UML-like Diagram

Generating a UML diagram from YANG can be done with pyang.

Information on pyang can be found here: https://1.ieee802.org/yangsters/yangsters-guidelines/yangsters-faq/#Tool_pyang

Pyang and instructions for installing pyang are found https://github.com/mbj4668/pyang.

I have installed pyang natively on a Windows PC running Windows Subsystem for Linux version 2 (WSL2), and on a VM running Ubuntu v16.04 or later. Following the install instructions found on the github site worked well. My only issue was after installing, I needed to run the "sudo ldconfig -v" command so pyang could find all the shared libraries. There are prereqs, so I installed python (2.7) and python3 (you need one or the other), pip, pip3, and git.

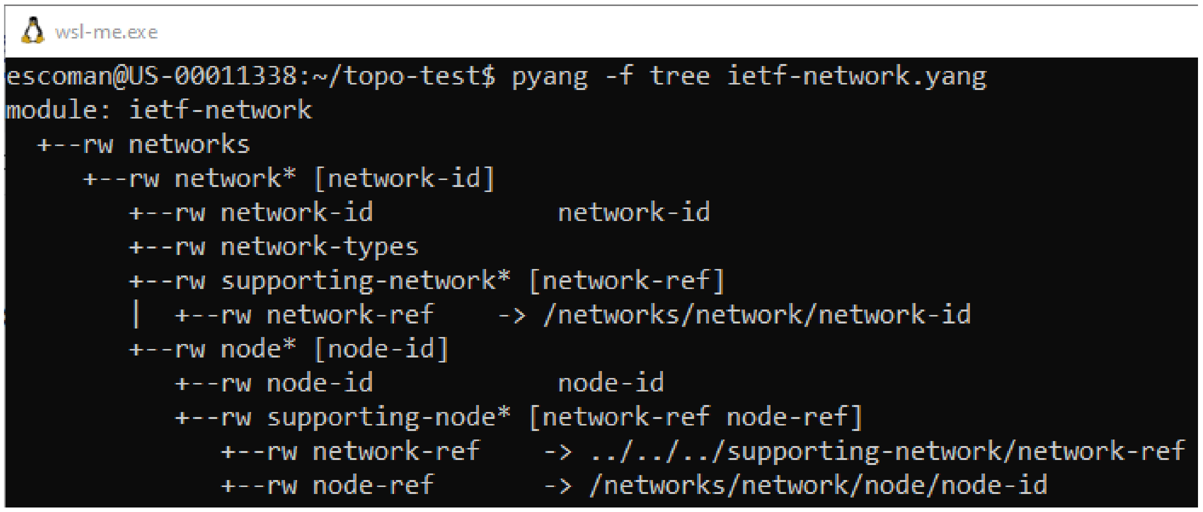
It is possible to use pyang with options to set the search path for import modules, but I find it easier to create a directory and copy all the modules that make up the model you want validate.

A simple example

The ietf-network.yang module defines a network of nodes. This is a very simple model which has a container for "networks" that contains a list (network) that contains a list (node). So there can be multiple instances of network each with it's own set of nodes. Hierarchy is supported through by a node having a supporting-node and/or a network having a supporting-network.

The set of modules that make up the networks model is small: ietf-network.yang module and the ietf-inet-types.yang module are all that are needed.

The yang tree for the ietf-network.yang module is created by running pyang:

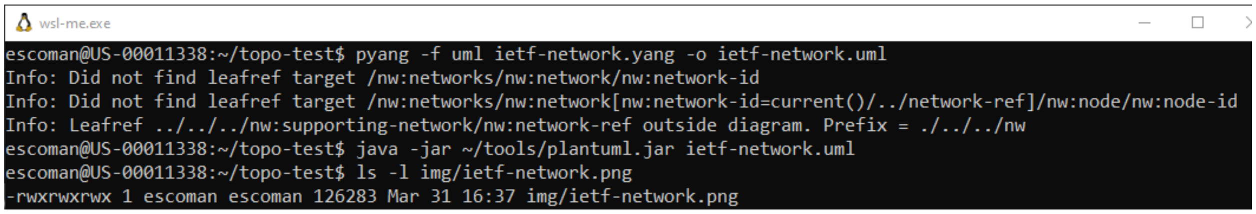


```
escoman@US-00011338:~/topo-test$ pyang -f tree ietf-network.yang
module: ietf-network
  +--rw networks
     +--rw network* [network-id]
        +--rw network-id            network-id
        +--rw network-types
        +--rw supporting-network* [network-ref]
        |  +--rw network-ref    -> /networks/network/network-id
        +--rw node* [node-id]
           +--rw node-id            node-id
           +--rw supporting-node* [network-ref node-ref]
              +--rw network-ref    -> ../../../supporting-network/network-ref
              +--rw node-ref       -> /networks/network/node/node-id
```

