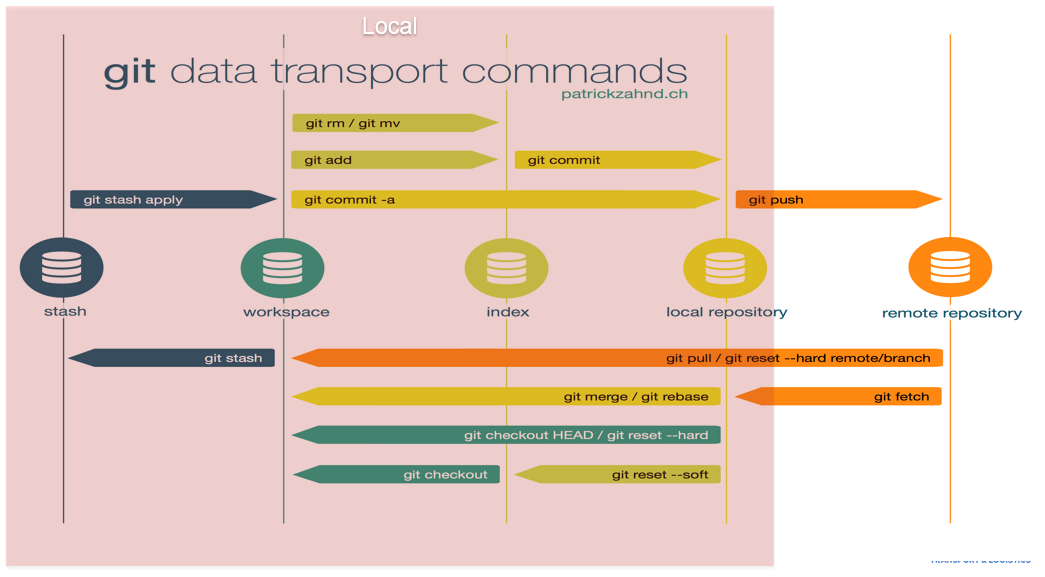
**Working with Git Branches**

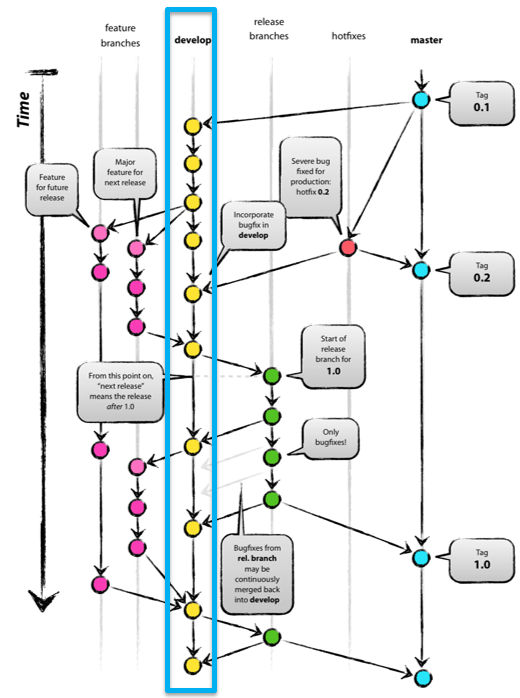


**Branches Naming Convention**

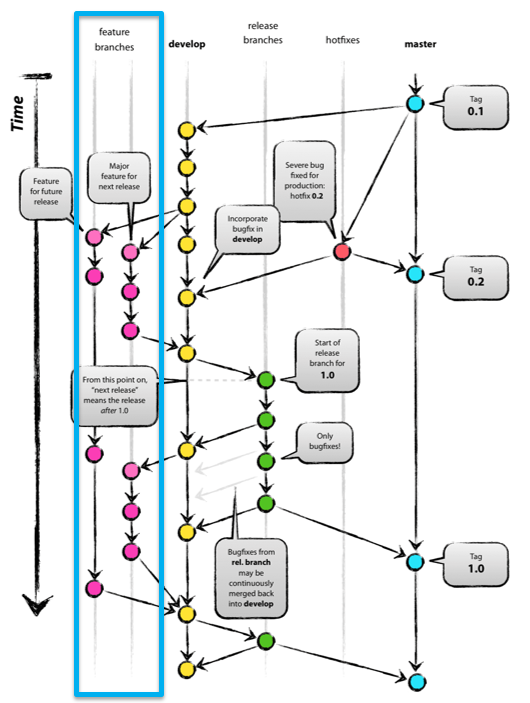
* develop (remote: origin/develop)
  + This is the integration branch – the branch containing all «DONE» user stories
* feature/\* (remote: origin/feature/\*)
  + This is a branch that contains all commits belonging to ONE user story
* release/\* (remote: origin/release/\*)
  + This is a branch created from develop when preparing ONE new release
* master (remote: origin/master)
  + This branch contains the version currently being in production
* hotfix/\* (remote: origin/hotfix/\*)
  + This branch contains hotfixes for production releases

