Robert Peschi, Nokia, BBF FAN Editor

October 30th, 2023

Achieving Scalable YANG Modules

in PON Access

This contribution is supported by the following individuals

- Joey Boyd, Adtran, BBF Common YANG Work Area Director
- Nick Hancock, Adtran, BBF Common YANG Editor
- Lincoln Lavoie, BBF Technical Committee Chair
- Kevin Noll, Vecima, BBF Member
- Sven Ooghe, Nokia, BBF Common YANG Work Area Director
- Ludwig Pauwels, Nokia, BBF Common YANG Editor
- Marta Seda, Calix, BBF FAN Work Area Director
- Frank van der Putten, Nokia, BBF Vice-chair Technical Committee and Board Member

Disclaimer: This contribution is offered to clarify the technical background and motivation for the Broadband Forum (BBF) WT-505 project. The statements made in this contribution do not constitute an official position of the BBF.

Introduction - History and goal of the contribution

June 2023: The Broadband Forum (BBF) sends a liaison to IEEE, IETF and ITU-T to inform them of their new project WT-505

“For Information: New Project for Addressing ONU Management at Scale”

- *“applicable portions of those [standard] modules are copied into new [BBF] modules”*

July 2023: The IEEE responds to the BBF's liaison

“Liaison response to BBF liaison on New Project for Addressing ONU Management at Scale”

- *“Taking the YANG modules from the standards documents and modifying them is not the model for cooperative development that has been established in the YANG community.”*
- “If the *BBF has identified use cases, requirements, or architectures related to constrained environments, IEEE 802.1 and IEEE 802.3 would like to understand that requirement so the working groups can consider how to satisfy those needs in a standardized fashion.”*
- *“The permission statement in the IEEE 802.1 and IEEE 802.3 YANG modules does not provide the right to copy and modify the YANG.”*

We realize that insufficient context may have been provided about the BBF WT-505 project.

We also believe that there may be a misunderstanding about the intentions of the project.

The goal of this contribution is to clarify these aspects.

Introduction – BBF context

The BBF’s focus

- To standardize solutions enabling the deployment of production/commercial telecom access networks
- To define YANG models to support such deployments

The BBF’s policy about YANG models for its deployment solutions

- Use existing standards from other SDOs (IEEE, IETF, ITU-T…) where appropriate
- The BBF also creates new YANG modules
    - To “augment” existing standards modules to support additional functions or
    - To cover domains not addressed by existing standards (e.g., PON transport) or
    - When existing standards are not fully adaptable to networks targeted by BBF standards

The WT-505 project intends to define new YANG modules including data nodes from IEEE / IETF / ITU-T standards.

- Why not reuse existing standards modules as is?
- What concrete YANG design is anticipated?
- Identifying possible technical collaboration between BBF and IEEE / IETF / ITU-T towards a common objective.

YANG: strong point and weak point

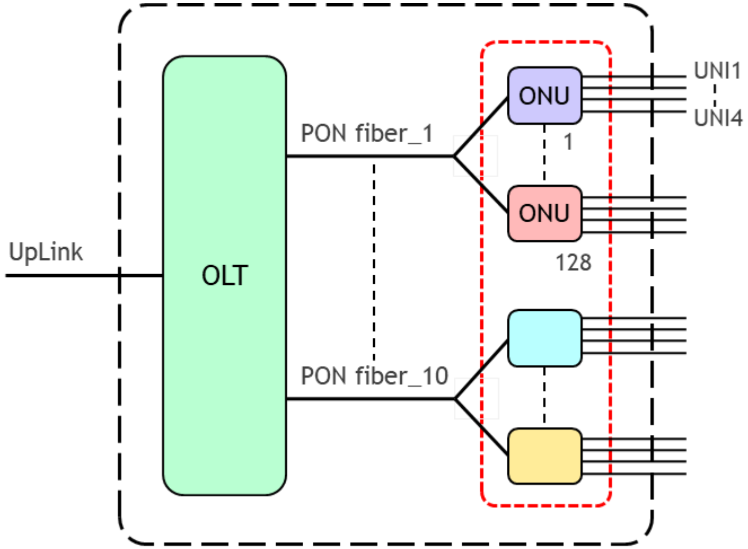
- One of the greatest assets of YANG is that it allows embedding semantic constraints in a model definition, e.g., ‘when’, ‘must’, etc.
    - any off-the-shelf YANG tool can automatically impose or verify the consistency of a configuration: no dependency on vendors software that are potentially prone to interpretation errors, jeopardizing interoperability

