UNIVERSITÄT DUISBURG ESSEN

Open-Minded

IEEE YANG Locations

Johannes Specht, University of Duisburg-Essen

Stephan Kehrer, Hirschmann

Content

UNIVERSITÄT DUISBURG ESSEN
Open-Minded

Objectives

- Overview of the structures, processes, roles, requirements YANG files related to IEEE 802.1 and beyond
- Proposals for aspects of structures, …
- Consider multiple perspectives:
  - Editor
  - User
  - Std. Draft Reviewer

Needs and Desires (a.k.a. Requirements)

UNIVERSITÄT DUISBURG ESSEN
Open-Minded

To Be Extended

Users

- YANG files published by the Standards Board for public use
- Bleeding edge YANG files of
  - a particular IEEE project
  - multiple IEEE projects, maybe all

Editors

- Clear/symmetric procedures for YANG
- Early conflict discovery (e.g., multiple IEEE projects in flight, dependent IETF YANG updates)
- Avoid automated/weakly motivated work

Reviewers

- Bleeding edge YANG files touched by project X in PDFs: File and Listing
- Changes since last draft

Administrative

- Don‘t conflict with IEEE-SA processes and procedures

Overview of Locations

Locations

UNIVERSITÄT DUISBURG ESSEN
Open-Minded

YANG Catalog (not covered)

Registry that allows users to find models relevant to their use cases from the large and growing number of YANG modules being published.

YANG Catalog GitHub Repository (yangcatalog::git)

- Stores IETF, IEEE, … YANG files, as found in the YANG Catalog
- IEEE uses the master branch, organized in a draft folder (yangcatalog::git::master::draft), intended to store the latest versions of YANG files, and a published folder (yangcatalog::git::master::published), intended to store the versions published for public use.

IEEE Standards PDFs

- Standards/Amendment PDFs, for ballots during a project (drafts) or approved and published at project end.

IEEE Overview Table (proposed)

- Provides a Web-based overview of published IEEE YANG files

IEEE Owned File Location (proposed)

- Stores published YANG files from IEEE on an IEEE owned server

Overview

IEEE Overview Table

Date (dd/mm/yyyy)	Standard	Filename
15/06/2019	802.1Q<*p0*>-2019	ieee802-types@2019-05-28.yang
15/06/2019	802.1Q<*p0*>-2019	ieee802-dot1q-types@2019-05-28.yang
15/07/2019	802.1Q<*p1*>-2019	ieee802-dot1q-types@2019-06-28.yang
15/07/2019	802.1Q<*p1*>-2019	ieee802-dot1q-bridge@2019-06-28.yang
15/07/2019	802.1Q<*p1*>-2019	ieee802-dot1q-sched@2019-06-28.yang
15/08/2019	802.1Q-2019	ieee802-types@2019-07-28.yang
15/08/2019	802.1Q-2019	ieee802-dot1q-types@2019-07-28.yang
15/08/2019	802.1Q-2019	ieee802-dot1q-bridge@2019-07-28.yang
15/08/2019	802.1Q-2019	ieee802-dot1q-sched@2019-07-28.yang

IEEE 802 Owned File Location

```
+ published
  + 802
    + ieee802-types@2019-05-28.yang
    + ieee802-types@2019-07-28.yang
  + 802.1
    + ieee802-dot1q-types@2019-05-28.yang
    + ieee802-dot1q-types@2019-06-28.yang
    + ieee802-dot1q-types@2019-07-28.yang
    + ieee802-dot1q-bridge@2019-06-28.yang
    + ieee802-dot1q-bridge@2019-07-28.yang
    + ieee802-dot1q-sched@2019-06-28.yang
    + ieee802-dot1q-sched@2019-07-28.yang
    + ...
  + ...
```

