Proposal: Enhancement Delivery Management in the Kubernetes Ecosystem

Introduction

As Kubernetes matures in both age and contribution density, a critical paradigm shift towards sustainability is underway that must be accounted for in a variety of different areas of concern. This proposal is focused on the scoping, implementation, and management of functional changes into the Kubernetes ecosystem via the newly-adopted Enhancement Proposal (KEP) process. In the current implementation, these changes are broadly referred to as “features” and the group responsible for managing their implementation is the Product Management Special Interest Group. The focus of this proposal is to shift away from “features” and instead focus on delivery of value.

It is the assertion of the author that this method of traditional product and project management is no longer in alignment with the governance model, nor with the needs of the Kubernetes community as a whole. To address this, we should look toward other successful models of massive-scale value delivery, as well as align with the current state such that a minimally-disruptive transition may take place.

Goals and Objectives

If fully implemented, this proposal should create a scalable, sustainable, and enduring model for the delivery of KEP-defined value into the Kubernetes ecosystem.

Core objectives are as follows:

- A clear and auditable chain of custody must be present for all value delivered into the Kubernetes ecosystem.
- The value delivery management process must enable contributors and stakeholders to channel resources into the project in the most efficient way possible.
- Elimination of waste will be a fundamental goal of the process. This includes limiting duplication of effort, no project resources directed towards unfunded mandates, avoidance of manual efforts where automation is possible, and overall reduction of management overhead.

- Whenever possible, SIGs should be the primary actors in any process since they are closest to the subject matter, accountable to relevant code changes, and oversee local technical governance.
- Product management should be focused on guiding usability, and other feedback from the user community toward the appropriate special interest groups, as well as tracking satisfaction over time using a tool like the Net Promoter Score.

Plan of Action, Methodology, and Design

The Kubernetes Enhancement Proposal (KEP) process is currently tasked with modeling the introduction of change and resulting value into the ecosystem. Its functional boundary is where idea and action intersect. This plan of action will take output from the KEP process and oversee its implementation. It will also ensure alignment between custody of the value and corresponding state changes in the KEP process.

In order to accomplish this, there are both roles and planning activities necessary that do not formally exist in the current project governance model. Also, the existing product management SIG will shift to a more strategic role that will be described in greater detail later.

The following section will describe the new roles and how they fit into the overall methodology.

Planning Facilitation Role

While grounded in the Agile definition of what a product owner does, Open Source development requires a shift in how the responsibilities get carried out. Each SIG delivering work into a release would need either a dedicated facilitator or at least someone filling the role for the duration of the release. The facilitator would represent the SIG at product management meetings, carrying status back and forth. The product management meeting provides the optimal moment for discussing blocked dependencies, slipped timelines, or adjustments in what was committed to. The facilitator is the primary representative from the SIG during planning events.

During release burndown, the facilitator coordinates the transition of the value described in the KEP to the community. This means aligning with the release team for issue management, coordinating the insertion of release notes into the draft, and acting as a conduit to the SIG for things like broken tests. Ideally, the facilitator would be working with the SIG to codify and prioritize technical debt, especially during “stabilization” releases. The facilitation role does not have to be highly technical, but may benefit from having deep experience in the SIG specialty. The role is not a “lead”, or imbued with extra privilege. On the contrary, it is a servant leadership role performed on behalf of the SIG for the entire community.

Release Planning Event

The purpose of the release planning event is to line up committed, funded work, and also identify areas where cross-SIG dependencies exist. Discussions should take place to untangle dependencies, work though the release schedule to ensure checkpoints exist across dependent SIGs, and if necessary, drop work that has unmet dependencies. A SIG should make their best effort to commit to work that has a reasonable chance of being completed within the release timeline.

At the end of the planning event, each SIG should have a prioritized, scheduled backlog of work identified via individual GitHub issues that tie back to the KEP proposal. Each issue in the milestone should describe the work necessary to complete the release commitments, as well as documentation necessary to create release notes. The release notes initial draft would be an artifact coming out of the event.

Event Mechanics

Planning events would occur during the week of code freeze, since facilitators should be focusing on the subsequent release cycle. This would also provide an opportunity for any last-minute negotiations required to ship features for the in-progress release.

Only facilitators representing impending features or dependent work need to attend, although it may be advisable to have SIG technical leads present if negotiations are required to unwind dependencies.

The meeting would ideally be in person, but that is aspirational. A virtual meeting is possible, and would require facilitators to have a list of KEPs and a breakdown of deliverable work for the release cycle (the work items below are essentially a bullet point list contained in the issue). The breakdown would be in terms of committed deliverables like so:

KEP ⇨	ISSUES ⇨	WORK ITEMS ⇨	PRs ⇨	MERGE

This would then form a rough schedule of deliverables over the course of the release in 2 week increments with an 8-week coding period. The committed increments would be actionable work in the issue milestone, and the percentage confidence in completing it would be at most diminished as in the following table. In high-dependency situations, those percentages may be lower, but never higher. This is to account for the inherent unknowns in work that is not fully described. It also gives facilitators and product management an idea about how likely the work is to ship.