- A weak point is that running YANG validation rules can however become very computationally intensive
    - For very large data stores, performance *collapses* far below practical levels!
    - *YANG models must be specially designed when they are to be used for large-scale devices*

- PON Access Networks targeted by BBF lead to huge YANG datastores (where greater than 10K interfaces need to be managed).
    - Many current standard models run into performance issues when scaled to that extreme
    - To address this, model structure must be adapted, and other modeling techniques used, to
        - Reduce the size of the datastore and
        - More efficiently organize data nodes (rather than define new ones).

PON Access Network: a matter of OLT and ONUs

- A PON Access Network is comprised of an OLT (Optical Line Termination) and ONUs (Optical Network Unit) entities each with their own data nodes (interfaces, hardware, etc…).
- A popular management mode is to consider the OLT and ONUs as *a single **large** device*.
  [BBF TR-385 "*OLT-ONUs Combined NE mode"*]
- Defining a scalable YANG model for this Combined "OLT-ONUs device" is crucial for achieving a usable and interoperable model that can be used by Northbound systems.
- The goal of the WT-505 project is to develop an *ONU management model* that scales efficiently for OLT-ONU Combined NE mode



Combined (OLT-ONUs) "device"

OLT-ONUs as a single large device

Dimension example:

- 3 VLAN interfaces per UNI
- 4 UNIs per ONU
- 128 ONUs per PON
- 10 $^{+}$ PONs per OLT

This results in 15K^{+} interfaces, just on the ONU side.

Moreover, each of these interfaces has a counterpart on the OLT side resulting in a grand total in the combined model of 30K^{+} interfaces related to ONUs.

Clustering ONUs in the datastore

- The typical YANG model for a combined (OLT-ONUs) does not separate OLT data nodes from ONU data nodes (e.g., in the case of interfaces, there is only one flat list randomly populated with OLT and ONUs interfaces).
- Moving to 30K^{+} interfaces, this lack of structure is very inefficient, for instance when it comes to running validation rules between interfaces or deleting an interface without leaving orphan interfaces behind, higher in the stack.
- The strong decoupling between OLT and ONUs in term of validation rules or deletion constraints is not exploited nor the potential configuration similarity between ONUs that may have identical service models applied to them.

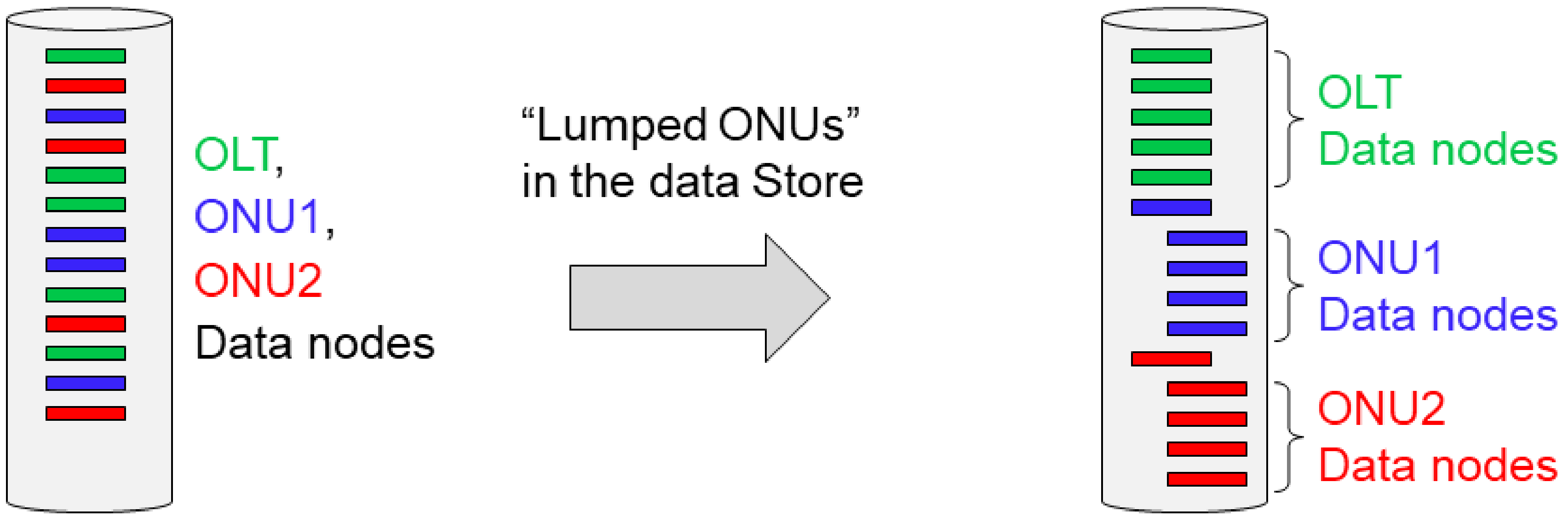
Analysis suggests that the best way to proceed would be to turn towards a model that **clusters** data nodes of individual ONUs, alongside the OLT’s. Doing this scopes ONU validation rules to a much smaller number of data nodes (e.g., think of interfaces).

