Anuket Release Process

Issues and Objectives

THE LINUX FOUNDATION

LF NETWORKING

Release Process Objectives

- For the Reference Model (RM) and each Reference Architecture (RA), deliver a compliant[†] Reference Conformance (RC) suite, and a Reference Implementation (RI) for use by the telecom industry for VNF / CNF development.
  - [†] Compliance is based on RM/RA/RC/RI specifications.
- Provide appropriate release artifacts, including documentation, such that the RM, RA, RC, and RI may be readily consumed by the telecom industry.
- Provide a path for release for projects that do not *currently* contribute directly to RC or RI.
- Coordinate releases with marketing and events to promote the Anuket project, bring awareness to the industry, encourage deployment, and attract contributors.

Overview

- Should specifications and software have an integrated, lock-step release process, or a loosely coupled release process?
- What should the release cadence be?
- What specific release artifacts will be delivered?
- What's the best means for software developers to provide input to specification development prior to publication/release?
- What level of compliance between software and specification is sufficient?
- What cross-project integration testing is required and how will it be accomplished?
- How should projects be released that do not *currently* have a direct contribution to RC or RI?
- What is the role of installers?
- Unified release names. What naming scheme?

Integrated vs. Loosely Coupled Release Process

- Integrated definition: a single release process for specifications and software.
- Integrated Pros:
  - Simple
  - Each release includes RC + RI, as well as associated specifications.
  - Immediate feedback between spec and sw development
- Integrated Cons:
  - Develop and agree upon new process steps and milestones
  - Possibly throttles specification development

Integrated vs. Loosely Coupled Release Process

- Loosely Coupled definition: separate release processes for specifications and software.
- Loosely Coupled Pros:
  - Separate release processes already exist
  - Specification development can proceed at its own pace
- Loosely Coupled Cons:
  - Over time, could have a broad divergence between specifications and software
  - Which specifications are selected for implementation?
  - Feedback between software and specification dev is less direct
  - Potentially confusing to consumers/doesn’t present a unified project stance.

Notes (TSC 01/12)

- Upstream dependencies? Loosely coupled less affected.
- RC and RI not coupled. Should have separate release streams.
- RC ⇒ RA ⇒ RM.
  - If no change to RA or RM, then RC stays the same (except for compliance improvements or bug fixes)

Release Cadence

- Should specifications and software have the same cadence? What is the implication if they have different cadences?
- OPNFV software has traditionally been released on a ~6 month cadence (twice per year). Continue?
- CNTT has released two major releases and two minor (optional) releases per year.

Release Artifacts

- Specifications (RM, RA, RC, RI)
  - RM/RA have requirements and specifications
  - RM ⇒ RA ⇒ RC
  - RA ⇒ RI
  - RA and RI may have more than one instance
  - One RC per RA, but multiple possible RIs
  - Specification related documentation
- Software
  - RC test suite and RI (one or more)
  - Project releases (Barometer, etc.)
  - Manifest (documents compliance of RC & RI to specifications)
  - RC and RI software documentation (user guide, installation, etc.)
  - Project documentation

Input to Specification Development

- What's the best means for software developers to provide input to specification development prior to publication/release?
  - Why? Avoid delays and change management overhead of discovering issues after publication.
  - Methods
    - Informal?
    - PR Process?

Level of Compliance

- What level of compliance between software and specification is sufficient?
  - Another way to ask this is, what is the MVP?
- For example, what if compliance is only 50%. Is it even worth releasing?

Integration Testing

- What cross-project integration testing is required and how will it be accomplished?
  - Jerma example

Projects Not Contributing to RC/RI

- How should software projects be released that do not *currently* have a direct contribution to RC or RI?
  - Self-release process? What would that look like?

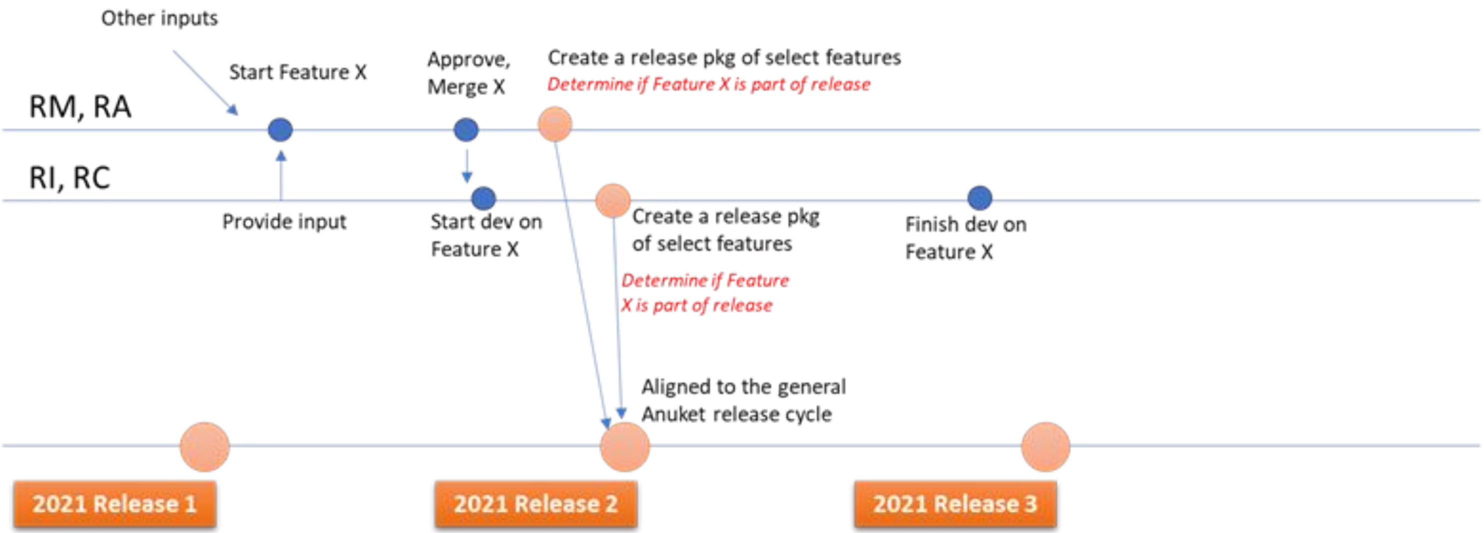
Installers

- What is the role of installers?
  - Currently, Airship is the only stand-alone installer project
    - BMRA being used by Kuberef for RI2 deployment
  - In the past, OPNFV had as many as 6 or 7 installers participating
  - Should installers be stand-alone projects?
  - What if another installer project wants to join Anuket? Does it depend?

Unified Release Naming

- What release naming scheme should we use?

Anuket Release Input



Other inputs

Start Feature X

Approve, Merge X

Create a release pkg of select features

Determine if Feature X is part of release

RM, RA

RI, RC

Provide input

Start dev on Feature X

Create a release pkg of select features

Determine if Feature X is part of release

Finish dev on Feature X

Aligned to the general Anuket release cycle

2021 Release 1

2021 Release 2

2021 Release 3

NOTE: This release cycle doesn't have to be 3 releases – TBD (eg, could be 2 or 4)
NOTE: May need to release in different "packages", such as Spec vs Impl vs Conformance