AW-SDX 2.0 Controller - Topology Data Model Specification

Versioning:

Version	Date	Description
1.0.0	07/09/2020	Specifies the Physical topology specification.

Introduction

The AW-SDX 2.0 Controller (a.k.a. SDX Controller) needs topology data from each participant Open Exchange Point (OXP) to compute the AtlanticWave-SDX topology database. The AtlanticWave-SDX topology database will have a few purposes:

1. Finding the best path between endpoints based on multiple user criteria;
2. Exporting the AtlanticWave-SDX topology database to external science applications to support their operations;
3. Monitoring the quality of the AtlanticWave-SDX network resources.

Each OXP's SDX Local Controller (a.k.a. SDX-LC[1]) must feed the SDX Controller with topology data using the data model specified in this document. The SDX Controller uses the topology data received from each SDX-LC to create a network graph. This network graph must be capable of supporting Constrained Shortest Path First (CSPF) algorithms using user-provided optional requirements, such as:

- Shortest propagation delay
- Fewest number of OXPs in the path
- Avoiding specific vendors or countries in the path
- Most reliable path
- Highest/specific bandwidth available

The approach to collect the topological data from the OXP Orchestrator (a.k.a. OXPO) is outside the scope of this document. However, to clarify for the reader's understanding, there are three main approaches for the SDX-LC to gather topology data from an OXPO:

1) The OXPO supports the AW-SDX 2.0 Topology Data Model specification (this document) and pushes the topology data to the SDX-LC via OpenAPI interfaces;

[1] The SDX Local Controller (SDX-LC) is a major component of the AW-SDX 2.0 architecture. A design objective of the SDX-LC is to abstract the distinct physical characteristics of a participant OXP.

2) The OXPO supports the AW-SDX 2.0 Topology Data Model specification, but the SDX-LC has to pull the topology data using the OXPO's API;
3) The OXPO does not support the AW-SDX 2.0 Topology Data Model specification. In this case, each SDX-LC has to pull topology data using the OXPO's API and existing OXPO's topology data model, and then convert the retrieved topology data to the AW-SDX 2.0 Topology Data Model specification.

The main outcome of the proposed topology data model is to support science drivers, such as Open Science Grid (OSG). With the available AW-SDX topology, OSG can make cache population decisions based on network characteristics, for instance, the closest OSG Cache node to an OSG Origin node.

Requirement Levels

The Topology Data Model specification provides the requirements and restrictions for the topology objects and their attributes. To avoid confusion and misinterpretation, the key words "must", "must not", "required", "shall", "shall not", "should", "should not", "recommended", "may", and "optional" in this document are to be interpreted as described in IETF RFC 2119 [https://www.ietf.org/rfc/rfc2119.txt].

Privacy and Security

For privacy and security reasons, since the SDX topology database will be available to SDX users, OXP operators will be able to define which topology attributes must be removed or anonymized before being exported by the SDX Controller to SDX users. This control will be made by adding to each topology object a property named "private" with the name of the properties to be considered private. When the private property is not empty, the SDX Controller must filter out the content of the attributes listed before exporting the topology to external users, including science drivers. This specification recommends that all attributes be public. Anonymization is outside of the scope of this specification.

Autonomic Network Architecture Properties

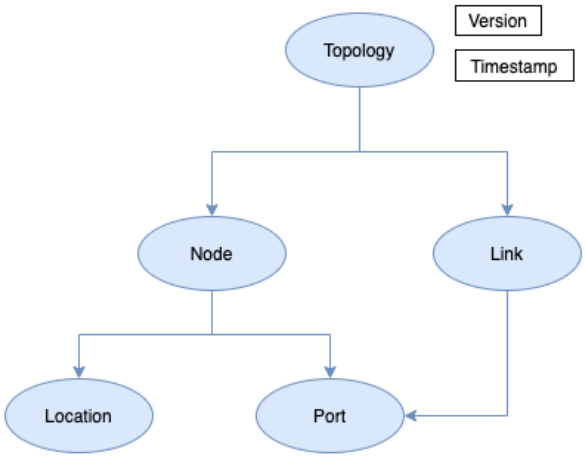
The AW-SDX 2.0 Topology Data Model specification is not affected by the Autonomic Network Architecture (ANA) since it is not a service or function but a data structure. The SDX Controller components involved in the topology data exchange should consider two ANA disciplines: *self-configuration* and *self-protection*. In the context of the topology data exchange, *self-configuration* applies when the components involved (OXPO and SDX-LC) need to discover their configurations dynamically. The connections between OXPO and SDX-LC should not be statically defined, for instance, via IP addresses. For *self-protection*, the SDX-LC should be protecting itself from bursts of topology updates coming from the OXPO due to link flaps and other updates at the OXP level. Neither should the SDX-LC burst updates to the SDX Controller. However, the implementation and enforcement of the ANA disciplines are outside of the scope of this document.