This would be achieved by defining a **list of ONUs** (“onus/onu”) with each entry regrouping all data nodes of the ONU.

- Applying the IETF Schema Mount method, although it appears an ideal solution, proved to be unsuitable, having more drawbacks than benefits (see back-up slides for more details)
- A better way to proceed would be to modify the XPath of standard ONU data nodes to move them inside the ONU list entry. For instance, the XPath of interfaces would change as illustrated on the next slide

Illustrating the change of the XPath for interfaces

`interfaces/interface` -> `onus/onu/interfaces/interface`



OLT,
ONU1,
ONU2
Data nodes

“Lumped ONUs”
in the data Store

OLT
Data nodes

ONU1
Data nodes

ONU2
Data nodes

Reducing datastore size

- Once ONUs are “clustered”, additional techniques can be applied to dramatically reduce datastore size:
    - ONU **shared profiles**
    - ONU **templates** (see back-up slides for a bit more details):
        - a list of **ONU instances** and a list of **ONU templates**
        - same data nodes except for **mandatory** and **default** statements

Note that YANG models specifically designed for scalability are more complex than typical YANG models.

There would be no point to use them when scalability is not an issue.

In other words, they *provide an alternative* rather than overrule current standards.

Practical realization

The big question is how to realize such scalable models in practice, while reusing as much as possible from existing standards?

The next slides propose three possible alternative ways forward that differ by the amount of direct involvement of IEEE / IETF / ITU-T (*).

