

OPNFV Release Process 2.0

 THE **LINUX** FOUNDATION

 **OLF** NETWORKING

Considerations

- › Reconcile CNTT requirements with OPNFV
- › Simplify and reduce the number of milestones
- › Support OPNFV level requirements planning
- › Improve release planning at the project level
- › Improve accountability across all project types
- › Enable project self releases independent of OPNFV cadence
- › Increase community engagement in the release process

Assumptions

- › The OPNFV TSC supports a regular OPNFV release
- › The release cadence will remain at approximately two per year
- › Supporting CNTT requirements will be the highest priority for OPNFV for the foreseeable future
- › Supporting CNTT requirements will require a coordinated response from OPNFV, sometimes affecting multiple projects
- › Requirements will also come from other sources besides CNTT

OPNFV Level Requirements Planning (MI)

- › Why is this important?
 - › We need a way to agree upon and to prioritize broad requirements that help to advance our mission, or to take respond to a concern that affects most projects.
 - › Example: Python 3 migration
 - › We need a way to address CNTT requirements affecting multiple projects.

OPNFV Level Requirements Planning (MI)

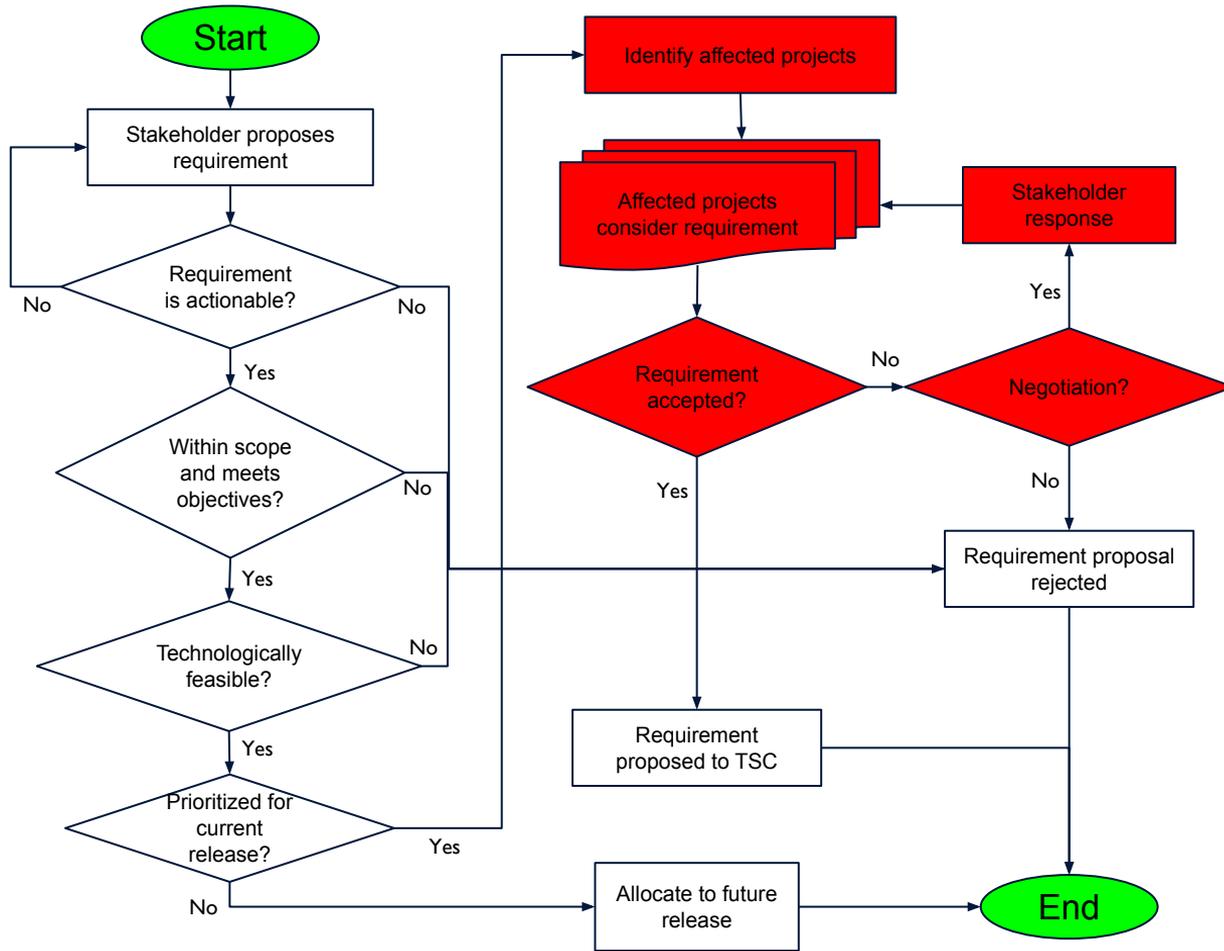
- › Overview
 - › Requirements working group or subcommittee gathers and vets requirements from multiple sources and makes recommendation to the TSC
 - › OPNFV Release Requirements approved by TSC at MI/2/3
 - › Requirement is de-scoped if not approved by the TSC
 - › Projects agree to prioritize OPNFV level requirements
 - › Each requirement has an owner and is documented in JIRA
 - › TSC approval at MI requires support commitment for each requirement from relevant projects
 - › Support documented in project release plan