Creating the topology

The version 1.0 of the AW-SDX 2.0 Topology Data Model specification focuses on the physical layer. Different versions will be created to add service-specific properties that will leverage the AW-SDX 2.0 Topology Data Model.

The AW-SDX 2.0 Topology Data Model specification was based on the GRENML model [https://code.canarie.ca/gren_map] to simplify future adoption. AW-SDX 2.0 extends the GRENML with properties to address its needs. Like the GRENML model, this specification is based on JSON, which stands for JavaScript Object Notation. JSON was chosen because of its simplicity to be read by humans and machines.

The diagram below will be used to explain the relationship between the topology objects.



Each topology object will be described in detail in the next subsections as well as its dependencies.

Topology Object

The Topology object is represented by the following attributes or properties:

- **name**
- **id**
- **version**
- **model version**
- **timestamp**
- **nodes**
- **links**

The **name** attribute is a string that represents the OXP name. **name** will be used to display the OXP name within the SDX web user-interface (UI). **name** is operator-defined. **name** must be limited to 30 (thirty) ASCII characters without special characters.

The **id** attribute is a Uniform Resource Name (URN) used to uniquely identify the OXP in the AW-SDX context. To guarantee the URN is unique, since two OXPs could have the same acronym or name, the OXP operator must use the OXP website or opertor's URL as part of the URN. The topology ID will follow the format: *"urn:sdx:topology:<oxp_url>"*, for instance *"urn:sdx:topology:amlight.net" for AmLight, "urn:sdx:topology:rnp.br" for SAX and "urn:sdx:topology:tenet.ac.za" for ZAOXI.*

The Topology Object has two properties to indicate changes to OXPO's topology and counters: **version** and **timestamp**:

- **version** is an unsigned integer and starts as 1 when the OXP is added to the AW-SDX and it will increment by 1 every time that there is an *administrative* change that MODIFIES the topology. Administrative changes are those created by the network operators. Below are some examples of physical changes that modify the topology and will increase the version number:
    - A new node is added or removed by the OXP operator..
    - A new link is added or removed.
    - A new user port is configured.
    - When the location of an item changes (node moved to a different address).
    - A link or node is set to *maintenance* mode and becomes unusable.
    - A link that was reconfigured to go through a different location (city or country).
    - A change on the port/interface bandwidth via configuration

- **timestamp** starts with a timestamp of when the topology was created. **timestamp** changes every time the topology changes (administrative or not), and when link counters, state, or status are updated:
    - A port, node, or link that changed from up to down or down to up because of a problem (power outage, fiber cut, damaged transceiver).
    - A change in the bandwidth utilization of a link.
    - An increase in packet loss or drops on a port.

version and **timestamp** will be used by the SDX-LC and SDX Controller to know how to process the topology update received and what kind of changes to expect.

The **model_version** attribute describes which topology data model specification version is in use, since updates might happen. The current version is the string "1.0.0".

The **nodes** and **links** attributes are list attributes. **nodes** is a list of Node objects and **links** is a list of Link objects. Both Node and Link objects are described in this specification.