Initial situation (schematic view)

```
ieee.yang module:
```

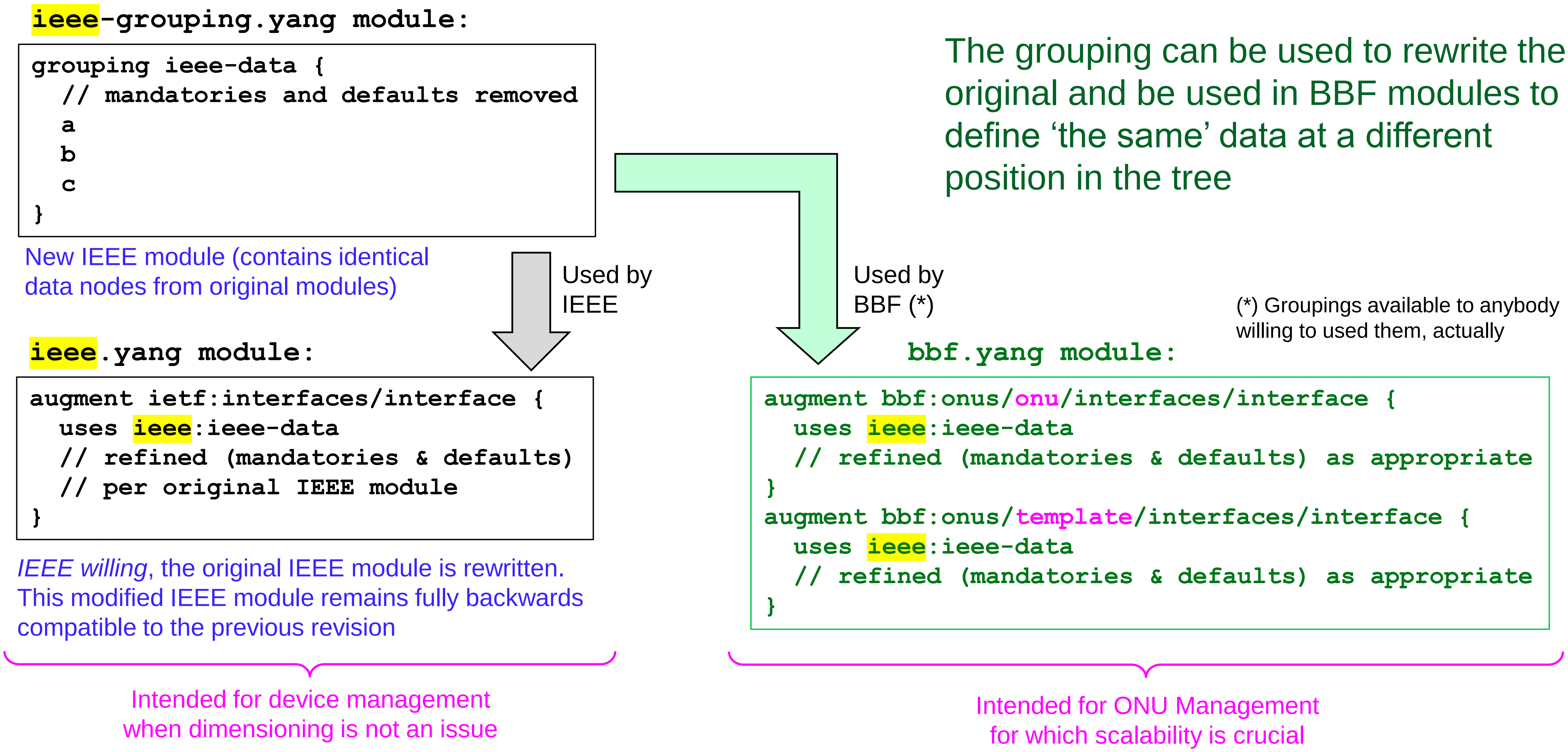
```
augment ietf:interfaces/interface {
  a
  b
  c
}
```

The original IEEE module

(*) For the sake of conciseness, the few next slides only mention IEEE in the schematic examples, but same principles apply to IETF and ITU-T as well.

Alternative 1- Step 1: *IEEE defines groupings; both IEEE and BBF use them*

(schematic view)



ieee-grouping.yang module:

```
grouping ieee-data {
  // mandatories and defaults removed
  a
  b
  c
}
```

New IEEE module (contains identical data nodes from original modules)

Used by IEEE

ieee.yang module:

```
augment ietf:interfaces/interface {
  uses ieee:ieee-data
  // refined (mandatories & defaults)
  // per original IEEE module
}
```

IEEE willing, the original IEEE module is rewritten. This modified IEEE module remains fully backwards compatible to the previous revision

Intended for device management
when dimensioning is not an issue

The grouping can be used to rewrite the original and be used in BBF modules to define ‘the same’ data at a different position in the tree

Used by BBF (*)