Release Sched.	WEEKS 1-2	WEEKS 3-4	WEEKS 5-6	WEEKS 7-8
Issues	Issue item 1	Issue item 1 & 2	Issue item 2	Issue item 3
Commit Confidence	100%	80%	60%	40%

Any work being merged into master during the release should be trackable via the parent issue, and in turn the original KEP. This will provide a “steel thread” for all committed work.

The Readout

As part of the event, each facilitator would do a verbal readout of the work they plan to deliver. This would include an explanation of the parent KEP, the individual issues that are slated for completion in the release, any dependencies, risks, and potential delays. Other facilitators would have an opportunity to ask questions or block depending on their ability to satisfy dependencies.

Other program-level attendees such as architecture, testing, and product management can also ask clarifying questions, and make sure that the planned work is in alignment with the long-term direction of the project. For example, if SIG-foo is planning on delivering new functionality into the cloud provider code, and in-tree providers are deprecated, the attendees should ask hard questions about why that should be going into the release at all. Ideally, this would have been satisfied in the KEP process, but sometimes these conflicts only emerge when work is at an implementation level. There may also be other dependencies that are not apparent, like deprecations, code breakout, and so forth. The intent is that once a readout is done, the work is ready to be committed to and delivered.

Once the readout is done, it is time for the confidence vote on the overall plan. The plan, for this purpose is a series of issues, organized by SIG, and labeled with the milestone of the upcoming release. A fist-to-five vote will be held, and if everyone is a 3 or above, the plan is approved. If anyone raises a 2 or less, those objections must either be satisfied so the person is a 3 or better, or the plan reworked to satisfy the concerns, and then re-voted.

Once the readout and vote are complete, facilitators would add the milestone-approved label to their issues, and remove any unapproved work from the milestone. The descriptions of the approved work would also go into the release notes draft, so it collectively represents the entirety of planned work for the release.

Unplanned work can, and will come up. These will be negotiated between product management and affected SIGs. **Anything outside the planning event will require an exception request**.

Implementation Challenges and Opportunities

The largest challenge in this process is identifying people willing to act as facilitators since it is a new role. This is also an opportunity to widen the scope of skills that can be added to the project. Ideally, we would reach out to parent companies in search of people willing to serve in this capacity. We would also solicit interest from SIGs.

Logistics would be another consideration, since planning events would be quarterly. Part of accepting the facilitator role for a SIG would be committing to a 4-hour block of time during the week of code freeze in the prior release. This would need to be scheduled far in advance.

Ideally the release lead and secondary would be identified prior to the planning event so they could attend. Product management would also be encouraged to participate. The role formerly known as the “features lead” would be the PM representative for the release, and would be a mandatory attendee.

SIG Facilitation pilot as a top-level concern

While “product management” facilitation is a core component of this proposal, it bears mentioning that other facilitation roles will become necessary over time in SIGs to help them remain productive and focused on delivering quality code instead of project administrivia. For example, roles like “note taker”, “retrospective facilitator”, “release testing point of contact”, “release document creator”, and so forth represent important activities that do not necessarily have defined roles at the SIG level.

As these helper roles become reliably staffed, SIG code contributors should be able to better focus on maximizing their input into the project. It also avoids the known antipatterns around context switching when you’re trying to solve a difficult problem.

This addendum is important because this proposal represents a pilot process for SIG facilitation, and it will be important to gather input on how to optimize the relationships, staff the role, document the expectations, define skills requirements, and retrospect overall outcomes. All of this will be critical for Kubernetes to scale and succeed over time.

TBD: Agile Project Management Tooling

- Multi-SIG work
- alpha/beta/stable workflow in KEP? Tooling?
- Would CNCF need to pay for a tool?

- Trello is a good candidate
- How do we make Kubernetes Org-wide project boards?

TBD: Asynch demos + publishing

- Mechanics for this? YouTube?
- What is the guidance around doing demos of “hard-to-demo” work? A: show a passing test as the worst case

TBD: Release tooling refresh & gaps between

-

TBD: Highlight the benefits

- vendor participation
- vendors can understand the commitments around their investments
- visibility overall of the work ~ you can look at anything in the repository and understand what top-level value it’s aligned with
- it helps the project as a whole scale since it helps eliminate another centralized process bottleneck
- it puts the responsibilities of delivery closer to the people doing the work, so much less is lost in translation
- this provides a VCS-centric way of differentiating planned work from unplanned work
- it opens the doors for SIGs to experiment with incremental delivery and other Agile/XP/lean processes at the team level
- ~~provides an example of how we can embed “facilitation” roles into SIGs to help them achieve their goals, favoring coding over administrivia~~

- SIG planning is asynchronous ~ detangling dependencies is synchronous
- What is the value, and how does it benefit?
    - Facilitating planning sessions wherever / whenever
    -
- “What is the motivation for introduction of change into the project?”
- Go experience reports
- How to make the aggregate release issues human readable?

Implementation:

- Issues for 1.10 (tie back to proposal)
- At KubeCon ~

- ~~Planning with SIG-Cluster-Lifecycle~~