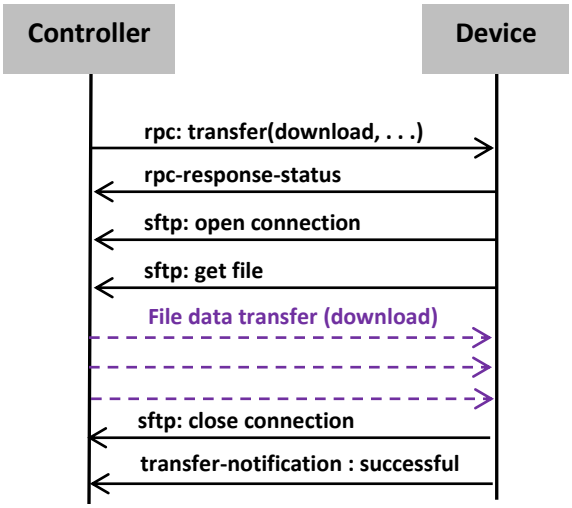


3.1.4 File transfer (download)

rpc: transfer

- [input] action: download (from Open ROADM Controller to device)
- [input] local-file-path: file on the device
- [input] remote-file-path: URI of file on the Open ROADM Controller including credentials

It was agreed that the operation be asynchronous with rpc-response-status is sent when the request is accepted.



3.1.5 Retrieve file list

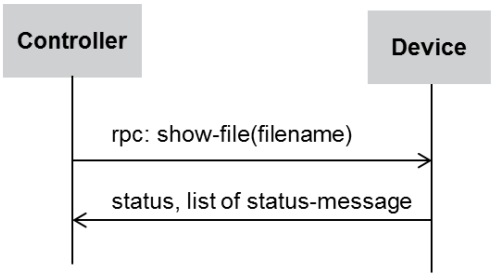
rpc: show-file

- [Input] filename: file to be listed (* is allowed as wild-card)
- [Output] status: Successful or Failed
- [Output] list of status-message, each one contains one filename

The file list operation is only applicable to the content of the flat Open ROADM directory structure. In terms of operations it was agreed that:

- A "*" or blank input would list all the files in the Open ROADM flat structure
- Only one filename can be specified as an input
- Partial wildcarding is not supported (ex: abc*)
- An empty list output would be represented by a success status with no file list

The MSA members will consider enhancements in a future release to show additional file information (ex: file size and timestamp).



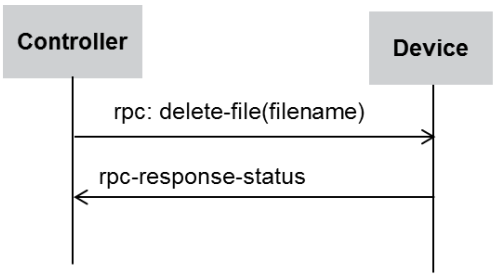
3.1.6 Delete file

rpc: delete-file

- [Input] filename: name of file to delete
- [Output] rpc-response-status

The file delete operation is only applicable to the content of the flat Open ROADM directory structure. In terms of operations it was agreed that:

- Input wildcarding (ex: “*”) is not supported
- Only one filename can be specified as an input



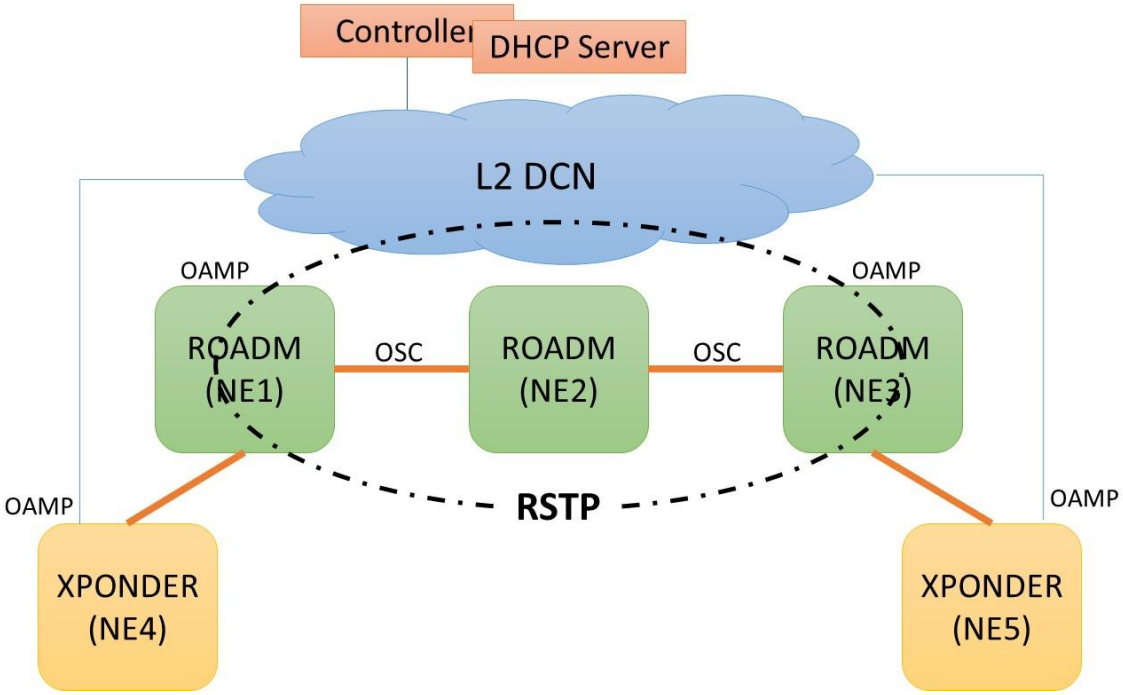
3.1.7 File operation notifications

Transient notifications were not fully supported in the Open ROADM Version 1.2.1 YANG models. As a short-term workaround, file operation transient notifications were sent as alarms with severity set to 'indeterminate’. These notifications did not have a corresponding clear notification unlike a normal alarm. Therefore, in Open ROADM Version 1.2.1, traditional alarms were not to be raised with the ‘indeterminate’ status.

This issue has been addressed in Open ROADM Version 2. In this release, specific notifications were added to support transient notifications. For example, notifications were added for create tech info complete, file transfer status, ODU SNCP protection group switch, RSTP topology, and software download/database operations events. In Version 2, the alarm severity ‘indeterminate’ should no longer be used to issue transient notifications.

3.2 DATA COMMUNICATION NETWORK (DCN)

Figure below shows the overview of the Open ROADM DCN architecture. Each device acts as a layer 2 bridge and runs Rapid Spanning Tree (RSTP) to provide loop free topology.



For the ROADM NE:

- Each ROADM NE is running in L2 bridging mode
- NE1 and NE3 have OAMP connection to DCN while NE2 is only reachable via OSC
- Single bridge with Bridge ports {OAMP, OSC1, OSC2, …}
- RSTP is enabled on OAMP and OSC ports to provide loop free topology
- NE1 and NE3 can be reached via OAMP if RSTP has not blocked that port otherwise reachable via OSC.
- NE2 is reachable via OSC
- Customer L2 DCN also participate in RSTP
    -

For Transponder Node:

- Has single management port OAMP.
- Acts as a host since there is only single management port.
- Does not need to participate in the RSTP exchange as the transponder node is a host

When a ROADM or transponder NE is powered up, it runs IPv4 and IPv6 DHCP clients and gets an IP address from the DHCP server sitting on the same LAN. The IP address may be either IPv4 or IPv6 depending on the operator's DCN configuration. This IP address is called temporary IP address. When DHCP server allocates temporary IP address for a NE, then Controller is notified for this new IP address allocation. Now controller can login to this NE using temporary address and can give the permanent IP address as per the carrier planning.

Provisioned IP address on the NE can be IPV6 or IPV4. NE also allows to provision the default gateway. The IP address, prefix length and default gateway should be specified in the same edit config operation.

3.3 NETCONF PROTOCOL

The Open ROADM network element is managed using the NETCONF protocol [RFC 6241] on TCP port 830. The default username and password for accessing a NETCONF network element is openroadm/openroadm.

The Open ROADM network element advertises its capabilities in the NETCONF Hello message. The Hello message provides an indication of support for standard features defined in NETCONF RFCs as well as support for specific namespaces.

Configuration data can be written directly to the running configuration datastore or written to the candidate configuration datastore with a subsequent commit to push the configuration to the running datastore. The specific mechanism supported by a Network Element is advertised in the Hello message. A vendor can support writing to the running configuration only, writing to the candidate configuration only, or support writing to both the running and candidate configurations.

NETCONF requires the support for subtree filtering on NETCONF messages. Network elements MUST also support XPATH filtering. An Open ROADM network element indicates its support for XPATH in the Hello message.

Operations on the network element should be idempotent when the NETCONF "merge" operation is used. If the data in the merge operation is the same as what already exists on the network element (even if it specifies only a partial set of the total attributes on the NE), the NE will accept the command with no changes. No database change notification would be sent under this case.

3.3.1 SSHv2 support

The NETCONF protocol runs over SSHv2 for security. It is recommended that implementations follow RFC 6242 which is the latest specification of NETCONF over SSH. RFC 6242 provides the procedure for interoperability with NETCONF implementations that support the older NETCONF over SSH described in RFC 4742.

Note: In RFC 4742, "]]>]]" was used as the NETCONF message delimiter. But an issue was identified where this character string could appear in the NETCONF body causing parsing issues. RFC 6264 modifies the message format to support message chunks with explicit message size identifiers to overcome this issue.

If multiple NETCONF sessions are established to a network element, those sessions should be established over separate SSH tunnels.

The Open ROADM network element uses the password authentication method for SSH. Once authenticated, the controller will request to open a channel of type "session" and invoke the "netconf" subsystem.

Devices may support IDLE timeout to clean up inactive sessions. In such case, keep alive messages are required to maintain connectivity. For example, an Open ROADM controller may send periodic retrieves of the info container using a ping mechanism.

3.3.2 Notification support

Open ROADM network elements must support NETCONF event notifications [RFC 5277]. Implementations may choose to output the Open ROADM notifications on the optional OPENROADM stream. If this stream is used, only Open ROADM notifications should be exposed on this stream. If the OPENROADM stream is not supported, then implementations must output the Open ROADM notifications on the default NETCONF stream.

Note that there is no guarantee that only Open ROADM notifications would appear on the default NETCONF stream.

A controller should retrieve the list of streams supported by an Open ROADM network element. If the NE supports the OPENROADM stream, the controller should subscribe to that stream. Otherwise it should subscribe to the NETCONF (default) stream.

Open ROADM devices must support the interleave option that allows both notifications and RPCs in the same session.

3.3.3 NETCONF monitoring

Open ROADM network elements must support NETCONF monitoring [RFC 6022] including support for retrieving the NETCONF capabilities, datastores, schema and session information. Implementations may support NETCONF monitoring statistics.

3.3.4 Change notifications.

The Open ROADM YANG model supports the change-notification to identify changes on the network element. This notification provides information about the change including the date and time of the change, who initiated the change (server or user), the datastore affected, the change operation, and the target of the change. The target points to the element in the data model that has changed. It is up to the controller to retrieve that element to determine the new value.

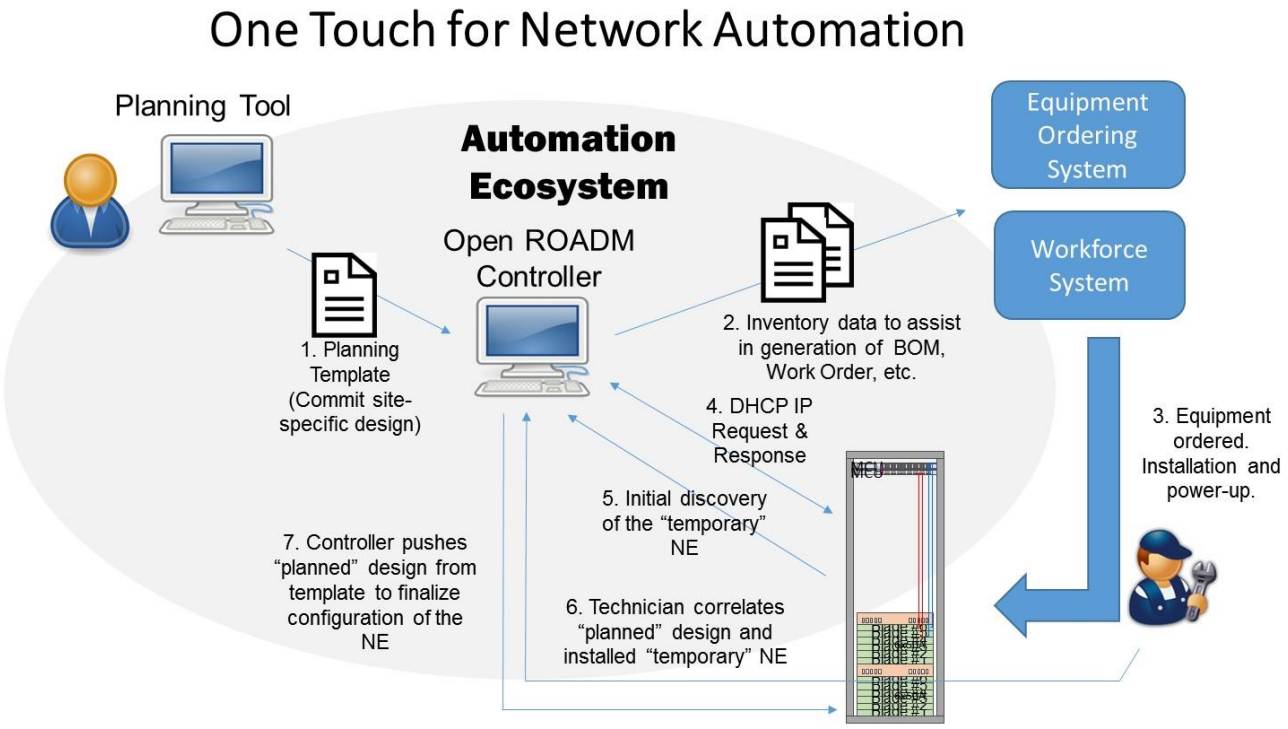
It is recommended that in the event of a large number of changes, that the controller implement a holdoff and a consolidation in order to reduce the number of queries made to the device. MSA members may discuss future enhancements to include the changed data in the notification itself to avoid subsequent queries to the device.

3.3.5 YANG modeling

The Open ROADM data models are based on YANG v1 (RFC 6020). The support for YANG v1.1 [RFC 7950] is under consideration for future versions of Open ROADM.

3.4 Open ROADM Discovery and Commissioning

The Open ROADM network element is provisioned through a one touch procedure to simplify the node commissioning step.

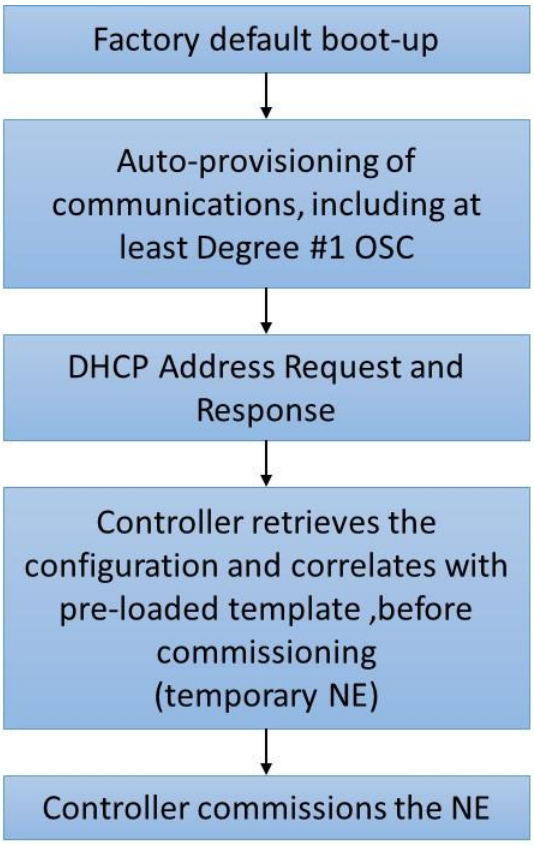


The steps for the commissioning include:

- Step 1: Loading the planning template into the Open ROADM controller (the specific method for loading the template is not standardized by Open ROADM). The planning template is not standardized by Open ROADM but provides data to the controller on how to commission the node using the Open ROADM device model. One possible implementation of the planning template is a JSON file containing a subset of the device model that would be needed to configure the Open ROADM device.
- Step 2: The Open ROADM controller may then generate inventory information that allows external ordering and work force systems to automate the ordering of equipment and installation.
- Step 3: The equipment is ordered. A field technician installs and powers the equipment.
- Step 4: The Open ROADM device initialization, autoprovisioning and IP address request via DHCP. The DHCP server responds with a temporary IP address.
- Step 5: The controller discovers the new IP address assignment by the DHCP server and attempts to connect and login to the device as an Open ROADM NE. If the device is an Open ROADM device, then the controller discovers the Open ROADM NE as a temporary NE.
- Step 6. The field technician provides the correlation between the controller discovered temporary NE and the pre-loaded planning template [One Touch]
- Step 7. The controller then pushes template configuration to the NE and rediscovers the NE (permanent NE)

Note: this section describes an example of a set of operations by an Open ROADM controller to commission a node. The MSA does not specify the operations required by an Open ROADM controller. Instead, it provides an example of how this could be done by a controller. It is possible that a controller could implement the commissioning steps differently than specified in this section.

The states on the network element are shown below:



3.4.1 Pre-planned device template

The controller is loaded with the template for the node that is to be commissioned. The method for loading this template (from planning tool, manually configured, vendor provided, etc.) is not described in this document.

The template contains the information to provision the NE beyond the autoprovisioning behavior. This would include the final node-id, permanent IP address, shelf/circuit-pack/port attributes, etc.

3.4.2 System Initialization and auto-provisioning

Upon system initialization, the Open ROADM NE will default the node-id to 'openroadm'. The default user account is 'openroadm' with password 'openroadm'.

Equipment should autoprovision to the point that communications can be established to the node, either through the external LAN port and/or the internal OSC. Both communication paths (external LAN and internal OSC) should be autoprovisioned as it will not be known which interface the NE will be accessed on.

Note: In general, the Open ROADM device should disable auto-provision with the exception of enabling the OSC for remote communications (minimum would be the OSC associated with degree #1). The reason to disable auto-provisioning is to prevent an accidental or incorrect auto-provisioning if the installer slots an equipment incorrectly. This would then create the entity associated with an incorrect slot that would be difficult to remedy as it is assumed the installer will not have access to the management interface on the Open ROADM device.

For OSC connectivity, it should be sufficient to autoprovision the OSC for degree #1.

For equipment (circuit-packs) provisioned either automatically or manually, it is expected that the ports under the circuit-packs are auto-created by the NE and queryable via NETCONF.

3.4.3 DHCP

The Open ROADM network element initializes with DHCP client enabled. The NE requests both an IPv4 and IPv6 address as it is not known which protocol is in use in the carrier network. Preference is given to IPv6. This is the mechanism to obtain the initial IP address and connectivity to the NE. The method for querying both IPv4 and IPv6 (parallel/serial, interleaved, timing, etc) is device dependent.

Note: It is expected that the operator will later change the DHCP address to a permanent IP address during node commissioning so that the IP address will be stable. Please refer to Section 3.4.7.2 for further details.

3.4.4 Controller discovery of new IP address assignment

The controller discovers the DHCP IP address assignment from the DHCP server. The operator may choose to have the DHCP server coexists with their Open ROADM controller or have it located separately from the controller.

The mechanism for DHCP IP address assignment discovery is outside the scope of the Open ROADM specification.

3.4.5 Initial discovery of the network element

The controller cannot tell if a given DHCP IP address belongs to an Open ROADM network element, or some other IP capable device that supports DHCP (e.g., a laptop connected onto the DCN network). Thus, the controller must first try to connect to this address using NETCONF and see if the device supports the Open ROADM YANG models.

