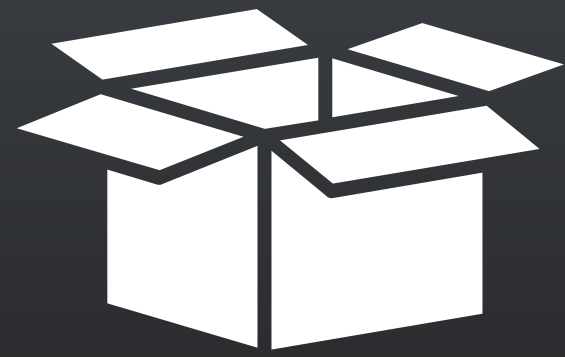# Committing to our Users

Product and Release Management in Apache Cassandra

Scott Andreas – NGCC 2019

Why ?

NGCC | Outline

Product + Release
Management

Evolving Cassandra:
Enhancement Process

User Community +
Communication

NGCC | Outline

Product + Release Management

Evolving Cassandra: Enhancement Process

User Community + Communication

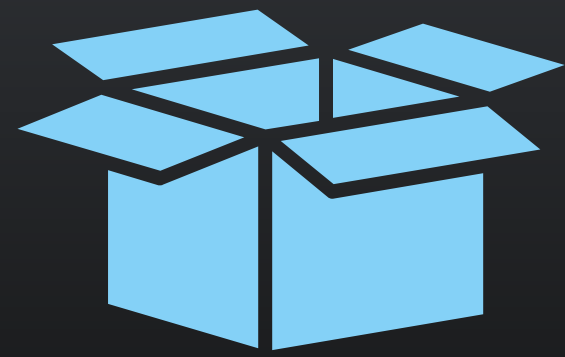# NGCC | Outline

Product + Release Management

Evolving Cassandra: Enhancement Process

User Community + Communication

NGCC | Outline

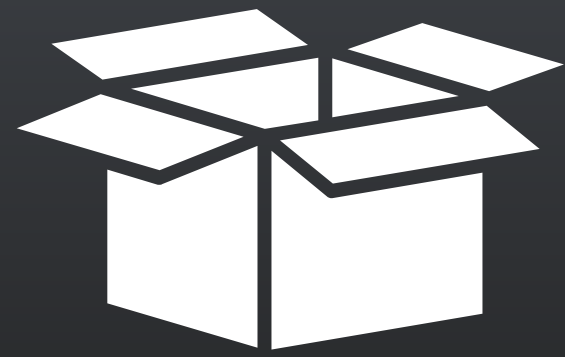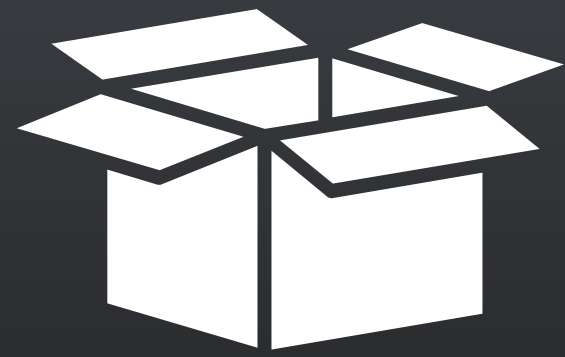Product + Release
Management

Evolving Cassandra:
Enhancement Process

User Community +
Communication

NGCC | Outline

Product + Release Management

Evolving Cassandra: Enhancement Process

User Community + Communication

NGCC | Product and Release Management

Why's this matter?

– Volunteers contributing to an open source project.

  – Most of us aren't working exclusively on Apache Cassandra.

– Users building some of the biggest systems in the world.

– Succeeding here is critical to growing the Cassandra ecosystem.

  – Ability for other projects to take dependencies on us (e.g., Zipkin).

NGCC | Product Management

- Release scope, timelines, and milestones

- (More) Predictable delivery

- Measurably high quality

- User communication

NGCC | Scope and Timeline

– Release Scope

  – What's in and what's out for major and minor releases.

