## Domain topology description and declaration.

Each domain, proxied by the customized SDX-LC who communicates between the SDX-controller and the domain (1) provisioning system (eg, Kytos) and (2) monitoring system (BAPM).

Therefore, in the whole SDX system, two types of topology models are needed:

1. Domain substrate description model: used by the intra-domain system.
2. Domain declaration/advertisement model: generated/passed by the SDX-LC to the SDX-controller for inter-domain topology assembly to support (a) inter-domain path computation; and (b) inter-domain path monitoring and reconfiguration. It would consist of three types of information: (1) Topology abstraction; (2) network resources available for inter-domain connections (eg, bw, vlan ranges, etc); (3) switching capability (eg, vlan, Q-in-Q, etc).



Fig. 1. Topology Description and Declaration

The conceptual model is depicted by Fig. 1. The AmLight is controlled by its Kytos provisioning system, that uses or generates (via OF topology discovery) an intra-domain topology model that describes its internal and external physical topology. Between Kytos and the SDX-LC, depending on the administrative policy, Kytos may pass to its SDX-LC (1) the **substrate model**, or (2) the **domain declaration model**. In the former case, the SDX-LC generates the domain declaration model.

**Design goals:**

1. The same domain abstraction function is used to generate the declaration model, either executed by the SDX-LC or the domain provisioning system like Kytos.
2. The same base objects/classes/schemas should be used for both models:grenml (<https://code.canarie.ca/gren_map>)

## Domain declaration model.

To maximize the potential performance optimization in computing/reconfiguring the inter-domain connections while respecting the administrative constraints from the underneath domain. This basic model is based on a full mesh abstraction of the domain substrate model, as shown in Fig.1, that consists o>"all domain border switches; (2) virtual intra-domain links between pairs of border switches; (3) inter-domain links to the neighboring domains.

Use the AmLight domain in Fig. 1 as an example, the declaration and performance update information on the two types of virtual links: **link(1,2)** and **link(1,3)**, would be beneficial to more intelligent computation of inter-domain paths.

The topology description is based on 5 object definitions, as shown in Fig. 2, whose JSON schemas are defined in <https://github.com/atlanticwave-sdx/datamodel>.



## Connection and Path model

These two objects are based on the JSON schemas in the above github repo.

The key performance metrics are:

1. Bandwidth (1 - 1,000,000 mbps): required and measured
2. Latency (1ms - 100,000ms: required and measured
3. Packet loss(0-100%): required and measured
4. Class(priority 0-7)
5. Multipath (0 or 1)
6. Backup path (0: no; 1: not disjoint; 2: link disjoint; 3: node disjoint)

## Appendix: Schemas

All the schemas are defined in JSON schema (<https://json-schema.org>).

<https://github.com/atlanticwave-sdx/datamodel>

Note 1. The schema file itself doesn’t allow comments.

Note 2. All objects have an ‘id’ field which ideally is a GUID

Note 3. All objects have an ‘name’ field which ideally is a well-defined urn. (<https://www.ogf.org/documents/GFD.202.pdf>)

### Topology

1. {
2. "title": "Topology"
3. "type": "object",
4. "properties": {
5. "domain\_service": {
6. "$ref": "repository://Service"
7. },
8. "id": { #This is supposed to be a guid.
9. "type": "string"
10. },
11. "name": { #This is supposed to be an urn.
12. "type": "string"
13. },
14. "version": {
15. "type": "number",
16. "minimum": 0
17. },
18. "time\_stamp": {
19. "type": "string",
20. "format": "date-time"
21. },
22. "nodes": {
23. "type": "array",
24. "items": [
25. {
26. "$ref": "repository://Node"
27. }
28. ]
29. },
30. "links": {
31. "type": "array",
32. "items": [
33. {
34. "$ref": "repository://Link"
35. }
36. ]
37. },
38. },
39. }

### Node

1. {
2. "title": "Node"
3. "type": "object",
4. "properties": {
5. "id": {
6. "type": "string"
7. },
8. "short\_name": {
9. "type": "string"
10. },
11. "name": {
12. "type": "string"
13. },
14. "location": {
15. "$ref": "repository://Location"
16. },
17. "ports": {
18. "type": "array",
19. "items": [
20. {
21. "$ref": "repository://Port"
22. }
23. ]
24. }
25. },
26. }

### Link

1. {
2. "title": "Link"
3. "type": "object",
4. "properties": {
5. "id": {
6. "type": "string"
7. },
8. "short\_name": {
9. "type": "string"
10. },
11. "name": {
12. "type": "string"
13. },
14. "ports": {
15. "type": "array",
16. "minItems": 2,
17. "maxItems": 2,
18. "uniqueItems": true,
19. "items": [
20. {
21. "$ref": "repository://Port"
22. }
23. ]
24. },
25. "total\_bandwidth": { #unit = 1mbps
26. "type": "number",
27. "minimum": 1,
28. "maximum": 1000000
29. },
30. "available\_bandwidth": { #unit = 1mbps
31. "type": "number",
32. "minimum": 1,
33. "maximum": 1000000
34. },
35. "latency": { #unit = 1ms
36. "type": "number",
37. "minimum": 1,
38. "maximum": 1000000
39. },
40. "packet\_loss": { #unit = %
41. "type": "number",
42. "minimum": 0,
43. "maximum": 100
44. },
45. "availability": { #unit = %
46. "type": "number",
47. "minimum": 0,
48. "maximum": 100
49. }
50. },
51. }