Figure 1 - ietf-network yang tree

In order to generate a uml-like diagram using pyang, another tool is needed (plantuml). The installation instructions for plantuml are found here: https://plantuml.com/starting and you will need both java and Graphviz install to support the generation of the uml-like class diagrams.

Once everything is installed (I put the jar file for plantuml in a "tools" directory in my home account), run the commands as below. FYI: I don't know why there are "Info" messages printed when I use the "-f uml" option, because pyang and pyang –lint both validate cleanly.



```
escoman@US-00011338:~/topo-test$ pyang -f uml ietf-network.yang -o ietf-network.uml
Info: Did not find leafref target /nw:networks/nw:network/nw:network-id
Info: Did not find leafref target /nw:networks/nw:network[nw:network-id=current()/../network-ref]/nw:node/nw:node-id
Info: Leafref ../../../nw:supporting-network/nw:network-ref outside diagram. Prefix = ./../../nw
escoman@US-00011338:~/topo-test$ java -jar ~/tools/plantuml.jar ietf-network.uml
escoman@US-00011338:~/topo-test$ ls -l img/ietf-network.png
-rwxrwxrwx 1 escoman escoman 126283 Mar 31 16:37 img/ietf-network.png
```

Figure 2 - uml and png generation

The result produced by the plantuml tool is attached, below is a zoom in on part of the diagram so it is readable:

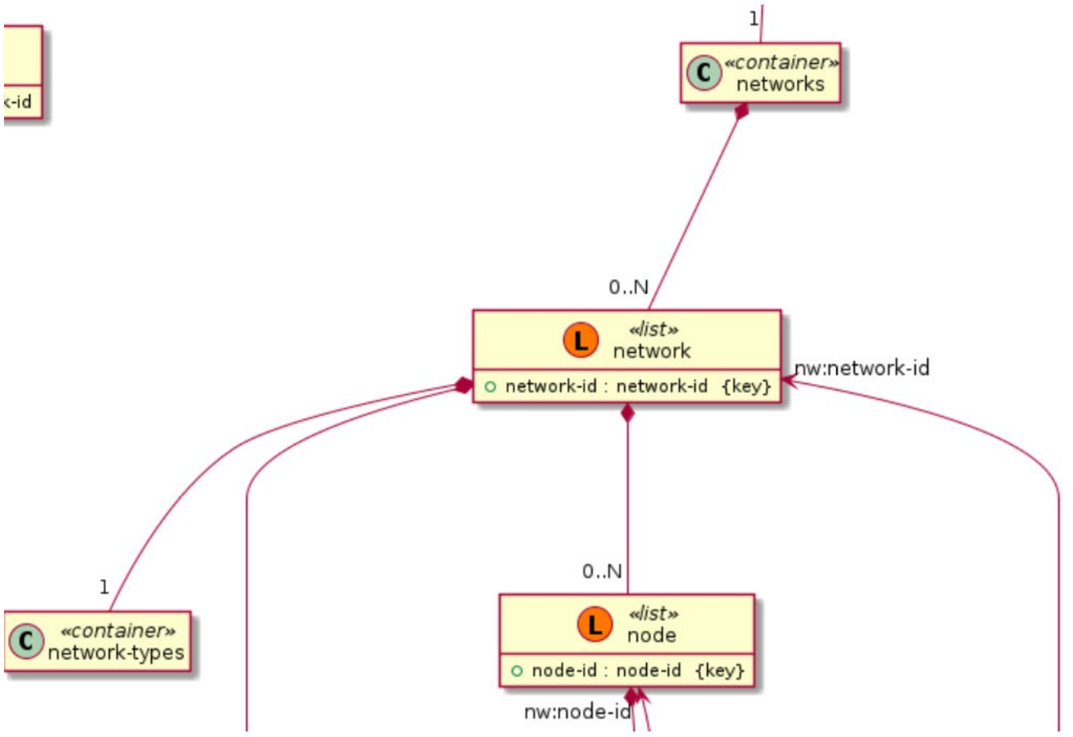


Figure 3 - Detail of ietf-network.yang uml-like diagram

The full image is here:

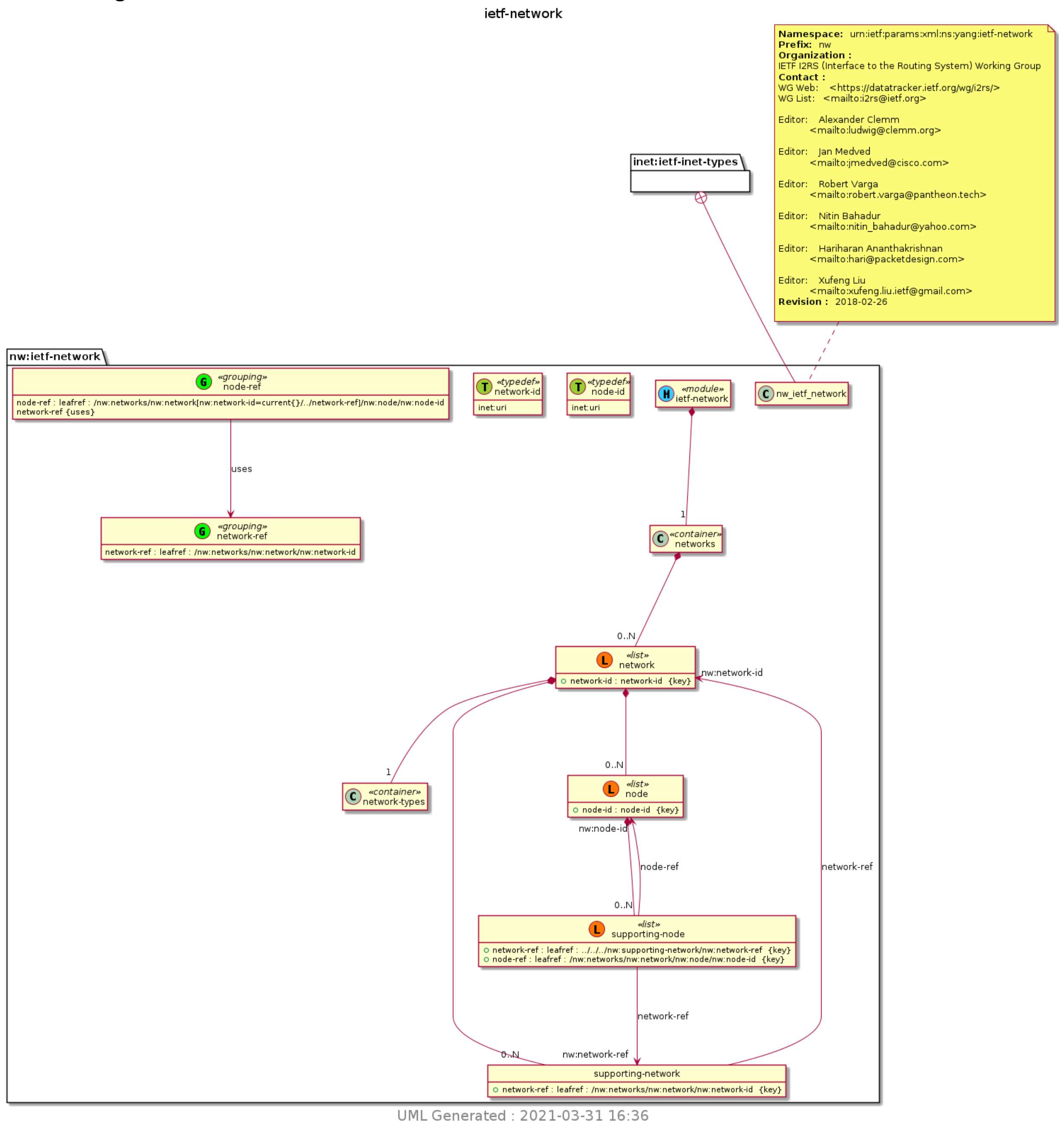

ietf-network
Namespace: urn:ietf:params:xml:ns:yang:ietf-network
Prefix: nw
Organization :
IETF I2RS (Interface to the Routing System) Working Group
Contact :
WG Web: <https://datatracker.ietf.org/wg/i2rs/>
WG List: <mailto:i2rs@ietf.org>
Editor: Alexander Clemm <mailto:ludwig@clemm.org>
Editor: Jan Medved <mailto:jmedved@cisco.com>
Editor: Robert Varga <mailto:robert.varga@pantheon.tech>
Editor: Nitin Bahadur <mailto:nitin_bahadur@yahoo.com>
Editor: Hariharan Ananthakrishnan <mailto:hari@packetdesign.com>
Editor: Xufeng Liu <mailto:xufeng.liu.ietf@gmail.com>
Revision : 2018-02-26
inet:ietf-inet-types
nw:ietf-network
«grouping» node-ref
node-ref : leafref : /nw:networks/nw:network[nw:network-id=current()/../network-ref]/nw:node/nw:node-id
network-ref {uses}
«typedef» network-id
inet:uri
«typedef» node-id
inet:uri
«module» ietf-network
nw_ietf_network
uses
«grouping» network-ref
network-ref : leafref : /nw:networks/nw:network/nw:network-id
1
«container» networks
0..N
«list» network
network-id : network-id {key}
nw:network-id
1
«container» network-types
0..N
«list» node
node-id : node-id {key}
nw:node-id
node-ref
network-ref
0..N
«list» supporting-node
network-ref : leafref : ../../../nw:supporting-network/nw:network-ref {key}
node-ref : leafref : /nw:networks/nw:network/nw:node/nw:node-id {key}
network-ref
0..N
nw:network-ref
supporting-network
network-ref : leafref : /nw:networks/nw:network/nw:network-id {key}
UML Generated : 2021-03-31 16:36

Figure 4 - Full ietf-network.yang uml-like diagram

Larger UML Example

CBcv provides a good example of a standard that has multiple modules. The modules can be considered separately. The example here is about Stream Identification which is found in ieee802-dot1cb-stream-identification.yang.

Stream Identification

The yang tree from pyang -f tree:

```
module: ieee802-dot1cb-stream-identification
  +-rw stream-identity* [index]
     +-rw index                                    uint32
     +-rw handle                                   uint32
     +-rw in-facing
     |  +-rw input-port*    if:interface-ref
     |  +-rw output-port*   if:interface-ref
     +-rw out-facing
     |  +-rw input-port*    if:interface-ref
     |  +-rw output-port*   if:interface-ref
     +-rw (parameters)
        +-:(null-stream-identification)
        |  +-rw null-stream-identification
        |     +-ro identification-type
        |     |  +-ro type-number?   dot1cb-sid-types:stream-id-function
        |     |  +-ro oui-cid?       string
        |     +-rw destination-mac?       ieee:mac-address
        |     +-rw tagged?                vlan-tag-identification-type
        |     +-rw vlan?                  vlan-identifier-type