The controller will attempt to connect to the IP address with NETCONF over SSH using port 830. The SSH authentication uses ‘openroadm’ as the username and ‘openroadm’ as the password. If the login is successful, the controller will retrieve the info container to obtain the network element's vendor, model and serial number attributes.

At this time, the node is considered to be in a temporary state.

3.4.6 Correlation of the planned template and the temporary discovered node

The operator in the field connects to the controller to provide the correlation between a planned node in the controller’s inventory and a newly discovered temporary node.

The operator would identify the planned node based on the site (CLLI) and node-id of the node he is installing.

The operator would identify the temporary discovered node based on the vendor, model and serial number of the equipment he installed.

Note: The operator needs to work with the equipment vendor about how to identify the serial number that will be used for correlation.

The operator would then correlate the planned node and discovered node to continue with the node commissioning.

3.4.7 Node commissioning.

Once the correlation has been made, the controller would begin to provision the network element. The template information is pushed to the network element using the NETCONF edit-config RPC with the merge operation. There may be processing involved in the controller, which takes both the template and the current state of the device as input and determines the set of operations that need to be performed on the device.

The use of the merge operation allows the provisioning to succeed even if the entity (e.g., shelf, circuit-pack, port, etc.) was auto-provisioned due to the idempotent behavior.

The controller commissions the network element in the following order based on the Open ROADM device YANG models:

- Initial Info Container Validation
- Set Info container including setting the permanent node-id and IP address
- Shelf
- Circuit-pack, including ports
- Degree
- SRG
- ILA
- Xponder
- Physical Links
- Interface (OAMP and OSC Ethernet, OTS, OMS)
- Protocol (LLDP and RSTP)
- User accounts
- Set date and time

Special handling is done for the user accounts and info container as described below.

3.4.7.1 Initial Validation

The controller will validate the device's vendor and model information in the info container against the template setting to ensure we are provisioning the correct vendor device. This is done before configuring the device. If the vendor and model does not match, then an error is raised and the node commissioning stops.

3.4.7.2 Node-ID, IP Address Provisioning

The first commissioning step is to update the info container, including setting the node-id, IP address (permanent IP address), prefix length and optional gateway.

The syntax for the node-id:

- length "7..20"
- pattern "([a-zA-Z][a-zA-Z0-9-]{5,18}[a-zA-Z0-9])"

When the IP address is changed, this will disable DHCP and remove the temporary IP address. Furthermore, by changing the IP address, the controller will re-establish the NETCONF session towards the permanent IP address.

After logging back into the NE, the device may require a restart using the following NETCONF RPC command:

```
<restart xmlns="http://org/openroadm/de/operations">
  <device>
    <node-id>openroadm</node-id>
  </device>
  <resource/>
  <resourceType>
    <type>device</type>
  </resourceType>
  <option>warm</option>
</restart>
```

Note that the node-id in the command above would be set to the actual node-id of the device that is being asked to be reset.

Following the reboot, the controller will relog back into the NE to continue node commissioning.

3.4.7.3 User account provisioning

The controller will provision the user accounts to be established on the network element, including the accounts (users) name, password and group. Currently only one group is defined called "sudo" that has full access to the NE.

The assumption under Open ROADM is that user security roles are managed at the controller, not on the NE.

The username is a string between 3-32 characters. The first character must be a lowercase letter. The remaining characters can be a lowercase letter or a number. The username may be treated as case insensitive on some devices.

The password is a string between 8-128 characters. Allowed characters in the password field include lowercase and uppercase letters, numbers and the special characters: ! $ % ^ ( ) _ + ~ { } [ ] . -

The following characters are not allowed in the password string: \ | ? # @ &

The default openroadm account will be deleted in a later step.

3.4.7.4 Date and Time setting

The controller will set the date and time directly on the NE. Note: the Open ROADM MSA does not require support for NTP or other time protocols in Open ROADM Version 1.2.1 (nor does the Open ROADM device model support provisioning of the address of time servers).

3.4.7.5 Delete default openroadm account

The controller will log off the NE under the 'openroadm' account, and relog back into the NE using one of the new accounts. The controller will then delete the openroadm account from the NE and close the session.

The reason for this procedure is that you cannot delete an account while connected to the NE using that account.

3.4.8 Permanent Node Discovery

After all commissioning steps are completed, the controller will discover the NE as a permanent NE under a new session.

At this point, the node is considered commissioned and fully discovered.

3.5 OMS Link Planning and Discovery

OMS link data is pushed to the controller using the Open ROADM network model[3]. The OMS link plan would include the near end and far end degree information on the two sides of the link.

The Open ROADM device supports the LLDP protocol over the Ethernet OSC. LLDP is used to support the discovery of the link in the network.

Two key fields are used for the link discovery in the LLDP message:

- SysName (TLV Type 5): contains the NE's node-id
- PortID (TLV Type 2, sub-type 5 interface name or sub-type 7 locally assigned): contains the NE's OSC interface name

Open ROADM implementations may choose to send either the sub-type 5 or sub-type 7 PortID format. Whichever format is selected, the value that is sent should match the Ethernet OSC's interface name.

Open ROADM implementations should be able to process the receipt of either the sub-type 5 or sub-type 7 PortID.

Open ROADM device implementations should support the ability to receive and ignore other LLDP mandatory and optional TLVs that are not directly used by Open ROADM.

An LLDP change notification (lldp-nbr-info-change notification) is issued from the Open ROADM device whenever the neighbor LLDP information is changed. The controller uses this information to correlate the OSC transmit LLDP information with the OSC receive LLDP information to discover the link connectivity.

[3] The Open ROADM network model resides on the Open ROADM controller (or other management system) and not on the device.

The discovered link is compared to the planned OMS link. If they match, then the controller will measure the span loss and take a baseline OTDR reading of the link to complete the OMS link commissioning.

3.6 DEVICE STATE DEFINITIONS AND TRANSITIONS

Three types of states are defined for OpenROADM device resources: administrative-state, operational-state and equipment-state.

Administrative-state is a read-write attribute with three enumerators (inService, outOfService and maintenance) that is used to support maintenance operations and control alarm notifications against a device resource. More specifically,

i. A resource in "inService" administrative-state will forward alarm notifications on the resource.
ii. A resource in "outOfService" administrative-state will suppress alarm notifications on the resource.
iii. A resource is required to be in "maintenance" administrative-state for the device to accept maintenance operations (i.e. loopback, test signal) against the resource.
iv. A device implementation shall support direct transition from one administrative-state to another.
v. A device implementation should not reject any edit or delete operation based on administrative-state value of itself or its parent or children. Controller should assume the responsibility following hierarchical rules to maintain system data structure integrity.

If there is a need for the device to prevent unintended data plane modifications (i.e. edit-config should not disrupt data plane), a future MSA model will have to identify data plane impacting attributes from those having no impact to data plane.

Operational-state is a read-only attribute with three enumerators (inService, outOfService and degraded) defined that reflects the functional status of a resource. A value of "inService" operational state indicates the resource is fully functional while a value of "outOfService" implies the resource fails to perform some or all of its functions. In general, it's expected a resource in outOfService operational state have outstanding alarm/event notification(s) that directly relates to this status. Future MSA release will explore mechanism to make this causal-effect relationship more explicit. The "degraded" operational-state value indicates a resource detected some anomaly with some of its functionality, yet it's not severe enough to trigger "outOfService" status. Some maintenance or path selection applications may consider the "degraded" state value too ambiguous for decision making, so any implementation of this operational state value should have sufficient documentation and justification for downstream users.

Equipment-state serves as a pass-through field for life-cycle management applications. The semantics of this state and application of it is outside the scope of the device model itself.

4 DEVICE SOFTWARE, FIRMWARE AND DATABASE

4.1 SOFTWARE UPGRADE

4.1.1 Manifest file

See Section 9: Appendix A: Manifest file for software download and database operations

4.2 SOFTWARE DOWNGRADE

A software downgrade is not supported in-service in Open ROADM devices.

4.3 FIRMWARE UPGRADE

Firmware upgrade is supported as of Open ROADM Version 2. Firmware is associated with the circuit packs. A device can provide information about the firmware on the circuit pack with the software-load-version attribute and a list of either circuit pack features or components. Implementations may describe their firmware as features, components or both.

The circuit pack features list the firmware as a series of features or capabilities that the firmware enables. For example, a feature could be OTN delay measurement. The feature list provides an indication if the feature is activated or not activated. If the feature is listed as not being activated, then this indicates that there is new firmware available for the circuit pack.

The component list provides the capability to describe the firmware as a series of firmware versions associated with different components on the circuit pack. This list provides the details of the current load version and a version to apply. If the version to apply is different from the current version, then this indicates that a new firmware is available for this circuit pack.

Firmware would be applied automatically to the circuit-pack if the loading of the new firmware is non-service affecting. For firmware that is service affecting, a cold reboot or power cycle of the circuit pack would be required to load the new firmware.

There is also a fw-update RPC that allows the user to manually load the firmware. This command can also be used to force the reload of a firmware into the circuit-pack.

Note that during software download, implementations may not support the upgrade of firmware while the new software is in the validation stage. In this case, firmware download would be supported after the load is committed.

4.4 DATABASE OPERATIONS

The Open ROADM API defines operations against the database, including the ability backup a database to an external store and restore a database from an external store. Also, change notifications are supported against the database to identify configuration changes to the system.

4.4.1 database backup

Database backup operation allows the local device configuration data to be backup into a file and stored in an offboard location. The procedure is to initiate a backup on the NE to store the database as a local file and then transfer the file off the NE into some external SFTP server.

More details on the database backup operation is provided in Section 9.1.2.

4.4.2 database restore

Database restore operation allows the service provider to restore a previously backup database. This may be that there was a device database corruption or configuration issue, and the service provider wants to revert to a previous known state. The procedure is to transfer the backup database file to the device and then initiate the restoration of the database. The NE would then load the database innot persistent memory, reboot, and restart using the saved database.

More details on the database restore operation is provided in Section 9.1.3.

4.4.3 database change notifications

Two types of change notifications are supported: database change notifications, or configuration change notifications, and operational change notifications. Change notifications are issued by the device as a notification of type change-notification.

Change notifications identify the time the change took place, who initiated the change (either the server or the specific user including session information), the datastore that was affected (e.g., running configuration), the operation that was done to the entity, and the target of the changed.

The details of the change itself is not provided by the change-notification. For instance, if an attribute has changed, the notification does not indicate the new value of the attribute. The user or system would need to retrieve the target to see the details of the change.

Change notifications are to be supported for any and all configuration data. Any time a configuration entity is created, modified or deleted, there should be a change-notification for that entity or a parent entity higher in the YANG data tree.

Only limited sets of operational data will result in change notifications. The MSA requires that the following operational data changes result in a change-notification:

For the controller to remain in sync with the NE, there are a few critical notifications of operational data required.

- Equipment plug in/out events (to understand when new equipment is added to the system, or removed from the system). This should result in a change-notification indicating that the physical inventory data has been changed (vendor, model, serial-id, type, product-code, manufacture-date, clei, and hardware-version)
- operational-state attribute change (to know when the operational status of an entity has changed)
- Capabilities change (to know when a capability announcement has been updated)..
    1. As a general rule for capabilities, if the change is attached to another entity and once created with the entity it never changed, then a separate notification is not

required. The change-notification for the parent (or higher) entity would cover the change-notification for the child capability
    - For example:
      pluggable-optics-holder-capability (associated with the parent cp-slot)
      supported-interface-capability (associated with the parent port or circuit-pack)
      port power capabilities (associated with the parent port or circuit-pack)
      port-capabilities (associated with the parent port or circuit-pack)
2. For capabilities, the key ones that require separate notifications would be ones not attached to another entity. This includes:
    - port-group-restrictions
    - connection-map
    - odu-switching-pools

Other critical operational data have specific notifications defined:

- LLDP neighbor change notification (for transport topology) via lldp-nbr-info-change
- RSTP state change notification (for dataplane topology) via rstp-topology-change and rstp-new-root

4.5 SYSLOG

More detail on Syslog will be added in the next version of this Whitepaper

5 PERFORMANCE MONITORING, ALARMS AND TCAS

5.1 PERFORMANCE MONITORING

The device model supports the retrieval of performance monitoring (PM) data. There are two primary lists associated with PM – currentPmlist and historicalPmlist.

5.1.1 Current PM list

The current PM list provides the PM data that is currently being collected and updated on the device. The PM is collected in 15 minute, 24 hour and untimed (granularity = NA) bins.

An example invocation using NETCONF to retrieve the current PM registers with a 15-minute granularity is as follows:

```
<get>
        <filter>
                <currentPmlist xmlns="http://org/openroadm/pm">
                        <currentPm>
                                <granularity>15min</granularity>
                        </currentPm>
                </currentPmlist>
        </filter>
</get>
```

An example invocation using NETCONF to retrieve the current PM registers with a 24-hour granularity is as follows:

```
<get>
        <filter>
                <currentPmlist xmlns="http://org/openroadm/pm">
                        <currentPm>
                                <granularity>24Hour</granularity>
                        </currentPm>
                </currentPmlist>
        </filter>
```

```
</get>
```

5.1.2 Historical PM list

The PM history list contains all the binned PM values collected by the device during some time window in 15-min intervals and 1-day intervals.

An example invocation using NETCONF to retrieve the first bin from the historical PM registers with a 15-minute granularity is as follows:

```
<get>
        <filter>
                <historicalPmlist xmlns="http://org/openroadm/pm">
                        <historicalPm>
                                <id/>
                                <resource/>
                                <layerRate/>
                                <binned-pm>
                                                <bin-number>1</bin-number>
                                </binned-pm>
                                 <granularity>15min</granularity>
                        </historicalPm>
                </historicalPmlist>
        </filter>
</get>
```

An example invocation using NETCONF to retrieve the first bin from the historical PM registers with a 24-hour granularity is as follows:

```
<get>
        <filter>
                <historicalPmlist xmlns="http://org/openroadm/pm">
                        <historicalPm>
                                <id/>
                                <resource/>
                                <layerRate/>
                                <binned-pm>
```

```
                                        <bin-number>1</bin-number>
                                    </binned-pm>
                                    <granularity>24Hour</granularity>
                                </historicalPm>
                            </historicalPmlist>
                        </filter>
                    </get>
```

Note: support for retrieving the historical PM list using a NETCONF get is optional for vendors to support.

5.1.2.1 File-based historical PM retrieval

Implementations must support the ability to retrieve the PM using a file transfer mechanism.

An asynchronous RPC mechanism is defined to support the ability to retrieve the historical PM via a file generated by the device.

The collect-historical-pm-file RPC is used to initiate the creation of the PM file. The input to the RPC indicates the start bin and end bin of the request, along with the granularity. Note that granularity is singular so separate RPC calls would be required to obtain the 15-min, 24-hour and untimed (notApplicable) granularities.

The response to the RPC provides the filename that will be used to store the PM data. The NE auto-generates the PM filename and it is recommended that this filename should be a unique name for every RPC call. This command should return immediately upon validating the RPC command and initiating the PM retrieval and file storage to avoid a timeout on the controller.

In the background, the device would collect the requested PM data and store it in the file. Upon completion of putting the data in the file, the device would issue a notification indicating that the PM file collection has been completed (historical-pm-collect-resulsts). This notification would indicate the success or failure of the operation, and if it failed, it would also provide an error indication in the status-message. The notification will also contain the filename to allow for correlation with the RPC command.

The controller will then transfer the file from the NE using the file transfer RPC. It is the responsibility of the controller to remove the pm file from the device once the file transfer has completed. This will be necessary to avoid exhausting the file store limits if multiple PM files are kept on the device.

5.1.3 PM behavior

5.1.3.1 Analog and Digital PM

PM monitor types can be classified as either analog or digital.

Analog PM are gauge or metered type parameters that can increase or decrease during a collection cycle. These PM monitor types can have an associated raw (instantaneous) reading as well as collect min, max and average statistics from the start of collection within the bin.

An example of an analog PM would be the optical power measurement. E.g., opticalPowerOutput, opticalPowerOutputMin, opticalPowerOutputMax, opticalPowerOutputAvg

Digital PM are counter type parameters that will be non-decreasing within a collection cycle.

Examples of digital PM include Coding Violations, Errored Seconds Section, In Frame, etc.

5.1.3.2 Granularity

The PM model supports three granularities: 15-minute, 24-hour and untimed (notApplicable).

Table 3 shows the requirements that implementations must meet in support of PM granularity.

Table 3: PM Granularity Support

Granularity	**Current/ Historical**	**Analog PM**		Digital PM
		Raw	**Min/Max/Ave**	
Untimed (notApplicable)	**Current**	*Optional*	*Optional*	*Mandatory*
	Historical	*Not Applicable*	*Not Applicable*	*Not Applicable*
15min	**Current**	*Mandatory*	*Mandatory*	*Mandatory*
	Historical	*Optional (snapshot at end of bin)*	*Mandatory*	*Mandatory*
24Hour	**Current**	*Mandatory*	*Mandatory*	*Mandatory*
	Historical	*Optional (snapshot at end of bin)*	*Mandatory*	*Mandatory*

Notes:

1. The analog raw PM would have identical values in the untimed, 15m and 24h current bins.
2. Analog raw PM was made mandatory to allow the full set of raw/min/max/ave to be retrieved together either under the 15m or 24h granularity. Otherwise the PM would have to be retrieved under both the untimed and 15m/24h granularities.
3. Other granularities (such as 1m, 1h) can be optionally supported by implementations. If supported, it is assumed that these granularities would follow the requirements for 15m and 24h granularities.

5.1.3.3 TCAs

TCAs can be set against the PM monitor types. TCAs are supported against both analog and digital PM.

Table 4Table 3 shows the requirements that implementations must meet in support of TCAs.

Table 4: TCA Support

Granularity	**Current/ Historical**	**Analog PM**		Digital PM
		Raw	**Min/Max/Ave**	
Untimed (notApplicable)	**Current**			
15min	**Current**	*TCA – Mandatory (high and low threshold where it makes sense)*		*Mandatory*
24Hour	**Current**			*Mandatory*

For analog PM, TCAs are only supported against the analog raw PM. It is set against the 15-min granularity. The TCA is a transient notification that would be raised when the value was less than the low threshold or greater than the high threshold. Only one transient notification for high and low threshold crossing would be issued per 15m time period. The TCA would be issued again the subsequent 15m time periods if the value was above/below the high/low threshold, or again crossed the threshold.

Analog raw TCAs should be consistent with the min and max readings. If a min reading goes below the low threshold or the max reading goes above the high threshold, then a TCA should have been issued for the analog raw PM.

For digital PM, the threshold can be independently set against the 15m or 24h granularity.

Note: It may be possible that implementations do not support TCAs for all PM monitor types. A future version of this document would clarify which PM monitor types must support TCAs.

5.1.3.4 Validity

The validity field can take the values complete, partial and suspect.

The behavior of the validity depends on the granularity and current/historical list for analog and digital PM:

1. Analog Raw PM – Untimed current, 15m current, 24h current, 15m historical and 24h historical:
    - Complete: Indicates that the current (instantaneous) or historical snapshot reading is valid.
    - Partial: Not used
    - Suspect: Indicates that the current (instantaneous) or historical snapshot reading is not valid
2. Analog Min/Max/Ave PM and Digital PM – untimed current
    - Complete: Indicates that there has been no invalid reading since the PM monitor type was created or last reset
    - Partial: Not used
    - Suspect: Indicates that there has been at least one invalid reading since the PM monitor type was created or last reset.
3. Analog Min/Max/Ave PM and Digital PM – 15m current and 24h current
    - Complete: Not used.
    - Partial: Indicates that PM collection is still occurring for this bin, the collection has been continuously been collected since the start of the bin, and there were no invalid readings in the bin.
    - Suspect: Indicates that the PM collection started after the bin start time, the PM was reset in the current bin, or that there was at least one invalid reading in the bin since the start of collection for the bin
4. Analog Min/Max/Ave PM and Digital PM – 15m current and 24h historical
    - Complete: Indicates that PM was collected for the entire duration of the bin and that there were no invalid readings in the bin.
    - Partial: Not used.
    - Suspect: Indicates that the PM collection did not collect data for the entire duration of the bin or that there was at least one invalid reading in the bin since the start of collection for the bin

When data is not available to be read, whether temporary or permanent, then the validity would be shown as suspect. This could be the case when PM is collected on remote modules and the device is unable to communicate with that module.