**Branches In Details**

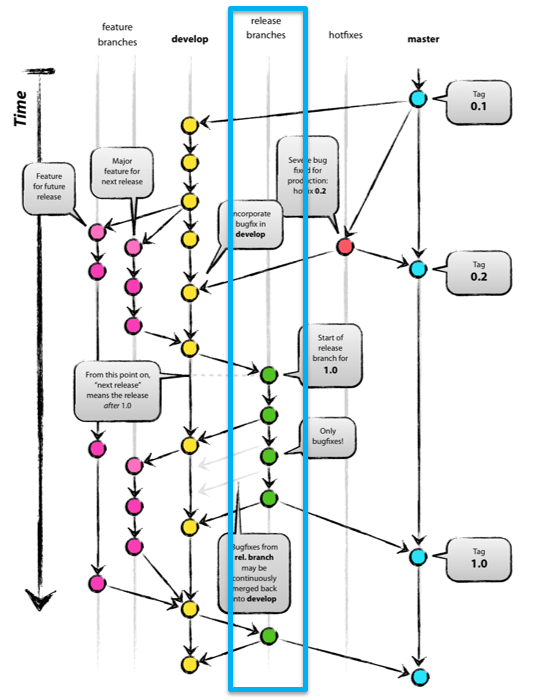
* develop



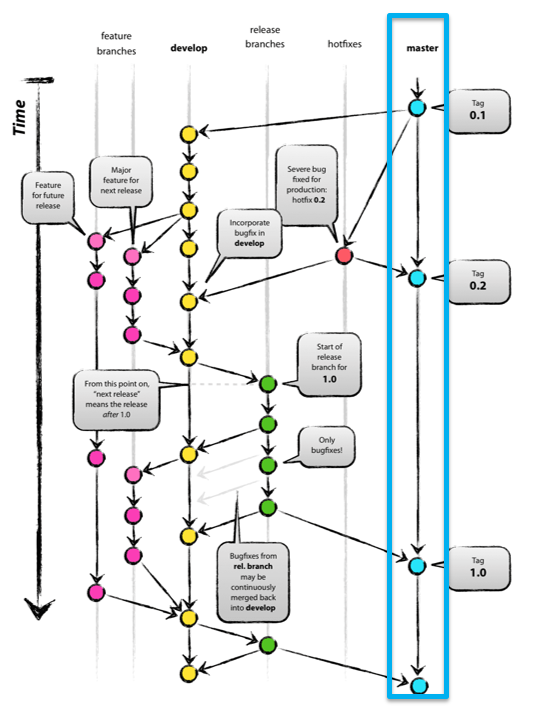
* + This is the integration branch – the branch containing all DONE user stories
  + All changes to this branch are introduced using «Pull Requests» – no direct commits
  + Typically we have a nightly build on our CI for this branch
  + In larger projects we should restrict users that are allowed to introduce changes into this branch (e.g. Technical Lead)
  + This branch is permanent
* feature/\*



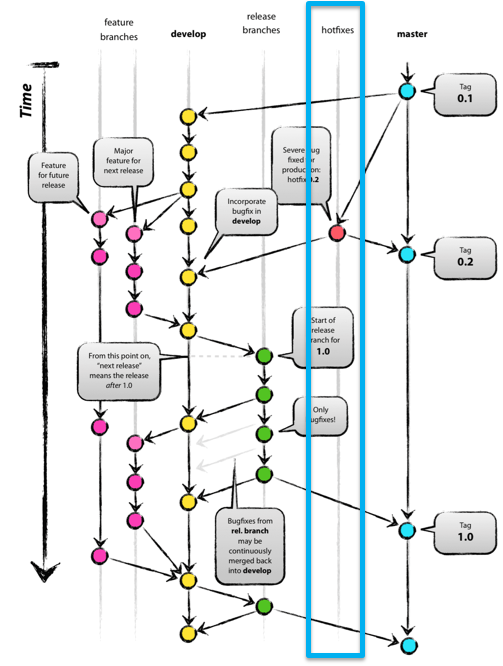
* + This is a branch that contains all commits belonging to ONE user story
  + Feature branches are merged into develop using «Pull Requests»
  + Typically we have a «build-on-push» CI job running at least the unit tests against this branch
  + All developers are allowed to create feature branches and push them to the remote
  + This branch is temporary
* release/\*



* + This is a branch created from develop when preparing ONE new release
  + This branch is used to prepare the next release and shall introduce only minor changes like documentation updates or fixes
  + The release branch is merged into master and develop after the release has been delivered
  + Typically we have a build-on-push and a nightly build on our CI for this branch
  + Usually all developers are allowed to push to this branch
  + This branch is temporary
* master



* + This branch contains the version currently being in production
  + This branch is used for tagging releases
  + All changes to this branch are introduced by merges from release and hotfixes branches
  + If we need to provide a hotfix our changes target this branch
  + Only the Release Manager is allowed to merge a release/\* branch into the master
  + This branch is permanent
* hotfix/\*



* + This branch is created from master
  + This branch contains hotfixes
  + This branch is merged back into master and develop (and potentially release/\*)
  + Typically all developers are allowed to push to hotfix branches however as the project version needs to be updated, the Release Manager has to be involved
  + This branch is temporary

**GIT workflow**

**Working with a feature branch**

1. Create a branch from develop, make and give it a name, such as feature/featureid-short-description
2. Code, commit your changes locally. Be able to have multiple commits for a bug fix or feature
3. Push your branch to the remote repository
4. Create a pull request so other people can review your changes. To incorporate feedback, you might need to make more commits and push more changes
5. Complete your pull request and resolve any merge conflicts from changes other people made after you created your branch
6. The feature branch shall be deleted after it has been merged

**Creating a release**

1. Leader creates a release branch from develop
2. Code and commit
3. Push your changes
4. Leader creates a pull request to merge it into master
5. Leader tags the release on master branch
6. Leader merges the release branch into develop
7. Leader deletes the release branch

**Creating a hotfix**

1. Leader creates a hotfix branch from master
2. Make the hotfix and commit the changes
3. Create a pull request
4. Leader tags the new release
5. Leader deletes the hotfix branch