        +-:(smac-vlan-stream-identification)
        |  +-rw smac-vlan-stream-identification
        |     +-ro identification-type
        |     |  +-ro type-number?   dot1cb-sid-types:stream-id-function
        |     |  +-ro oui-cid?       string
        |     +-rw source-mac?            ieee:mac-address
        |     +-rw tagged?                vlan-tag-identification-type
        |     +-rw vlan?                  vlan-identifier-type
        +-:(dmac-vlan-stream-identification)
        |  +-rw dmac-vlan-stream-identification
        |     +-ro identification-type
        |     |  +-ro type-number?   dot1cb-sid-types:stream-id-function
        |     |  +-ro oui-cid?       string
        |     +-rw down
        |     |  +-rw destination-mac?   ieee:mac-address
        |     |  +-rw tagged?            vlan-tag-identification-type
        |     |  +-rw vlan?              vlan-identifier-type
        |     |  +-rw priority?          dot1qtypes:priority-type
        |     +-rw up
        |        +-rw destination-mac?   ieee:mac-address
        |        +-rw tagged?            vlan-tag-identification-type
        |        +-rw vlan?              vlan-identifier-type
        |        +-rw priority?          dot1qtypes:priority-type
        +-:(ip-stream-identification)
        |  +-rw ip-stream-identification
        |     +-ro identification-type
        |     |  +-ro type-number?   dot1cb-sid-types:stream-id-function
        |     |  +-ro oui-cid?       string
        |     +-rw destination-mac?       ieee:mac-address
        |     +-rw tagged?                vlan-tag-identification-type
        |     +-rw vlan?                   vlan-identifier-type
        |     +-rw ip-source?              inet:ip-address
        |     +-rw ip-destination?         inet:ip-address
        |     +-rw dscp?                   inet:dscp
        |     +-rw next-protocol?          enumeration
        |     +-rw source-port?            inet:port-number
        |     +-rw destination-port?       inet:port-number
        +-:(organization-specific)
           +-rw organization-specific
              +-rw identification-type
                 +-rw type-number?   int32
                 +-rw oui-cid?       string
  augment /if:interfaces/if:interface/if:statistics:
    +-ro stream-id
       +-ro per-port-counters
       |  +-ro input-pkts?    uint64
       |  +-ro output-pkts?   uint64
       +-ro per-port-per-stream-counters* [direction-out-facing handle]
          +-ro direction-out-facing    dot1cb-sid-types:direction
          +-ro handle                  -> /stream-identity/handle
          +-ro input-pkts?             uint64
          +-ro output-pkts?            uint64
```