  – Proposed by contributors, agreed to on dev list with exception process managed by PMC, and published on the website.

– Release Timeline and Milestones

  – Timeline and forecast for these milestones.

  – Done isn't a date – but working to hit "done" on published dates.

  – **Definitions:** Alpha, Beta, Release Candidate, and GA.

NGCC | Scop[e]

- Release Scope[...]
  - What's in and [...]
  - Proposed by c[...]
    process mana[...]

- Release Timeli[ne]
  - Timeline and fo[...]
  - Done isn't a d[...]
  - Definitions: Al[...]

C* Release Lifecycle

File   Edit   View   Insert   Format   Tools   Add-ons   Help

Apache Cassandra Release Lifecycle

The purpose of this document is to define the product stages as part of C* release life cycle and the expectations and commitments which exist at each stage. Scope of this document is limited to the lifecycle and stability of the product at large and not about any individual feature. The following stages have been identified and are described in detail below: Snapshot, Alpha, Beta, Release Candidate (RC), General Availability (GA), Maintenance, Deprecated, EOL.

Snapshot
- A snapshot artifact is published for every commit to trunk.
- A retention window is associated with the snapshot artifacts.
- Should not be used without collaboration/consultation with the developer community.
- **Will** be discontinued / rewritten even if the idea validated.
- Expectation that components of the core machinery may change, resulting in required code changes from the early adopters. These breaking changes should be well-communicated.
- As commits are made into trunk, contributors are expected to have completed extensive testing/validation to ensure that no critical or severe bugs exist (e.g., data loss, consistency violations, incorrect responses to queries, etc)

Alpha
- An alpha release is made to have early adopters try it out and provide feedback in terms of the following
  - Compatibility with tooling and automation
  - Compatibility with client libraries
  - Validation of newly added features
  - Smoke Test of existing features for catching any regression
  - Performance evaluation
  - Stability at different levels of workload (throughput)
- From a customer's perspective, the system is mostly "feature complete" [Feature List is determined by polling the community to begin with, followed by PMC deciding on the final list of features for the release and documented in the confluence page]
- Usability is unpolished, and potentially changing on a frequent and regular basis.
- Not ready for any customer-impacting production usage.
- Should not be used without collaboration/consultation with the developer community.
- A new major version is associated with the release, as it progresses from Alpha through EOL.
- A new branch is created for this release with the new major version, limiting any new

---

https://is.gd/VqzZph

NGCC | Quality and Communication

– Measurably high quality

- Test and qualification plans

- Release quality metrics

– User Communication

- Announcements, release notes, blog posts, and public presentations.

- Regular updates on release progress and milestones.

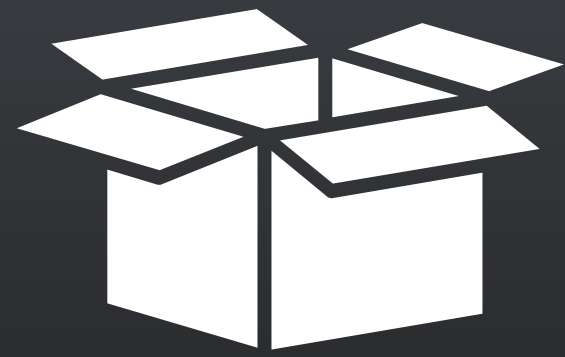- All indexed and available via the website (not just email).

NGCC | Release Management

- Managing open items against scope and timeline.

- Ensuring convergence of open items against that target.

- Raising issues and flagging scope questions/ concerns to the dev list and PMC.

- Communication of release status to our user community.

NGCC | Outline

Product + Release
Management

Evolving Cassandra:
Enhancement Process

User Community +
Communication

NGCC | **Evolving Cassandra:** Enhancement Process

- **Motivation:** Lightweight structure for our work, and making it clear to the community.

- **Precedent** in other projects: KIP / SIP / KEP, swift-evolution, etc.