yangcatalog::git::master

```
+ draft
  + 802
    + ieee802-types.yang
  + 802.1
    + ieee802-dot1q-types.yang
    + ieee802-dot1q-tsn-types.yang
    + ieee802-dot1q-sched.yang
    + ieee802-dot1q-preemption.yang
    + ieee802-dot1q-cfm.yang
    + ieee802-dot1q-cfm-bridge.yang
    + ieee802-dot1q-cfm-mib.yang
    + ieee802-dot1q-vlan-bridge.yang
    + ieee802-dot1q-bridge.yang
    + ...
  + ...

+ published
  + 802
    + ieee802-types@2019-05-28.yang
    + ieee802-types@2019-07-28.yang
  + 802.1
    + ieee802-dot1q-types@2019-05-28.yang
    + ieee802-dot1q-types@2019-06-28.yang
    + ieee802-dot1q-types@2019-07-28.yang
    + ieee802-dot1q-bridge@2019-06-28.yang
    + ieee802-dot1q-bridge@2019-07-28.yang
    + ieee802-dot1q-sched@2019-06-28.yang
    + ieee802-dot1q-sched@2019-07-28.yang
    + ...
  + ...
```

IEEE Standards Documents

IEEE 802.1Q<*p0*> Amendment Project

P802.1Q<*p0*> DRAFT X.X PDF
ieee802-types.yang
ieee802-dot1q-types.yang
ieee802-types.yang
ieee802-dot1q-types.yang

802.1Q<*p0*>-2019 PDF
ieee802-types@2019-05-28.yang
ieee802-dot1q-types@2019-05-28.yang

IEEE 802.1Q<*p1*> Amendment Project

P802.1Q<*p1*> DRAFT X.X PDF
ieee802-dot1q-types.yang
ieee802-dot1q-bridge.yang
ieee802-dot1q-sched.yang
ieee802-dot1q-bridge.yang
ieee802-dot1q-sched.yang

802.1Q<*p1*>-2019 PDF
ieee802-dot1q-types@2019-06-28.yang
ieee802-dot1q-bridge@2019-06-28.yang
ieee802-dot1q-sched@2019-06-28.yang

IEEE 802.1Q Revision Project

P802.1Q-REV DRAFT X.X
ieee802-types.yang
ieee802-dot1q-types.yang
ieee802-dot1q-bridge.yang
ieee802-dot1q-sched.yang
ieee802-dot1q-bridge.yang
ieee802-dot1q-sched.yang

802.1Q-2019
ieee802-types@2019-07-28.yang
ieee802-dot1q-types@2019-07-28.yang
ieee802-dot1q-bridge@2019-07-28.yang
ieee802-dot1q-sched@2019-07-28.yang

Dates

Filenames (only the YANG files touched by an IEEE project)

Overview Table

Revision Nodes in YANG Files

Overview of Dates

IEEE Overview Table

Date (dd/mm/yyyy)	Standard	Filename
15/06/2019	802.1Q<*p0*>-2019	ieee802-types@2019-05-28.yang
15/06/2019	802.1Q<*p0*>-2019	ieee802-dot1q-types@2019-05-28.yang
15/07/2019	802.1Q<*p1*>-2019	ieee802-dot1q-types@2019-06-28.yang
15/07/2019	802.1Q<*p1*>-2019	ieee802-dot1q-bridge@2019-06-28.yang
15/07/2019	802.1Q<*p1*>-2019	ieee802-dot1q-sched@2019-06-28.yang
15/08/2019	802.1Q-2019	ieee802-types@2019-07-28.yang
15/08/2019	802.1Q-2019	ieee802-dot1q-types@2019-07-28.yang
15/08/2019	802.1Q-2019	ieee802-dot1q-bridge@2019-07-28.yang
15/08/2019	802.1Q-2019	ieee802-dot1q-sched@2019-07-28.yang

IEEE 802 Owned File Location

```
+ published
  + 802
    + ieee802-types@2019-05-28.yang
    + ieee802-types@2019-07-28.yang
  + 802.1
    + ieee802-dot1q-types@2019-05-28.yang
    + ieee802-dot1q-types@2019-06-28.yang
    + ieee802-dot1q-types@2019-07-28.yang
    + ieee802-dot1q-bridge@2019-06-28.yang
    + ieee802-dot1q-bridge@2019-07-28.yang
    + ieee802-dot1q-sched@2019-06-28.yang
    + ieee802-dot1q-sched@2019-07-28.yang
    + ...
  + ...
```