The following is a zoom in on some of the detail of the stream-identification model.

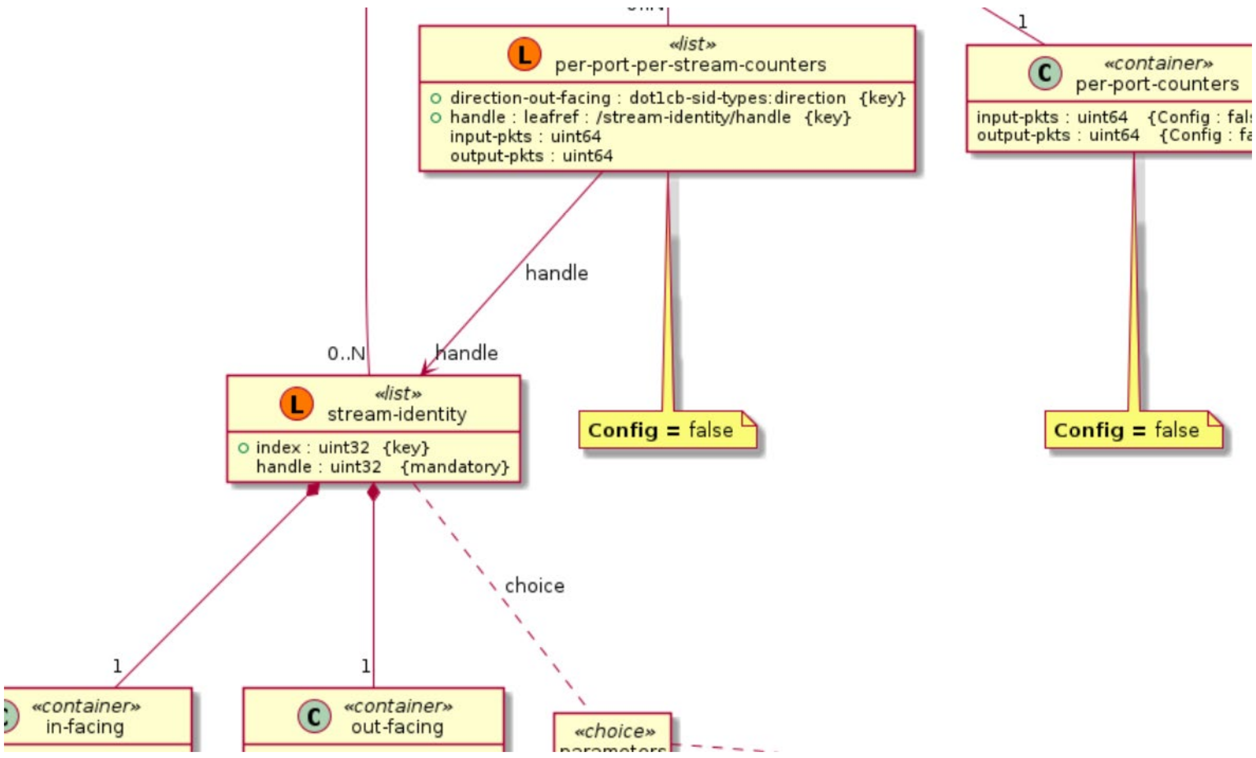

«list»
per-port-per-stream-counters
direction-out-facing : dot1cb-sid-types:direction {key}
handle : leafref : /stream-identity/handle {key}
input-pkts : uint64
output-pkts : uint64
1
«container»
per-port-counters
input-pkts : uint64 {Config : fal
output-pkts : uint64 {Config : fa
handle
handle
0..N
«list»
stream-identity
index : uint32 {key}
handle : uint32 {mandatory}
Config = false
Config = false
choice
1
1
«container»
in-facing
«container»
out-facing
«choice»

Figure 5 - Detail of ieee802-dot1cb-stream-identification.yang

The following figure is the full uml diagram produced by plantuml after running pyang -f uml on ieee802-dot1cb-stream-identification.yang.