(*) Groupings available to anybody willing to used them, actually

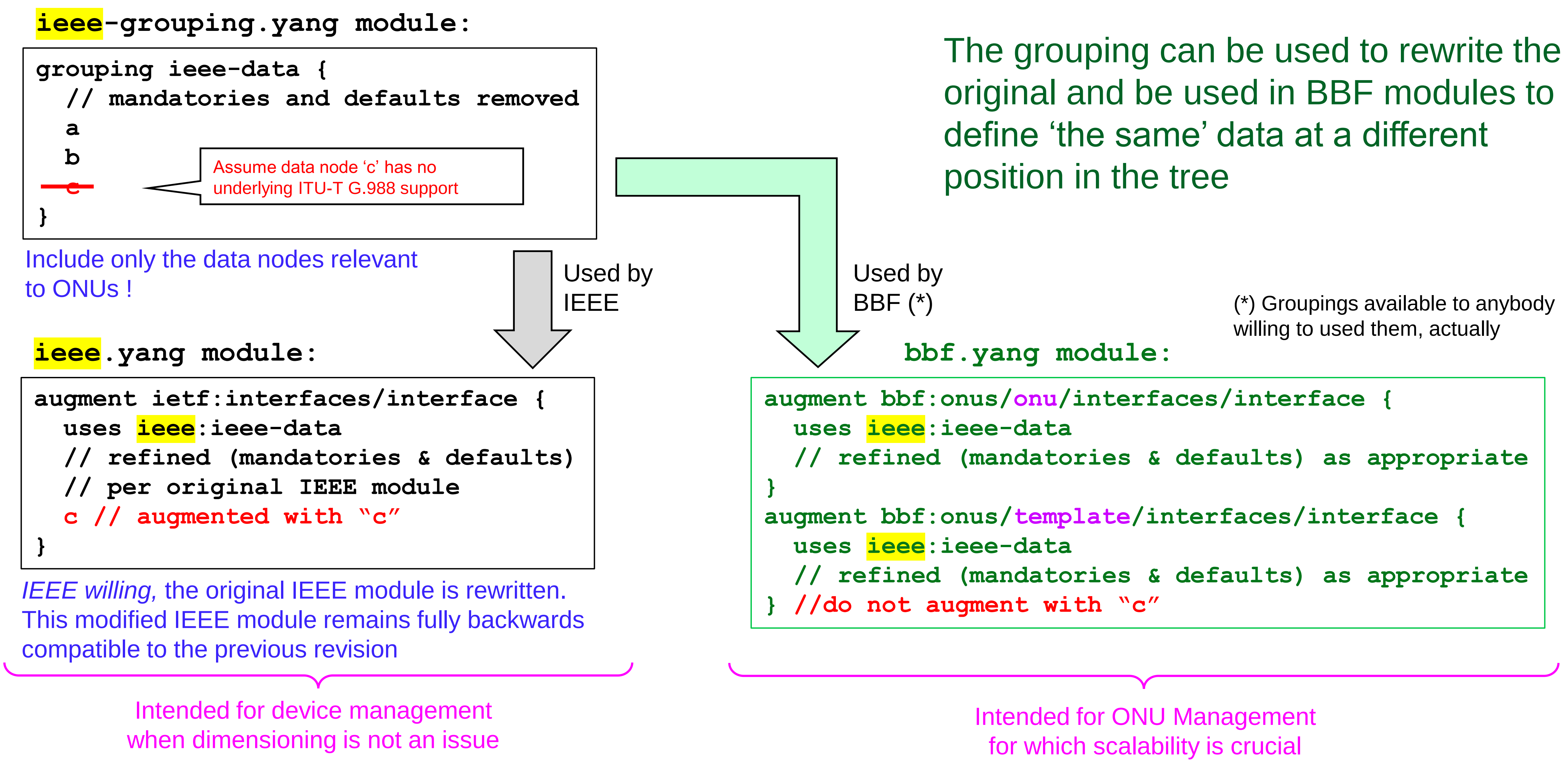
bbf.yang module:

```
augment bbf:onus/onu/interfaces/interface {
  uses ieee:ieee-data
  // refined (mandatories & defaults) as appropriate
}
augment bbf:onus/template/interfaces/interface {
  uses ieee:ieee-data
  // refined (mandatories & defaults) as appropriate
}
```

Intended for ONU Management
for which scalability is crucial

Alternative 1 - Step 2 (& Done): *Tune groupings, to contain only data nodes applicable to ONUs*

(these are the ones supported by ITU-T G.988)



ieee-grouping.yang module:

```
grouping ieee-data {
  // mandatories and defaults removed
  a
  b
  ~~c~~
}
```

Assume data node ‘c’ has no underlying ITU-T G.988 support

Include only the data nodes relevant to ONUs !