yangcatalog::git::master

```
+ draft
  + 802
    + ieee802-types.yang
  + 802.1
    + ieee802-dot1q-types.yang
    + ieee802-dot1q-tsn-types.yang
    + ieee802-dot1q-sched.yang
    + ieee802-dot1q-preemption.yang
    + ieee802-dot1q-cfm.yang
    + ieee802-dot1q-cfm-bridge.yang
    + ieee802-dot1q-cfm-mib.yang
    + ieee802-dot1q-vlan-bridge.yang
    + ieee802-dot1q-bridge.yang
    + ...
  + ...

+ published
  + 802
    + ieee802-types@2019-05-28.yang
    + ieee802-types@2019-07-28.yang
  + 802.1
    + ieee802-dot1q-types@2019-05-28.yang
    + ieee802-dot1q-types@2019-06-28.yang
    + ieee802-dot1q-types@2019-07-28.yang
    + ieee802-dot1q-bridge@2019-06-28.yang
    + ieee802-dot1q-bridge@2019-07-28.yang
    + ieee802-dot1q-sched@2019-06-28.yang
    + ieee802-dot1q-sched@2019-07-28.yang
    + ...
  + ...
```

```
revision 2019-05-28 {
  description
    „Some description …";
  reference
    "IEEE Std 802.1Q<p0>-2019";
}
```

IEEE Standards Documents

IEEE 802.1Q<*p0*> Amendment Project

P802.1Q<*p0*> DRAFT X.X PDF
ieee802-types.yang
ieee802-dot1q-types.yang
ieee802-types.yang
ieee802-dot1q-types.yang

802.1Q<*p0*>-2019 PDF
ieee802-types@2019-05-28.yang
ieee802-dot1q-types@2019-05-28.yang

IEEE 802.1Q<*p1*> Amendment Project

P802.1Q<*p1*> DRAFT X.X PDF
ieee802-dot1q-types.yang
ieee802-dot1q-bridge.yang
ieee802-dot1q-sched.yang
ieee802-dot1q-bridge.yang
ieee802-dot1q-sched.yang

802.1Q<*p1*>-2019 PDF
ieee802-dot1q-types@2019-06-28.yang
ieee802-dot1q-bridge@2019-06-28.yang
ieee802-dot1q-sched@2019-06-28.yang

IEEE 802.1Q Revision Project

P802.1Q-REV DRAFT X.X
ieee802-types.yang
ieee802-dot1q-types.yang
ieee802-dot1q-bridge.yang
ieee802-dot1q-sched.yang
ieee802-dot1q-bridge.yang
ieee802-dot1q-sched.yang

802.1Q-2019
ieee802-types@2019-07-28.yang
ieee802-dot1q-types@2019-07-28.yang
ieee802-dot1q-bridge@2019-07-28.yang
ieee802-dot1q-sched@2019-07-28.yang

Dates?

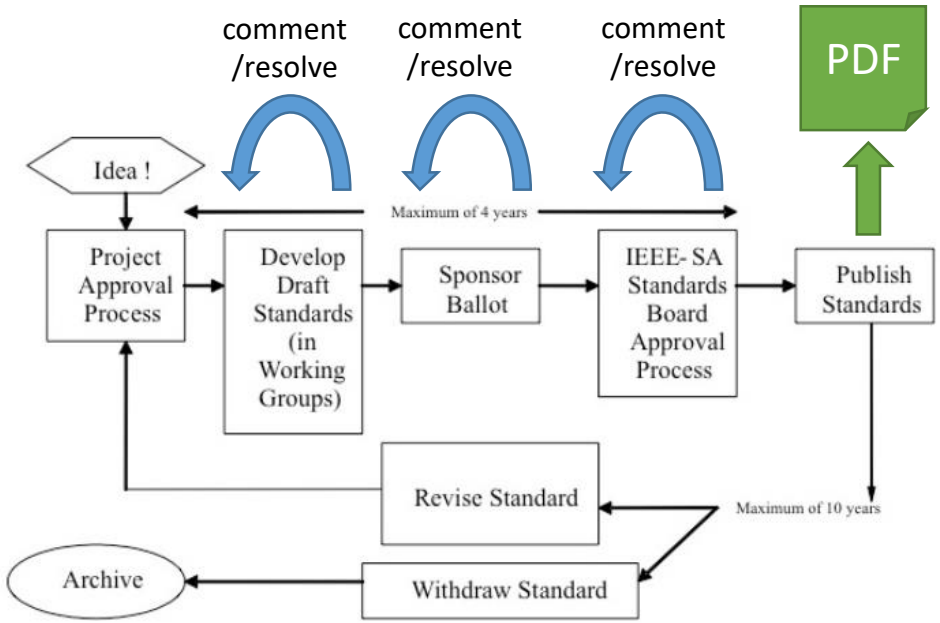
UNIVERSITÄT DUISBURG ESSEN
Open-Minded

Question

- What date should be in the date tagged filenames AND the revision node?

