Release Cadence Proposal

This is just a draft version put together to capture the feedback provided during the meeting and present my point of view based on my experience in other open source projects

Current state

- Patches are submitted via gerrit
- Architecture is described on a wiki and (sometimes) later migrated to read the docs
- We have a waterfall-ish development model with clearly distinguish planning, designing, implementation and testing phases
- There is a lot of paper work to be done at every milestone
- Features for every release are described globally (per release) and approved by TSC in the beginning of each release
- We have 2-3 releases per year

Glossary

Feature - a high level design of new functionality that impacts multiple components. Has to be approved by subcommittees, impacted PTLs and TSC. Just technical details, no resource allocation.

Spec - a detailed level design design of a change that is planned but focused on a single component. Has to be approved by impacted project Team. Just technical details, no resource allocation.

ONAP best practice - "coding standard" (may be code related, security related, configuration related etc.) recognized by the community that should be followed. Approved by TSC, has to be followed by any new code entering the tree. Enforced for a code arriving for a review after approval.

Global requirement - best practice, non functional requirement chosen by the TSC to be applied to whole ONAP code base during a given release. Enforced since beginning of release X for whole ONAP code base. Stays forever

Assumptions

1. DON'T BREAK ANY THE MASTER!
2. All approved best practices are checked in gerrit review and enforce for any new code that is entering the tree
3. Global requirements are mandatory for all projects. If project fails to deliver global requirements, it is descoped from the release (previous release containers are used).
4. TSC defines the vision and sets the direction for ONAP project
5. Based on recommendations from subcommittees, TSC makes the final decision whether the new feature should be approved or not.
6. TSC makes the final decision whether the best practice should be approved or not.
7. Architecture team consists of ONAP specialists (both on functional and implementation view) that can clearly assess whether the proposed feature is aligned with long term ONAP vision and is aligned with ONAP design principles
8. Security Team consists of ONAP security specialists that can clearly assess whether the proposed feature does not compromise ONAP security standards
9. All other subcommittees consists of ONAP specialists in a given domain who help to asses the proposal if it touches their domain of interest
10. Anyone is free to propose a new feature into ONAP at any point of the time but it has to be scoped to the release before the deadline in order to be considered as active
11. Anyone is free to assess the proposed feature from a technical point of view
12. Anyone is free to propose new best practice at any point of time
13. Before project meets all global requirements for a given release, it cannot make backward incompatible changes that are impacting other components.
14. Project may release a new docker image at any point of the release (taken into account previous point)
15. PTL is free to define additional rules and quality metrics that has to be met before the patch can be merged
16. Anyone is free to submit any patch at any point of the time
17. It's PTL & committers responsibility to make sure that patches they merge for a given branch obey to the rules set by TSC (best practices, release phase)
18. PTL and committers are responsible for the project condition
19. PTL and committers have a full control over what and when should be merged (ie. They can prevent functional patches from being merged until global requirements for their project are met)
20. It's up to the feature/spec owner to organize resources to implement it.
21. Features can be worked on since approval until they are ready, no matter if it takes a release of 5 years. The release is always on time just like a train, always on time. You missed one, no issues, you can release new docker as a part of the next release as soon as you are ready.

Development

- Keep 3 releases in a year:
    - W8
    - W24
    - W43
- Divide the release in 3 sections:
    - R - 16 Release kickoff (example October 2020 for Honolulu right after Guilin signoff)
        - Start estimating which global requirements can be met in this release
        - Start socializing global requirements proposals with PTLs so that every one knows what has to be done in this release
        - Features and specs work is ongoing during this phase as well.

    - R - 14 Global requirements defined
        - TSC approves a list of global requirements that has to be address by the teams in this release
    - R - 10 Spec freeze (example December 2020 for Honolulu)
        - All features and specs that are going to be implemented in this release should be approved and marked as active
        - At this point we know what we will get in this release (at most)
    - R - 5 Feature freeze (example February 2021 for Honolulu)
        - End of accepting patches containing new features for this release
        - Start of bug fixing period
        - **Release branch is being created for all participating projects**
            - particularly OOM so that we can start to focus on the release package of helm charts and corresponding docker containers
    - R - 3 Beginning of RC-phase (example February 2020 for Honolulu)
        - **RC every end of week**
        - Bug fixes submitted for both master and release branch
        - Feature development may continue for approved specs in the master branch
    - R - 0 - final testing, release notes and Sign-off (example March 2020 for Honolulu)
    - R - 16 of next release (example November 2020 for H release)
- Every spec has below stages in its life cycle:
    - Proposed
        - The spec has been proposed by the author for a review by affected parties
    - Approved
        - The spec has been reviewed by interested parties and approved
        - In this point we agree that the design is good and if agreed we find the resources so it can be implemented
    - Scoped for the release
        - There is a volunteer to work on this particular spec within this release
        - The work may be continued in next release with PTL and TSC consent.
    - Implemented
        - The spec has been fully implemented and should be mentioned in the release notes
    - Dropped
        - The spec has been dropped because no one is willing to work on it or it became out of the date

Recording from 08.07.2020:

Please move the calendar to March to see the impact of our proposal on the release cycle.

Team Calendars