5.1.3.5 Other PM behavior notes

For analog raw historical PM data, the value represents the instantaneous value at the end of the bin.

The bin start and stop times are referenced to UTC time on the device. Note that Open ROADM only supports UTC time.

- For the 15m bins, the start time would be at xx:00:00, xx:15:00, xx:30:00, and xx:45:00 UTC. The end times would be xx:14:59, xx:29:59, xx:44:59, and xx:59:59 UTC, respectively.
- For the 24h bins, the start time would be at 00:00:00 UTC and the end time would be at 23:59:59 UTC.

When retrieving data from the current bin, the retrievalTime represents the time of the retrieval at which the current PM data is retrieved and returned.

For historical bin data, only the end time is reported (completionTime) which represents the end time of the bin.

Partial only applies to current binned data to indicate the bin data is valid but hasn't completed the full collection cycle for the bin. When a new bin is started, the validity resets to partial and will either remain partial or transition to suspect once data begins collection. If the bin is still partial at the end of the current bin, then it transitions to complete when propagating into the historical bin.

Once a bin goes suspect, it would typically always be suspect unless the bin is re-initialized (untimed granularity only). Note: implementations that support a concept of "data temporarily not available" due to a temporary access issue (e.g., to a remote circuit-pack) could report the data as suspect and then back to partial/complete once the access issue is resolved.

Changing date/time that could cause a bin to be long or short is reported as suspect. This affects the 15m and 24h current PM data for analog min/max/ave and digital PM. The marking of data as suspect due to long or short may be performed at the end of the bin before rolling into into the historical list. This would allow implementations to check the total collection time to see if the data was long or short at the bin end time.

5.1.4 PM Future Enhancements

The following features are not supported in v2.2 but are enhancements that may be considered for a future release:

- A configurable option to enable/disable PM event forwarding for any specific resource or group of resources under a common resource node
- A configurable option to enable/disable PM event forwarding for any specific or combination of fault type (s) and time intervals (under pm node);
- RPCs used to support configuration of PM event forwarding and PM event sync (after failed communication to northbound is restored or on-demand).

5.1.5 PM Tables

MSA PM	MW Wr OSC	Reported Against…	Used by Controller	Tide Marks	Open ROADM yang version 1.2.1		Comment
					Type	extension	
Optical Power Output (OPOUT-OTS)	MW	OTS-IF	x		opticalPowerOutput		total optical signal power with OSC, includes VOA attenuation
				min	vendorExtension	opticalPowerOutputMin	
				max	vendorExtension	opticalPowerOutputMax	
				avg	vendorExtension	opticalPowerOutputAvg	
Optical Power Input (OPIN-OTS)	MW	OTS-IF	x		opticalPowerInput		total optical signal power with OSC
				min	vendorExtension	opticalPowerInputMin	
				max	vendorExtension	opticalPowerInputMax	
				avg	vendorExtension	opticalPowerInputAvg	
Optical Power Output (OPOUT-OMS)	MW	OMS-IF			opticalPowerOutput		total optical signal power without OSC, includes VOA attenuation
				min	vendorExtension	opticalPowerOutputMin	
				max	vendorExtension	opticalPowerOutputMax	
				avg	vendorExtension	opticalPowerOutputAvg	
Optical Power Input (OPIN-OMS)	MW	OMS-IF			opticalPowerInput		total optical signal power without OSC
				min	vendorExtension	opticalPowerInputMin	
				max	vendorExtension	opticalPowerInputMax	
				avg	vendorExtension	opticalPowerInputAvg	

MSA PM	MW Wr OSC	Reported Against…	Used by Controller	Tide Marks	Open ROADM yang version 1.2.1		Comment
					Type	extension	
Optical Return Loss (ORL-OTS)	MW	OTS-IF on tx port	x		opticalReturnLoss		at MW port(s) B (including OSC reflection) No tide markings required (e.g. min, max, avg) Either ORL-OTS or ORL-OMS is supported depending upon vendor implementation. direction, if included is rx (not tx)
Optical Return Loss (ORL-OMS)	MW	OMS-IF on tx port	x		opticalReturnLoss		at MW port(s) B (not including OSC reflection) No tide markings required (e.g. min, max, avg) Either ORL-OTS or ORL-OMS is supported depending upon vendor implementation. direction, if included is rx (not tx)
OSC Optical	MW	OTS-IF			vendorExtension	opticalPowerOutputOSC	OSC Transmit

MSA PM	MW Wr OSC	Reported Against…	Used by Controller	Tide Marks	Open ROADM yang version 1.2.1		Comment
					Type	extension	
Power Transmit (OPT-OSC)							power on MW port
				min	vendorExtension	opticalPowerOutputOSCMin	
				max	vendorExtension	opticalPowerOutputOSCMax	
				avg	vendorExtension	opticalPowerOutputOSCAvg	
OSC Optical Power Receive (OPR-OSC)	MW	OTS-IF			vendorExtension	opticalPowerInputOSC	OSC Receive power on MW port
				min	vendorExtension	opticalPowerInputOSCMin	
				max	vendorExtension	opticalPowerInputOSCMax	
				avg	vendorExtension	opticalPowerInputOSCAvg	
Optical Channel Power Transmit (OPT-OCH)	MW	OCH-IF	x		opticalPowerOutput		individual optical signal power on MW port
				min	vendorExtension	opticalPowerOutputMin	
				max	vendorExtension	opticalPowerOutputMax	
				avg	vendorExtension	opticalPowerOutputAvg	
Optical Channel Power Receive (OPR-OCH)	MW	OCH-IF			opticalPowerInput		individual optical signal power on MW port, includes VOA attenuation
				min	vendorExtension	opticalPowerInputMin	
				max	vendorExtension	opticalPowerInputMax	
				avg	vendorExtension	opticalPowerInputAvg	
Optical Power Receive (OPR)	Wr	SRG client port	x		opticalPowerInput		optical total signal power on Wr port (from transponder) –

MSA PM	MW Wr OSC	Reported Against…	Used by Controller	Tide Marks	Open ROADM yang version 1.2.1		Comment
					Type	extension	
							single wavelength
				min	vendorExtension	opticalPowerInputMin	
				max	vendorExtension	opticalPowerInputMax	
				avg	vendorExtension	opticalPowerInputAvg	
Optical Power Transmit (OPT)	Wr	SRG client port			opticalPowerOutput		optical total signal power on Wr port (to transponder) – multiple wavelengths
				min	vendorExtension	opticalPowerOutputMin	
				max	vendorExtension	opticalPowerOutputMax	
				avg	vendorExtension	opticalPowerOutputAvg	
Code Violations (CV-PCS)	OSC	Eth-IF			codeViolations		IEEE 802.3ah, Section 45.2.1.44; 8B/10B errors
Errored Seconds Section (ES-PCS)	OSC	Eth-IF			erroredSecondsSection		IEEE 802.3ah, Section 45.2.1.46
Severely Errored Seconds (SES-PCS)	OSC	Eth-IF			severlyErrorSeconds *(typo)*		IEEE 802.3ah, Section 45.2.1.48 *(typo will be fixed in next release)*
Unavailable Seconds PCS (UAS-PCS)	OSC	Eth-IF			unavailableSecondsPCS		IEEE 802.3ah, Section 45.2.1.52

MSA PM	MW Wr OSC	Reported Against…	Used by Controller	Tide Marks	Open ROADM yang version 1.2.1		Comment
					Type	extension	
In frames (INFRAMES-E)	OSC	Eth-IF			inFrames		
In frames errored (INFRAMESERR-E)	OSC	Eth-IF			inFramesErrored		
Out frames (OUTFRAMES-E)	OSC	Eth-IF			outFrames		
Errored Seconds Ethernet (ES-E)	OSC	Eth-IF			erroredSecondsEthernet		IEEE 802.3ah, Section 45.2.1.46
Severely Errored Seconds Ethernet (SES-E)	OSC	Eth-IF			severelyErroredSecondsEthernet		IEEE 802.3ah, Section 45.2.1.48
Unavailable Seconds Ethernet (UAS-E)	OSC	Eth-IF			unavailableSecondsEthernet		IEEE 802.3ah, Section 45.2.1.52
OSC Optical Power Receive (OPR-OSC)	OSC	port			opticalPowerInput		OSC Receive power on OSC port
				min	vendorExtension	opticalPowerInputMin	
				max	vendorExtension	opticalPowerInputMax	
				avg	vendorExtension	opticalPowerInputAvg	
OSC Optical Power Transmit (OPT-OSC)	OSC	port			opticalPowerOutput		OSC Transmit power on OSC port
				min	vendorExtension	opticalPowerOutputMin	
				max	vendorExtension	opticalPowerOutputMax	
				avg	vendorExtension	opticalPowerOutputAvg	

MSA PM	MW Wr OSC	Reported Against…	Used by Controller	Tide Marks	Open ROADM yang version 1.2.1		Comment
					Type	extension	
Current Output Power	W	port			opticalPowerOutput		Single wavelength to Wr
				min	vendorExtension	opticalPowerOutputMin	
				max	vendorExtension	opticalPowerOutputMax	
				avg	vendorExtension	opticalPowerOutputAvg	
Current Input Power	W	OCH-IF			opticalPowerInput		Single tuned wavelength
				min	vendorExtension	opticalPowerInputMin	
				max	vendorExtension	opticalPowerInputMax	
				avg	vendorExtension	opticalPowerInputAvg	
Total Input Power	W	port			totalOpticalPowerInput		Multiple wavelengths from Wr
				min	vendorExtension	totalOpticalPowerInputMin	
				max	vendorExtension	totalOpticalPowerInputMax	
				avg	vendorExtension	totalOpticalPowerInputAvg	
pFECcorrErr	W	OTU-IF on transponder network RX port	x		vendorExtension	preFECCorrectedErrors	Ref) G798 : 6.5.1.3
RX_ Forward Error Correction Correctable Blocks	W	OTU-IF			vendorExtension	FECCorrectableBlocks	count of corrected FEC Blocks direction = rx
RX_ Forward Error Correction Uncorrectable Blocks	W	OTU-IF			vendorExtension	FECUncorrectableBlocks	count of uncorrected FEC Blocks direction = rx

MSA PM	MW Wr OSC	Reported Against…	Used by Controller	Tide Marks	Open ROADM yang version 1.2.1		Comment
					Type	extension	
pN_EBC (BIP-8)	W	OTU-IF ODU-IF			erroredBlockCount		Ref) G.798 : 6.5.1.1 location = nearEnd
	W	ODU-IF			vendorExtension	erroredBlockCountTCM1	
	W	ODU-IF			vendorExtension	erroredBlockCountTCM2	
	W	ODU-IF			vendorExtension	erroredBlockCountTCM3	
	W	ODU-IF			vendorExtension	erroredBlockCountTCM4	
	W	ODU-IF			vendorExtension	erroredBlockCountTCM5	
	W	ODU-IF			vendorExtension	erroredBlockCountTCM6	
pF_EBC (BEI)	W	OTU-IF ODU-IF			erroredBlockCount		Ref) G.798 : 6.5.1.1 location = farEnd
	W	ODU-IF			vendorExtension	erroredBlockCountTCM1	
	W	ODU-IF			vendorExtension	erroredBlockCountTCM2	
	W	ODU-IF			vendorExtension	erroredBlockCountTCM3	
	W	ODU-IF			vendorExtension	erroredBlockCountTCM4	
	W	ODU-IF			vendorExtension	erroredBlockCountTCM5	
	W	ODU-IF			vendorExtension	erroredBlockCountTCM6	
pN_delay	W	ODU-IF			delay		Ref) G798 : number of frames between a DMValue toggle event and the received DMp signal value toggle event location = nearEnd
	W	ODU-IF			vendorExtension	delayTCM1	
	W	ODU-IF			vendorExtension	delayTCM1	
	W	ODU-IF			vendorExtension	delayTCM1	
	W	ODU-IF			vendorExtension	delayTCM1	
	W	ODU-IF			vendorExtension	delayTCM1	
	W	ODU-IF			vendorExtension	delayTCM1	
RX_ Errored blocks	W	ETH-IF			erroredBlockCount		Ref) IEEE 802.3ba Section

MSA PM	MW Wr OSC	Reported Against…	Used by Controller	Tide Marks	Open ROADM yang version 1.2.1		Comment
					Type	extension	
							45.2.3.12.4 direction = rx
RX_ BIP error counter	W	ETH-IF			vendorExtension	BIPErrorCounter	Ref) IEEE 802.3ba Section 45.2.3.36 Ref) IEEE 802.3ba Section 45.2.3.37 direction = rx
TX_ Errored blocks	W	ETH-IF			erroredBlockCount		Ref) IEEE 802.3ba Section 45.2.3.12.4 direction = tx
TX_ BIP error counter	W	ETH-IF			vendorExtension	BIPErrorCounter	Sum of each lane is notified in FW9500. Ref) IEEE 802.3ba Section 45.2.3.36 Ref) IEEE 802.3ba Section 45.2.3.37 direction = tx

5.2 ALARMS

MSA Alarm	comment	cause	extension	resource resourceType	severity	direction *	location *
W Port Alarms							
OTU - LOS 100GE Signal Det	OTN : It is shown that the input power drop. Ethernet : Ref) IEEE 802.3ba Section 86.2 Physical maps to port level LOS Ethernet - linkDown at eth-if, LOS at sub-ports; LOS at parent port if all sub-ports LOS. (sub-port LOS is suppressed when port level LOS is raised. sub-port LOS is cleared and not retrievable when port level LOS is raised.)	portLossOfLight		port			
	OCH LOS at transponder only (specific to tuned wavelength); Reported on both the network and client och;	lossOfSignal		och-if			

	the network och alarm is used by the controller..)						
		linkDown		eth-if			
LOF	Ref) G798 : 6.2.5.1	lossOfFrame		otu-if			
LOM	Ref) G798 : 6.2.5.2	lossOfMultiframe		otu-if			
BDI	Ref) G798:6.2.6.6	backwardsDefectIndication		otu-if odu-if			
		backwardsDefectIndicationTCM1		odu-if			
		backwardsDefectIndicationTCM2		odu-if			
		backwardsDefectIndicationTCM3		odu-if			
		backwardsDefectIndicationTCM4		odu-if			
		backwardsDefectIndicationTCM5		odu-if			
		backwardsDefectIndicationTCM6		odu-if			
DEG	Ref) G798:6.2.3 *(typo will be fixed in next release)*	degradedEffect *(typo)*		otu-if odu-if			
		degradedDefectTCM1		odu-if			
		degradedDefectTCM2		odu-if			
		degradedDefectTCM3		odu-if			
		degradedDefectTCM4		odu-if			
		degradedDefectTCM5		odu-if			
		degradedDefectTCM6		odu-if			
BIAE	Ref) G798:6.2.6.11 * indicates non-alarmed standing condition in the MSA. Non-alarmed is being	backwardIncomingAlignmentError		otu-if	warning		FEND

	interpreted to mean "warning" severity.	backwardIncomingAlignmentErrorTCM1		odu-if	warning		FEND
		backwardIncomingAlignmentErrorTCM2		odu-if	warning		FEND
		backwardIncomingAlignmentErrorTCM3		odu-if	warning		FEND
		backwardIncomingAlignmentErrorTCM4		odu-if	warning		FEND
		backwardIncomingAlignmentErrorTCM5		odu-if	warning		FEND
		backwardIncomingAlignmentErrorTCM6		odu-if	warning		FEND
IAE	Ref) G798:6.2.6.10 * indicates non-alarmed standing condition in the MSA. Non-alarmed is being interpreted to mean "warning" severity.	incomingAlignmentError		otu-if	warning		FEND
		incomingAlignmentErrorTCM1		odu-if	warning		FEND
		incomingAlignmentErrorTCM2		odu-if	warning		FEND
		incomingAlignmentErrorTCM3		odu-if	warning		FEND
		incomingAlignmentErrorTCM4		odu-if	warning		FEND
		incomingAlignmentErrorTCM5		odu-if	warning		FEND
		incomingAlignmentErrorTCM6		odu-if	warning		FEND
TTI	Ref) G798:6.2.2.1	trailTraceIdentifierMismatch		otu-if odu-if			
		trailTraceIdentifierMisMatchTCM1		odu-if			
		trailTraceIdentifierMisMatchTCM2		odu-if			
		trailTraceIdentifierMisMatchTCM3		odu-if			
		trailTraceIdentifierMisMatchTCM4		odu-if			
		trailTraceIdentifierMisMatchTCM5		odu-if			
		trailTraceIdentifierMisMatchTCM6		odu-if			
AIS	Ref) G798:6.2.6.3.2	alarmIndicationSignal		otu-if odu-if			
		alarmIndicationSignalTCM1		odu-if			
		alarmIndicationSignalTCM2		odu-if			

		alarmIndicationSignalTCM3		odu-if			
		alarmIndicationSignalTCM4		odu-if			
		alarmIndicationSignalTCM5		odu-if			
		alarmIndicationSignalTCM6		odu-if			
OCI	Ref) G798:6.2.6.8	openConnectionIndication		odu-if			
		openConnectionIndicationTCM1		odu-if			
		openConnectionIndicationTCM2		odu-if			
		openConnectionIndicationTCM3		odu-if			
		openConnectionIndicationTCM4		odu-if			
		openConnectionIndicationTCM5		odu-if			
		openConnectionIndicationTCM6		odu-if			
LCK	Ref) G798:6.2.6.9	lockedDefect		odu-if			
		lockedDefectTCM1		odu-if			
		lockedDefectTCM2		odu-if			
		lockedDefectTCM3		odu-if			
		lockedDefectTCM4		odu-if			
		lockedDefectTCM5		odu-if			
		lockedDefectTCM6		odu-if			
LTC	Ref) G798:6.2.1.4	lossofTandemConnectionTCM1		odu-if			
		lossofTandemConnectionTCM2		odu-if			
		lossofTandemConnectionTCM3		odu-if			
		lossofTandemConnectionTCM4		odu-if			
		lossofTandemConnectionTCM5		odu-if			
		lossofTandemConnectionTCM6		odu-if			
PLM	Ref) G798 : 6.2.4.1	payloadMismatch		opu-if			
CSF	Ref) G798:6.2.10	clientSignalFailDefect		opu-if			
Loss of FEC Alignment	Only BjFEC=ON (i.e 100GBASE-SR4) Ref) IEEE 802.3bj Section 45.2.1.92b.2	lossOfFECAlignment		eth-if			

Loss of Sync (Rcv)	Ref) IEEE 802.3ba Section 45.2.3.11.5	lossOfSynchronization		eth-if		rx	
Rx_BASE-R PCS high BER	Ref) IEEE 802.3ba Section 45.2.3.11.4	highBER		eth-if		rx	
Loss of Alignment (Rcv)	OR of each lane is notified. Ref) IEEE 802.3ba Section 45.2.3.16c/d/e/f	lossOfAlignment		eth-if		rx	
Rx_Local Fault	Ref) IEEE 802.3ba Section 81.3.4	localFault		eth-if		rx	
Rx_Remote Fault	Ref) IEEE 802.3ba Section 81.3.4	remoteFault		eth-if		rx	
Loss of Sync (Tx)	Ref) IEEE 802.3ba Section 45.2.3.11.5	lossOfSynchronization		eth-if		tx	
Tx_BASE-R PCS high BER	Ref) IEEE 802.3ba Section 45.2.3.11.4	highBER		eth-if		tx	
Loss of Alignment (Tx)	OR of each lane is notified. Ref) IEEE 802.3ba Section 45.2.3.16c/d/e/f	lossOfAlignment		eth-if		tx	
Tx_Local Fault	Ref) IEEE 802.3ba Section 81.3.4	localFault		eth-if		tx	
Tx_Remote Fault	Ref) IEEE 802.3ba Section 81.3.4	remoteFault		eth-if		tx	
MW Mux (Out) Ports							
Automatic Shutoff/ Automatic Laser Shutdown	shut off amplifier laser due to safety, OSC will never be shut off	automaticLaserShutdown		oms-if			