- **Durable record** of project artifacts for major contributions: consensus, decisonmaking, architecture, and development.

NGCC | **Evolving Cassandra:** Enhancement Process

- Goals:

  - Transparency in major changes planned, and a forum for discussion.

  - PMC-chartered working groups for major changes. Regular status updates and public comment periods.

  - Empower working groups to design, lead, and execute projects.

  - Reduce asymmetry of effort on behalf of proposer and commenters through participation and stake.
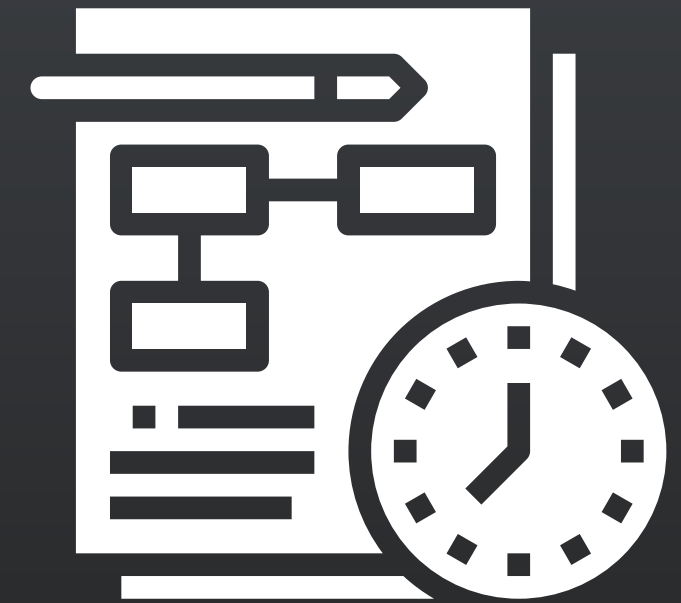
NGCC | **Evolving Cassandra:** Enhancement Process

- Non-Goals:

  - Overly-broad application to every type of change.

  - Heavyweight design process that requires all aspects of a project be designed up front.

  - Mountains of paperwork before anything can get done.

  - Establishing fiefdoms (communication and opportunities to participate are open).
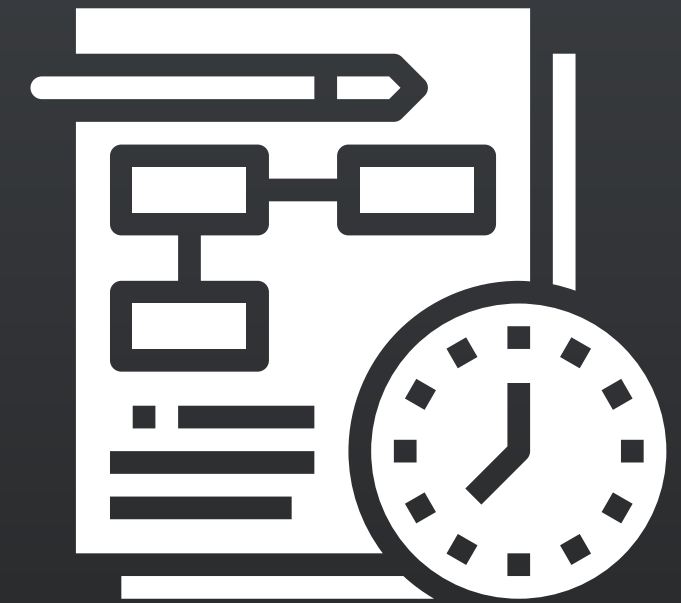
NGCC | Evolving Cassandra

- **Potential Approach:** Similar to JEP

  – Lightweight doc: scope, goals, approach, timeline, mailing list / Slack channels, umbrella JIRA ticket.

  – **Example:** JEP-169 Value Objects (Valhalla)

  – Once "accepted," working group is established, chartered by PMC, and project + communication artifacts prepared (JIRA, list, channels, etc).

  – **Participation is open**, with periods of public comment. But **contribution requires commitment**.

