A YANG Data Model for module revision management

draft-wang-netmod-module-revision-management-00

Author: Michael Wang

Qin Wu

Motivation & Objective

Motivation

- To support the NMDA, many modules may need to be updated or restructured especially for some published non-NMDA modules.
- How to support backward-compatible and indicate the module's changes is an issue.

Objective

- This document provide a yang data model which can
  1. provide a design aid to the user to indicate the module's update details;
  2. help user to manage all the revisions of that modules;
  3. to track the module's updates.

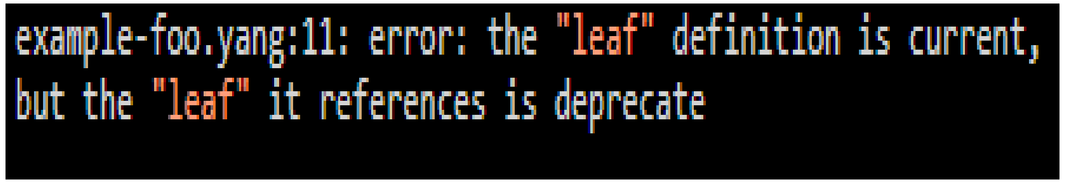
Scenarios I : A module design aid

This module can present different revisions' update details. It will be useful to module designer, for example:

```
module example-foo {
  namespace "urn:ietf:params:xml:ns:yang:example-foo";
  prefix foo;
  import ietf-interfaces {
    prefix if;
  }
  container foo {
   leaf a {
    type string;
   }
   leaf foo-interfaces-state {
    type leafref {
    path "/if:interfaces-state/if:interface/if:name";
    }
    config false;
   }
  }
}
```

//Exists multiple revisions

Run result:



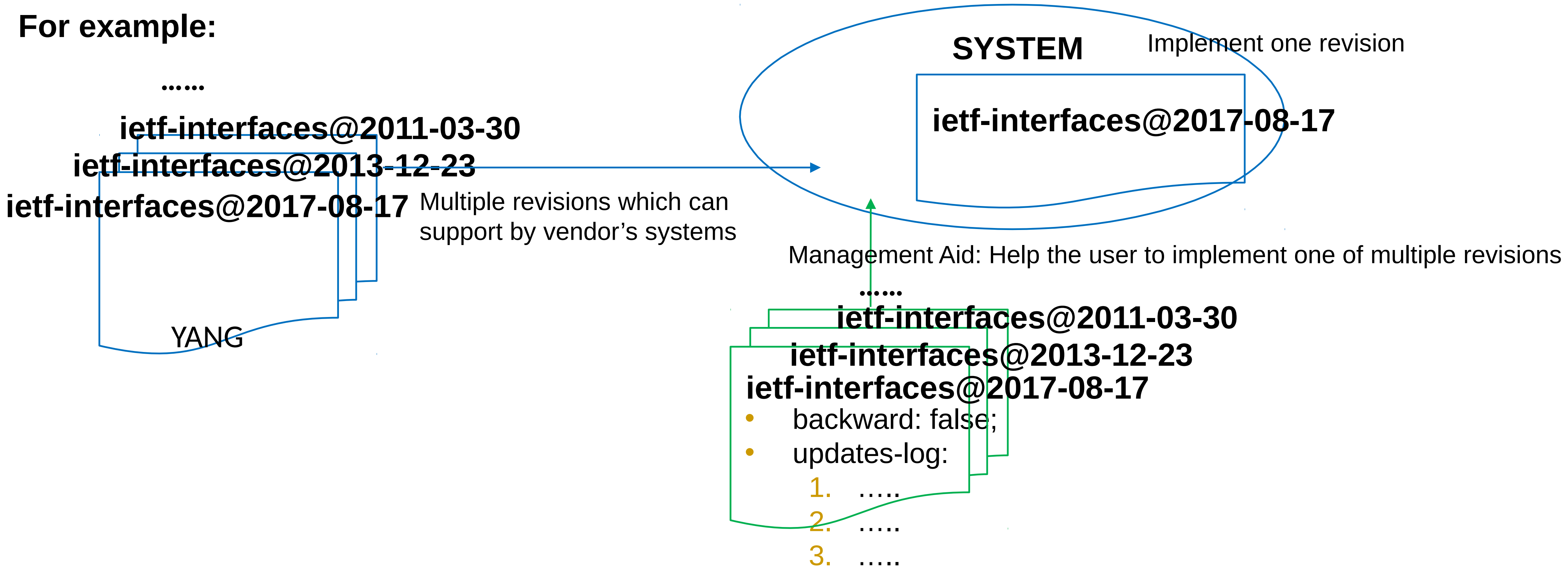
```
example-foo.yang:11: error: the "leaf" definition is current,
but the "leaf" it references is deprecate
```

- **If we can provide some tools to indicate the details of each module's updates comparing with previous revision, it will greatly improve the module designer's efficiency to design new module(e.g., decide which module revision should be imported, or which piece of module is deprecated)**

Scenarios II : A system management aid

This module also can serve as a management aid. It can be used by vendors who support multiple revisions of the same YANG module in their systems to decide which revision of a module should be implemented.