Used by IEEE

ieee.yang module:

```
augment ietf:interfaces/interface {
  uses ieee:ieee-data
  // refined (mandatories & defaults)
  // per original IEEE module
  c // augmented with “c”
}
```

IEEE willing, the original IEEE module is rewritten. This modified IEEE module remains fully backwards compatible to the previous revision

Intended for device management
when dimensioning is not an issue

The grouping can be used to rewrite the original and be used in BBF modules to define ‘the same’ data at a different position in the tree

Used by BBF (*)

(*) Groupings available to anybody willing to used them, actually

bbf.yang module:

```
augment bbf:onus/onu/interfaces/interface {
  uses ieee:ieee-data
  // refined (mandatories & defaults) as appropriate
}
augment bbf:onus/template/interfaces/interface {
  uses ieee:ieee-data
  // refined (mandatories & defaults) as appropriate
} //do not augment with “c”
```

Intended for ONU Management
for which scalability is crucial

Alternative 2: *IEEE defines groupings but does not use them, BBF uses them (and anybody else willing)*

ieee-grouping.yang module:

```
grouping ieee-data {
  // mandatories and defaults removed
  a
  b
  ~~c~~
}
```

Assume data node ‘c’ has no underlying ITU-T G.988 support

New IEEE module (contains ONU relevant data nodes from original modules)

Not Used by IEEE!

ieee.yang module:

```
augment ietf:interfaces/interface {
  a
  b
  c
}
```

No changes to the original module

The grouping is used in BBF modules

Used by BBF (*)

(*) Groupings available to anybody willing to used them, actually

bbf.yang module:

```
augment bbf:onus/onu/interfaces/interface {
  uses ieee:ieee-data
  // refined (mandatories & defaults) as appropriate
}
augment bbf:onus/template/interfaces/interface {
  uses ieee:ieee-data
  // refined (mandatories & defaults) as appropriate
} //do not augment with “c”
```

Intended for device management when dimensioning is not an issue

Intended for ONU Management for which scalability is crucial

Alternative 3: *BBF defines and publishes groupings based on IEEE data nodes*
(with agreed header, copyright, …)

`ieee.yang module:`

```
augment ietf:interfaces/interface {
  a
  b
  c
}
```

No changes to the original module

`bbf-grouping.yang module:`

```
grouping ieee-data {
  // mandatories and defaults removed
  a
  b
}
```

New (contains ONU relevant data nodes from original IEEE modules)

Used by BBF (*)

(*) groupings available to anybody willing to used them, actually

`bbf.yang module:`

```
augment bbf:onus/onu/interfaces/interface {
  uses bbf:ieee-data
  // refined (mandatories & defaults) as appropriate
}
augment bbf:onus/template/interfaces/interface {
  uses bbf:ieee-data
  // refined (mandatories & defaults) as appropriate
}
```

Intended for device management when dimensioning is not an issue

Intended for ONU Management for which scalability is crucial

Some additional considerations

- Leafrefs: absolute XPath references must become relative XPaths.

  Options that can be considered, on a case by case whatever is most practical:
  - Option 1: convert absolute XPath references into relative XPaths as functional equivalent
  - Option 2: remove the absolute XPath data node from the grouping and
    - when rewriting the original *IEEE* module, add the data node with the *absolute* XPath after the “uses” statement
    - in the *BBF* module, add the data node with a *relative* XPath when using the grouping

Note, this issue is:

- Not applicable to *ieee802-ethernet-interface*
- Applicable to *ietf-interfaces*
- Not yet studied for other IEEE modules

The way forward ?