JEP 169: Value Objects

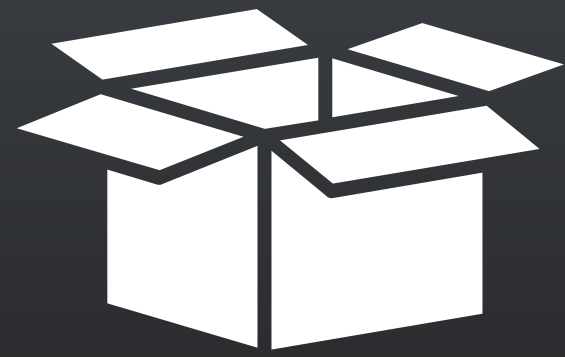|	
Owner	John Rose
Type	Feature
Scope	SE
Status	Draft
Component	hotspot / compiler
Discussion	mlvm dash dev at openjdk dot java dot net
Effort	L
Duration	L
Created	2012/10/22 20:00
Updated	2015/05/29 13:51
Issue	8046159

Summary

Provide JVM infrastructure for working with immutable and reference-free objects, in support of efficient by-value computation with non-primitive types.

Goals

- Support user-defined and library-defined abstract data types with performance profiles similar to Java primitive types.
- Bring the semantics of `int` and `java.lang.Integer` closer together.
- Allow representation of ubiquitous types currently not well supported on the JVM, including complex numbers, vector values, and tuples.
- Increase shareability of Java data structures.
- Provide clear and explicit semantics for shared read-only array data.
- Enable functional-style computation with pure data, for optimized parallel computations.
- Increase safety and security and decrease "defensive copying" in applications which must share structured data across trust boundaries.
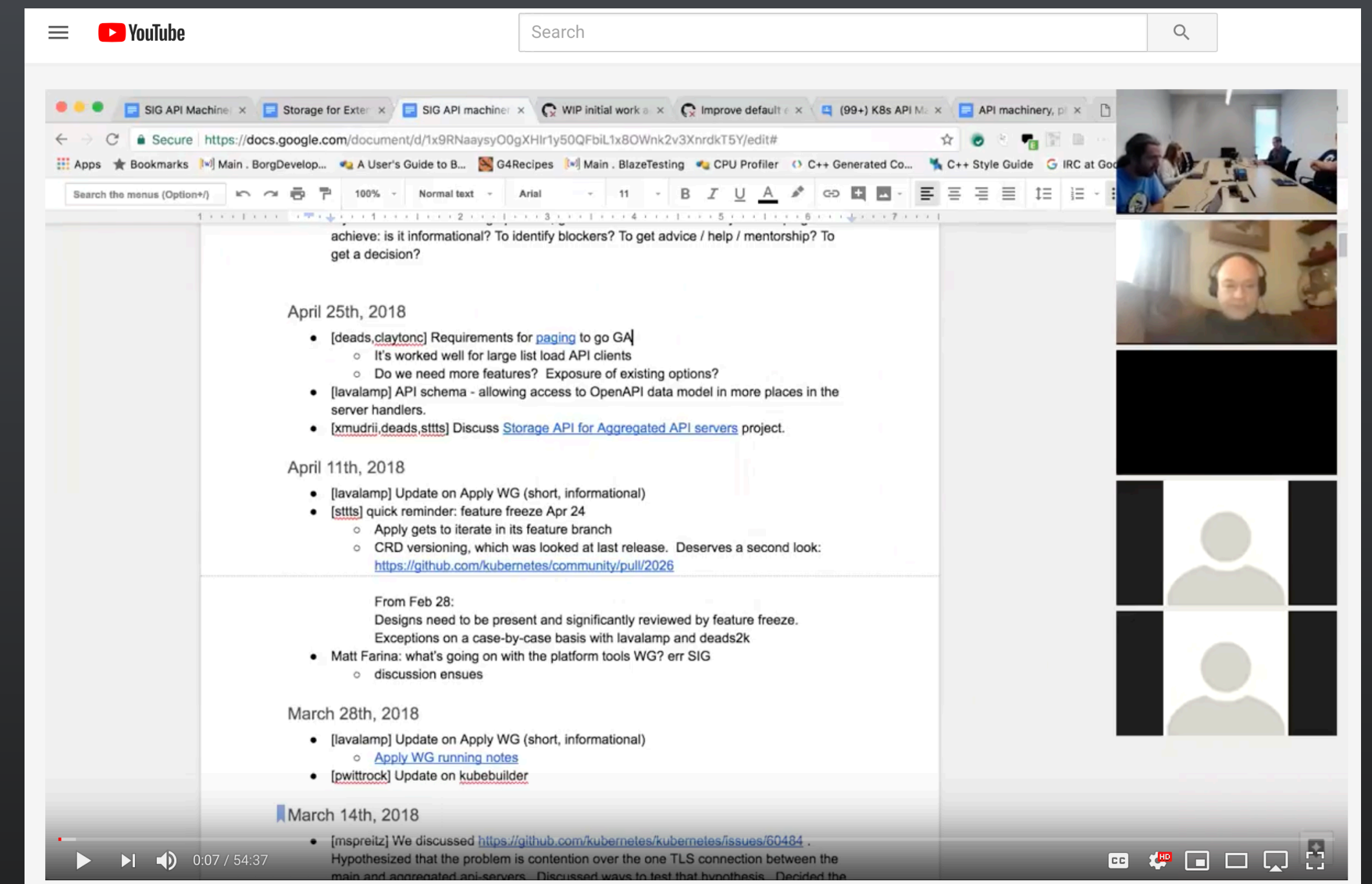
NGCC | User Community + Communication