Automatic Shutoff Disabled	RPC was triggered to shut off laser safety, standing alarm	automaticShutoffDisabled		oms-if			
Automatic Power Reduction Active (optional)	during high reflection event, power is lowered, but not shut off	vendorExtension	automaticPowerReduction	port			
High Reflection / Low Optical Return Loss	reflectionTooHigh on ots-if is used by the controller Reported on ots-if or oms-if based on vendor implementation (ots-if if OSC is included in the measurement; oms-if, otherwise.	reflectionTooHigh		ots-if or oms-if			
Loss of Signal	May be used by the controller; Not masked since there is no LOS on OTS (tx); Optional for device to report	lossOfSignal		och-if		tx	
MW Demux (In) Ports							

Loss of Signal	LOS on OTS includes OSC and OMS; LOS on OTS masks LOS on OMS but does not mask LOS on OSC and OMS LOS on OTS is used by the controller.	lossOfSignal		ots-if		rx	
	LOS on OMS	lossOfSignal		oms-if		rx	
	LOS on OSC	vendorExtension	lossOfSignalOSC	ots-if		rx	
	LOS on OCH Masked by LOS on OTS (rx); Optional for device to report	lossOfSignal		och-if		rx	
Link Down	Link Down on OSC	linkDown		eth-if		rx	
Low Optical Power Warning (Optical Power Degraded)		opticalPowerDegraded		ots-if		rx	

Optical Line Fail	combines OMS LOS and linkDown on OSC into one alarm; Optical Line Fail masks OMS LOS but does not mask linkDown on OSC and Ethernet linkdown	opticalLineFail		otsoms-if		rx	NEND
Wr Mux (In) Ports							
Loss of Signal		portLossOfLight		port		rx	
Optical Power Degraded		opticalPowerDegraded		port		rx	
The remaining causes do not map to any MSA Alarms currently							
		administrativeDown					
		certificateNotInstalled					
		databaseCorruption					
		databaseLockedDbAlarmPresent					
		databaseLockedDbRestoreInProgress					
	To be removed in version 2.0	databaseLockedIlfViolation					
		databaseLockedShelfProvModePresent					
		databaseLockedSoftwareUpgradeInProgress					

	To be removed in version 2.0	databaseLockedSysInitInProgress					
		databaseVersionMismatch					
		equipmentFault					
		equipmentInterConnectFailure					
		equipmentLedOn					
		equipmentMiscabledConnection					
		equipmentMismatch					
		equipmentRemoved					
		equipmentWarmup					
		facilityLoopback2Active					
		facilityLoopbackActive					
		facilityTestsignalActive					
		fanCoolingFail					
		firmwareBackwardCompatibleAll					
		firmwareBackwardCompatibleLimited					
		firmwareDownloadOrActivationFailure					
		firmwareInitInProgress					
		firmwareVersionMismatch					
		forwardDefectIndication					
	To be removed in version 2.0	ilfViolationCritical					
	To be removed in version 2.0	ilfViolationMajor					
		incompatibleFirmware					
		lampTest					
		oscPowerOutOfSpecificationHigh					
		oscPowerOutOfSpecificationLow					
		otsSpanlossPowerOutOfSpecificationHigh					
	To be removed in	payloadMissingIndication					

	version 2.0						
	To be removed in version 2.0	postBlockSpanAdjustmentInProgress					
	To be removed in version 2.0	postBlockManualLaserShutdown					
		powerOutOfSpecificationHigh					
		powerProblemA					
		powerProblemB					
	To be removed in version 2.0	preBlockSpanAdjustmentInProgress					
		serverSignalFail					
		shelfProvisioningMode					
		softwareReset					
		softwareStageInProgress					
		softwareVersionMismatch					
		sysNameChanged					
		sysNtpNotSynchronized					
		terminalLoopbackActive					
		terminalTestsignalActive					

5.3 THRESHOLD CROSSING ALARMS (TCAS)

TCAs are supported in Open ROADM Version 2.

- Potential TCA List provides the list of TCAs supported per resource
  - Thresholds are set using the potential TCA list
  - Threshold values are absolute threshold values. Relative thresholds are not supported on the device in this version of the MSA
  - Low and High thresholds are supported. Note that not all PM types will support both a high and low threshold. For example, counter type PMs (digital PMs) will only support a high threshold.
- TCA notifications provide the notification when a high or low threshold is crossed.
  - TCA notifications are transient in nature (no persistent state on threshold crossing and no clear notifications are sent).

6 DEVICE SPECIFIC TOPICS

6.1 ROADM

6.1.1 Flexible grid

The Open ROADM model has been extended to support flexible grid capabilities using MC (Media Channel) and NMC (Network Media Channel) constructs to allow for:

- Multiple NMC per MC
- Mix of fixed grid and flexible grid capable ROADM hardware
- Mix of media channel widths

6.1.1.1 Media Channel Capabilities

For each Degree and SRG, the ROADM node advertises mc-capabilities which consist of:

```
|  +--ro mc-capabilities
|     +--ro slot-width-granularity?    org-openroadm-common-types:frequency-GHz
|     +--ro center-freq-granularity?   org-openroadm-common-types:frequency-GHz
|     +--ro min-slots?                 uint32
|     +--ro max-slots?                 uint32
```

The following table shows sample mc-capabilities for fixed grid and flexible grid capable ROADM nodes:

Node Type	slot-width-granularity	center-freq-granularity	min-slots	max-slots
Fixed grid	50 GHz	50 GHz	1	1
Flexible grid	12.5 GHz	6.25 GHz	3	384
Flexible grid	6.25 GHz	3.125 GHz	6	768

These figures consider a total frequency range of 4800 GHz from 191.325 THz to 196.125 THz, corresponding to 96 channels for the Fixed Grid case.

These parameters are used by the controller to determine the wavelength provisioning flexibility that is supported in each degree and SRG within the ROADM node.

6.1.1.2 Media Channel Interfaces (mc-ttp)

Media channels are terminated at every degree of a ROADM with a trail termination point, MC-TTP, which allows access to the NMC(s) contained within the MC. Media channels are not terminated / modeled on line amplifier nodes. For ease of operations, the Open ROADM model is restricted to the same MC size over a WDM span. MC-TTP interfaces are defined as follows:

```
       |  +--rw org-openroadm-media-channel-interfaces:mc-ttp
       |  |  +--rw org-openroadm-media-channel-interfaces:min-freq?      org-
openroadm-common-types:frequency-THz
       |  |  +--rw org-openroadm-media-channel-interfaces:max-freq?      org-
openroadm-common-types:frequency-THz
       |  |  +--ro org-openroadm-media-channel-interfaces:center-freq?   org-
openroadm-common-types:frequency-THz
       |  |  +--ro org-openroadm-media-channel-interfaces:slot-width?    org-
openroadm-common-types:frequency-GHz
```

MC-TTP are a child of a degree port. A degree port can have one or more MC-TTPs and the frequency ranges of MC on a degree port cannot overlap.

Media channels interfaces are created by provisioning the min/max frequency (THz). These are determine based on the MC capabilities provided by the ROADM. The MC must include the bandwidth of the NMC(s) and guard bands (dead bands) at the extremities (4 GHz).

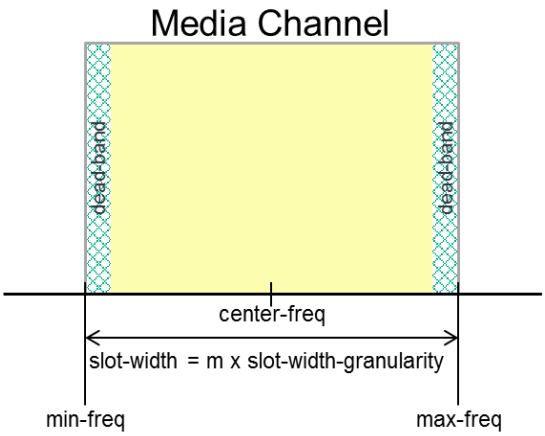
min-freq (THz) = 193.1 + (center-freq-granularity * *n* - slot-width-granularity * *m* / 2) / 1000
max-freq (THz) = 193.1 + (center-freq-granularity * *n* + slot-width-granularity * *m* / 2) / 1000

where: *min-slots* $\leqslant m \leqslant$ *max-slots* and *n* is a positive or negative integer including 0.

The value of *n* should be selected in order to guarantee for each media channel that the minimum frequency is $\geqslant$ 191.325 THz and the maximum frequency $\leqslant$ 196.125 THz. In particular for the Fixed Grid case - 35 $\leqslant n \leqslant$ *60;* while for the Flex Grid cases *n* depends on the number and frequency width of the provisioned MCs.

Slot width and center frequency being read only data are calculated according to these formulas:

slot-width (GHz) = slot-width-granularity * *m*
center-freq (THz) = 193.1 + center-freq-granularity / 1000 * *n*



For the 100G fixed grid services upgraded from v1 to v2 or created in v2, the mc-ttp attributes would be set as follows:

- min-freq (THz) = 193.1 + (50 * n - 25) / 1000
- max-freq (THz) = 193.1 + (50 * n + 25) / 1000
- center-freq (THz) = 193.1 + 50 / 1000 * *n*
- slot-width = 50 GHz
- where: - 35 $\leqslant n \leqslant 60$.

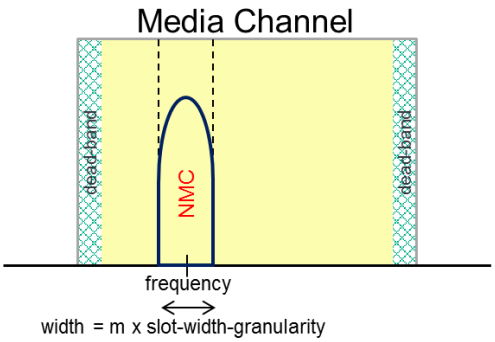
6.1.1.3 Network Media Channel Interfaces (nmc-ctp)

ROADM cross connections are done between NMC connection termination points, NMC-CTP. NMC-CTP interfaces are defined as follows:

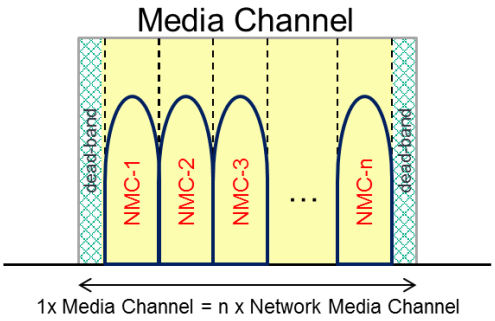
```
       |  +--rw org-openroadm-network-media-channel-interfaces:nmc-ctp
       |  |  +--rw org-openroadm-network-media-channel-interfaces:frequency?   org-
openroadm-common-types:frequency-THz
       |  |  +--rw org-openroadm-network-media-channel-interfaces:width?       org-
openroadm-common-types:frequency-GHz
```

width (GHz) = slot-width-granularity * *m – 4*2*, where *min-slots* $\leqslant m \leqslant$ *max-slots and 4 GHz are considered for the dead bands*

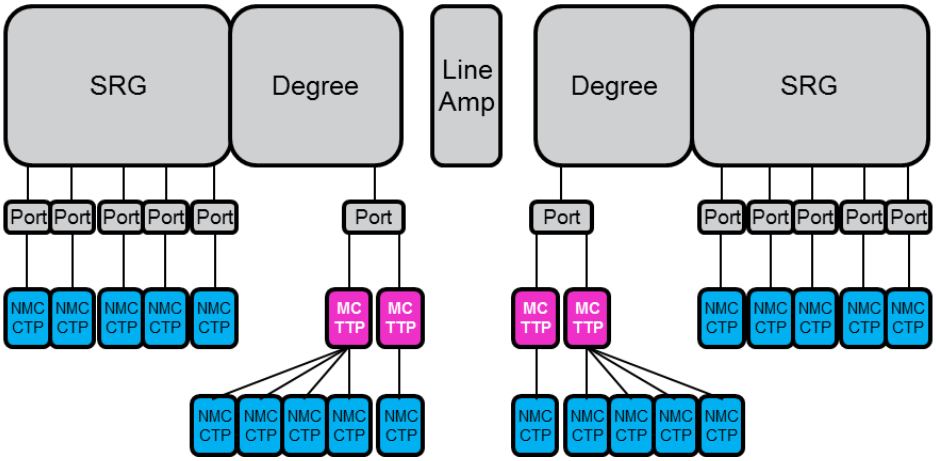
frequency (THz) = 193.1 + center-freq-granularity / 1000 * *n*, where *n is a positive or negative integer including 0.*



NMC-CTP are a child of a MC-TTP or a child of a SRG port. An SRG port can have only one NMC-CTP while an MC-TTP can have one or more NMC-CTP. NMCs within an "MC" must not overlap and no dead-bands are needed between adjacent NMCs.



The following figure shows the relationship of the MC-TTP and NMC-CTP to each other and the ports of a degree and SRG.



NMC-CTP are only used by the ROADM device. XPDR devices will continue to use OCh and center frequency and signal bandwidth are not regulated by the slot-width-granularity or center-freq-granularity. The OCh width (ie. signal bandwidth + modulation guardbands) must be ≤ the NMC-width.

For the 100G fixed grid services upgraded from v1 to v2 or created in v2, the nmc-ctp attributes would be set as follows:

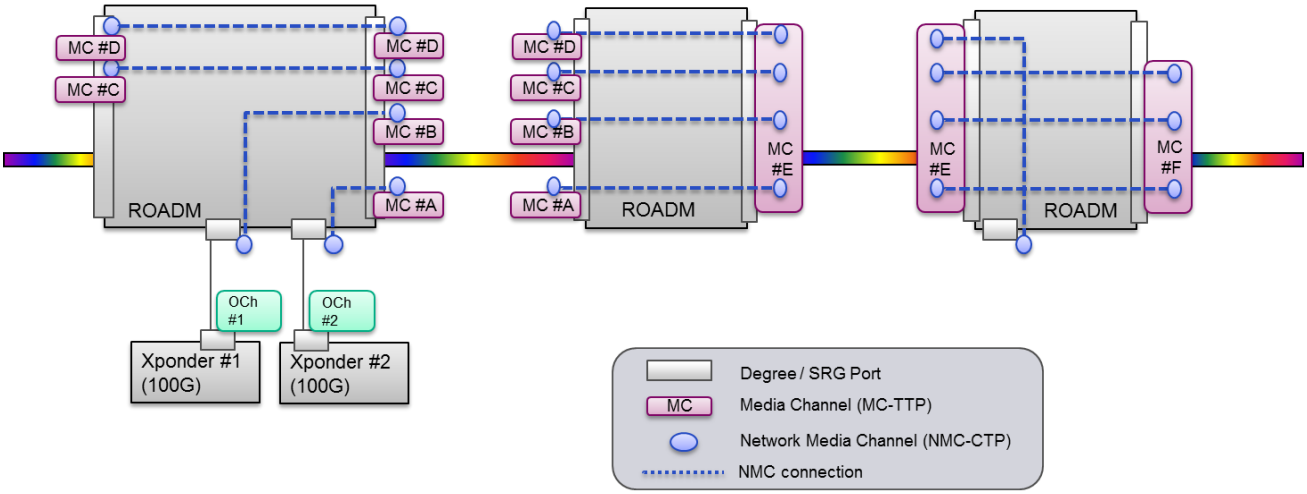
- frequency (THz) = 193.1 + 50 / 1000 * n
- width = 40 GHz (Note: it was decided to use 40 GHz for the width instead of 42 GHz that the formula listed above would suggest for the fixed grid channels from v1.2.1)
- where: - 35 $\leqslant n \leqslant 60$.

Note that for the OCH settings on transponders, the OCH frequency and width would have the same values as the nmc-ctp (e.g., OCH width = 40 GHz).

6.1.1.4 ROADM Cross Connections

Instead of using OCh to OCh connections, the flexible grid model uses NMC-CTP to NMC-CTP connections to create ROADM cross connects. When cross connecting NMC-CTPs, the frequency and width must match.

```
+--rw roadm-connections* [connection-name]
|  +--rw connection-name         string
|  +--rw opticalControlMode?     org-openroadm-common-types:optical-control-mode
|  +--rw target-output-power?    org-openroadm-common-types:power-dBm
|  +--rw source
|  |  +--rw src-if     -> /org-openroadm-device/interface/name
|  +--rw destination
|     +--rw dst-if     -> /org-openroadm-device/interface/name
```



The ROADM device does not model NMC groups (NMCG). The SDN controller is responsible for maintaining NMC groups and ensure all NMC's within the group are routed together.

6.1.2 OTDR scan

6.1.3 Interfaces

also need to talk about GitHub issue 108, MPO modelling

<< Need to add the hierarchy for interfaces on the ROADM

6.2 TRANSPONDER

6.2.1 Interfaces

<< Need to add the hierarchy for interfaces on the ROADM

6.3 PLUGGABLE

In V12, no pluggable stand-alone device (e.g. WDM line side pluggable directly in a router) exists yet in the device model. This will be tackled in V4 and beyond.

6.4 Modeling Notes

6.4.1 Circuit pack and ports uniqueness and restriction

The circuit pack's circuit-pack-name is unique within the node. But the port's port-name under the circuit-pack are unique only within the context of the circuit pack's circuit-pack-name.

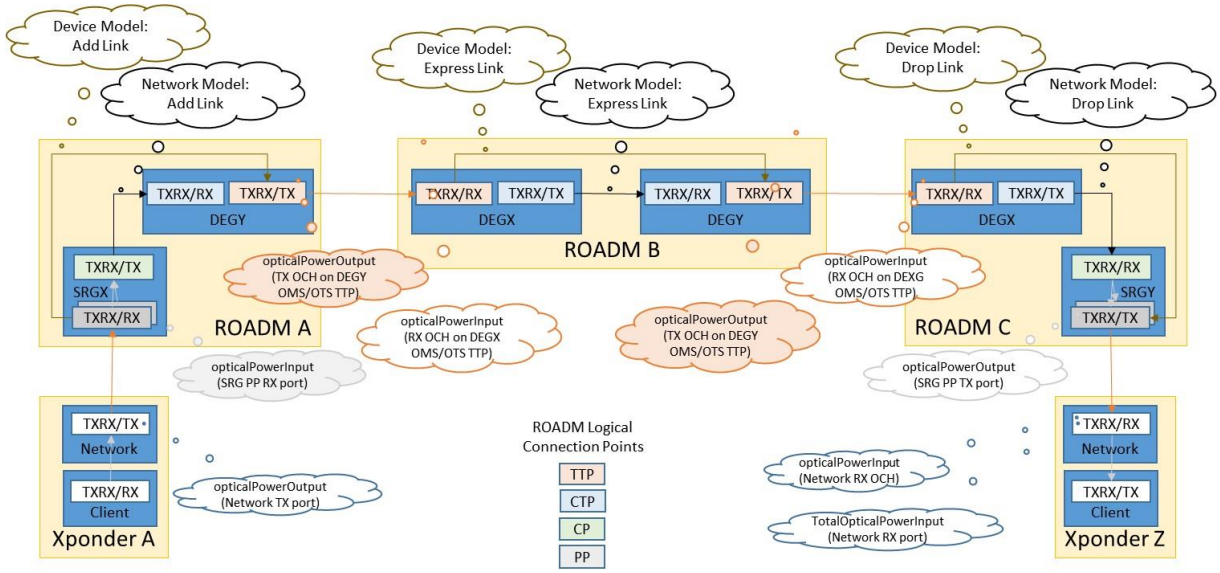
Some management systems require a node-wide unique name for all entities (e.g., for alarming purposes). The recommendation is to concatenate the circuit-pack-name and port-name with the "##" character sequence in the management systems:

- Circuit-pack-name = "cpA"
- Port-name = "portB"
- Concatenate name for the port would be "cpA##portB"

Note that the circuit-pack-name and the port-name should not contain the "##" pattern to allow a management system to separate the circuit-pack-name and port-name from the concatenated node-wide unique port-name.