Answer

- Ideal
  When the final PDF is approved by the Standards Board for public use …
- Caveat
  Who should do this? The project is done by then, we can‘t update YANG files/their contents/their listing in the PDF.



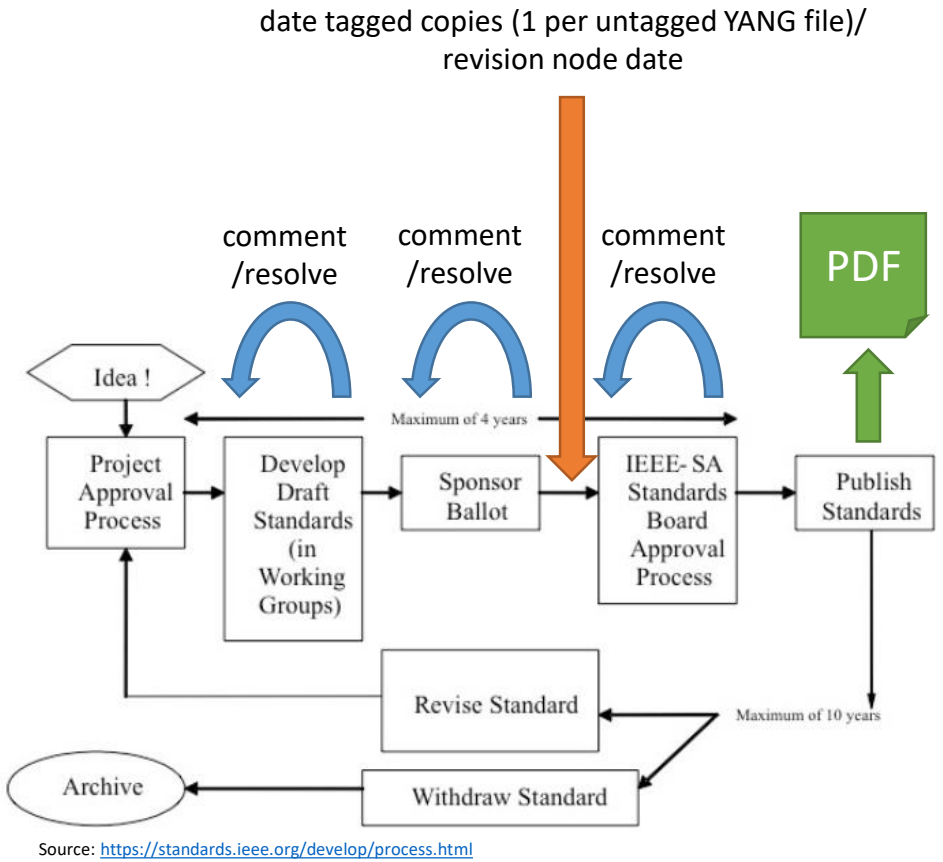
Source: https://standards.ieee.org/develop/process.html

Dates: Ideal Solution Approximated

UNIVERSITÄT DUISBURG ESSEN
Open-Minded

As late as possible (to be discussed)

- End of sponsor ballot (submission to RevCom):
  - Submission date in revision node
  - Date tagged file copies created
- Before:
  - Temporary date(s) in the revision node (the current date)
  - NO date tagged file copies
- Afterwards, during Stds. Board Approval Process (options):
  1. First RevCom Submission
     Every YANG file created or modified by the project; no changes at 2nd, 3rd, etc. RevCom submission.
  2. First RevCom Submission AND on YANG update
     Like before, but in addition for each subsequent RevCom submission that required changes in at least one YANG file created or modified by the project
  3. Every RevCom Submission
     Update revision node and date tagged copies each time. May cause a series of deprecated date tagged file copies?!



