Release Management Principles for LFN Projects

THE LINUX FOUNDATION

LF NETWORKING

Introduction

- The purpose of this presentation is to give an overview of some basic release management techniques and strategies to consider for LFN projects
- Every project is different, so these techniques will need to be adapted as needed

Cadence

- Most projects choose to release on a periodic basis, say twice per year, or every 6 months.
- You may feel that you want to be more aggressive, but remember that this is an open source project and the participants are typically working on a part-time basis.
- Also, be careful about maintenance releases, particularly if you are thinly resourced
  - Distracts from the next major release
  - Often requires an independent CI workflow

Release Phases

- Once you have decided on a cadence, you can then break that time period into different phases. Again, each project will be a little different, but typically, these include:
  - Requirements collection and project release planning
  - Development
  - Component test and debug
  - Integration test
- As you are determining these phases, also think about who will be doing the work. For example,
  - Requirements subcommittee
  - Integration team
  - Release manager

Requirements Gathering

- You will want to decide on a set of requirements for the release that support your project's high level purpose and goals
- These requirements should be analyzed to determine which project teams are affected
  - It's essential that the projects affected by the requirement agree to do the work. If not, then the requirement should be altered or deferred.
- It's not unusual to have more requirements than capacity. This means that the requirements will need to be prioritized, so that the lower priority issues may be deferred
- Finally, the TSC should vote on whether to approve the proposed requirements for the release

(Sub)Project Release Planning

- Each project participating in the release must submit a project release plan that includes:
  - Scope
  - Features (typically a list of Jira epics)
  - Test & verification
  - Dependencies
  - Deliverables
- This helps the TSC, other projects, the integration team, the release manager, and marketing function understand what the project intends to accomplish
- It also provides some assurance that the project team has put thought into all aspects of the release
- Use a template

Release Milestones

- Release milestones are useful for gauging progress relative to the schedule.
- Milestones are placed at the end of each phases
- Keep your milestones fairly general. Avoid tying milestones to specific technology or projects.
  - This will keep your release process from going stale as technology and projects change.
- Each milestone requires effort by the entire team, therefore, keep the number to a minimum, say 5 - 7

Release Milestones

- To be effective, each milestone should have measurable criteria. A milestone might sound good on paper, but it's useless if it can't be measured
- Use Jira to document the criteria as a set of tasks for each project team.
- The release manager may then check the status of the project teams by reviewing the status of the release management tasks assigned in Jira
- The TSC should approve that the release criteria has been met before moving on.
  - This may require adjusting the schedule or agreeing to exceptions, or to move certain tasks until later in the release.

Example - ONAP Guilin

- Schedule
- Status