(Software)
Revision Systems

What is it

- Keeps history of code changes
- Necessary for team based code development
- Very useful even if you code alone - rollback
- Allows branching - alternative versions
- Resolves conflicting code changes
- Development cycle: check out, change code, update, resolve problems, commit

Popular Systems

- CVS
- Subversion
- Git
- Mercurial

What to store

- source code
- configuration files
- Makefiles
- No compiled files (binaries)
- No (large) data files
- No external libraries

How does it work

- meta information stored in hidden directory .git/ or .svn/
- stores previous version of files
- also used for exclusion and repository location

Centralized

- CVS and SVN are centralized systems
- administration and repository in one place
- common code base and branches for all users

Distributed

- local user code base
- local branches
- can be synced with a common remote code base
- users can exchange versions without a central repository

Example Workflow SVN

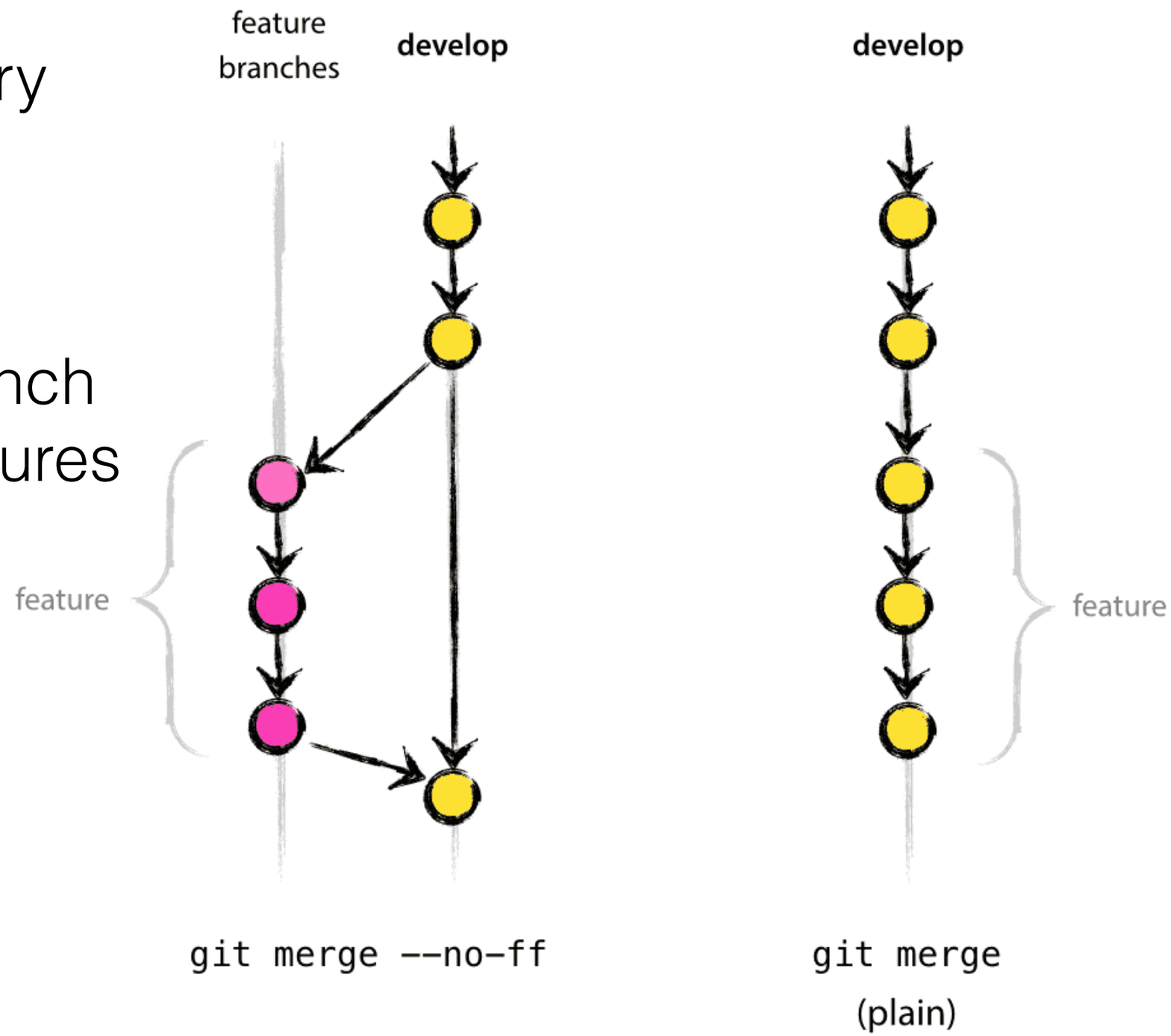
- svn checkout
- add or modify files
- update to latest version
- commit
- merge and fix collisions