What does “vetting” mean

- › A requirements group is needed to vet proposed requirements. Why?
 - › We want to focus our time and energy in a given release cycle on requirements that:
 - › Are actionable, i.e., clearly defined with sufficient detail for implementation
 - › Are within scope, and meet OPNFV objectives, as defined by the Charter, the Governing Board, and the TSC
 - › Are technologically feasible
 - › *Have support and sufficient resources in the affected projects as agreed to by the PTL*
 - › Have priority relative to other eligible requirements
- › The requirements group will apply the above criteria to requirements proposed by stakeholders and make a final proposal to the TSC.

More on verifying support among affected projects

- › The requirements group helps stakeholders identify and contact the affected projects
- › The affected projects consider the proposed requirement and indicate whether they will support the requirement, or not
- › If the project does not support the requirement, for example, due to a shortage of resources, then this may start a negotiation between the stakeholder and the project.
- › Note that the requirements group *facilitates* communication with affected projects. It's up to the stakeholder to determine support.



Stakeholder Responsibility

- › Prepare and present a detailed, well defined requirement proposal to the requirements subcommittee
- › Respond to questions and direction from the requirements subcommittee
- › Work with the requirements subcommittee to determine which OPNFV projects are affected by the requirement
- › Engage with the affected projects to determine whether they will support the requirements. If possible, negotiate issues such as resource constraints.
- › Once the requirement is accepted by the TSC for the release, monitor and report status back to the TSC at each milestone, or upon request.
- › Work with the projects to overcome blocking issues to successfully complete the requirement for the release.

Requirements Example

OPNFV Level Requirements Planning (MI)

- › Requirements Working Group or Subcommittee
 - › Gathers requirements recommendations from community
 - › Allocates requirements to releases
 - › Ensures that requirements have support from affected projects
 - › Ensures that requirements have an owner and are well documented in JIRA
 - › Recommends a set of prioritized requirements to the TSC for approval for the current release

Project Release Plans

- › Template based
- › Reviewed and approved as part of release process
- › Commitment to OPNFV-level requirements
 - › Document how requirement will be met
- › Other project objectives for the release
- › Specify deliverables
- › All work documented in JIRA and assigned to release

Independent Projects

- › Definition
 - › Not dependent on, or a dependency of, any other project in OPNFV
- › Self-declared (if applicable)
- › Requirements
 - › OPNFV repo
 - › CI using OPNFV resources
 - › Documentation (TBD) and release notes
 - › Self verification (project asserts readiness to release)

Self Release

- › All projects will maintain internal versioning
 - › OPNFV release versioning will follow current practice of using the prefix “opnfv-” on version numbers to distinguish them.
- › Projects will be required to contribute to OPNFV releases, approximately every 6 months, that meet OPNFV requirements established by the TSC and the release process.
- › In addition, projects may release independently, using a Self-Release Process (TBD)

Documentation

- › The current documentation is organized around the traditional OPNFV concept of “scenarios,” which is no longer a prominent aspect of OPNFV.
- › Need to reconcile CNTT documentation with OPNFV documentation organization and process.
- › Ask the DOCS project lead an effort, along with other stakeholders, to develop and propose new documentation structure to the TSC.
- › Continue current practice of having milestone requirements for preliminary and final documentation as part of release process.

Integration and Gating

- › An integration project will track project test and integration status, and will report this information to the release manager and to the TSC.