- **Alternative 1 and Alternative 2 would involve technical resources of the IEEE / IETF / ITU-T**
  - The IEEE / IETF / ITU-T may not have the need to invest technical resource in the requirements of a PON access network

- **Alternative 3 would only involve minimum involvement of the IEEE / IETF / ITU-T**
  - The IEEE / IETF / ITU-T do not publish new modules, this is left to others
  - Make sure that the legal notice and copyright is satisfactory

- **Because of existing scaling issues in the field, a timely solution is requested by the industry. Alternative 3 might be an interesting candidate in this case.**

Takeaways

- To support BBF requirements on the management of access network equipment, there is a need for YANG modules that efficiently scale to support large-scale systems.
- Unfortunately, current IEEE / IETF / ITU-T standard YANG modules are proving unpractical to deploy in such large-scale systems.
- It is proposed to consider new YANG modules reusing data nodes defined in IEEE / IETF / ITU-T standards, with edited XPath expressions, using relative rather than absolute XPath leaf references, having targeted application of default and mandatory statements and with some other minor tunings.
- *There is no intention of changing existing IEEE, IETF nor ITU-T standard YANG modules(*).* Instead, *complementary* YANG modules are proposed that extend the domain of IEEE / IETF / ITU-T data nodes to support deployments beyond what was originally assumed.
- This contribution proposes several practical *realization alternatives*, with different levels of involvement by IEEE / IETF / ITU-T:
  - IEEE / IETF / ITU-T define “groupings” modules, on which sponsors of the contribution could collaborate, both usable:
    - in current IEEE / IETF / ITU-T standard modules (optional and fully backward compatible) and
    - in new BBF modules (possibly also other SDOs), without editing any IEEE / IETF / ITU-T files neither copying any part of.
  - Another interesting possibility that would not require any IEEE / IETF / ITU-T work involvement besides copyright and legal notice agreements would be that BBF defines the needed groupings itself, extracting the required data nodes from IEEE / IETF / ITU-T modules.

(*) Editing existing IEEE / IETF / ITU-T module as per alternative 1 is left to IEEE / IETF / ITU-T discretion.

Thank You.

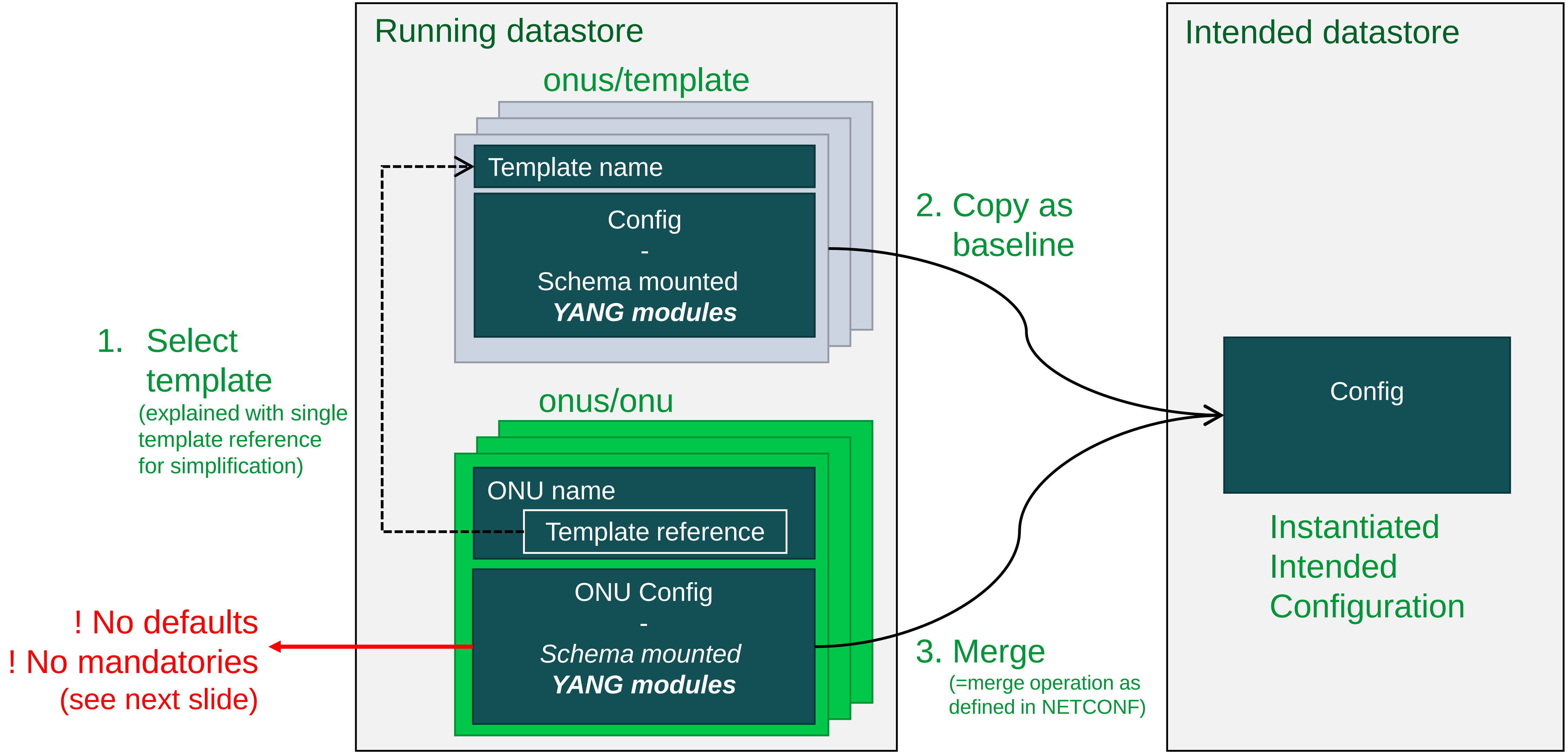
Back-up slide, 1/3: Why IETF Schema-mount (RFC 8528) has proved unsuitable