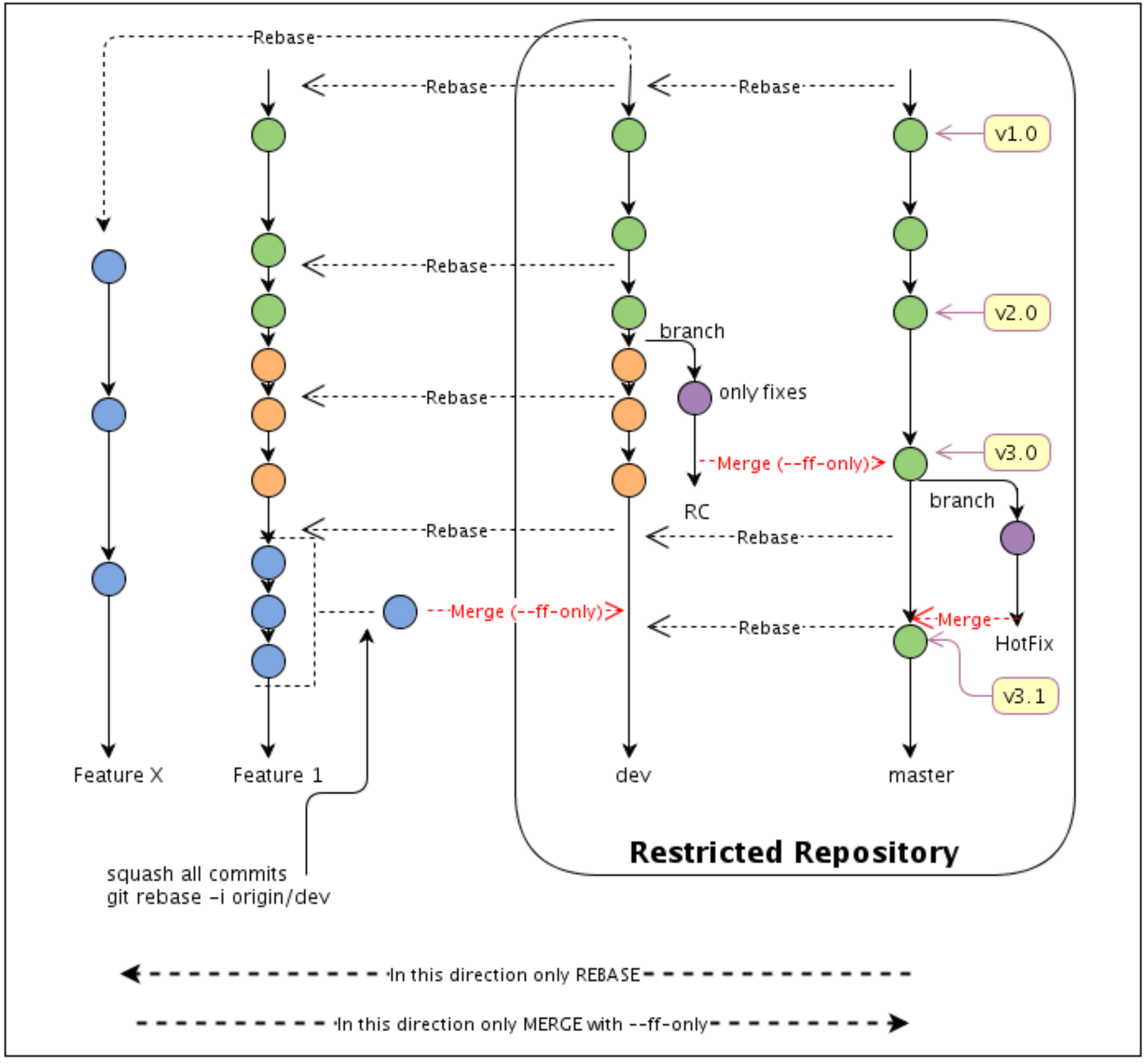
git

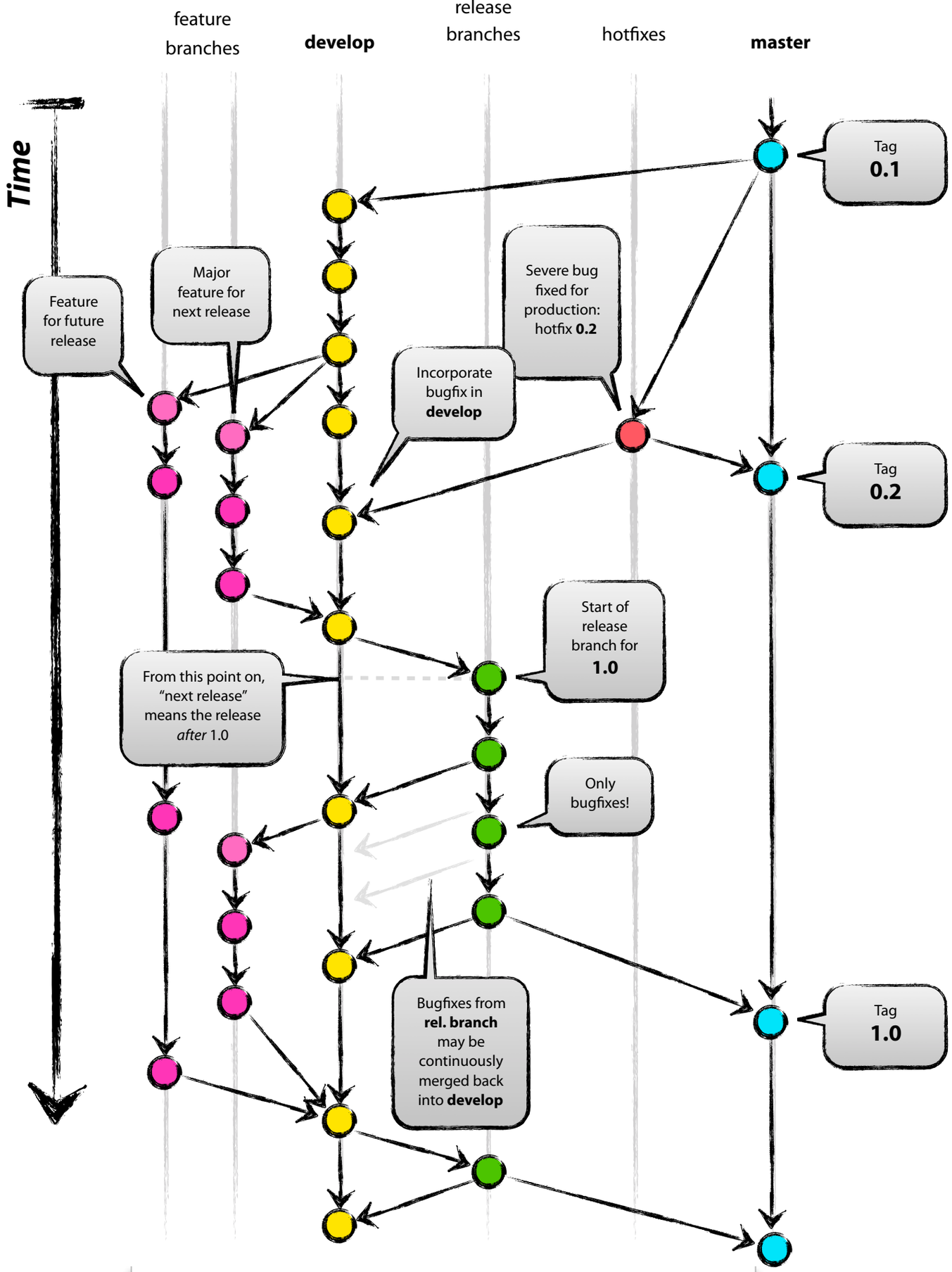
- developed by Linux Torvalds for Linux kernel development
- highly distributed development
- each commit has a unique hash value as identifier
- master is the latest version (pointer to last commit)
- origin/master in case of remote repository
- branching is encouraged

Example Workflow git

- git clone repository
- add mofify files
- create (local) branch for additional features
- git pull
- git commit
- merge/rebase




Rebase
Rebase
Rebase
Rebase
Rebase
Rebase
Rebase
Rebase
Rebase
v1.0
v2.0
v3.0
branch
only fixes
Merge (--ff-only)
RC
branch
HotFix
Merge
Merge (--ff-only)
v3.1
Feature X
Feature 1
dev
master
Restricted Repository
squash all commits
git rebase -i origin/dev
In this direction only REBASE
In this direction only MERGE with --ff-only



Author: Vincent Driessen
Original blog post: http://nvie.com/posts/a-succesful-git-branching-model
License: Creative Commons BY-SA

https://try.github.io