Source: https://standards.ieee.org/develop/process.html

YANG: Revision Nodes?

UNIVERSITÄT DUISBURG ESSEN
Open-Minded

Question

- When to create a revision node in the YANG files in progress, and what should be contained?

Answer

- When:
  - Only once (AND early) during the course of a project seems reasonable.
  - The text can remain identical until project end.
  - Adding one revision node per PDF draft version, or even at additional points in time, seems unreasonable:
    - If two projects A and B share a file, how would this look like?
    - … And in which order do the intermediate version revision nodes interleave in a shared file in yangcatalog::git::master::draft?
    - Who needs the history of all drafts of multiple projects in the YANG revision nodes, after several projects tweaked the same file?
    - … And if it‘s needed, why is it needed?
- What:
  To be discussed – ideally, we have a template (proposal on next slide).

YANG: Revision Node Proposal

```
revision 2019-05-28 {
  description
    "Initial revision from IEEE P802.1<p0>.";
  reference
    "IEEE Std 802.1Q<p0>-2019";
}
```

```
revision 2019-06-28 {
  description
    "Revision from IEEE P802.1Q<p1>.";
  reference
    "IEEE Std 802.1Q<p1>-2019";
}
```

```
revision 2019-07-28 {
  description
    "Revision from IEEE P802.1Q-Rev.";
  reference
    "IEEE Std 802.1Q-2019";
}
```

Drafts: No @YYYY-MM-DD?

UNIVERSITÄT DUISBURG ESSEN
Open-Minded

Question

1. Why is there no data tag (@YYYY-MM-DD) in filenames in
   1. yangcatalog::git::master::draft
   2. P802.1*-D*.* PDFs?

Answers

1. Having dates here adds zero information:
   1. The latest is attached to P802.1*-D*.* PDFs
   2. Likewise, git helps to find a particular version by proper commit messages
   3. the latest draft YANG modules, NOT considering particular 802.1 projects but 802.1 as a whole, can be obtained by using untagged files from yangcatalog::git::master::draft (i.e., no need to manually find out the most recent date)
2. Adding them makes it harder for users and reviewers
   Multiple "interleaved" copied of a shared file in yangcatalog::git::master::draft, where one subset belongs to project A, and another subset to project B, would make usage of the draft folders content hard/questionable (e.g., "lets get the latest for project A" → latest = most recent date → "oops, got the one from project B").
3. Adding them makes it harder for editors
   If the date would be added to draft filenames in yangcatalog::git::master::draft, editors of parallel IEEE projects can easier oversee conflicts in shared files – these would be distinct files in GIT, omitting conflict notifications.

However

If there is the need/desire to add dates to draft YANG files, we may re-consider the purpose of a version control system like git …

Bleeding Edge Versions of Unfinished Projects

Users: Get the most recent files under development

Editors: Check (early) for conflicts with other projects

yangcatalog::git::master::draft: No bleeding edge?

UNIVERSITÄT DUISBURG ESSEN
Open-Minded

Question

- Why doesn‘t (or at least, shouldn’t …) yangcatalog::git::master::draft contain the most recent file versions of IEEE projects in flight? This seems to be not the way git is supposed to be used.

Answer

- We can‘t/we shouldn’t:
  In case of files shared amongst multiple active projects, the YANG files in a published PDF of project A would contain *fragments of other active projects* (e.g., project B).
  - This would exceed the PAR scope of project A
  - Not violating the PAR scope would be error prone and tedious
    Unfortunately, we do not know how to automatically *exclude* the fragments of project B from files for PDF release of project A. The editors of projects would have to do this manually, for each PDF draft, and the final PDF at completion of each project.

yangcatalog::git::master::draft: Why?

UNIVERSITÄT DUISBURG ESSEN
Open-Minded

Question

- Why is yangcatalog::git::master::draft needed, if updates on yangcatalog::git::master should only happen at project end?