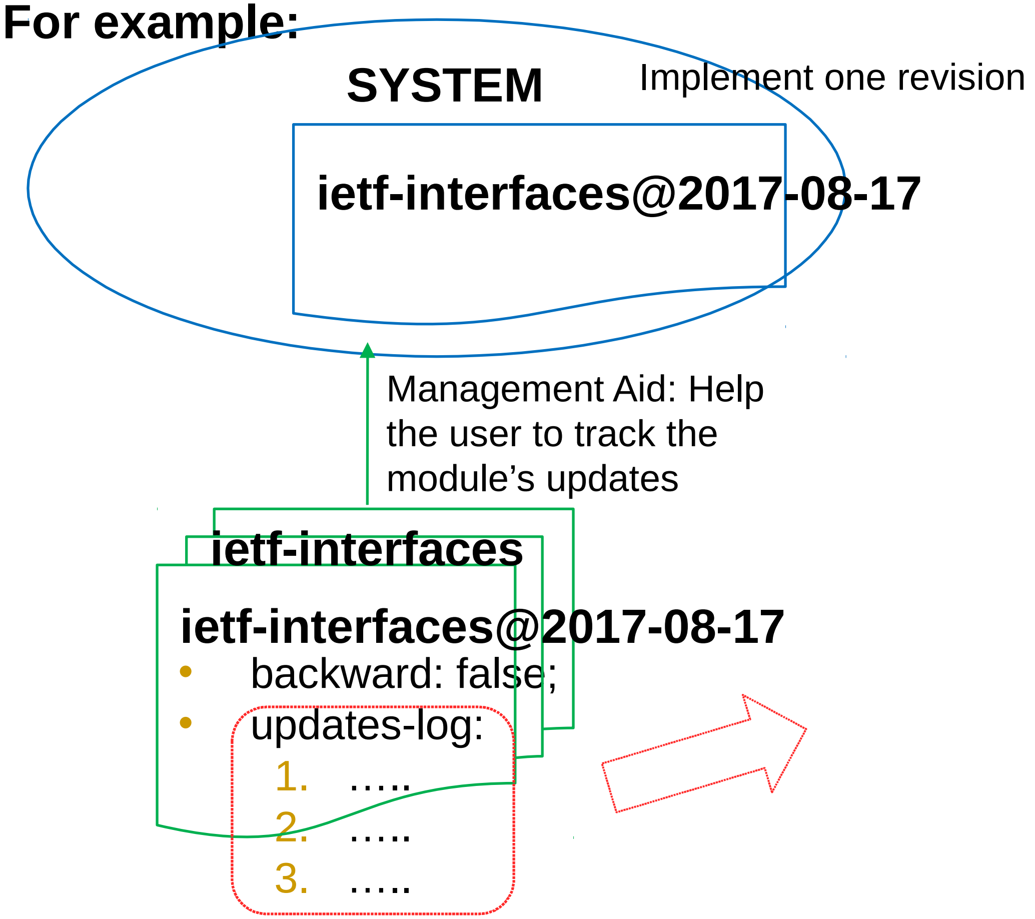
For example:

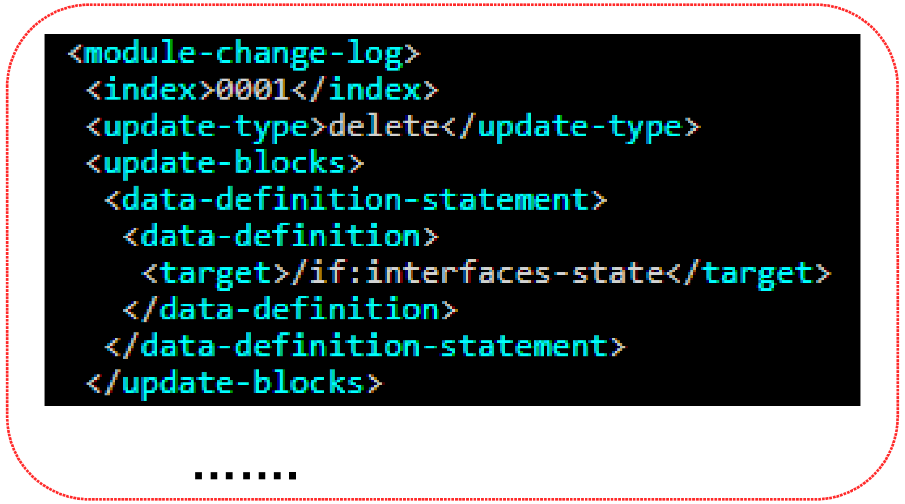

……
ietf-interfaces@2011-03-30
ietf-interfaces@2013-12-23
ietf-interfaces@2017-08-17
YANG
Multiple revisions which can support by vendor's systems
SYSTEM
Implement one revision
ietf-interfaces@2017-08-17
Management Aid: Help the user to implement one of multiple revisions
……
ietf-interfaces@2011-03-30
ietf-interfaces@2013-12-23
ietf-interfaces@2017-08-17
backward: false;
updates-log:
1. …..
2. …..
3. …..