7 Provisioning Action Use Cases

The following figure is used to provide context for this section.



7.1 Connection Management

In this section, the different basic operations needed to configure interfaces and connections on the ROADM transponder nodes are presented. Interface and connections deletion is presented in a second step. These actions are performed through device's NETCONF interfaces. XML encoding is not detailed and only basic NETCONF operations are presented in order to simplify and alleviate the description.

In the same way, because some ports/interfaces can be either uni (TX/RX) or bi-directional (TXRX), we try to be as generic as possible in the action sequence description. Consequently, such ports will be specified as [TXRX/TX] or [TXRX/RX].

7.1.1 Xponder interfaces creation (TX)

Configures the ports and interfaces on both line and client sides of a transponder in the TX direction. Xponder interfaces are supposed to be bi-directional. However, we describe the sequence of actions following the provisioning methodology; where the connections are configured sequentially across the pre-calculated path in one direction first, and then in the reverse direction. Thus, we only give precision on the elements and configuration actions that will be engaged at a time for a specific purpose. Real implementation may differ from the proposed sequence of action. As another example, we consider applying power settings just after the connection is created, but they could be performed in a second step. This example considers the case of a Transponder device mapping a single 100GE client signal into a 100G OTU4 DWDM signal, in accordance to paragraph. 2.2.3. Modulation format is QPSK.

Input parameters:

- node-id (xponder X)
- xponder network circuit-pack-name (xponder X network circuit pack)
- xponder network port-name (xponder X network port)
- xponder client circuit-pack-name (xponder X client circuit pack)
- xponder client port-name (xponder X client port)
- frequency
- rate
- modulation-format
- transmit-power

Main actions sequence:

Get-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<xponder X network circuit pack>”]/ports[port-name= “<xponder X network port>”]/)

retrieve xponder X network port transponder-port/port-power-capability (min/max tx)

retrieve xponder X network port supported-interface-capability

Get-config (org-openroadm-device/interface[supporting-circuit-pack-name=”< xponder X network circuit pack >” and supporting-port=”< xponder X network port >”])

Check if OCH interface present on network port and the status of the port

If NO OCH corresponding interface created on the Xponder network port

(either bi-directional port or uni-directional TX port)

equipment-state of circuit-pack and admin-state of network-port shall be set appropriately :

edit-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<xponder X network circuit pack>”]) nc:operation=replace

set equipment-state of OCH interface supporting circuit-pack to not-reserved-inuse

edit-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<xponder X network circuit pack>”]/ports[port-name= “<xponder X network port>”]) nc:operation=replace

set admin state of OCH interface supporting port to inService

OCH interface shall be created on network port:

edit-config (org-openroadm-device/interface>"]) nc:operation=create

create an OCH interface XPDR<n>-NETWORK<m>-[TXRX/TX]-[frequency] on supporting port

set name = "<XPDR<n>-NETWORK<m>-[TXRX/TX]-[frequency] >"

set type to opticalChannel

set supporting-circuit-pack-name="< xponder X network circuit pack >"

set supporting-port ="< xponder X network port >"

set rate, frequency, modulation-format (QPSK)

set admin state of OCH interface to inService

OTU4 interface shall be created on supporting OCH interface:

edit-config (org-openroadm-device/interface) nc:operation=create

create an otnOtu interface *OTU4* on supporting port/OCH interface

set name = "<OTU4-XPDR<n>-NETWORK<m>-TXRX/TX>"

set type to otnOtu

set supporting-circuit-pack-name="< xponder X network circuit pack >"

set supporting-port ="< xponder X network port >"

set supporting-interfaces ="< OCH interface>"

set admin state of OTU4 interface to inService

ODU4 interface shall be created on supporting OTU4 interface:

edit-config (org-openroadm-device/interface) nc:operation=create

create an otnOdu interface *ODU4* on supporting port/OTU4 interface

set name = "<ODU4-XPDR<n>-NETWORK<m>-TXRX/TX>"

set type to otnOdu

set supporting-circuit-pack-name="< xponder X network circuit pack >"

set supporting-port ="< xponder X network port >"

set supporting-interfaces "<OTU4-XPDR<n>-NETWORK<m>-TXRX/TX>"

set admin state of ODU4-1 interface to inService

Output power shall be configured on the OCH interface:

edit-config (org-openroadm-device/interface[name= "<XPDR<n>-NETWORK<m>-[TXRX/TX]-[frequency]>"]/och/transmit-power>") nc:operation=replace

set transmit-power to output-power

The controller verifies that the output power reaches the set value (verify pmParameterName/type = opticalPowerOutput):

Get-config (/currentPmlist/currentPm/currentPmlist/currentPm[resource/resource/port-name"< xponder X network port >"and granularity="15min"])

The controller verifies that connection between client and network xponder circuit packs and ports exists:

Optional: Get-config (org-openroadm-device/connection-map[source/circuit-pack-name=”<xponder X network circuit pack >”and source/port-name=”<xponder X client port>” and destination/circuit-pack-name=”<xponder X client circuit pack > and destination/port-name=”<xponder X network port>”])

Check if interface present on client port and the status of the port

Get-config (org-openroadm-device/interface[supporting-circuit-pack-name=”< xponder X client circuit pack >” and supporting-port-name=”< xponder X client port >”])

If NO corresponding interface created on the Xponder client port

(either bidirectional port or uni-directional RX port)

equipment-state of circuit-pack and admin-state of network-port shall be set appropriately :

edit-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<xponder X client circuit pack>”]) nc:operation=replace

set equipment-state of 100GE interface supporting circuit-pack to not-reserved-inuse

edit-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<xponder X client circuit pack>”]/ports[port-name= “<xponder X client port>”]) nc:operation=replace

set admin state of 100GE interface supporting port to inService

100GE Ethernet interface shall be created on client port:

edit-config (org-openroadm-device/interface) nc:operation=create

create a 100GE interface 100GE on supporting port

set name = “<100GE-XPDR<n>-CLIENT<m>-TXRX/TX>”

set type to ethernetCsmacd

set supporting-circuit-pack-name=”< xponder X client circuit pack >”

set supporting-port =”< xponder X client port >”

set speed, fec, duplex, mtu, auto-negotiation, curr-speed

set admin state of 100GE interface to inService

END of sequence

7.1.2 Add-Link creation

Configures the ports, interfaces and connections for an added link from a SRG-PP to Degree TTP logical point. One shall note that the power settings could be applied in a second step, after the interfaces and the connection have been created. Some actions are optional and may differ from one implementation to another. This example considers the case of a ROADM connection between two nmc-ctp where on the degree side the nmc-ctp interface is the only child of a mc-ttp interface. Furthermore nmc-ctp and mc-ttp interfaces have the same center frequency and NMC width is included inside the MC width.

Input parameters:

- node-id (SRGX, DEGY)
- circuit-pack (SRGX CircuitPack, DEGY CircuitPack)
- supporting-port (SRGX-PPN, DEGY-TTP)
- min-freq and max-freq for the mc-ttp interface
- frequency and width for the nmc-ctp interface

- calculated output power

Main actions sequence:

When optional is mentioned, it usually refers to a step where the controller performs some verification steps that are not mandatory.

Get-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<SRGX CircuitPack >”]/ports [port-name= “<SRGX-PPN >”]/interfaces)

If NO nmc-ctp (bidirectional) corresponding interface created on the SRGX-PPN

edit-config(org-openroadm-device/circuit-packs[circuit-pack-name= “<SRGX CircuitPack >”]/ports [port-name= “<SRGX-PPN >”]/) nc:operation=replace

set admin state of nmc-ctp interface supporting PP port to inService

edit-config (org-openroadm-device/interface) nc:operation=create

create nmc-ctp interface 1 (SRGX-PPN-[TXRX/RX]-[frequency]) on supporting PP port

set name = “<SRGX-PPN-[TXRX/RX]-[frequency] >”

set type to networkMediaChannelConnectionTerminationPoint

set supporting-circuit-pack-name= “< SRGX CircuitPack >”

set supporting-port = “< SRGX-PPN >”

set frequency

set width

set admin state of nmc-ctp interface inService

Get-config (org-openroadm-device/ circuit-packs[circuit-pack-name= “<DEGY-CircuitPack>”]/ports[port-name= “<DEGY-TTP>/interfaces”)

If NO OTS (bidirectional) corresponding interface created on the DEGY-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create OTS-interface (OTS-DEGY-TTP-[TXRX/TX]) on supporting TTP port

set name = “<OTS-DEGY-TTP-[TXRX/TX]>”

set type to opticalTransport

set supporting-circuit-pack-name= “< DEGY CircuitPack >”

set supporting-port = “< DEGY-TTP >”)

set admin state of OTS-interface to inService

If NO OMS (bidirectional) corresponding interface created on the DEGY-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create OMS-interface (OMS-DEGY-TTP-[TXRX/TX]) on supporting TTP port

set name = “<OMS-DEGY-TTP-[TXRX/TX]>”

set type to openROADMOpticalMulitplex

set supporting-circuit-pack-name= “< DEGY CircuitPack >”

set supporting-port = “< DEGY-TTP >”)

set supporting-interfaces =”<OTS-DEGY-TTP-[TXRX/TX]>”

set admin state of OMS-interface to inService

If NO mc-ttp (bidirectional) corresponding interface created on the DEGY-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create mc-ttp interface (mmc-ttp-DEGY-TTP-[TXRX/TX]-[frequency]) on supporting TTP port

set name = “<mmc-ttp-DEGY-TTP-[TXRX/TX]-[frequency]>”

set type to mediaChannelTrailTerminationPoint

set supporting-circuit-pack-name= “< DEGY CircuitPack >”

set supporting-port = “< DEGY-TTP >”)

set supporting-interfaces =”<OMS-DEGY-TTP-[TXRX/TX]>”

set min-freq

set max-freq

set admin state of mc-ttp-interface to inService

If NO nmc-ctp (bidirectional) corresponding interface created on the DEGY-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create nmc-ctp-interface2 (nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]) on supporting TTP port

set name = “<nmc-ttp-DEGY-TTP-[TXRX/TX]-[frequency]>”

set type to networkMediaChannelConnectionTerminationPoint

set supporting-circuit-pack-name= “< DEGY CircuitPack >”

set supporting-port = “< DEGY-TTP >”)

set supporting-interfaces =”<mc-ttp-DEGY-TTP-[TXRX/TX]-[frequency]>”

set frequency

set width

set admin state of nmc-ctp-interface2 to inService

edit-config (org-openroadm-device/roadm-connections) nc:operation=create

Set connection from nmc-ctp-Interface1 to nmc-ctp-Interface 2

Set connection-name = “<SRGX-PPN-[TXRX/RX]-[frequency]-to-nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]>”

set source/src-if = “<SRGX-PPN-[TXRX/RX]-[frequency] >”

set destination/dst-if = “<nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]>”

Power setting and control-mode change could be made in a second step after all connections are created

edit-config (org-openroadm-device/roadm-connections[connection-name= “<SRGX-PPN-[TXRX/RX]-[frequency]-to-nmc-ctpDEGY-TTP-[TXRX/TX]-[frequency]>”]) nc:operation=replace

set optical control-mode = power

set target-output-power = calculated output power

In a final step, the controller verifies that the output power reaches the set value (verify pmParameterName/type = opticalPowerOutput):

Get-config (/currentPmlist/currentPm[resource/resource/interface-name=”< nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]>” and granularity=”15min”])

And once current-output-power has converged towards expected output power, control-mode is moved to gainLoss

edit-config (org-openroadm-device/roadm-connections[connection-name= “<SRGX-PPN-[TXRX/RX]-[frequency]-to- nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]>”]) nc:operation=replace

Connection from nmc-ctp-Interface 1(PP) to OCH-Interface2 (TTP)

set optical control-mode = gainLoss

End of sequence

7.1.3 Express link creation

Configures the ports, interfaces and connections for an express link between 2 Degree TTP logical points in one direction. One shall note that the power settings could be applied in a second step, after the interfaces and the connections have been created. Some actions are optional and may differ from one implementation to another. This example considers the case of a ROADM connection between two nmc-ctp where on both degree sides the nmc-ctp interfaces are the only children of a mc-ttp interface. Furthermore nmc-ctp and mc-ttp interfaces have the same center frequency and NMC width is included in MC width.

A bidirectional express link will be created in 2 steps (connections are unidirectional) applying this sequence in both directions.

Input parameters:

- node-id (DEGX, DEGY)
- circuit-pack (DEGX CircuitPack
- DEGY CircuitPack)
- supporting-port (DEGX-TTP, DEGY-TTP)
- min-freq and max-freq for mc-ttp interfaces
- frequency and width for nmc-ctp interfaces
- expected input power
- calculated output power

Main actions sequence:

Get-config (org-openroadm-device/ circuit-packs[circuit-pack-name= “<DEGX-CircuitPack>”]/ports[port-name= “<DEGX-TTP>”]/interfaces)

If NO OTS (bidirectional) corresponding interface created on the DEGX-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create OTS-interface (OTS-DEGX-TTP-[TXRX/RX]) on supporting TTP port

set name = “<OTS-DEGX-TTP-[TXRX/RX]>”

set type to opticalTransport

set supporting-circuit-pack-name= “< DEGX CircuitPack >”

set supporting-port = “< DEGX-TTP >”)

set admin state of OTS-interface to inService

If NO OMS (bidirectional) corresponding interface created on the DEGX-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create OMS-interface (OMS-DEGX-TTP-[TXRX/RX]) on supporting TTP port

set name = “<OMS-DEGX-TTP-[TXRX/RX]>”

set type to openROADMOpticalMulitplex

set supporting-circuit-pack-name= “< DEGX CircuitPack >”

set supporting-port = “< DEGX-TTP >”)

set supporting-interfaces =”<OTS-DEGX-TTP-[TXRX/RX]>”

set admin state of OMS-interface to inService

If NO mc-ttp (bidirectional) corresponding interface created on the DEGX-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create mc-ttp-interface1 (mc-ttp-DEGX-TTP-[TXRX/RX]-[frequency]) on supporting TTP port

set name = “<mc-ttp-DEGX-TTP-[TXRX/RX]-[frequency]>”

set type to mediaChannelTrailTerminationPoint

set supporting-circuit-pack-name= “< DEGX CircuitPack >”

set supporting-port = “< DEGX-TTP >”)

set supporting-interfaces =”<OMS-DEGX-TTP-[TXRX/RX]>”

set min-freq

set max-freq

set admin state of mmc-ttp-interface1 to inService

If NO nmc-ctp (bidirectional) corresponding interface created on the DEGX-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create nmc-ctp-interface1 (DEGX-TTP-[TXRX/RX]-[frequency]) on supporting TTP port

set name = “<nmc-ctp-DEGX-TTP-[TXRX/RX]-[frequency]>”

set type to networkMediaChannelConnectionTerminationPoint

set supporting-circuit-pack-name= “< DEGX CircuitPack >”

set supporting-port = “< DEGX-TTP >”)

set supporting-interfaces =”<mc-ttp-DEGX-TTP-[TXRX/RX]-[frequency]>”

set frequency

set width

set admin state of nmc-ctp-interface1 to inService

The following step may be optional and aims at verifying that input power is compliant with MSA specifications (verify pmParameterName/type = opticalPowerInput):

Optional : Get-config (/currentPmlist/currentPm[resource/resource/interface-name=”<nmc-ctpDEGX-TTP-[TXRX/TX]-[frequency]>” and granularity=”15min”])

retrieve opticalPowerInput

compare to expected input power derived from MSA specifications : per channel output power of preceding span (per channel output power offset diagram)- span loss

Get-config (org-openroadm-device/ circuit-packs[circuit-pack-name= “<DEGY-CircuitPack>”]/ports [port-name= “<DEGY-TTP>]/interfaces”)

If NO OTS (bidirectional) corresponding interface created on the DEGY-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create OTS-interface (OTS-DEGY-TTP-[TXRX/TX]) on supporting TTP port

set name = “<OTS-DEGY-TTP-[TXRX/TX]>”

set type to opticalTransport

set supporting-circuit-pack-name= “< DEGY CircuitPack >”

set supporting-port = “< DEGY-TTP >”)

set admin state of OTS-interface to inService

If NO OMS (bidirectional) corresponding interface created on the DEGY-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create OMS-interface (OMS-DEGY-TTP-[TXRX/TX]) on supporting TTP port

set name = “<OMS-DEGY-TTP-[TXRX/TX]>”

set type to openROADMOpticalMulitplex

set supporting-circuit-pack-name= “< DEGY CircuitPack >”

set supporting-port = “< DEGY-TTP >”)

set supporting-interfaces =”<OTS-DEGY-TTP-[TXRX/TX]>”

set admin state of OMS-interface to inService

If NO mc-ttp (bidirectional) corresponding interface created on the DEGY-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create mc-ttp-interface2 (DEGY-TTP-[TXRX/TX]-[frequency]) on supporting TTP port

set name = “<mc-ttp-DEGY-TTP-[TXRX/TX]-[frequency]>”

set type to mediaChannelTrailTerminationPoint

set supporting-circuit-pack-name= “< DEGY CircuitPack >”

set supporting-port = “< DEGY-TTP >”)

set supporting-interfaces =”<OMS-DEGY-TTP-[TXRX/TX]>”

set min-freq

set max-freq

set admin state of mmc-ttp-interface2 to inService

If NO nmc-ctp (bidirectional) corresponding interface created on the DEGY-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create nmc-ctp-interface2 (DEGY-TTP-[TXRX/TX]-[frequency]) on supporting TTP port

set name = “<nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]>”

set type to networkMediaChannelConnectionTerminationPoint

set supporting-circuit-pack-name= “< DEGY CircuitPack >”

set supporting-port = “< DEGY-TTP >”)

set supporting-interfaces =”<mc-ttp-DEGY-TTP-[TXRX/TX]-[frequency]>”

set frequency

set width

set admin state of nmc-ctp-interface2 to inService

edit-config (org-openroadm-device/roadm-connections) nc:operation=create

set connection from nmc-ctp-Interface1 to nmc-ctp-Interface 2

set connection-name = “<nmc-ctp-DEGX-TTP-[TXRX/RX]-[frequency]-to-nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]>”

set source/src-if = “<nmc-ctp-DEGX-TTP-[TXRX/TX]-[frequency] >”

set destination/dst-if = “<nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]>”

Power setting and control-mode change could be made in a second step after all connections are created

edit-config (org-openroadm-device/roadm-connections[connection-name= “<nmc-ctp-DEGX-TTP-[TXRX/RX]-[frequency]-to-nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]>”]) nc:operation=replace

set optical control-mode = power

set target-output-power = calculated output power

The output power is calculated from MSA specifications: per channel output power offset diagram)

In a final step, the controller verifies that the output power reaches the set value (verify pmParameterName/type = opticalPowerOutput):

Get-config (/currentPmlist/currentPm[resource/resource/interface-name=”< nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]>” and granularity=”15min”])

And once current-output-power has converged towards expected output power, control-mode is moved to gainLoss

edit-config (org-openroadm-device/roadm-connections[connection-name= “<nmc-ctp-DEGX-TTP-[TXRX/RX]-[frequency]-to-nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]>”]) nc:operation=replace

Change optical-control-mode of connection from nmc-ctp-Interface 1(TTP) to nmc-ctp-Interface2 (TTP)

set optical control-mode = gainLoss

End of Sequence

7.1.4 Drop-Link creation

Configures the ports, interfaces and connections for drop link between a TTP and a PP logical points. One shall note that the power settings could be applied in a second step, after the interfaces and the connections have been created. Some actions are optional and may differ from one implementation to another. This example considers the case of a ROADM connection between two nmc-ctp where on the degree side the nmc-ctp interface is the only child of a mc-ttp interface. Furthermore nmc-ctp and mc-ttp interfaces have the same center frequency and NMC width is included in MC width.