### Port

1. {
2. "type": "object",
3. "properties": {
4. "id": {
5. "type": "string"
6. },
7. "short\_name": {
8. "type": "string"
9. },
10. "name": {
11. "type": "string"
12. },
13. "node": {
14. "$ref": "repository://Node"
15. },
16. "inter\_domain": {
17. "type": "boolean"
18. },
19. "type": { #encoding
20. "type": "string",
21. "enum": [
22. "10GE",
23. "1GE",
24. "100GE"
25. ]
26. },
27. "encapsulation": {
28. "type": "string",
29. "enum": [
30. "Q-in-Q",
31. "Mac-in-Mac",
32. "VLAN"
33. ]
34. },
35. "label": {
36. "type": "string",
37. "enum": [
38. "vlan"
39. ]
40. },
41. "swapping\_capability": { #needs to be distinguished from the label type?
42. "type": "string",
43. "enum": [
44. "vlan"
45. ]
46. },
47. "label\_range": {
48. "type": "array",
49. "items": {
50. "type": "array",
51. "minItems": 1,
52. "maxItems": 4096,
53. "uniqueItems": true,
54. "items": {
55. "type": "string"
56. }
57. }
58. },
59. "mtu": { #needs to be enum?
60. "type": "number",
61. "minimum": 1000,
62. "maximum": 10000
63. },
64. "status": {
65. "type": "string",
66. "enum": [
67. "up",
68. "down"
69. ]
70. }
71. },
72. "title": "Port"
73. }

### Location

1. {
2. "type": "object",
3. "properties": {
4. "address": {
5. "type": "string"
6. },
7. "latitude": {
8. "type": "number",
9. "minimum": -90,
10. "maximum": 90
11. },
12. "longitude": {
13. "type": "number",
14. "minimum": -180,
15. "maximum": 180
16. }
17. },
18. "title": "Location"
19. }

### Service

1. {
2. "type": "object",
3. "properties": {
4. "owner": {
5. "type": "string"
6. },
7. "provisioning\_system": {
8. "type": "string"
9. },
10. "provisioning\_url": {
11. "type": "string"
12. },
13. "vendor": {
14. "not": [
15. {
16. "type": "array",
17. "items": {
18. "type": "string"
19. }
20. }
21. ]
22. }
23. },
24. "title": "Service"
25. }

### Connection

1. {
2. "type": "object",
3. "properties": {
4. "id": {
5. "type": "string"
6. },
7. "name": {
8. "type": "string"
9. },
10. "class": {
11. "type": "integer",
12. "minimum": 0,
13. "maximum": 7
14. },
15. "ingress\_port": {
16. "$ref": "repository://Port"
17. },
18. "egress\_port": {
19. "$ref": "repository://Port"
20. },
21. "end\_time": {
22. "type": "string",
23. "format": "date-time"
24. },
25. "start\_time": {
26. "type": "string",
27. "format": "date-time"
28. },
29. "multi\_path": {
30. "type": "boolean"
31. },
32. "preempt": {
33. "type": "boolean"
34. },
35. "backup\_path\_type": {
36. "type": "string",
37. "enum": [
38. "0",
39. "1",
40. "2",
41. "3"
42. ]
43. },
44. "exclusive\_links": {
45. "type": "array",
46. "items": [
47. {
48. "$ref": "#/definitions/Link"
49. }
50. ]
51. },
52. "inclusive\_links": {
53. "type": "array",
54. "items": {
55. "$ref": "repository://Link"
56. }
57. },
58. "bandwidth\_required": {
59. "type": "number",
60. "minimum": 1,
61. "maximum": 1000000
62. },
63. "bandwidth\_measured": {
64. "type": "number",
65. "minimum": 1,
66. "maximum": 1000000
67. },
68. "latency\_required": {
69. "type": "number",
70. "minimum": 1,
71. "maximum": 1000000
72. },
73. "latency\_measured": {
74. "type": "number",
75. "minimum": 1,
76. "maximum": 1000000
77. },
78. "packetloss\_required": {
79. "type": "number",
80. "minimum": 0,
81. "maximum": 100
82. },
83. "packetloss\_measured": {
84. "type": "number",
85. "minimum": 0,
86. "maximum": 100
87. },
88. "availability\_required": {
89. "type": "number",
90. "minimum": 0,
91. "maximum": 100
92. },
93. "availability\_measured": {
94. "type": "number",
95. "minimum": 0,
96. "maximum": 100
97. },
98. "paths": {
99. "type": "array",
100. "items": [
101. {
102. "$ref": "repository://Path"
103. }
104. ]
105. },
106. "status": {
107. "type": "string",
108. "enum": []
109. }
110. },
111. "title": "Connection"
112. }

### Path

1. {
2. "type": "object",
3. "properties": {
4. "id": {
5. "type": "string"
6. },
7. "name": {
8. "type": "string"
9. },
10. "bandwidth\_required": {
11. "type": "number",
12. "minimum": 1,
13. "maximum": 1000000
14. },
15. "bandwidth\_measured": {
16. "type": "number",
17. "minimum": 1,
18. "maximum": 1000000
19. },
20. "latency\_required": {
21. "type": "number",
22. "minimum": 1,
23. "maximum": 1000000
24. },
25. "latency\_measured": {
26. "type": "number",
27. "minimum": 1,
28. "maximum": 1000000
29. },
30. "packetloss\_required": {
31. "type": "number",
32. "minimum": 0,
33. "maximum": 100
34. },
35. "packetloss\_measured": {
36. "type": "number",
37. "minimum": 0,
38. "maximum": 100
39. },
40. "links": {
41. "type": "array",
42. "items": [
43. {
44. "$ref": "#/definitions/Link"
45. }
46. ]
47. }
48. },
49. "title": "Path"
50. }