- It has been long hoped that IETF Schema Mount (RFC 8528) would be ideal for scaling ONU YANG models
  - It allows to deal with much fewer data nodes when managing a specific ONU
  - It gives the promise to recursively use *unmodified* YANG modules.

But this is not sufficient !

- With Schema Mount, each ONU datastore must be self-consistent hence a lot of data, although identical for many ONUs (e.g., profiles), must be repeated per ONU, leading to an *increase* of the datastore size.
  - RFC 8528 foresees the possibility that a mounted device refers to a parent data node, but after analysis, it has been anticipated that it will fall short due to conditions and limitations expressed in the RFC.
- Most significant problem: Schema Mount is incompatible with the *ONU template* technique, the key for scaling YANG models, *without editing the YANG modules*, hence defeating the very advantage of Schema Mount
  - An ONU template contains the data nodes of a typical ONU configuration (the YANG model of an ONU template has the same data nodes as the YANG model of an actual ONU).
  - Configuring an actual ONU instance is done by reference to an ONU template (to make a copy of the template) and only modifying the value of data nodes unique for a given ONU (e.g., TC layer identifiers)
  - The YANG model of an **ONU instance may not contain default statements** (would otherwise overrule specific values coming from the ONU template)
  - The YANG model of the **ONU instance may not contain mandatory data nodes** (might already come from the template)

Back-up slide, 2/3: Principles of using ONU templates



1. Select template (explained with single template reference for simplification)

2. Copy as baseline

3. Merge (=merge operation as defined in NETCONF)

! No defaults
! No mandatories
(see next slide)

Back-up slide, 3/3: The problem with mandatories / defaults and templates

Illustrated with ietf-interfaces: ‘type’ is mandatory, ‘enabled’ has default ‘true’

Data node characteristic	Running DS > ONU Template	Running DS > ONU instance	Intended DS (with source of the data)	Issue
Mandatory	*Configure*	*Configure (same as template)*	ONU instance	Unintended redundant configuration (memory)
	Configure	Configure (≠ from template)	ONU instance	Intended overrule
default	*Configure*	Configure	ONU instance	Intended overrule
	Configure	*Default*	ONU instance	Unintended overrule (silent side effects only known by YANG specialists)
	Default	Configure	ONU instance	Intended overrule
	Default	*Default*	ONU instance	Unintended redundant configuration (memory)
optional	Configure	Configure	ONU instance	Intended overrule
	Configure	Not configured	Template	Intended
	Not configured	Configure	ONU instance	Intended overrule
	Not configured	Not configured	Not existing	Intended not existing

The data from the template will never be used

We cannot modify IEEE / IETF / ITU-T modules.

Even if the module would be a BBF module we would not be able to remove defaults.

Solving the issues requires new modules for different handling of mandatories and defaults.

As said previously, this defeats the very advantage of RFC 8528 Schema Mount !