

Release Process

Introduction

The purpose of this document is to describe the cadence, processes, milestones, and associated tasks used in the ONAP release cycle. See [Project Status](#) and [Release Planning](#) for information and schedule for specific releases.

Overview

ONAP releases typically happen twice per year at approximately 6 month intervals. Releases are named after major world cities in alphabetical order. See the [Long Term Roadmap](#) for a list of previous and future releases.

The current release process was developed by community volunteers and rolled out with the Honolulu release. Releases are administered and tracked by a release manager. Release status is discussed at the [weekly PTL meeting](#).

Requirements

Requirements are submitted to the Requirements Subcommittee for evaluation in the weeks leading up to the M2 milestone. Requirements include specifications, features, use cases, best practices, global requirements, and Proof of Concept (PoC). These are described on the [Requirement Types](#) page.

Requirements are documented as epics in the [REQ Jira project](#). These epics will then have release management tasks attached to them for each release milestone, which are the responsibility of the requirement owner to complete. In addition, for features, specifications, and use cases, the requirement owner will also attach one or more integration test tasks. The status of these test tasks is closely tracked from milestone M4 until the end of the release.

Time Based Releases

As of the Honolulu release, ONAP releases are time based. This means that requirements are evaluated at M4 for readiness for integration testing. Those that are ready proceed with the release. Those that are not ready are deferred to the following release.

Release Management Tasks

In order to track release status, release management tasks are assigned to ONAP projects and to requirement owners using Jira. Release management tasks for projects are assigned using the project team's own Jira project. Release management tasks for requirement owners are attached to the requirement epic in the [REQ project](#). Release status at each milestone is determined largely based on the status of the release management tasks.

Exception Process

If a release milestone, best practice, or general requirement cannot be met, then the project team or requirement owner may file an [exception request](#). Exception requests may be prepared using the exception request page contained within the project status area for each release. Exception requests are periodically reviewed by the TSC.

Release Milestones

 **Note:** Project tasks UPDATED 13 Jul 2023 to reflect the [task review performed by the PTLs in 1H23](#) and approved by the TSC.

As mentioned above, release milestones, and their associated tasks, are used to track the status of the release. The milestones and management tasks used in the current release process are described in the table below.

Milestone	Description	Project Tasks	Requirement Owner Tasks	Subcommittee Tasks
M0	Start of release cycle			

- [Introduction](#)
- [Overview](#)
 - [Requirements](#)
 - [Time Based Releases](#)
 - [Release Management Tasks](#)
 - [Exception Process](#)
- [Release Milestones](#)

M1	General requirements approved	<ul style="list-style-type: none"> Request an architectural subcommittee review Document API issues in the requirement description DOCS: create documentation tracking page and pre-fill information. 	<ul style="list-style-type: none"> Get approval and identify impact of Use Case, Feature, Spec and any new TSC Best Practices and Global Requirements Document API issues in the requirement description 	
M2	Specification freeze <ul style="list-style-type: none"> Freeze the scope Review approved Global Requirements Verify that specifications and features are approved by affected projects PTLs determine impact of features and specs on APIs 	<ul style="list-style-type: none"> Complete release planning template Review license scan issues Complete Architectural subcommittee review Data models shared with Modeling subcommittee Verify that there are no merge requests older than 36 hours Communicate API changes to other projects Color code Impact View Per Component page 	<ul style="list-style-type: none"> Finalize documentation of API changes and share with affected PTLs 	
M3	Final Code Submission	<ul style="list-style-type: none"> Verify that test coverage goals have been met Review license scan issues Verify information in documentation tracking page. Update as necessary. 	None	
M4	Feature freeze	<ul style="list-style-type: none"> Start OOM review with updated container image Assign Jira issues to the release Complete preliminary documentation Review and update INFO.yaml DOCS: confirm that PTL repo changes in M2 (new/removed repos) and M4 (preliminary doc) are represented in master and RTD 	<ul style="list-style-type: none"> Document integration test tasks in Jira Complete preliminary documentation Review and update scope status 	
RC	Release Candidate 0 Integration and Test	<ul style="list-style-type: none"> Create a release branch Complete key updates page Review license scan issues Finalize documentation DOCS: verify that repo branch exists, verify that RTD branch exists, verify that project release notes have been finalized 	<ul style="list-style-type: none"> Update integration test tasks in Jira Complete key updates page Create preliminary release notes 	DRAFT: Need to discuss with Arch Subcommittee <ul style="list-style-type: none"> Arch subcommittee <ul style="list-style-type: none"> Finalize architectural description Finalize architectural diagram

Sign Off	Approval for release by the TSC	<ul style="list-style-type: none"> • Complete project testing • JIRA Cleanup • DOCS: prepare composite release notes. 	<ul style="list-style-type: none"> • Complete E2E testing • JIRA cleanup • Finalize Use Case, Specification, and Feature documentation, including release notes 	
----------	---------------------------------	--	--	--