Milestones

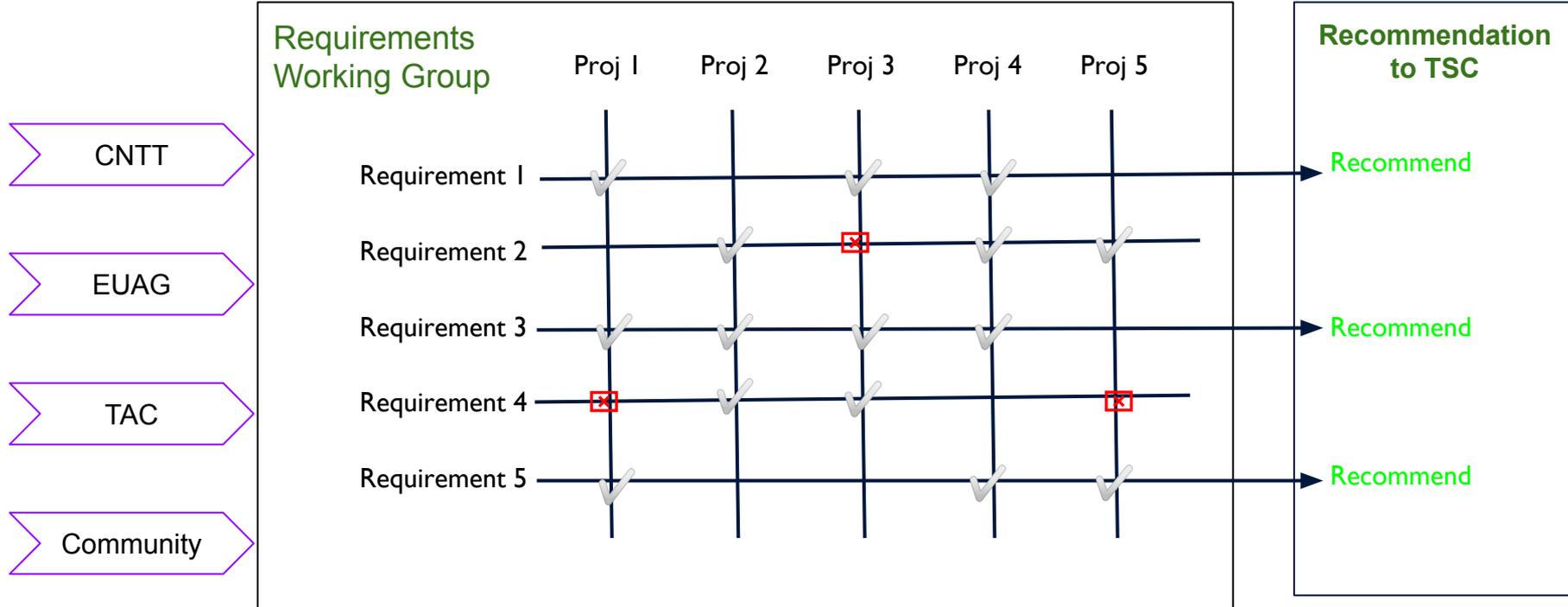
- › Projects must complete tasks at each milestone to be approved to proceed in the release
- › Requirements are evaluated at each milestone to determine whether they remain feasible
- › Milestones:
 - › M0 - Start of Release
 - › M1 - Planning Complete
 - › M2 - API / Functional Freeze
 - › M3 - Code Freeze
 - › RC0 - First Release Candidate
 - › RCn - Final Release Candidate

ToDo - Prerequisites for initiating new process

- › Establish requirements working group
- › Establish integration management project
- › Determine new documentation layout/organization
- › Develop self-release process
- › Develop project release plan template

BACKUP

OPNFV Level Requirements Planning (MI)



Milestones: Planning (M0 ⇒ M1)

- › Requirements gathered, reviewed, and approved by TSC
- › Project release plans completed, reviewed, and approved
- › All work planned for the release is documented in JIRA and assigned to the release (fix version field)
- › Risks documented

Milestones: API & Functional Freeze (M1 ⇒ M2)

- › Resolve integration blocking issues
- › Resolve license scan issues
- › Update risks documentation

Milestones: Code Freeze (M2 ⇒ M3)

- › Resolve high priority JIRA issues
- › Complete preliminary documentation
- › Update risks documentation

Milestones: Release Candidate 0, 1, ..., x

- › Verify release plan, including all planned testing, has been completed
- › Resolve high priority JIRA issues
- › Prepare and verify release artifacts
- › Complete final documentation
- › Complete tagging