Restrictions:

1. **name, id, version, model_version, timestamp, nodes, links** attributes must be provided when creating the topology data.
2. **name, id, version, model_version, timestamp, nodes, links** attributes must not be empty.
3. **name** must be an ASCII string with length shorter than 30 characters.
4. **name** must not include special characters.
5. **id** must follow the format *"urn:sdx:topology:<url>"* where *<url>* is the OXP's website main domain name.
6. **version** must be an unsigned integer that starts with 1 and increments by 1 when there are administrative topology changes.
7. **version** will be 0 only when the SDX-LC is added for the first time to AW-SDX.
8. **timestamp** attribute must be the UNIX timestamp and be updated when there are non-administrative topology changes.
9. **timestamp** must be a string and must follow the format "YYYYMMDD-HHmmSS", where YYYY stands for four-digit year, MM stands for two-digit month, DD stands for two-digit day of the month, HH stands for two-digit hour using 24 hours, mm stands for two-digit minutes, and SS stands for two-digit seconds. And "-" (dash) separates days from time.
10. **timestamp** must be based on UTC.
11. **model_version** must be "1.0.0".
12. **nodes** attribute must be a non-empty list of Node objects.
13. **links** attribute must be a non-empty list of Link objects.
14. The Topology Object has no attribute that can be set to private since all attributes are essential for the SDX operation.

Example:

Example of a topology object, where the attributes **nodes** and **lists** are removed to simplify the representation. Examples of **nodes** and **lists** are provided in the next subsections.

```
{
      "name": "AmLight-OXP",
      "id": "urn:sdx:topology:amlight.net",
      "version": 2,
      "time_stamp": "20210707-211940",
      "model_version": "1.0.0",
      "nodes": [ {...}, {...} ],
      "links": [ {...}, {...} ]
}
```

Node Object

The Node object is represented by the following attributes or properties of a network device, such as a switch or a router:

- **name**
- **id**
- **location**
- **ports**

The **name** attribute is a string that represents the node name. **name** will be used to display the node name within the SDX web user-interface (UI). **name** is operator-defined. **name** must be limited to 30 (thirty) ASCII characters without special characters.

The **id** attribute is a Uniform Resource Name (URN) used to uniquely identify the node in the AW-SDX context. The OXP operator is responsible for guaranteeing the uniqueness of the URN. The node ID will follow the format: *"urn:sdx:node:<oxp_url>:<node_name>".* The <*oxp_url*> is the OXP website or operator's URL, the same used for the Topology Object. The *<node_name>* represents the name of the node and should be derived from the attribute **name**, entirely or a subset of it. It is up to the OXP operator to make this definition. Some examples of IDs:

- *"urn:sdx:node:redclara.net:switch_01"*
- *"urn:sdx:node:amlight.net:juniper_router01"*
- *"urn:sdx:node:sax.net:s1"*
- *"urn:sdx:node:tenet.za.ac:tor"*

location is used to represent the physical location of the node. The Location object is used and it must not be empty.

ports is a list of ports that belong to the node. The content for **ports** is a list of Port objects. Each port has a set of attributes to reflect the current network state and status. The Port Object is described in the next sections.

Restrictions:

1. **name, id, location,** and **ports** must be provided when creating the node object.
2. **name, id, location,** and **ports** must not be empty.
3. **name** must be an ASCII string with length not to exceed 30 characters.
4. **name** must not include special characters.
5. **id** must follow the format *"urn:sdx:node:<oxp_url>:<node_name>"* where *<oxp_url>* is the OXP's website or operator's website domain name.
6. **location** must be a Location object.
7. **ports** must be a non-empty list of Port Objects.

8. The Node Object has no attributes that can be set to private since all attributes are essential for the SDX operation. However, the Location Object attributes can be manipulated to not provide the exact location. More details can be found in the Location Object section.

Example:

Example of a Node object, where the attribute **ports** is removed to simplify the representation. Examples of **ports** are provided in the Port Object subsection.

```
{
            "name": "switch01",
            "id": "urn:sdx:node:amlight.net:switch01",
            "location": {
                  "address": "Miami,FL,USA",
                  "latitude": "25.761681",
                  "longitude": "-80.191788"
            },
            "ports": [ {...}, {...} ]
}
```

Port Object

The Port object is represented by the following attributes or properties of a network device's port (or interface):

- **name**
- **id**
- **node**
- **type**
- **mtu**
- **nni**
- **status**
- **state**
- **services**

The **name** attribute is a string that represents the name of the port and it will be used to display the node name within the SDX portals. It is operator-defined. The only restriction created for the **name** attribute is its length of 30 (thirty) characters.