Answer

The question is valid (!!!):

- A “draft” folder, only containing “published” files, appears confusing.
- Alternative:
  - The files currently in yangcatalog::git::master::draft (i.e., without date tags in filenames) could be directly located in yangcatalog::git::master::published, plus those with date tags …
  - … as an extra, co-existing date tagged and non-date tagged “hard” files avoid symlink related issues (some operating systems) …
  - … if everything is in yangcatalog::git::master::published, one folder level (draft and published) may be eliminated

Where is the bleeding edge?

UNIVERSITÄT DUISBURG ESSEN
Open-Minded

Question

- If yangcatalog::git::master::draft does not contain the bleeding edge versions, where can I find it?

Answer

- Bleeding edge of a single active project A:
  In the git branch of project A, AND in the projects most recent draft PDF.
  Note that shared files with any parallel active project B do not contain the latest changes from project B.
- Bleeding edge of all active projects:
  Users/editors can merge the git branches of multiple unfinished projects locally.
  Note that the merge result is ambiguous: The editor may make different merge decisions for the final publication at project end, especially if conflict resolution of shared files is needed.

Branches in yangcatalog::git?

UNIVERSITÄT DUISBURG ESSEN
Open-Minded

Question

- Can project editors effectively use branches in yangcatalog::git?

Answer

- IIRC (JS) from November 2018, they can‘t
- If so, yangcatalog::git can‘t be used for developing YANG files during the course of an IEEE project. Development has to happen somewhere else, followed by a single pull request to yangcatalog::git::master at project end…
- No git‘s dependency tracking to create local merges of multiple project branches to get a bleeding edge version …

What can we do?

- Editor‘s have an individual copy, work on it, pull request to yangcatalog::git::master at project end
  → Getting the bleeding edge is rough, i.e., collect YANG files attached to PDFs and merge without git‘s dependency tracking
- Have a second space (git repository)
  - every IEEE Std project can have a branch
  - Single pull request to yangcatalog::git::master each time an IEEE project completes
  - Some options:
    - An IEEE server (open source?), or
    - another github project (pull requests may be easier)

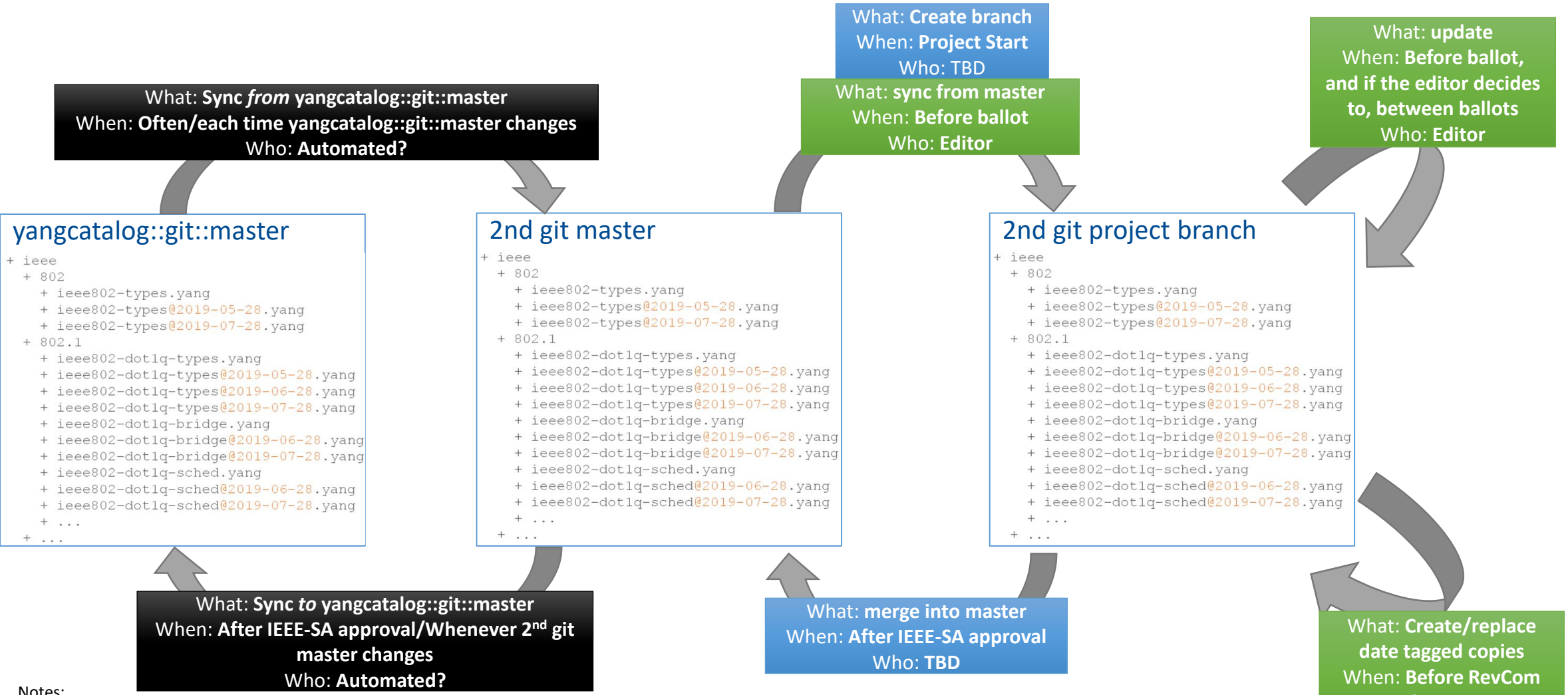
Can we Improve?