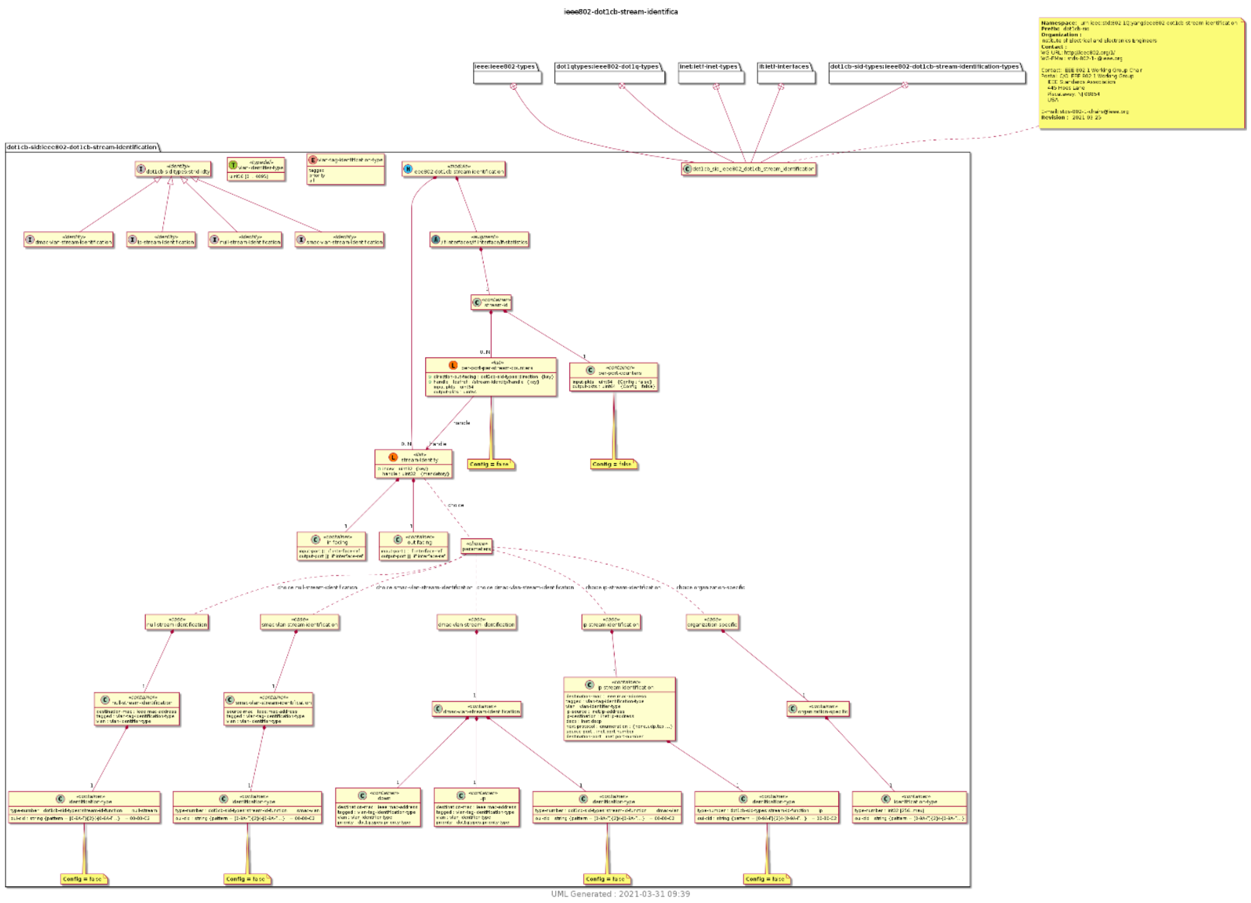


Figure 6 - Full ieee802-dot1cb-stream-identification.yang