Input parameters:

- node-id (DEGX, SRGY)
- circuit-pack (DEGX CircuitPack, SRGY CircuitPack)
- supporting-port (DEGX-TTP, SRGY-PPM)
- min-freq and max-freq for mc-ttp interface
- frequency and width for nmc-ctp interfaces
- calculated output power
- expected input power

Main actions sequence:

Get-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<DEGX-CircuitPack>”]/ports[port-name= “<DEGX-TTP>”]/interfaces)

If NO OTS (bidirectional) corresponding interface created on the DEGX-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create OTS-interface (OTS-DEGX-TTP-[TXRX/RX]) on supporting TTP port

set name = “<OTS-DEGX-TTP-[TXRX/RX]>”

set type to opticalTransport

set supporting-circuit-pack-name= “< DEGX CircuitPack >”

set supporting-port = “< DEGX-TTP >”)

set admin state of OTS-interface to inService

If NO OMS (bidirectional) corresponding interface created on the DEGX-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create OMS-interface (OMS-DEGX-TTP-[TXRX/RX]) on supporting TTP port

set name = “<OMS-DEGX-TTP-[TXRX/RX]>”

set type to openROADMOpticalMulitplex

set supporting-circuit-pack-name= “< DEGX CircuitPack >”

set supporting-port = “< DEGX-TTP >”)

set supporting-interfaces =”<OTS-DEGX-TTP-[TXRX/RX]>”

set admin state of OMS-interface to inService

If NO mc-ttp (bidirectional) corresponding interface created on the DEGX-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create mc-ttp-interface (mc-ttp-DEGX-TTP-[TXRX/RX]-[frequency]) on supporting TTP port

set name = “<mc-ttp-DEGX-TTP-[TXRX/RX]-[frequency]>”

set type to mediaChannelTrailTerminationPoint

set supporting-circuit-pack-name= “< DEGX CircuitPack >”

set supporting-port = “< DEGX-TTP >”)

set supporting-interfaces =”<OMS-DEGX-TTP-[TXRX/RX]>”

set min-freq

set max-freq

set admin state of mc-ttp-interface to inService

If NO nmc-ctp (bidirectional) corresponding interface created on the DEGX-TTP

edit-config (org-openroadm-device/interface) nc:operation=create

create nmc-ctp-interface1 (DEGX-TTP-[TXRX/RX]-[frequency]) on supporting TTP port

set name = “<nmc-ctp-DEGX-TTP-[TXRX/RX]-[frequency]>”

set type to networkMediaChannelConnectionTerminationPoint

set supporting-circuit-pack-name= “< DEGX CircuitPack >”

set supporting-port = “< DEGX-TTP >”)

set supporting-interfaces =”<mc-ttp-DEGX-TTP-[TXRX/RX]-[frequency]>”

set frequency

set width

set admin state of nmc-ctp-interface1 to inService

The following step may be optional and aims at verifying that input power is compliant with MSA specifications (verify pmParameterName/type = opticalPowerInput):

Optional: Get-config (/currentPmlist/currentPm[resource/resource/interface-name="<nmc-ctp-DEGX-TTP-[TXRX/RX]-[frequency]>" and granularity="15min"])

retrieve opticalPowerInput

compare to expected input power derived from MSA specifications : per channel output power of preceding span (per channel output power offset diagram)- span loss

Get-config (org-openroadm-device/circuit-packs[circuit-pack-name= "<SRGY CircuitPack >"]/ports[port-name= "<SRGY-PPN >"]/interfaces)

If NO nmc-ctp (bidirectional) corresponding interface created on the SRGY-PPN

edit-config(org-openroadm-device/circuit-packs/[circuit-pack-name= "<SRGY CircuitPack >"]/ports /[port-name= "<SRGY-PPN >"]/) nc:operation=replace

set admin state of nmc-ctp interface supporting PP port to inService

edit-config (org-openroadm-device/interface) nc:operation=create

create nmc-ctp-interface 2 (SRGY-PPN-[TXRX/TX]-[frequency]) on supporting PP port

set name = "<SRGY-PPN-[TXRX/TX]-[frequency] >"

set type to networkMediaChannelConnectionTerminationPoint

set supporting-circuit-pack-name= "< SRGY CircuitPack >"

set supporting-port = "< SRGY-PPN >"

set frequency

set width

set admin state of nmc-ctp interface 2 to inService

edit-config (org-openroadm-device/roadm-connections) nc:operation=create

Set connection from nmc-ctp-interface1 to nmc-ctp-interface2 nmc-ctp-DEGX-TTP-[TXRX/RX]-[frequency]-to- SRGY-PPM-[TXRX/TX]-[frequency]

Set connection-name = "<nmc-ctp-DEGX-TTP-[TXRX/RX]-[frequency]-to- SRGY-PPM-[TXRX/TX]-[frequency]>"

set source/src-if = "<nmc-ctp-DEGX-TTP-[TXRX/RX]-[frequency]>"

set destination/dst-if ="<SRGY-PPN-[TXRX/TX]-[frequency] >"

set optical control-mode = power

In a final step, the controller verifies that the output power reaches the set value (verify pmParameterName/type = opticalPowerOutput):

Get-config (/currentPmlist/currentPm[resource/resource/port-name="<SRGY-PPN>" and granularity="15min"])

retrieve SRG opticalPowerOutput

End of sequence

7.1.5 Xponder interfaces creation (RX)

Configures the ports and interfaces on both line and client sides of a transponder in the RX direction. This example considers the case of a Transponder device mapping a single 100GE client signal into a 100G OTU4 DWDM signal, in accordance to paragraph. 2.2.3. Modulation format is QPSK.

Input parameters:

- node-id (xponder X)
- xponder network circuit-pack-name (xponder X network circuit pack)
- xponder network port-name (xponder X network port)
- xponder client circuit-pack-name (xponder X client circuit pack)
- xponder client port-name (xponder X client port)
- SRGY supporting port (SRGY-PPN)
- frequency
- rate
- modulation format

Main actions sequence:

Get-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<xponder X network circuit pack>”]/ports[port-name= “<xponder X network port>”]/transponder-port)

retrieve xponder X network port **port-power-capability** (min/max rx)

retrieve xponder X network port **supported-interface-capability**

Optional: Check SRGY-PPN opticalPowerOutput retrieved in drop-link creation is compatible with min-max rx-port-power-capability

Get-config (org-openroadm-device/interface[supporting-circuit-pack-name=”< xponder X network circuit pack >” and supporting-port-name=”< xponder X network port >”])

Check if OCH interface present on network port and the status of the port

If NO OCH corresponding interface created on the Xponder network port

(either bi-directional port or uni-directional RX port)

equipment-state of circuit-pack and admin-state of network-port shall be set appropriately :

edit-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<xponder X network circuit pack>”]) nc:operation=replace

set equipment-state of OCH interface supporting circuit-pack to not-reserved-inuse

edit-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<xponder X network circuit pack>”]/ports[port-name= “<xponder X network port>”]) nc:operation=replace

set admin state of OCH interface supporting port to inService

OCH interface shall be created on network port:

edit-config (org-openroadm-device/interface) nc:operation=create

create an OCH interface XPDR<n>-NETWORK<m>-[TXRX/RX]-[frequency] on supporting port

set name = “<XPDR<n>-NETWORK<m>-[TXRX/RX]-[frequency] >”

set type to opticalChannel

set supporting-circuit-pack-name=”< xponder X network circuit pack >”

set supporting-port =”< xponder X network port >

set rate, frequency, modulation-format (QPSK)

set admin state of OCH interface to inService”

OTU4 interface shall be created on supporting OCH interface:

edit-config (org-openroadm-device/interface) nc:operation=create

create an otnOtu interface *OTU4* on supporting port/OCH interface

set name = “<OTU4-XPDR<n>-NETWORK<m>-TXRX/RX>”

set type to otnOtu

set supporting-circuit-pack-name=”< xponder X network circuit pack >”

set supporting-port =”< xponder X network port >”

set supporting-interfaces =”< OCH interface>”

set admin state of OTU4 interface to inService

ODU4 interface shall be created on supporting OTU4 interface:

edit-config (org-openroadm-device/interface) nc:operation=create

create an otnOdu interface *ODU4* on supporting port/OTU4 interface

set name = “<ODU4-XPDR<n>-NETWORK<m>-TXRX/RX>”

set type to otnOdu

set supporting-circuit-pack-name=”< xponder X network circuit pack >”

set supporting-port =”< xponder X network port >”

set supporting-interfaces =”<OTU4-XPDR<n>-NETWORK<m>-TXRX/RX>”

set admin state of ODU4 interface to inService

The controller verifies that connection between client and network xponder circuit packs exists:

Optional: Get-config (org-openroadm-device/connection-map[source/circuit-pack-name=”<xponder X network circuit pack >”and source/port-name=”<xponder X network port>” and destination/circuit-pack-name=”<xponder X client circuit pack > and destination/port-name=”<xponder X client port>”])

Check if interface present on client port and the status of the port

Get-config (org-openroadm-device/interface[supporting-circuit-pack-name=”< xponder X client circuit pack >” and supporting-port-name=”< xponder X client port >”])

If NO corresponding interface created on the Xponder client port

(either bidirectional port or uni-directional TX port)

equipment-state of circuit-pack and admin-state of network-port shall be set appropriately :

edit-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<xponder X client circuit pack>”]) nc:operation=replace

set equipment-state of 100GE interface supporting circuit-pack to not-reserved-inuse

edit-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<xponder X client circuit pack>”]/ports[port-name= “<xponder X client port>”]) nc:operation=replace

set admin state of 100GE interface supporting port to inService

100GE Ethernet interface shall be created on client port:

edit-config (org-openroadm-device/interface) nc:operation=create

create a 100GE interface 100GE on supporting port

set name = “<100GE-XPDR<n>-CLIENT<m>-TXRX/TX>”

set type to ethernetCsmacd

set supporting-circuit-pack-name=”< xponder X client circuit pack >”

set supporting-port =”< xponder X client port >”

set speed, fec, duplex, mtu, auto-negotiation, curr-speed

set admin state of 100GE interface to inService

End of sequence

7.1.6 Xponder interfaces deletion (TX/RX)

Deletes the interfaces and un-configures the ports on both line and client sides of a transponder in both RX and TX directions.

Input parameters:

- node-id (xponder X)
- xponder network circuit-pack-name (xponder X network circuit pack)
- xponder network port-name (xponder X network port)
- xponder client circuit-pack-name (xponder X client circuit pack)
- xponder client port-name (xponder X client port)

Main actions sequence:

edit-config (org-openroadm-device/interface[name= “<XPDR<n>-NETWORK<m>-[TXRX/TX]-[frequency]>”]/och/transmit-power) nc:operation=replace

set transmit power to minimum level

set transmit-power to -5 dBm

edit-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<xponder X network circuit pack>”]) nc:operation=replace

set equipment-state of OCH interface supporting circuit-pack to not-reserved-available

edit-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<xponder X network circuit pack>”]/ports[port-name= “<xponder X network port>”]) nc:operation=replace

set admin state of network port to outOfService

Get-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<xponder X network circuit pack>”]/ports[port-name= “<xponder X network port>”]/interfaces)

Check interfaces configured on the line port(s) (either 1 bidirectional or 2 unidirectional ports)

If OCH interface is unique and bidirectional

edit-config (org-openroadm-device/interface[name= “<ODU4-XPDR<n>-NETWORK<m>-TXRX >”]) nc:operation=delete

delete an ODU4 interface

edit-config (org-openroadm-device/interface[name= “<OTU4-XPDR<n>-NETWORK<m>-TXRX”]) nc:operation=delete

delete an OTU4 interface

edit-config (org-openroadm-device/interface[name= “<XPDR<n>-NETWORK<m>-[TXRX]-[frequency] >”]) nc:operation=delete

delete an OCH interface XPDX-LINEN-TXRX-[frequency]

If 2 OCH unidirectional interfaces are present

edit-config (org-openroadm-device/interface[name= “<ODU4-XPDR<n>-NETWORK<m>-TX>”]) nc:operation=delete

delete first ODU4 interface

edit-config (org-openroadm-device/interface[name= “<ODU4-XPDR<n>-NETWORK<m>-RX>”]) nc:operation=delete

delete second ODU4 interface

edit-config (org-openroadm-device/interface[name= “<OTU4-XPDR<n>-NETWORK<m>-TX>”]) nc:operation=delete

delete first OTU4 interface

edit-config (org-openroadm-device/interface[name= “<OTU4-XPDR<n>-NETWORK<m>-RX>”]) nc:operation=delete

delete second OTU4 interface

edit-config (org-openroadm-device/interface[name= “<XPDR<n>-NETWORK<m>-[TX]-[frequency] >”]) nc:operation=delete

delete first OCH interface

edit-config (org-openroadm-device/interface[name= “<XPDR<n>-NETWORK<m>-[RX]-[frequency] >”]) nc:operation=delete

delete second OCH interface

edit-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<xponder X client circuit pack>”]) nc:operation=replace

set equipment-state of 100GE interface supporting circuit-pack to not-reserved-available

Get-config (org-openroadm-device/interface[supporting-circuit-pack-name=”< xponder X client circuit pack >” and supporting-port-name=”< xponder X client port >”])

Check interfaces configured on the client port(s) (either 1 bidirectional or 2 unidirectional ports)

edit-config (org-openroadm-device/circuit-packs[circuit-pack-name= “<xponder X client circuit pack>”]/ports[port-name= “<xponder X client port>”]) nc:operation=replace

set admin state of 100GE interface supporting port to outOfService

If client interface is unique and bidirectional

edit-config (org-openroadm-device/interface[name= “<100GE-XPDR<n>-NETWORK<m>-TXRX>”]) nc:operation=delete

If 2 OCH unidirectional interfaces are present

edit-config (org-openroadm-device/interface[name= “<100GE-XPDR<n>-NETWORK<m>-TX >”]) nc:operation=delete

edit-config (org-openroadm-device/interface[name= “<100GE-XPDR<n>-NETWORK<m>-RX >”]) nc:operation=delete

End of sequence

7.1.7 Add-link deletion

Configures ports, deletes connections and interfaces for an add link from a SRG-PP to a Degree TTP logical point. . This example considers the case of a ROADM connection between two nmc-ctp where on the degree side the nmc-ctp interface is the only child of a mc-ttp interface, thus, both nmc-ctp and mc-ttp are deleted.

Input parameters:

- node-id (SRGX, DEGY)
- circuit-pack (SRGX-CircuitPack, DEGY-CircuitPack)
- supporting-port (SRGX-PPN, DEGY-TTP)
- calculated-output-power
- interface (SRGX-PPN-[TXRX/RX]-[frequency]
- interface nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency])
- connection SRGX-PPN-[TXRX/RX]-[frequency]-to-nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]

Main actions sequence:

edit-config (org-openroadm-device/roadm-connections/[connection-name= “<SRGX-PPN-[TXRX/RX]-[frequency]-to- nmc-ctpDEGY-TTP-[TXRX/TX]-[frequency]>”]) nc:operation=replace

Set connection power from SRGX-PPN-[TXRX/RX] -[frequency] to nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency] to the minimum acceptable value:

set target-output-power = -60dBm

edit-config(org-openroadm-device/circuit-packs/[circuit-pack-name= “<SRGX CircuitPack >”]/ports /[port-name= “<SRGX-PPN>”]/) nc:operation=replace

set admin state of nmc-ctp interface supporting PP port to outOfService

edit-config (org-openroadm-device/roadm-connections/[connection-name= “<SRGX-PPN-[TXRX/RX]-[frequency]-to- nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]>”])nc:operation=delete

delete connection SRGX-PPN-[TXRX/RX]-[frequency]-to-nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]

Get-config (org-openroadm-device/roadm-connections/)

check if an nmc-ctp-DEGY-TTP-[TXRX/RX]-[frequency]-to-SRGX-PPN-[TXRX/TX]-[frequency] connection is present

If (SRGX-PPN & nmc-ctp-DEGY-TTP are unidirectional OR

NO nmc-ctp-DEGY-TTP-[TXRX/RX]-[frequency]-to-SRGX-PPN-[TXRX/TX]-[frequency] connection present)

edit-config (org-openroadm-device/interface/[name= “<SRGX-PPN-[TXRX/RX]-[frequency] >”]) nc:operation=delete

delete an nmc-ctp- interface on supporting SRGX PP port

edit-config (org-openroadm-device/interface/[name= “<nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency] >”]) nc:operation=delete

delete an nmc-ctp interface on supporting DEGY TTP port

edit-config (org-openroadm-device/interface/[name= “<mc-ttp-DEGY-TTP-[TXRX/TX]-[frequency] >”]) nc:operation=delete

delete an mc-ttp interface on supporting DEGY TTP port

End of sequence

7.1.8 Express link deletion

Configures the ports and deletes the interfaces and the connections for an express link between 2 Degree TTP logical points. Express link in this example is deleted in 1 step. Interfaces can be either bi (TXRX) or uni-directional (RX or TX) so that when we state [TXRX/TX] , [TXRX/RX] or [TXRX/RX/TX], only one (TXRX) or the other option (TX or RX) is valid depending on the type of interface . If the express link connections were deleted in 2 steps, then in the case of bi-directional nmc-ctp interfaces, we would delete these last-only in the second step.

Input parameters:

- node-id (DEGX, DEGY)
- circuit-pack (DEGX CircuitPack, DEGY CircuitPack)
- supporting-port (DEGX-TTP, DEGY-TTP)
- connection nmc-ctp-DEGX-TTP-[TXRX/ RX]-[frequency]-to-nmc-ctp-DEGY-TTP-[TXRX/ TX]-[frequency]

Main actions sequence:

edit-config (org-openroadm-device/roadm-connections/[connection-name= “<nmc-ctp-DEGX-TTP-[TXRX/RX]-[frequency]-to-nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]>”]) nc:operation=merge

Set Connection nmc-ctp-DEGX-TTP-[TXRX/ RX]-[frequency]-to-nmc-ctp-DEGY-TTP-[TXRX/ TX]-[frequency] to low power

set target-output-power = -60dBm

edit-config (org-openroadm-device/roadm-connections/[connection-name= “<nmc-ctp-DEGX-TTP -[TXRX/RX]-[frequency]-to-nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]>”]) nc:operation=delete

Delete connection nmc-ctp-DEGX-TTP-[TXRX/RX]-[frequency]-to-nmc-ctp-DEGY-TTP-[TXRX/TX]-[frequency]

edit-config (org-openroadm-device/roadm-connections/[connection-name= “<nmc-ctp-DEGY-TTP-[TXRX/RX]-[frequency]-to-nmc-ctp-DEGX-TTP-[TXRX/TX]-[frequency]>”]) nc:operation=merge

Set Connection nmc-ctp-DEGY-TTP-[TXRX/RX]-[frequency]-to- nmc-ctp-DEGX-TTP-[TXRX/TX]-[frequency] to low power

set target-output-power = -60dBm

edit-config (org-openroadm-device/roadm-connections/[connection-name= “<nmc-ctp-DEGY-TTP-[TXRX/RX]-[frequency]-to- nmc-ctp-DEGX-TTP-[TXRX/TX]-[frequency]>”]) nc:operation=delete

Delete connection nmc-ctp-DEGY-TTP-[TXRX/RX]-[frequency]-to- nmc-ctp-DEGX-TTP-[TXRX/TX]-[frequency]

edit-config (org-openroadm-device/interface/[name= “<nmc-ctp-DEGX-TTP-[TXRX/TX/RX]-[frequency]>”) nc:operation=delete

delete nmc-ctp interface(s) on supporting DEGX TTP port

edit-config (org-openroadm-device/interface/[name= “<nmc-ctp-DEGY-TTP-[TXRX/TX/RX]-[frequency]>”) nc:operation=delete

delete nmc-ctp interface(s) on supporting DEGY TTP port

edit-config (org-openroadm-device/interface/[name= “<mc-ttp-DEGX-TTP-[TXRX/TX/RX]-[frequency]>”) nc:operation=delete

delete mc-ttp interface(s) on supporting DEGX TTP port

edit-config (org-openroadm-device/interface/[name= “mc-ttp-DEGY-TTP-[TXRX/TX/RX]-[frequency] >”) nc:operation=delete

delete mc-ttp interface(s) on supporting DEGY TTP port

End of sequence

7.1.9 Drop link deletion

Configures ports, deletes connections and interfaces for drop link between a TTP and a PP logical points. This example considers the case of a ROADM connection between two nmc-ctp where on the degree side the nmc-ctp interface is the only child of a mc-ttp interface, thus, both nmc-ctp and mc-ttp are deleted.

