

# Bonnes pratiques et modalités de développements

# GIT Workflow on GITLAB inria

<https://gitlab.inria.fr/croco-ocean>

The screenshot displays the GitLab web interface for the **croco-ocean** project. The top navigation bar includes the GitLab logo, a search bar, and various utility icons. The left sidebar contains navigation links for Projects, Groups, Your work, Explore, CI/CD, Settings, and more. The main content area shows the project's name, a green banner with a message, and statistics such as 467.6 MB Project Storage and 7 Releases. Below this, there are buttons for finding files, using the Web IDE, and cloning the repository. The bottom section features a table with columns for Name, Last commit, and Last update, showing a commit for the AGRIF project.

**Switch to**

- Projects
- Groups
- Your work
- Explore

**Frequently visited**

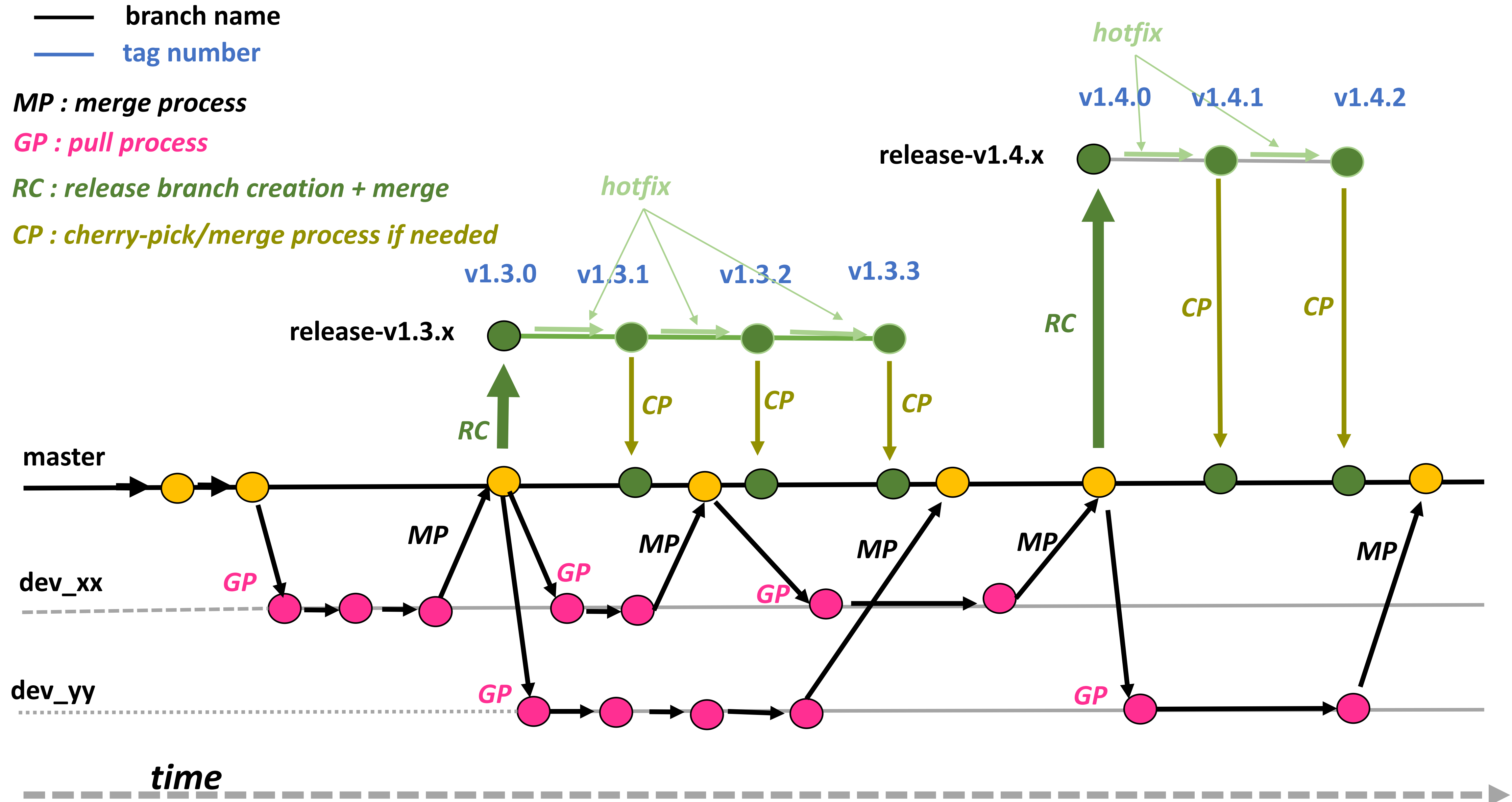
- croco (croco-ocean)
- croco\_tools (croco-ocean)
- croco (croco-ocean / ... / gcambon)
- croco\_pytools (croco-ocean)
- my-tools (croco-ocean / ... / gcambon)

**Project Details:**

- 467.6 MB Project Storage
- 7 Releases
- Star 8
- Fork 9
- Commit: 1aaf4589

Name	Last commit	Last update
AGRIF	AGRIF: add compilation option for conv to av...	1 year ago

