Release Exit Criteria

Goals

To produce releases that:

- solve real use cases
- can be easily consumed by new and old users alike
- the developers feel confident that their code does what it says it does / is supposed to do
- can confidently be used in production scenarios

Exit Criteria

Exit criteria for a major or minor release shall be determined by consensus of the maintainers that there is:

- all of the agreed supported features are complete
- features should comprise in total a meaningful set of functionality
- adequate test coverage (unit and functional/integration) to have confidence that there aren’t likely to be any gotchas lurking in the untested code,
- maintainers have reviewed all remaining open bugs and agree on severity/priority,
- zero high or highest priority bugs remain open,
- the preceding release candidate will have been published for a minimum of two weeks
- no bugs that affect security of the system remain open unless they have mitigating workarounds published in release notes,
- that there have been security scans (static and dynamic), code audit and penetration testing with all significant findings reported in JIRA (major releases only),
- code included in the release has been scanned for license infringements and no infringements have been found,
- crypto code included in the release has been audited for crypto export compliance,
- release will not require users/operators to compile code to utilize based Fabric (Note this does not include their own code),
- release has met all the criteria for CII Best Practices Badge
- documentation sufficient to ensure that users/operators have clear guidance on how to get started and how to configure and operate. The operational aspects need to be correct and independently user tested,
- any features that we choose not to support, because there is risk or they are incomplete, need to have a build tag or runtime feature flag to disable said feature and clear guidance as to the risks attendant to enabling it,
- signed release code (not sha1, which is what git currently uses as that is no longer secure), published images (DTR) and binaries,
- future goal - automated release notes similar to those published by mature open source projects such as Kubernetes, or Cloud Foundry

Review Process & Criteria for v1.0

As discussed here the Fabric maintainers will observe the following policy when reviewing CRs while in a release cycle following a code-freeze.

There are three broad categories of change that will be considered to be merged:

1. Changes that do not affect production code. This includes documentation, new or improved tests, bug fixes to non production code
2. Bug fixes that affect production code, but which have minimally complex changes
3. Changes that affect production code but which have a more complex resolution

Changes that belong to the first two categories will be reviewed and merged provided they meet the criteria:

- the CR identifies the associated JIRA item in its title
- the JIRA item is complete in its description of the problem and the resolution, and is appropriately labeled, categorized and identifies the current release in Affects Version/s and the pending release in Fix Version/s
- the CR has test coverage for the defect being resolved (as appropriate) and the documentation is updated accordingly as needed.

Please note that changes that belong to the third category will be subject to extra scrutiny (by the maintainers).