Input parameters:

- node-id (DEGX, SRGY)
- circuit-pack (DEGX CircuitPack, SRGYCircuitPack)
- supporting-port (DEGX-TTP, SRGY-PPM)
- connection nmc-ctp-DEGX-TTP-[TXRX/ RX]-[frequency]-to-SRGY-PPN-[TXRX/ TX]-[frequency]

Main actions sequence:

edit-config(org-openroadm-device/circuit-packs/[circuit-pack-name= “<SRGY CircuitPack >”]/ports/[port-name= “<SRGY-PPM>”]/) nc:operation=replace

set admin state of nmc-ctp interface supporting PP port to outOfService

edit-config (org-openroadm-device/roadm-connections/[connection-name=“< nmc-ctp-DEGX-TTP-[TXRX/RX]-[frequency]-to- SRGY-PPM-[TXRX/TX]-[frequency]>”]) nc:operation=delete

delete connection nmc-ctp-DEGX-TTP-[TXRX/RX]-[frequency]-to-SRGY-PPN-[TXRX/TX]-[frequency]

Get-config (org-openroadm-device/roadm-connections/)

checks if an SRGY-PPN-[TXRX/RX]-[frequency]- to- nmc-ctp-DEGX-TTP-[TXRX/TX]-[frequency] connection is present

If (SRGY-PPN & DEGX-TTP are unidirectional OR NO* SRGY-PPN-[TXRX/ RX]-[frequency]- to-DEGX-TTP-[TXRX/ TX]-[frequency] *connection present)

edit-config (org-openroadm-device/interface/[name= “<nmc-ctp-DEGX-TTP-[TXRX/RX]-[frequency] >”]) nc:operation=delete

delete an nmc-ctp interface on supporting DEGX TTP port

edit-config (org-openroadm-device/interface/[name= “<mc-ttp-DEGX-TTP-[TXRX/RX]-[frequency] >”]) nc:operation=delete

delete an mc-ttp interface on supporting DEGX TTP port

edit-config (org-openroadm-device/interface/[name= “<SRGY-PPN-[TXRX/TX]-[frequency] >”]) nc:operation=delete

delete an nmc-ctp interface on supporting SRGY PP port

End of sequence

7.1.10 Note on State Models

Open ROADM Version 1.2.1 does not specify a specific state model behavior with respect to equipment and interfaces. Thus, there may be state model dependencies

- E.g., port state interaction between ports and interfaces such that a port cannot be placed in administrative down unless the supported interfaces are put in administrative down state) by a vendor’s implementation.
- E.g., an entity may not be allowed to be deleted unless the administrative state is down

The equipment state model (reserved-for-facility-planned, not-reserved-planned, reserved-for-maintenance-planned, etc.) is managed by the controller and stored on the device. The actual equipment-state may or may not drive behavior on the Open ROADM device.

7.2 INCREMENTAL HARDWARE

7.2.1 Degree Growth

Degrees are to be deployed from degree number 1 to degree number N sequentially, up to the maximum number of degrees supported by the NE. This means that degree ***m*** would not be deployed before degree ***(m-1)***.

The growth of degrees is independent of the number of SRGs currently deployed.

In order to deploy a new degree, the following provision would take place:

- Provision new shelves as necessary
- Provision new circuit packs that will support the new degrees, and configure the ports associated with the circuit packs
  - Including setting the TTP and CTP logical connection point attributes (against the port entities)
- Provision the degree container for the new degree which include:
  - The list of circuit packs associated with the new degree.
  - The list of (external) connection ports associated with the degree
- Provision the new physical links associated with the degree. This may include:
  - Physical links internal to the degree, including the physical link for the OSC jumper that connects to the OSC pluggable circuit pack.

- Physical links connecting the new degree to the existing degrees
- Physical links connecting the new degree to the existing SRGs
- Physical links to the OTDR unit

7.2.1.1 Create Entities

Creates new shelf entities associated with a degree augmentation, if required.

Input parameters:

- node-id
- shelf-name
- shelf-type
- rack location
- shelf position within the rack
- administrative state
- equipment state
- due-date

Node-id, shelf-name and shelf-type are mandatory. All other parameters are optional.

- Vendors define the type of shelf based on the shelf-type.
- The rack and shelf-position identifies the location of the shelf within an office. The rack is the rack identifier, and the shelf-position identifies the location of the shelf within the rack. The naming scheme for these identifiers are up to the operator.
- Equipment-state tracks the lifecycle states of the equipment. It takes on the following values:
  - reserved-for-facility-planned: equipment is planned for use by a service
  - not-reserved-planned: equipment is planned by not reserved for any purpose
  - reserved-for-maintenance-planned: equipment is planned for use as a maintenance spare
  - reserved-for-facility-unvalidated: equipment is reserved for use by a service but not validated against planned equipment
  - not-reserved-unvalidated: equipment is not reserved for any purpose and not validated against planned equipment
  - unknown-unvalidated: unknown equipment not validated against planned equipment
  - reserved-for-maintenance-unvalidated: equipment is to be used for use as a maintenance spare but not validated against planned equipment
  - reserved-for-facility-available: reserved for use by a service and available
  - not-reserved-available: not reserved for use by a service and available
  - reserved-for-maintenance-available: reserved as a maintenance spare and available
  - reserved-for-reversion-inuse: equipment that is reserved as part of a home path for a service that has been temporarily re-routed

- not-reserved-inuse: equipment in use for a service
- reserved-for-maintenance-inuse: maintenance spare equipment that is in use as a maintenance spare

Read-only data:

- Physical inventory data is provided for the shelf including the vendor, model, serial-id, type (typically should match the shelf-type), product-code, manufacture-date, CLEI and hardware version.
- The slots container identifies the slots on a shelf that can host circuit-packs. The label field would be optionally present to represent the silkscreen of the slot if the slot-name does not match the silkscreen. The provisioned-circuit-pack would be present if there is a circuit pack provisioned that is plugged into the slot. In addition, the slot-status provides information about the operational status of the slot including if there is a circuit-pack plugged into the slot and if there is a mismatch.

edit-config (/org-openroadm-device/shelves) nc:operation=create

Set the shelf-type and other shelf attributes.

Set the shelf-name

Set the shelf-type to the vendor-specific shelf type

Set the administrative-state of the shelf to inService

Set the rack, shelf-position, equipment-state and due date, if desired. These attributes are for tracking purposes only.

End of sequence

7.2.1.2 Create Circuit-Pack Entities

Creates new circuit-pack entities associated with a degree augmentation.

Input parameters:

- node-id
- circuit-pack-name
- circuit-pack-type
- circuit-pack-product-code
- circuit-pack-mode
- shelf
- slot
- subSlot
- parent circuit-pack-name
- parent cp-slot-name
- administrative state
- equipment state
- due-date

Node-id, circuit-pack-name, circuit-pack-type, shelf, and slot are mandatory. All other parameters are optional.

- Vendors define the type of circuit pack based on the circuit-pack-type (mandatory) and circuit-pack-product-code (optional). The combination of these two attributes (type, product code) should uniquely identify a circuit-pack hardware.
- If a circuit pack hardware can take on different "personalities", then this is provisioned using the circuit-pack-mode attribute (e.g., REGEN mode vs TRANSPONDER mode).
- Circuit packs refer to their location in the shelf via the shelf, slot and optional subSlot attribute. SubSlot should be set the same value as the cp-slot-name in the parent-circuit-pack container. In general, the hierarchical location of circuit packs should use the shelf and slot fields for circuit packs that plug directly into a shelf (in this case, there will not be a parent-circuit-pack container), or use the cp-slot and parent-circuit-pack fields for circuit packs that plug into other circuit packs (e.g., pluggable optics that plug into a circuit pack).
- If circuit packs are plugged into a parent circuit pack, then the parent-circuit-pack container is mandatory. In this case, the parent is specified by providing both the parent's circuit-pack-name and cp-slot-name for the cp-slot that the circuit pack is plugging into. For example:

```
"circuit-packs": [
 {
  "circuit-pack-name": "1/1",
  "circuit-pack-type": "CPTYPE1",
  "shelf": "1",
  "slot": "1",
 },
 {
  "circuit-pack-name": "1/1/1",
  "circuit-pack-type": "CPTYPE2",
  "shelf": "1",
  "slot": "1",
  "subSlot": "1",
  "parent-circuit-pack": {
          "circuit-pack-name": "1/1"
          "cp-slot-name": "1"
  }
 },
```

- Equipment-state tracks the lifecycle states of the equipment. The values are the same as indicated for the shelf.

Read-only data:

- Physical inventory data is provided for the circuit-pack including the vendor, model, serial-id, type (typically should match the circuit-pack-type), product-code (typically

should match the circuit-pack-product-code), manufacture-date, CLEI and hardware version.

- Circuit-pack-category provides additional classification of the circuit-pack by the vendor.
- The cp-slots container identifies the “slots” on a circuit-pack that can host other circuit-packs. The label field would be optionally present to represent the silkscreen of the cp-slot if the slot-name does not match the silkscreen. The slot-type indicates if the slot is used to hold pluggable optics (pluggable-optics-holder) or other (non-optical-pluggable circuit packs). The provisioned-circuit-pack would be present if there is a child circuit pack provisioned that is plugged into the cp-slot. In addition, the slot-status provides information about the operational status of the slot including if there is a circuit-pack plugged into the slot and if there is a mismatch.
    - The cp-slots container also contains capability information representing information about what can be used and created from the cp-slot.
- The is-pluggable-optics indicates if this circuit-pack represents an optical pluggable module.
- Firmware information including the software-load-version, firmware features list and firmware component lists. See Section 4.3.

edit-config (/org-openroadm-device/circuit-packs) nc:operation=create

Set the circuit-pack-type and other circuit-pack attributes.

Set the circuit-pack-name

Set the circuit-pack-type (mandatory) and circuit-pack-product-code (optional) to the vendor-specific values

Set the circuit-pack-mode as necessary based on vendor-specific values

Set the shelf, slot and subSlot to identify the location of the circuit-pack.

Also set the parent-circuit-pack circuit-pack-name and cp-slot-name if this circuit-pack plugs into another circuit pack.

Set the administrative-state of the shelf to inService

Set the equipment-state and due date if desired. These attributes are for tracking purposes only.

End of sequence

7.2.1.2.1 Configure Circuit-Pack Port Entities

This substep configures the port entities associated with the circuit pack. Ports are auto-created upon creation of the circuit-pack.

Note: Recommendation to use the “merge” operation on editing port attributes.

Input parameters:

- node-id
- circuit-pack-name
- port-name

- port-type
- port-qual
- circuit-id
- administrative-state
- logical-connection-point
- For OTDR ports, launch-cable-length and port-direction

Node-id, circuit-pack-name and port-name are mandatory. All other parameters are optional.

- Vendors define the port-type for the port. This value is not standardized in the Open ROADM specifications. This field is optional.
- ForROADM degree, the port-qual can take the value of roadm-external, roadm-internal, otdr or the attribute is not present. The general guideline is that the ROADM network port would be set to roadm-external and any internal ports on the data path would be set to roadm-internal. The ports on an OTDR are set to otdr. For all other ports, the port-qual is not present or optionally set to roadm-internal.
- Circuit-id is a user's defined field to identify the circuit that may be associated with this port. Its use is optional.
- The logical connection point is used to support the mapping at the controller level between the device model and network model.
    - For the ROADM degree network port, the logical connection point should be set to the format "DEG<n>-TTP-RX", "DEG<n>-TTP-TX" or "DEG<n>-TTP-TXRX" depending on if the port-direction is unidirectional receive, unidirectional transmit, or bidirectional, respectively, where <n> is the degree number. For example for degree 1 with unidirectional ports, there should be two ports – one with logical connection point = "DEG1-TTP-RX" and the other with logical connection point = "DEG1-TTP-TX"
    - For the ROADM degree internal ports that connect to another degree or SRG, the logical connection point should be set to the format "DEG<n>-CTP-RX", "DEG<n>-CTP-TX" or "DEG<n>-CTP-TXRX" depending on if the port-direction is unidirectional receive, unidirectional transmit, or bidirectional, respectively, where <n> is the degree number. For example for degree 1 with unidirectional ports that connect to degree 2, there should be two internal ports – one with logical connection point = "DEG1-CTP-RX" and the other with logical connection point = "DEG1-CTP-TX"
- For OTDR ports, the OTDR launch cable length sets the length of the launch cable and the port-direction indicates if port is associated with the degree receive or transmit port. Currently, Open ROADM runs the OTDR over the degree receive port.

Read-only data:

- Port-wavelength-type is set to multi-wavelength for the ROADM network degree ports. This attribute may not be present for other ports.
- Port-direction indicates if the port is modelled as a unidirectional or bidirectional port, and if unidirectional, whether the port is transmit or receive. The values for port-direction are "tx", "rx" or "bidirectional"

- Label is meant to convey the silkscreen on the device to aid in physically locating the port on the device (e.g., provided to the local technician for troubleshooting). If the port-name matches the silkscreen, then label is not necessary and may be omitted by the vendor. Otherwise, the label should be present.
- Supported-interface-capability is used to indicate which interface(s) can be built on the port. The ROADM network port list of interface capabilities include: if-OTS, if-OMS and if-OCH for ROADM degree.
- The partner-port identifies the associated port when the ports are unidirectional.
- The interface container contains a list of provisioned interfaces associated with the port.
- The roadm-port capabilities details the minimum and maximum aggregate powers supported by the port. This field applies to the ROADM network port for the degree.

Main actions sequence:

edit-config (/org-openroadm-device/circuit-packs[circuit-pack-name]/ports) nc:operation=merge

Edit the port configuration as necessary.

Set the port-name

Set the port-type to the vendor-specific port type

Set the port-qualifier and logical-connection-point as above. Note that some ports may not have these attributes set.

Set the administrative-state of the port to inService

For OTDR ports, set the launch-cable-length and port-direction.

End of sequence

7.2.1.3 Create Degree Entity

Creates new degree entity that defines the circuit-packs associated with the degree, and the connection ports for the external connections.

Input parameters:

- node-id
- degree-number
- list of circuit-packs associated with the degree
- list of connection-ports associated with the degree (external ports)
- For OTDR application, the OTDR's circuit-pack-name and port-name that is used to monitor this degree

Node-id, degree-number, circuit-pack-list, and connection-port-list are mandatory. All other parameters are optional.

- The circuit-packs container lists all circuit-packs associated with the degree.

    - Note that if there are shared circuit-packs that is used for more than one degree (or SRG), then that circuit-pack should also be listed under the new degree as well as the old degree (or SRG).
- The connection-ports container lists the ports that connect externally to the line transmission fibers.
    - The connection ports should refer to the physical connection to the external line fibers. This may also be the logical connections where the OTS interfaces are constructed.
- The otdr-port container provides the OTDR port that will monitor the degree. The OTDR should monitor the degree transmit port. The circuit-pack-name and port-name refer to the port on an OTDR.

Read-only data:

- The max-wavelength identifies the maximum number of wavelengths supported by the degree. This value should be set to 96.

Main actions sequence:

edit-config (/org-openroadm-device/degree) nc:operation=create

Set the degree attributes.

Set the degree-number

Set the list of circuit-packs associated to the degree

Set the list of connection ports associated to the degree

Configure the otdr-port that would monitor the degree

End of sequence

7.2.1.4 Create Physical Link Entities

Creates the physical link entities that describe the internal ROADM fibering and/or cabling within the degree, between the degree and common units, between the degree and other existing degrees, and between the degree and SRGs.

Input parameters:

- node-id
- physical-link-name
- source and destination port idenifiers

Node-id, physical-link-name, source circuit-pack-name, source port-name, destination circuit-pack-name and destination port-name are mandatory.

- Physical links are always unidirectional
    - Source should indicate the TX port
    - Destination should indicate the RX port
- Physical links can represent a single bidirectional cable (e.g., Ethernet cable). In this case, the physical link is still modeled as two unidirectional entities.

- Physical links can also represent MPO connections. Both the physical connector and the logical connectors (child ports within the MPO) can be modeled

Main actions sequence:

edit-config (/org-openroadm-device/ physical-link) nc:operation=create

Set the physical link end points.

Set the physical-link-name

Set the physical link source circuit-pack-name and port-name (TX)

Set the physical link destination circuit-pack-name and port-name (RX)

End of sequence

This completes the commissioning of a new degree.

7.2.2 SRG Growth

SRGs are the add/drop units on the ROADM device. SRGs are to be deployed from SRG number 1 to SRG number N sequentially, up to the maximum number of SRGs supported by the NE. This means that SRG m would not be deployed before SRG (m-1).

The growth of SRGs are independent of the number of degrees currently deployed.

The deployment of SRGs is similar to the deployment of degrees. In order to deploy a new SRG, the following provision would take place:

- Provision new shelves as necessary
- Provision new circuit packs that will support the new SRGs, and configure the ports associated with the circuit packs
  - Including setting the TTP and CTP logical connection point attributes (against the port entities)
- Provision the shared-risk-group container for the new SRG which include:
  - The list of circuit packs associated with the new SRG.
- Provision the new physical links associated with the SRG. This may include:
  - Physical links internal to the SRG
  - Physical links connecting the new SRG to the existing degrees
  - Physical links connecting the new SRG to the existing SRGs

7.2.2.1 Create Shelf Entities

Creates new shelf entities associated with a SRG augmentation, if required.

Input parameters:

- node-id
- shelf-name
- shelf-type
- rack location

- shelf position within the rack
- administrative state
- equipment state
- due-date

Node-id, shelf-name and shelf-type are mandatory. All other parameters are optional.

Refer to “Create Shelf Entities” in the degree section, Section 7.2.1.1, for a description of the provisionable and read-only attributes for shelves.

Main actions sequence:

edit-config (/org-openroadm-device/shelves) nc:operation=create

Set the shelf-type and other shelf attributes.

Set the shelf-name

Set the shelf-type to the vendor-specific shelf type

Set the administrative-state of the shelf to inService

Set the rack, shelf-position, equipment-state and due date, if desired. These attributes are for tracking purposes only.

End of sequence

7.2.2.2 Create Circuit-Pack Entities

Creates new circuit-pack entities associated with a SRG augmentation.

Input parameters:

- node-id
- circuit-pack-name
- circuit-pack-type
- circuit-pack-product-code
- circuit-pack-mode
- shelf
- slot
- subSlot
- parent circuit-pack-name
- parent cp-slot-name
- administrative state
- equipment state
- due-date

Node-id, circuit-pack-name, circuit-pack-type, shelf, slot are mandatory. All other parameters are optional.

Refer to “Create Circuit-Pack Entities” in the degree section, Section 7.2.1.2, for a description of the provisionable and read-only attributes for shelves.

Main actions sequence:

edit-config (/org-openroadm-device/circuit-packs) nc:operation=create

Set the circuit-pack-type and other circuit-pack attributes.

Set the circuit-pack-name

Set the circuit-pack-type (mandatory) and circuit-pack-product-code (optional) to the vendor-specific values

Set the circuit-pack-mode as necessary based on vendor-specific values

Set the shelf, slot and subSlot to identify the location of the circuit-pack.

Also set the parent-circuit-pack circuit-pack-name and cp-slot-name if this circuit-pack plugs into another circuit pack.

Set the administrative-state of the shelf to inService

Set the equipment-state and due date if desired. These attributes are for tracking purposes only.

End of sequence

7.2.2.2.1 Configure Circuit-Pack Port Entities

This substep configures the port entities associated with the circuit pack. Ports are auto-created upon creation of the circuit-pack.

