

Auto release mechanism

Implementation:

- Tooling to check that every PR has an entry in `.changelog/*.md` and there is a commit with the trailers.
-

Design:

What needs to change in PRs

To be able to release automatically a new version, different information needs to be present at the moment of releasing that it is now manually introduced by the maintainer/core-team member that runs the PR:

Generating changelog information

PR author will need to write the changelog information

- ✓ Release manager don't need to do it anymore. Changelog information is more complete.
- ↓ Friction for small changes or external contributors

Problems using a single file as `Changelog.md`. Parallel PRs will conflict all the time on merges:

- Do nothing. Author resolves merges.
 - ↓ Level of friction increases
 - ✓ No tooling needed
- 3W new changelog entries per file per PR (i.e: `Changelog_PR####`)

The problem of who creates those files. Can not be created at push time since the PR number is not known:

 - Users deal with the problem anyway
 - ↓ Level of friction increases
 - ✓ No tooling needed
 - Create the PR file automatically on PR creation
 - ↓ Need implementation with write access
 - ↓ Dealing with external forks ?
 - ✓ No tooling needed

Where to store it?

- PR Descriptions like react
- Git commits using git
-

How does this work with backports / forward ports?

Be careful with PR Git stash?

Steven suggested using git commit metadata to handle version bumps and host changelogs entries in the repository.

-

Bump version in CMakeLists.txt and package.xml

✓ Release manager don't need to do it anymore

1. PR author is the one that deals with info about bumps
 - a. ↓ Level of friction increases
 - b. Might use some info in github description template
 - i. ↓ Difficult to parse at the release moment
 - c. Might use git trailers
 - i. ↓ Might be difficult for external users
 2. Lib maintainer is the one that deals with info about bumps
 - a. Github labels that indicate the minor / patch
 - i. ↓ Difficult to parse at the release moment
- Also change package.xml
 - ↓ Tooling to bump version to change this
 - ✓ Release manager don't need to do it anymore

When to trigger the release




Defined a fixed period of time to run releases: weekly, bi-weekly, monthly

- ↓ Changes merged close to release-time receive fw time to be tested in from source builds, CI or advanced users.
- ✓ Changes are no longer waiting forever to be released


CI tools to change

release.py execution:





No PR to bump changelog and version anymore

-  Release manager don't need to do it anymore
-  No PRs for releasing anymore
-  Require tooling / coding to integrate the changelog generation and the direct commit operation

Automatic releasing

 Release manager role disappear, releases are doing frequently in an expected cadence

Could be based on nightly:

-  Tooling to schedule of the releases, calling the new versions and deal with no-changing repositories
-  Credentials needed to write commits and / or create PRs
- Dealing with -release repositories
 -  Tooling needed to integrate the current call to spawn_changelogs
- ros_vendor packages
 - Code currently opens a PR in the ros_vendor repositories
 -  More automation to deal with automatic blooming