Accelerate ONAP Release Cadence

We would like to suggest 3 different approaches - these can be combined or performed independently from each other.

- Divide and Conquer
- Speed up on what we have
- The 'Facebook' way

1. Divide and Conquer

Approach

Split release types
E.g. : Core (Kernel – Orchestration/instantiation?) releases VS Apps (OAM, CL, …) release
E.g. : Main Releases VS Service releases (like ODL)

-Main Releases can introduce new projects, new extended features
-Service Releases can introduce smaller features, smaller extension, can be done much faster

Focus

Operating System Model (piece by piece)

Assumptions

-Robust test suites and coverage to certify release
-Agree on “Quality Gates” that can certify a release / a service release
-Split Features into several categories i.e. Core, Service releases, Security releases etc.
-Because the cycles are shorter, and downstream projects more isolated from upstream, there are fewer Milestones with fewer requirements
-Imply to maintain more branches but for a short period (~4 weeks), clever selection of long lived branches (Long Term Support ?)
-Clear Configuration Management Strategy

2. Speed up on what we need

Approach

ONAP is already built on a daily basis
-All teams have implemented a daily release build, get everything from Master and build Staging artifacts including potentially shippable release containers
-Add a ‘weekly’ stable build candidate – Keep main release target but allow for features to be released prior the ‘official’ release date (e.g. alpha, beta builds)
-Dashboard Check on Monday and try to fix the open issues by Thursday EOD to release as an “experimental” release on Friday.

Focus

Features Centric

Assumptions

-Robust test suites and coverage to certify release
-Agree on “Quality Gates” that can certify a release / a service release – Decide core test suites, and priority focus on Ready-to-Go features
-Can be combined with a shorter cycle, could even allow to move content around releases (if the ONAP TSC accepts that scope can move)

3. The "Facebook" way

Create an immutable Release pipeline, that continuously build and push releases provided that :
-No breakage is detected during various test cycle
-No ‘Emergency stop’ is triggered by Dev or Test or ONAP TSC or anyone to stop the release
-Allow for feature branches to introduce more disruptive changes (maybe limit them per product to limit overhead)

Focus

Defects, small improvements

Assumptions

-Robust test suites and coverage to certify release
-Pull everything from Master, always move forward (no patch, no service release) – ideally this model can deliver multiple times a week
-Real Feature thinking, work on a smaller set of features - no ‘Big Name’ release anymore
-Additional LF infrastructure resources are available
-“Intelligence Release Robot” to be developed