The **id** attribute is a Uniform Resource Name (URN) used to uniquely identify the port in the AW-SDX context. The OXP operator is responsible for guaranteeing the uniqueness of the URN. The port ID will follow the format: *"urn:sdx:port:<oxp_url>:<node_name>:<port_name>".*

The <*oxp_url*> is the same URL used to create the Topology Object ID. The <*node_name*> is the same URL used to represent the Node Object ID. The <*port_name*> represents the name of the port and should be derived from the attribute **name**, entirely or a subset of it. It is up to the OXP operator to make this definition. Some examples of valid port **id**s are:

- *"urn:sdx:port:amlight.net:switch_01:port_1"*
- *"urn:sdx:port:amlight.net:tor:131"*
- *"urn:sdx:port:rnp.br:juniper_router01:amlight_100G"*
  "urn:sdx:port:zaoxi.ac.za:s1:port_to_brazil"

The **node** attribute is a Uniform Resource Name (URN) used to uniquely identify which node the port belongs to in the AW-SDX context.

The **type** attribute represents the technology and bandwidth of the physical port (or interface). **type** is an enum with only one value acceptable. For version 1.0.0 of the Topology data model specification, the only technology supported is Ethernet. The **type** enum is 100FE, 1GE, 10GE, 25GE, 40GE, 50GE, 100GE, 400GE, and *Other.* When the value *Other* is chosen, no bandwidth guaranteed services will be supported in this port. The value *Other* was created to enable flexibility when the port is not on the enum. In case *Other* becomes recurrent, the SDX team must increase the specification subversion and add the correct bandwidth to the **type** enum. The specification version table must be updated with such info.

The **mtu** attribute is the port's maximum transmission unit (MTU) or the max size of a packet supported by the port in bytes. **mtu** is a kind of attribute that could become a challenge to dynamically retrieve from a node. For this reason, this attribute is considered optional, but recommended.

The **nni** attribute is used to describe whether the port is a Network to Network Interface (NNI). NNI will be used to qualify the port as an endpoint of an intra-domain (internal) or an inter-domain (external) link. If **nni** is not set (an empty string), the port is considered an UNI (User-Network Interface), meaning a user port. From the SDX perspective, a R&E network that is not operated by the AtlanticWave-SDX Controller is considered a user port. If the port is a NNI, then the **nni** attribute must be set with the Link ID (URN to represent the Link), if it is an intra-domain link; otherwise, the **nni** attribute must be set with the remote OXPs Port ID, if it is an inter-domain. For example, if the port is a NNI part of the link "*Novi03/p2_Novi02/p3*" at the AmLight OXP, then the **nni** attribute is set to *"urn:sdx:link:amlight.net:Novi03/p2_Novi02/p3".* If the port is an AmLight port connected to ZAOXI OXP, via link named "*sacs_sub_link"* then the **nni** attribute on the AmLight topology side is set to *"urn:sdx:link:zaoxi.ac.za:sacs_sub_link".*

The **status** attribute represents the current operational status of the port. **Status** is an enum with the following values: "down" if the port is not operational, "up" if the port is operational, 'error' when there is an error with the interface.

The **state** attribute represents the current administrative state of the port. **State** is an enum with the following values: "enabled" if the port is in administrative enabled mode, "disabled" when the

port is in administrative disabled mode (a.k.a. *shutdown)*, and "maintenance" when in under maintenance (not available for use).

The **services** attribute describes the services supported and their attributes. **services** is set as an empty string when no services are supported or declared for this port. The usage of **services** will be available in future versions of this specification.

Restrictions:

1. **name, id, node, type, status,** and **state** must be provided when creating the node object.
2. **name, id, node, type, status,** and **state** must not be empty.
3. **name** must be an ASCII string with length not to exceed 30 characters.
4. **name** must not include special characters.
5. **id** must follow the format *"urn:sdx:port:<oxp_url>:<node_name>:<port_name>"* where *<oxp_url>* is the OXP's website or operator's website domain name, *<node_name>* is the node's name, and *<port_name>* is the port's name.
6. When **mtu** is not set, the port's MTU is considered to be 1,500 bytes.
7. **mtu** is an integer with minimum value of 1,500 and maximum of 10,000.
8. When **nni** is not set (empty string), the port is considered an UNI.
9. **status** is an enum and only supports one of the following values: "up", "down", or "error"
10. **state** is an enum and only supports one of the following values: "enabled", "disabled", or "maintenance"
11. From the Port Object, **mtu**, **status** and **state** can be set as private attributes although it is highly recommended to keep them public.