- **Public video calls**

  - Moderated, published agenda, recorded, and hosts rotated.

  - Emcee owns timekeeping and execution of meeting agenda.

  - Low-fi: light on prep and production value.

  - Could be applied either among contributors or Cassandra users.

NGCC | User Community + Communication

- **SIGs as "Working Groups"**

  - Scalability and empowerment through chartered roles: release management, subsystems, docs, testing, and community management.

  - "Release Management" and "Documentation" as potential groups.

    - Precedent: GSoD contributions

  - Clear path to contribution for new contributors interested in making non-code contributions.
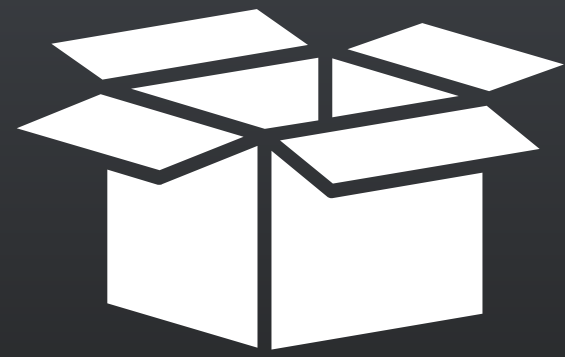
NGCC | User Community + Communication

- **Marketing**

  - Apache Cassandra blog

    - Areas: Release status, features, success stories, community highlights, others?

  - Twitter account: promotion of this content to our audience.

  - Public speaking engagements (and re-sharing of this content on our website).

  - Giving our website and docs some love.

NGCC | Outline

Product + Release Management

Evolving Cassandra: Enhancement Process

User Community + Communication

NGCC | Recap

- **Big Idea:** Exercising underrepresented functions in the project beyond engineering.

- **Approach:** Expanding the role of the PMC to charter working groups to lead particular areas: large projects, release management, and user communication / marketing.

- **Creating Opportunities** for contributors whose primary role isn't writing code.

- **Examples:**

  - **Project Management:** Formalizing release management (scope, status, and quality)

  - **Development:** Lightweight JEP-like design docs and working groups.

  - **Communication:** Dev / User Video Calls, Documentation, and Marketing

Committing to our Users

Product and Release Management in Apache Cassandra

Scott Andreas – NGCC 2019