Scenarios III : A tracker

This module can serve as a tracker, which can help user to track the module's updates between any two revisions of the same module.

For example:


SYSTEM
Implement one revision
ietf-interfaces@2017-08-17
Management Aid: Help the user to track the module's updates
ietf-interfaces
ietf-interfaces@2017-08-17
backward: false;
updates-log:
1. …..
2. …..
3. …..



```
<module-change-log>
 <index>0001</index>
 <update-type>delete</update-type>
 <update-blocks>
  <data-definition-statement>
   <data-definition>
    <target>/if:interfaces-state</target>
   </data-definition>
  </data-definition-statement>
 </update-blocks>
```

…….

Model design

```
+--ro yang-modules
   +--ro module* [name revision]
      +--ro name                     yang:yang-identifier
      +--ro revision                 yanglib:revision-identifier
      +--ro backward-compatible?     boolean
      +--ro module-change-log* [index]
         +--ro index                 uint32
         +--ro change-operation      identityref
         +--ro (update-blocks)?
            +--:(data-definition-statement)
            |  +--ro data-definition
            |     +--ro target-node         yang:xpath1.0
            |     +--ro location-point?     yang:xpath1.0
            |     +--ro where?              enumeration
            |     +--ro data-definition?
            +--:(other-statement)
               +--ro other-statement
                  +--ro statement-name?          identityref
                  +--ro statement-definition?
                  +--ro substatements* [statement-name]
                     +--ro statement-name            identityref
                     +--ro substatement-definition?
```

Indicate the “change type”, such as:

- Delete statements;
- Move data node to a new location;
- Add new statements;
- Modify statements
- ……

Present the updates which locate in:

- Leaf definition & subdefinition
- List definition & subdefinition;
- Container definition & subdefinition;
- Choice/case definition & subdefinition;
- ……

Present the updates which locate in:

- Identity definition & subdefinition
- feature definition & subdefinition;
- typedef definition & subdefinition;
- grouping definition & subdefinition;
- ……

```
rpcs:
   +---x module-revision-change
      +---w input
      |  +---w module-name         yang:yang-identifier
      |  +---w source-revision?    yanglib:revision-identifier
      |  +---w target-revision?    yanglib:revision-identifier
      +--ro output
         +--ro data-nodes* [data-node-name]
            +--ro data-node-name      string
            +--ro is-new-node?        boolean
            +--ro change-operation?   identityref
```

Retrieve the schema data node changes between any two revisions of the same module, for example:

- the data node that get updated or newly added during module revision change.

Usage Example: RFC7223bis

- **Change-log 1: Delete “interfaces-state”:**

```
<index>0001</index>
<update-type>delete</update-type>
<update-blocks>
 <data-definition-statement>
  <data-definition>
   <target>/if:interfaces-state</target>
  </data-definition>
 </data-definition-statement>
</update-blocks>
```

- **Change-log 2: Add “if-mib” feature:**

```
<index>0002</index>
<update-type>create</update-type>
<update-blocks>
 <other-statement>
  <statement>
   <statement-name>feature-statement</statement-name>
   <statement-definition>
    feature if-mib {
     description
     "This feature indicates that the device implements
      the IF-MIB.";
     reference
     "RFC 2863: The Interfaces Group MIB";
    }
   </statement-definition>
  </statement>
 </other-statement>
</update-blocks>
```

- **Change-log 3: Add new statement—”if-feature if-mib”:**

```
<index>0003</index>
<update-type>modify</update-type>
<update-blocks>
 <data-definition-statement>
  <data-definition>
   <target>
   /if:interfaces/if:interface/if:link-up-down-trap-enable
   </target>
   <data-node>
    leaf link-up-down-trap-enable {
    if-feature if-mib; // add if-feature statement
     type enumeration {
     ....
   </data-node>
  </data-definition>
 </data-definition-statement>
</update-blocks>
```

- **Change-log 4: move “admin-status” to a new location:**

```
 <index>0004</index>
 <update-type>move</update-type>
 <update-blocks>
  <data-definition-statement>
   <data-definition>
    <target>
    /if:interfaces-state/if:interface/if:admin-status
    </target>
   <location-point>
    /if:interfaces/if:interface/link-up-down-trap-enable
   </location-point>
   <where>after</where>
  </data-definition-statement>
 </update-blocks>
```

Next Steps:

- The authors appreciate thoughts, feedback, and text on the content of the documents.
- And then prepare another version.

Thank you