Example:

```
{
            "id": "urn:sdx:port:amlight.net:s3:s3-eth2",
            "name": "s3-eth2",
            "node": "urn:sdx:node:amlight.net:s3",
            "type": "10GE",
            "mtu": 10000,
            "status": "up",
            "state": "enabled",
            "nni": "urn:sdx:link:amlight.net:Novi03/2_s3/s3-eth2",
            "services": "",
            "private": ["state", "mtu"]
}
```

Location Object

The Location object is represented by the following attributes or properties of a physical location:

- **address**
- **latitude**
- **longitude**

The **address** attribute is a string that represents the physical location. It can be a full address, the name of a city or a country. **address** will be used to display a node's address within the SDX web user-interface (UI). **address** is operator-defined. **address** must be limited to 255 (two hundred and fifty five) ASCII characters.

The **latitude** attribute is the geographic coordinate that specifies the north–south position of a node on the Earth's surface.

The **longitude** attribute is the geographic coordinate that specifies the east–west position of a node on the Earth's surface.

Restrictions:

12. **address, latitude,** and **longitude** must be provided when creating the Location object.
13. **address, latitude,** and **longitude** must not be empty.
14. **latitude** and **longitude** must be represented as a string with a float number from -90.0 to 90.0.
15. **address** must be an ASCII string with length shorter than 255 characters.
16. For privacy reasons, **address, latitude,** and **longitude** can be provided with content that doesn't show the exact location of a node.

Examples:

```
{
            "address": "Miami, FL, USA",
            "latitude": "25",
            "longitude": "-80"
}

{
            "address": "Equinix MI3, Boca Raton, FL, USA",
            "latitude": "26.35869",
            "longitude": "-80.0831"
}
```

Links Object

The Link object is represented by the following attributes or properties of a network connection between two network devices:

- **name**
- **id**
- **ports**
- **type**
- **bandwidth**
- **residual_bandwidth**
- **latency**
- **packet_loss**
- **availability**
- **status**
- **state**

The **name** attribute is a string that represents the name of the link and it will be used to display the link name within the SDX web user interface (UI). It is operator-defined. The only restriction created for the **name** attribute is its length of 30 (thirty) characters.

The **id** attribute is a Uniform Resource Name (URN) used to uniquely identify the link in the AW-SDX context. The OXP operator is responsible for guaranteeing the uniqueness of the URN. The link ID will follow the format: *"urn:sdx:link:<oxp_url>:<link_name>"*. The <*oxp_url*> is the same URL used to create the Topology Object ID. The <*link_name*> represents the name of the link. Some examples of valid link **id**s are:

- *"urn:sdx:link:amlight.net:saopaulo_miami"*
- *"urn:sdx:link:ampath.net:lsst_100G"*
- *"urn:sdx:link:rnp.br:ana_100G_dc_paris"*
- *"urn:sdx:link:zaoxi.ac:link_to_amlight"*

The **ports** attribute lists the Port object IDs that create the link. For the scope of the AtlanticWave-SDX, all links will be point-to-point. However, since the **ports** attribute is a list, the SDX team has the flexibility for future specifications. For the topology data model specification version "1.0.0", the **ports** attribute has two Port objects only.

The **type** attribute describes if a Link object represents an intra-OXP link (internal) or an inter-OXP link (external). **Type** is an enum with acceptable values either "intra" for intra-OXP or "inter" for inter-OXP.

The **bandwidth** attribute describes the maximum capacity in terms of bandwidth of a Link object. The bandwidth of a link could be the interface's bandwidth or a leased capacity provided by a carrier to the OXP. bandwidth must represent how much bandwidth capacity is accessible to be used by the SDX community in units of Gbps. For instance, a 50 Gbps link must have the attribute **bandwidth** set to 50. **bandwidth** accepts a fraction value. For instance, for a 500

Mbps or 3250 Mbps link, **bandwidth** must be converted to Gbps, with values 0.5 and 3.25 respectively.