Note: Recommendation to use the “merge” operation on editing port attributes.

Input parameters:

- node-id
- circuit-pack-name
- port-name
- port-type
- port-qual
- circuit-id
- administrative-state
- logical-connection-point

Node-id, circuit-pack-name and port-name are mandatory. All other parameters are optional.

Refer to “Create Shelf Entities” in the degree section, Section 7.2.1.2.1, for a description of the provisionable and read-only attributes for shelves. Differences specific to SRGs are described below.

- For ROADM SRGs, the port-qual can take the value of roadm-external, roadm-internal, or the attribute is not present. The general guideline is that the ROADM SRG add/drop port would be set to roadm-external and any internal SRG ports on the data path would be set to roadm-internal. For all other ports, the port-qual is not present or optionally set to roadm-internal.

- The logical connection point is used to support the mapping at the controller level between the device model and network model.
    - For the ROADM SRG add/drop port, the logical connection point should be set to the format “SRG<n>-PP<m>”, where <n> is the SRG number and <m> is the add/drop port pair identifier. For example for SRG 1 add/drop port #7 would have the logical connection point set to SRG1-PP7.
    - For the ROADM SRG internal ports that connect to another degree or SRG, the logical connection point should be set to the format “SRG<n>-CP-RX”, “SRG<n>-CP-TX” or “SRG<n>-CP-TXRX” depending on if the port-direction is unidirectional receive, unidirectional transmit, or bidirectional, respectively, where <n> is the SRG number. For example for SRG 1 with unidirectional ports that connect to degree 1, there should be two internal ports – one with logical connection point = “SRG1-CP-RX” and the other with logical connection point = “SRG1-CP-TX”

Read-only data:

- Port-wavelength-type is set to wavelength for the ROADM SRG add/drop port. This attribute may not be present for other ports.
- Supported-interface-capability is used to indicate which interface(s) can be built on the port. The ROADM SRG add/drop port, the list of interface capabilities include: if-OCH.
- The roadm-port capabilities details the minimum and maximum aggregate powers supported by the port. This field applies to the ROADM SRG add/drop ports.

Main actions sequence:

edit-config (/org-openroadm-device/circuit-packs[circuit-pack-name]/ports) nc:operation=merge

Edit the port configuration as necessary.

Set the port-name

Set the port-type to the vendor-specific port type

Set the port-qualifier and logical-connection-point as above. Note that some ports may not have these attributes set.

Set the administrative-state of the port to inService

End of sequence

7.2.2.3 Create Shared Risk Group Entity

Creates new shared-risk-group entity that defines the circuit-packs associated with the SRG.

Input parameters:

- node-id
- srg-number
- list of circuit-packs associated with the SRG

Node-id, srg-number, and circuit-pack-list are mandatory.

- The circuit-packs container lists all circuit-packs associated with the SRG.
    - Note that if there are shared circuit-packs that are used for more than one SRG (or degree), then that circuit-pack should also be listed under the new SRG as well as the old SRG (or degree).

Read-only data:

- The max-add-drop-ports identifies the maximum number of SRG add/drop ports supported by the SRG, regardless if all of the circuit-packs associated with the maximum SRG configuration are provisioned.
- The current-provisioned-add-drop-ports identify the maxiumum number of SRG add/drop ports supported by the SRG based on the current provisioning of the SRG.

Main actions sequence:

edit-config (/org-openroadm-device/shared-risk-group) nc:operation=create

Set the SRG attributes.

Set the srg-number

Set the list of circuit-packs associated to the SRG

End of sequence

7.2.2.4 Create Physical Link Entities

Creates the physical link entities that describe the internal ROADM fibering and/or cabling within the SRG, between the SRG and common units, between the SRG and other degrees, and between the SRG and other existing SRGs.

Input parameters:

- node-id
- physical-link-name
- source and destination port identifiers

Node-id, physical-link-name, source circuit-pack-name, source port-name, destination circuit-pack-name and destination port-name are mandatory.

- Physical links are always unidirectional
    - Source should indicate the TX port
    - Destination should indicate the RX port
- Physical links can represent a single bidirectional cable (e.g., Ethernet cable). In this case, the physical link is still modeled as two unidirectional entities.
- Physical links can also represent MPO connections. Both the physical connector and the logical connectors (child ports within the MPO) can be modeled

Main actions sequence:

edit-config (/org-openroadm-device/ physical-link) nc:operation=create

Set the physical link end points.

Set the physical-link-name

Set the physical link source circuit-pack-name and port-name (TX)

Set the physical link destination circuit-pack-name and port-name (RX)

End of sequence

This completes the commissioning of a new SRG.

7.2.3 Xponder Growth

Xponders represent the transponder, muxponders and other xponder types supported by the network element.

Xponders have network port(s) that connects to the ROADM's SRG add/drop ports.

Xponders have client port(s) that connects to Routers or other client equipment.

The deployment of xponders is similar to the deployment of degrees and SRGs with the exception there is not an explicit "xponder" container. In order to deploy new xponder equipment, the following provision would take place:

- Provision new shelves as necessary
- Provision new circuit packs that will support the new xponder, and configure the ports associated with the circuit packs
  - Including setting the logical connection point attributes (against the port entities)
- Provision the xponder container for the new transponder, muxponder, etc. which include:
  - The xponder type (xpdr-type)
  - For regenerators, indicates if the regenerator supports wavelength change using the recolor attribute
  - The list of network and client ports associated with the xponder.
  - The equipment SRG shared risk (eqpt-srg-id) associated with the ports on the xponder. This identifier is a locally significant identifier (not globally unique) that identifies the ports that share a common equipment SRG when the ports have the same eqpt-srg-id.
- Provision the new physical links associated with the xponder as applicable. The physical links would be needed if the xponder had multiple circuit packs with physical cabling between the circuit-packs.

7.2.3.1 Create Shelf Entities

Creates new shelf entities associated with a xponder augmentation, if required.

Input parameters:

- node-id
- shelf-name
- shelf-type

- rack location
- shelf position within the rack
- administrative state
- equipment state
- due-date

Node-id, shelf-name and shelf-type are mandatory. All other parameters are optional.

Refer to “Create Shelf Entities” in the degree section, Section 7.2.1.1, for a description of the provisionable and read-only attributes for shelves.

edit-config (/org-openroadm-device/shelves) nc:operation=create

Set the shelf-type and other shelf attributes.

Set the shelf-name

Set the shelf-type to the vendor-specific shelf type

Set the administrative-state of the shelf to inService

Set the rack, shelf-position, equipment-state and due date, if desired. These attributes are for tracking purposes only.

End of sequence

7.2.3.2 Create Circuit-Pack Entities

Creates new circuit-pack entities associated with a xponder augmentation.

Input parameters:

- node-id
- circuit-pack-name
- circuit-pack-type
- circuit-pack-product-code
- circuit-pack-mode
- shelf
- slot
- subSlot
- parent circuit-pack-name
- parent cp-slot-name
- administrative state
- equipment state
- due-date

Node-id, circuit-pack-name, circuit-pack-type, shelf, slot are mandatory. All other parameters are optional.

Refer to “Create Circuit-Pack Entities” in the degree section, Section 7.2.1.2, for a description of the provisionable and read-only attributes for shelves.

Main actions sequence:

edit-config (/org-openroadm-device/circuit-packs) nc:operation=create

Set the circuit-pack-type and other circuit-pack attributes.

Set the circuit-pack-name

Set the circuit-pack-type (mandatory) and circuit-pack-product-code (optional) to the vendor-specific values

Set the circuit-pack-mode as necessary based on vendor-specific values

Set the shelf, slot and subSlot to identify the location of the circuit-pack.

Also set the parent-circuit-pack circuit-pack-name and cp-slot-name if this circuit-pack plugs into another circuit pack.

Set the administrative-state of the shelf to inService

Set the equipment-state and due date if desired. These attributes are for tracking purposes only.

End of sequence

7.2.3.2.1 Configure Circuit-Pack Port Entities

This substep configures the port entities associated with the circuit pack. Ports are auto-created upon creation of the circuit-pack.

Note: Recommendation to use the “merge” operation on editing port attributes.

Input parameters:

- node-id
- circuit-pack-name
- port-name
- port-type
- port-qual
- circuit-id
- administrative-state
- logical-connection-point

Node-id, circuit-pack-name and port-name are mandatory. All other parameters are optional.

Refer to “Create Shelf Entities” in the degree section, Section 7.2.1.2.1, for a description of the provisionable and read-only attributes for shelves. Differences specific to SRGs are described below.

- For xponders, the port-qual can take the value of xpdr-network, xpdr-client, or the attribute is not present. The general guideline is that the xponder network ports would be set to xpdr-network and xponder client ports would be set to xpdr-client. For all other ports, the port-qual is not present.
- The logical connection point is used to support the mapping at the controller level between the device model and network model.

- For the xponder network port, the logical connection point should be set to the format XPDR<n>-NETWORK<m>.
- For the xponder client ports, the logical connection point should be set to the format XPDR<n>-CLIENT<m>.
- Where <n> is the logical transponder/muxponder/xponder number, and <m> are the network and client ports that make up the transponder/muxponder/xponder. Typically, the network <m> will be 1 for transponders and muxponers. For transponders, the client <m> will be 1 and for muxponders the client <m> will be 1 .. m where m = max number of client muxponder ports.

Read-only data:

- Port-wavelength-type is set to wavelength for the xponder network and client ports. This attribute may not be present for other ports.
- Supported-interface-capability is used to indicate which interface(s) can be built on the port.
  - The xponder network port, the list of interface capabilities include: if-OCH. (Note: MSA v1.2.1 does not support an explicit identity for if-OTU4 nor if-ODU4)
  - The xponder client port, the list of interface capabilities may include: if-100GE and if-OCH. The client supports both 100GE and OTU4. (Note: MSA v1.2.1 does not support an explicit identity for if-OTU4 nor if-ODU4)
- The transponder-port capabilities details the minimum and maximum aggregate powers supported by the port. This field applies to the xponder network and client ports.

Main actions sequence:

edit-config (/org-openroadm-device/circuit-packs[circuit-pack-name]/ports) nc:operation=merge

Edit the port configuration as necessary.

Set the port-name

Set the port-type to the vendor-specific port type

Set the port-qualifier and logical-connection-point as above. Note that some ports may not have these attributes set.

Set the administrative-state of the port to inService

End of sequence

7.2.3.3 Create Xponder Entity

Creates new xponder entity that defines the xponder type and the ports associated with the xponder.

Input parameters:

- node-id
- xpdr-number
- xpdr-type

- list of ports associated with the xponder
- eqpt-srg-id for the ports

Node-id, xpdr-number, and xpdr-port list are mandatory.

- The xpdr-type can be tpdr (transponder), mpdr (muxponder), switch (OTN switch or switchponder), regen (regenerator) or regen-uni (unidirectional regenerator)
- The xpdr-port container lists all ports associated with the xponder.
- Eqpt-srg-id identifies the shared risk of the ports based on common equipment utilization. This value is locally significant to the node.

Read-only data:

- Recolor indicates if a regenerator supports wavelength recoloring. This only applies if xpdr-type is regen or regen-uni.

Main actions sequence:

edit-config (/org-openroadm-device/xponder) nc:operation=create

Set the xponder attributes.

Set the xpdr-number

Set the xpdr-type

Set the list of xpdr-ports associated to the xponder including the eqpt-srg-id associated with the port

End of sequence

7.2.3.4 Create Physical Link Entities

Creates the physical link entities that describe the internal xponder fibering and/or cabling, if applicable.

Input parameters:

- node-id
- physical-link-name
- source and destination port identifiers

Node-id, physical-link-name, source circuit-pack-name, source port-name, destination circuit-pack-name and destination port-name are mandatory.

- Physical links are always unidirectional
  - Source should indicate the TX port
  - Destination should indicate the RX port
- Physical links can represent a single bidirectional cable (e.g., Ethernet cable). In this case, the physical link is still modeled as two unidirectional entities.
- Physical links can also represent MPO connections. Both the physical connector and the logical connectors (child ports within the MPO) can be modeled

Main actions sequence:

edit-config (/org-openroadm-device/ physical-link) nc:operation=create

Set the physical link end points.

Set the physical-link-name

Set the physical link source circuit-pack-name and port-name (TX)

Set the physical link destination circuit-pack-name and port-name (RX)

End of sequence

This completes the commissioning of a new xponder unit(s).

8 MAINTENANCE ACTION USE CASES

8.1 RESET/RESTART AT SYSTEM OR CIRCUIT PACK LEVEL

More detail on resets and restarts will be added in the next version of this Whitepaper, including sections on system-level non-service affecting reset, circuit pack level non-service affecting reset (data plane intact) and circuit pack level power cycle reset (data plane down)

8.2 SET OTU/ODU TTI FIELDS (SAPI/DAPI MESSAGE) FOR CONNECTIVITY MANAGEMENT

More detail on TTI will be added in the next version of this Whitepaper, including describing the mechanism and semantics to set OTU/ODU interface TTI fields needed for connectivity management.

8.3 OPERATE/RELEASE LOOPBACK ON AN INTERFACE

More detail on loopback operaitons will be added in the next version of this Whitepaper, including describing the mechanism and semantics to operate/release loop-backs:

Operator/Release a loop-back toward the external (fiber) connections on a (100G) client interface

Operator/Release a loop-back toward the internal connections on a (100G) client interface

Operator/Release a loop-back toward the external (fiber) connections on an OTU4 network interface

Operator/Release a loop-back toward the internal connections on an OTU4 network interface

8.4 GENERATE/COLLECT DEBUG DUMP FILE(S) FOR IN-DEPTH FAILURE ANALYSIS

To be addressed in Open ROADM V3

8.5 RESTORE REMOTE COMMUNICATIONS TO THE DEVICE FROM CORRUPTED OR INCONSISTENT CONFIGURATION

More detail on restoring communications to a device will be added in the next version of this Whitepaper, including describing the mechanism to restore communications to the device from corrupted or inconsistent configuration, assuming power cycle of the processor or management blade is ineffective in this case.

8.6 Re-sync with northbound controller/applications after extended period of communication loss

More detail on controller resync will be added in the next version of this Whitepaper, including describing the mechanism and semantics for the controller to re-sync up with the device after extended period of communication loss, to the extent that network buffer is overflow and controller has to recollect the events from device log file(s).

8.7 Missing topics (to be addressed in future version of Whitepaper)

- What does “admin-state” actually means
    - Suppression of alarms only, no impact to service state
        - "inService" - Services requested to be up, alarm reporting is active, no maintenance operation allowed.
        - "outOfService" - Services requested to be up, alarms suppressed, no maintenance operation allowed.
        - "maintenance" - Services requested to be up, alarms suppressed, maintenance operation allowed.
- What failures should drive operational states
- Alarm suppression rules and guidelines
- RPCs
    - Disable automatic shutoff
    - Connection port trail
- Wavelength map descriptions
    - 0 is reserved and means “laser off”
- Min-power / max-power interpretations and usage
- Internal link usage and examples’
- Span loss changes on existing system

9 APPENDIX A: MANIFEST FILE FOR SOFTWARE DOWNLOAD AND DATABASE OPERATIONS

9.1 INTRODUCTION

This section describes the manifest file for software download, database backup and database restore operations. It includes details of how an Open ROADM implementation would specify the file transfer, staging, backup, restore and activation of software loads and database archives.

The manifest file allows vendor to specify vendor-specific behavior in a common format for software download and database operations. This allows the controller to automatically adjust procedures to the vendor specific requirements.

The manifest file is expected to be provided out-of-band (i.e., the file is not provided directly with the network element but provided offline by the vendor to the controller).

There may be multiple versions of the manifest file distinguished by the vendor, model and sw-version. It is expected that the database operations would apply to the current software version running on the Open ROADM device. For software download operations, the sw-version would be the software load that the operation would upgrade to (e.g., sw-version = "2.0" when the current software version is "1.2.1"). Note that the sw-version is the vendor software version and not necessarily the Open ROADM version.

9.1.1 Software Download

Software download is expected to take place as a series of operations as specified in the manifest file. There should be a manifest file created for the new software load. The manifest file specifies one or more instruction-sets. Each instruction-set would apply to one or more from-sw-versions that would represent the current sw-version running on the device. This allows different instructions based on the upgrade from different current software versions.

The typical order of operations for software download is as follows:

1) Transfer one or more software files from an SFTP server to the device.
2) Stage the software file
    a. As part of staging, additional software files may be transferred from the SFTP server to the device. The details of what software files to transfer would be vendor-dependent and the file information should be in the software file that was staged. These software files can be in a flat or hierarchical sub-directory structure on the SFTP server.
3) If needed, delete the software file.
4) Steps 1-3 may repeat if the device storage does not support transferring all software files at once.
5) Activate the software with an optional validation timer
6) Typically it is expected that the device would reboot at this time (reboot should be automatic).

7) When the device comes up, the operator would evaluate the software load in the validation time (if supported and specified in the manifest file).
    a. During this time, access to the database may be restricted to allow for rollback. Thus, some provisioning activities (e.g., new service activation) may be restricted.
8) If the software load is acceptable, the operator would cancel the validation time and accept the software load.
9) The device is now fully functional at the new software release

The operator may also cancel the validation and not accept the new software load. Under this case, the device will reboot and revert back to the original software load.

Note that if the device did not support a validation period, then reversion to the original software load may not be supported.

9.1.2 Database Backup

The database backup operation will backup the current database from the device and transfer it to an SFTP server for offline storage.

The typical order of operations for database backup is as follows:

1) Initiate the database backup operation and specify the filename. The device will create the database file on the device for file transfer.
2) When the database file is completed (either as a synchronous command or a an asynchronous command with an event notification), the controller would transfer the file from the device to the SFTP server.
3) The controller would then delete the database file from the device.

The database backup manifest file defines two variables: `__LOCAL-FILE-PATH` and `__REMOTE-FILENAME`

- __LOCAL-FILE-PATH is the local filename (and path) for the creation of the database backup file
- __REMOTE-FILENAME is the remote filename for storage on the SFTP server to be used in the transfer-file RPC

It is expected that the user or controller will provide the value of these attributes.

9.1.3 Database Restore

The database restore operation will restore a previously saved (backup) database file from an SFTP server.

The typical order of operations for database restore is as follows:

1) Transfer the previously backed up database file from the SFTP server to the device
2) Initiate a database restore operation on the device.
3) Once the database restore operation completes, the database file can be deleted from the device.
4) Activate the database restored file with an optional rollback timer (similar to the software download validation timer)

5) Typically it is expected that the device would reboot at this time (reboot should be automatic).
6) When the device comes up, the operator would evaluate the device with the restored database in the rollback time (if supported and specified in the manifest file).
    a. During this time, access to the database may be restricted to allow for rollback. Thus, some provisioning activities (e.g., new service activation) may be restricted.
7) If the database restore is acceptable, the operator would cancel the rollback timer and accept the database.
8) The device is now fully functional at the point of the restored database file

The operator may also cancel the rollback timer and not accept the database restore. Under this case, the device will reboot and revert back to the database in the state before the database restore.

Note that if the device did not support a rollback period, then reversion to the previous database may not be supported.

The database restore manifest file defines three variables: `__LOCAL-FILE-PATH`, `__REMOTE-FILENAME`, and `__NODE-ID-CHECK`

- __LOCAL-FILE-PATH is the local filename (and path) where the database file will be stored on the device for restoration
- __REMOTE-FILENAME is the remote database filename on the SFTP server to be used in the transfer-file RPC to the device
- __NODE-ID-CHECK is to have the device validate that the current node-id and the node-id in the database file match to ensure the correct database file is being restored

It is expected that the user or controller will provide the value of these attributes.

9.2 MANIFEST FILE YANG MODULE

9.2.1 Tree View

The Manifest file YANG module tree view is included as part of the Open ROADM v2 YANG repository (under the common directory).

9.2.2 YANG module (this module will be finalized and provided as part of Open ROADM Version 2)

The Manifest file YANG module is included as part of the Open ROADM v2 YANG repository (under the common directory).