Maintenance Item 0264: Erroneous xpath statements

Don Fedyk dfedyk@labn.net
xpath for ieee802-dot1q-bridge.yang and ieee802-dot1q-pb.yang

module: ieee802-dot1q-bridge
  +--rw bridges
    +--rw bridge* [name]
      +--rw name dot1q-types:name-type
      +--rw address ieee:mac-address
      +--rw bridge-type identityref
      +--ro ports? uint16
      +--ro up-time? yang:zero-based-counter32
      +--ro components? uint32
    +--rw component* [name]
      +--rw name string
      +--rw id? uint32
      +--rw type identityref
      +--rw address? ieee:mac-address
      +--rw traffic-class-enabled? boolean
      +--ro ports? uint16
      +--ro bridge-port* if:interface-ref
      +--ro capabilities

**augment /if:interfaces/if:interface:
  +--rw bridge-port
    +--rw component-name string

**

when "./component-name != 'd-bridge-component'" ← Erroneous xpath statements of this form in the YANG file.
Two problems

1. Component-name is not a reference pointer
2. Need to test the type that is under component[name]
How to fix this?

1. **Use an absolute path**
   - Most in line with current yang
   - One extra leaf ref in the interface tree

2. **Use a relative path**
   - Technically correct but current tool set does not completely support
   - One extra leaf ref in the interface tree

3. **Create a local type in the interfaces tree that is a reference to the type in the bridge tree.**
   - Two extra leaf refs in the interface tree

4. **Remove the xpath statements**
   - Actually my favorite

5. **Move the bulk of the logic to the bridges tree**
   - Alternative response from the YANG doctors – Non starter
Absolute xpath for ieee802-dot1q-bridge.yang and ieee802-dot1q-pb.yang Tested fix

module: ieee802-dot1q-bridge
   +--rw bridges
      +--rw bridge* [name]
         +--rw name dot1q-types:name-type
         +--rw address ieee:mac-address
         +--rw bridge-type identityref
         +--ro ports? uint16
         +--ro up-time? yang:zero-based-counter32
         +--ro components? uint32
         +--rw component* [name]
            +--rw name string
            +--rw id? uint32
            +--rw type identityref
            +--rw address? ieee:mac-address
            +--rw traffic-class-enabled? boolean
            +--ro ports? uint16
            +--ro bridge-port* if:interface-ref
            +--ro capabilities

     ***

augment /if:interfaces/if:interface:
   +--rw bridge-port
      +--rw bridge-name                  -> /bridges/bridge/name
      +--rw component-name              -> /bridges/bridge[dot1q:name=current()]/../bridge-name]/component/name

     ***

when "/dot1q:bridges/dot1q:bridge[dot1q:name=current()]/../dot1q:bridge-name]/dot1q:component[name=current()]/../dot1q:component-name]/dot1q:type != 'd-bridge-component'"
Alternative Relative paths using deref()

Feedback Absolute path is bulky hard to read and error prone:

```
when "/dot1q:bridges/dot1q:bridge[dot1q:name=current()]/../dot1q:bridge-name]/dot1q:component[name=current()]/../dot1q:component-name]/dot1q:type != 'd-bridge-component'
```

Since component_name is a ref-pointer to the referenced node, we can reference the type this way:

```
when "deref(../dot1q:component-name)/../dot1q:type != 'dot1q:d-bridge-component'"
```

Currently does not work in the modules I tested in Yuma123. (Works in some cases). It works in theory.
Works in confd but requires tailf extension to work.
Local type

module: ieee802-dot1q-bridge
   +++rw bridges
      +++rw bridge* [name]
         +++rw name        dot1q-types:name-type
         +++rw address     ieee:mac-address
         +++rw bridge-type identityref
         +++ro ports?      uint16
         +++ro up-time?    yang:zero-based-counter32
         +++ro components? uint32
      +++rw component* [name]
         +++rw name         string
         +++rw id?          uint32
         +++rw type         identityref
         +++rw address?     ieee:mac-address
         +++rw traffic-class-enabled? boolean
         +++ro ports?       uint16
         +++ro bridge-port* if:interface-ref
         +++ro capabilities

augment /if:interfaces/if:interface:
   +++rw bridge-port
      +++rw bridge-name => /bridges/bridge/name
      +++rw component-name => /bridges/bridge[dot1q:name=current()]/../bridge-name]/component/name
      +++rw component-type
         name]/component[dot1q:name=current()]/../component-name]/type

When "../component-type != 'd-bridge-component'"

Requires extra data entry. Validate ensure the ref pointers are correct.
Alternative: Remove the when statements.

- We (802.1) are creating work for ourselves putting in conditional YANG.
- What a conditional statement means in yang is certain parts of the tree are not valid for certain conditions – in this case component types.
- When a validate is performed the YANG conditions are tested
  - Depending on the tool set configuration of the items is blocked (but not completely)
  - Validate may fail, Informing the user config cannot be committed.
- After validation next step is to commit
  - During this phase back end code also does checks and is is not going to trust the YANG validate so validation tests are redundant – they express intent but the backend code does much more.
- IEEE are providing YANG tests that need to be maintained and are at best hints of what should be configured.
- Debugging xpath turned out to be a big headache
- Adding new components will require visiting all affected statements
- I don’t expect this alternative to be adopted but the nature of the beast and the simplest solution.
Resolution?