The **residual_bandwidth** attribute describes the average bandwidth available for the Link object. The representation of the **residual_bandwidth** must be provided in percentage from 0 to 100 of the **bandwidth** attribute. The OXP operator is responsible for defining the time interval to be based, for instance, the last 30 days, the last day, or the last 12 hours. This specification *suggests* that **residual_bandwidth** to be based on the last 7 to 14 days for better accuracy and decision making.

The **latency** attribute describes the delay introduced by the Link object in milliseconds in the end-to-end path. In optical networks or lit services, latency represents the propagation delay between two endpoints and tends to be deterministic. In Carrier Ethernet and MPLS networks, latency reports the service delay and varies according to the carrier's network state at the moment. **latency** accepts a fraction value.

The **packet_loss** attribute describes a percentage of packet loss observed for the Link object. The representation of the **packet_loss** must be provided in percentage from 0 to 100. **packet_loss** accepts a fraction value. The OXP operator is responsible for defining the time interval to be based, for instance, the last 14 days, the last day, or the last 12 hours. This specification *suggests* that **packet_loss** to be based on the last 24 hours or less for better accuracy and decision making. This specification leaves it for the OXP operator to decide the approach to retrieve the Link's packet loss. As a suggestion, OXP operators could use OWAMP installed in perfSONAR nodes, IP SLA, OAM, or similar technologies.

The **availability** attribute describes the percentage of time the link has been available for data transmission. Also known as reliability, the **availability** attribute is a metric used by the SDX Controller to select the best path when provisioning and re-provisioning services based on the criticality of the service requested. For instance, real-time and interactive applications should be provisioned using links with the best **availability** possible. The representation of the **availability** must be provided in percentage from 0 to 100. The OXP operator is responsible for defining the time interval and the formula to be used when computing the availability. This specification *suggests* that **availability** to be based on the last 14 days or less for better accuracy and decision making. This specification *suggests* that **availability** takes into consideration both full outage as well as flaps when calculating the resilience of the link.

The **status** attribute represents the current operational status of the link. **Status** is an enum with the following values: "down" if the link is not operational, "up" if the link is operational, 'error' when there is an error with the interface.

The **state** attribute represents the current administrative state of the link. **State** is an enum with the following values: "enabled" if the link is in administrative enabled mode, "disabled" when the link is in administrative disabled mode (a.k.a. *shutdown)*, and "maintenance" when link in under maintenance (not available for use).

Restrictions:

1. **name, id, ports, bandwidth, type, status,** and **state** must be provided when creating the link object.
2. **name, id, ports, bandwidth, type, status,** and **state** must not be empty.
3. **name** must be an ASCII string with length shorter than 30 characters.
4. **name** must not include special characters.
5. **id** must follow the format *"urn:sdx:link:<oxp_url>:<link_name>"* where *<oxp_url>* is the OXP's website or operator's website domain name and *<link_name>* is the link's name.
6. t**ype** is an enum with acceptable values either "intra" for intra-OXP or "inter" for inter-OXP.
7. **bandwidth** must be a numerical value greater than 0 and to be provided as a unit in Gbps.
8. **residual_bandwidth** must be provided as a numerical percentage value from 0 to 100 of the **bandwidth** attribute.
9. **packet_loss** must be provided as a numerical percentage value from 0 to 100.
10. **availability** must be provided as a numerical percentage value from 0 to 100.
11. **residual_bandwidth, latency, packet_loss,** and **availability** must be provided as 100, 0, 0, and 100 respectively when collecting these counters is not possible to the OXP Operator. They accept fraction values.
12. **status** is an enum and only supports one of the following values: "up", "down", or "error".
13. **state** is an enum and only supports one of the following values: "enabled", "disabled", or "maintenance".
14. From the Link Object, **residual_bandwidth**, **latency, packet_loss** and **packet_loss** can be set as private attributes although it is highly recommended to keep them public.

Schemas

The data model schemas in this specification are provided at [1] for easy implementation and validation.

[1] https://github.com/atlanticwave-sdx/datamodel/blob/main/schemas/