UNIVERSITÄT DUISBURG ESSEN
Open-Minded

- The entire topic is as complicated as it is …
  - Dates
  - Folder structures
  - Files shared between Projects
  - Redundant files
  - …
- If yes, this could be done as follows:
  1. **No date tags in draft filenames**
     Explained before. Instead, date tagged copies only at projects end – one corresponding revision entry in these files.
  2. **Merge ”draft“ and ”published“ folders**
     Explained before.
  3. **Use a second git repository**
     For development during projects – allows branches for projects, while tracking shared files, getting bleeding edge versions, etc. is supported
  4. **Avoid an additional IEEE owned file location**
     The second git repository IS the IEEE owned location

How a Second Git Repository Could Work (Discussion)

UNIVERSITÄT DUISBURG ESSEN
Open-Minded



Notes:

- Frequent sync. between yangcatalog::git::master is needed because YANG modules from other SDOs are referenced by IEEE modules
- Two folders may still be used, e.g. “latest” and “history”. However, implications (e.g., for existing YANG tools) need to be considered
- Automatic sync between both repos. may be simple: No file conflicts, basic “fork” mechanisms of git, etc.

Thank you for your Attention!

UNIVERSITÄT DUISBURG ESSEN
Open-Minded

Questions, Opinions, Ideas?

Johannes Specht
Dipl.-Inform. (FH)

Dependability of Computing Systems
Institute for Computer Science and
Business Information Systems (ICB)
Faculty of Economics and
Business Administration
University of Duisburg-Essen

Schuetzenbahn 70
Room SH 502
45127 Essen
GERMANY
T +49 (0)201 183-3914
F +49 (0)201 183-4573

Johannes.Specht@uni-due.de
http://dc.uni-due.de

Other Questions/Answers

“experimental“ vs. “standard“?

UNIVERSITÄT DUISBURG ESSEN
Open-Minded

Question

- Why is naming/structure/content in GitHub directory “standard“ different, compared to “experimental“?

Answers

- The IEEE 802.1 project editor and/or the IEEE 802.1 group decided that naming/structure/content in “standard“ should be different than in “experimental“.
- There is no obligation to use contents “experimental“ in approved projects.

No IEEE project ID in filenames?

UNIVERSITÄT DUISBURG ESSEN
Open-Minded

Question

- Why are IEEE YANG files not called like the IEEE project ID (e.g. ieee802-dot1-qa?.yang)?

Answer

- Not user friendly in the long run
  Even for a user involved in Stds. development, it‘s hard to remember the scope of a project (e.q., … what was Qa? about?)
- Not good at all: Amendments vs. YANG
  - Amendments can implement maintainance items, thus tweaking deeply in the YANG tree introduced in a former amendment. This can result in a flood of „augment“ nodes, making the YANG file harder to read. IIRC, there were some restrictions, too.
  - More worse, removing or replacing former nodes is not possible in YANG. It would thus be required to touch the files of already finished projects anyway.