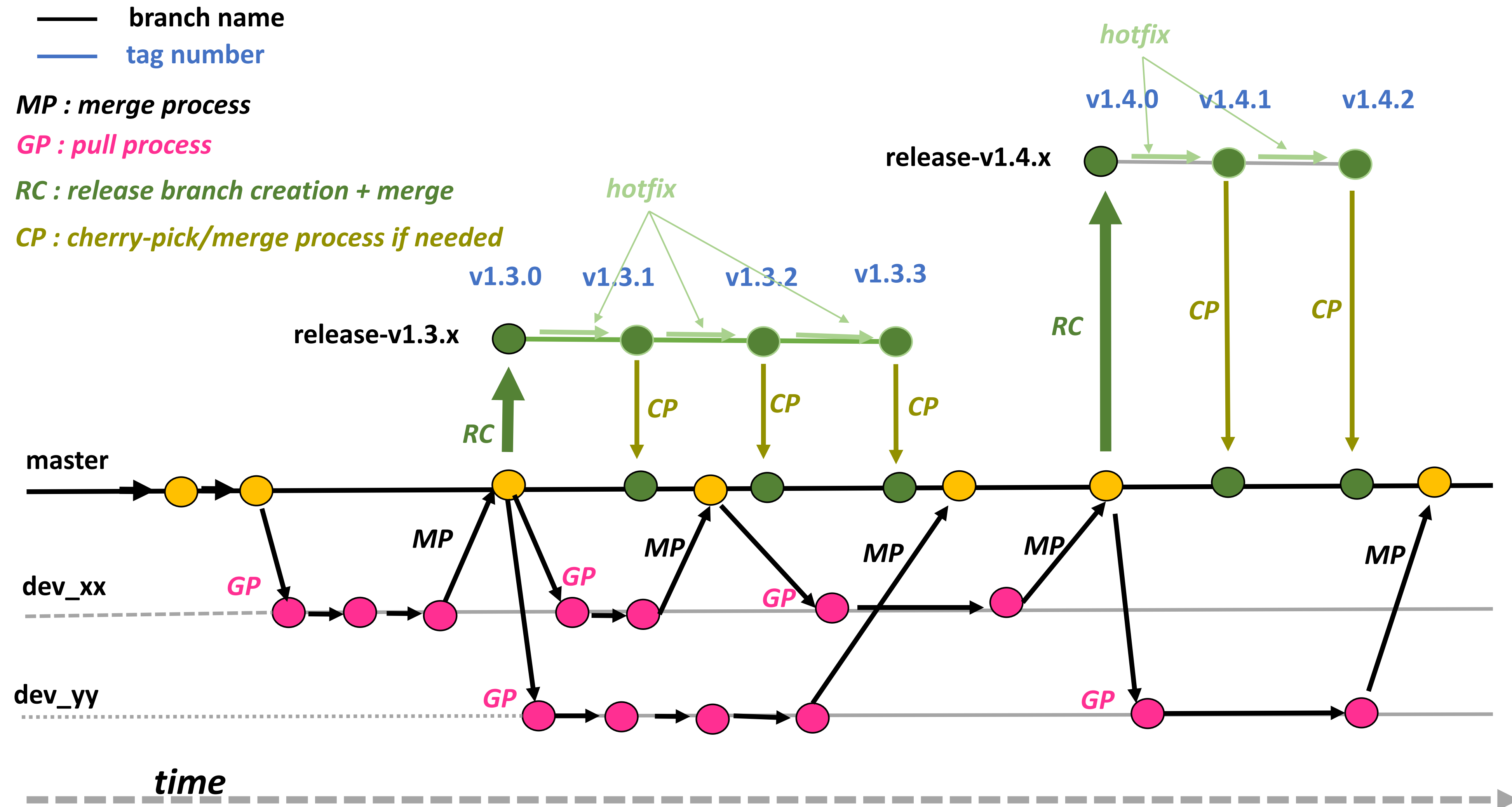
# Git workflow : Schéma de principe



# Avant de pusher des modifications : quelques habitudes

- je ne pushe pas directement sur le master
- je commence par synchroniser ma branche locale avec la branche distante
- je fais mes commits en local
- je m'astreins à passer les tests d'integration continue en local ( see wiki , #Automatic testing)
  - cas analytiques (ANA) [légers]
  - cas réaliste (REG) [plus lourds]
- je prepare un commit global thématique avec un message clair (see wiki : DEVELOPMENT GUIDE, #Squash : merging several commits before push)
- si je crée une nouvelle branche, j'ouvre une issue descriptive et je cree un label dédié (je n'oublie pas de fermer si nécessaire et d'enlever le label)
- **Il s'agit d'un dev (feature)**
  - je pushe sur ma branche de dev\_yy
  - je merge ensuite dans le master
- **Il s'agit d'un bug fix (hotfix) :**
  - j'ouvre une issue sur gitlab et je la documente (wiki : DEVELOPMENT GUIDE, # Issue)
  - en premier lieu je commites dans la branche stable : release\_va.b.x (finie tjr par x, c'est la « famille" de tag va.b)
  - j'indique dans le commit le numero de l'issue #XX
  - je propage dans la branche master avec un cherry-pick ou un merge (CP)
  - je ferme l'issue en indiquant les numéros de commit dans les branches release\_va.b.x ET master.

# Git workflow : Schéma de principe



## dev-xx branch :

May branch off from: master (GP)

Must merge back into: master (MP)

## release-va.b.x branch (ex release-v1.3.x or release-v1.4.x)

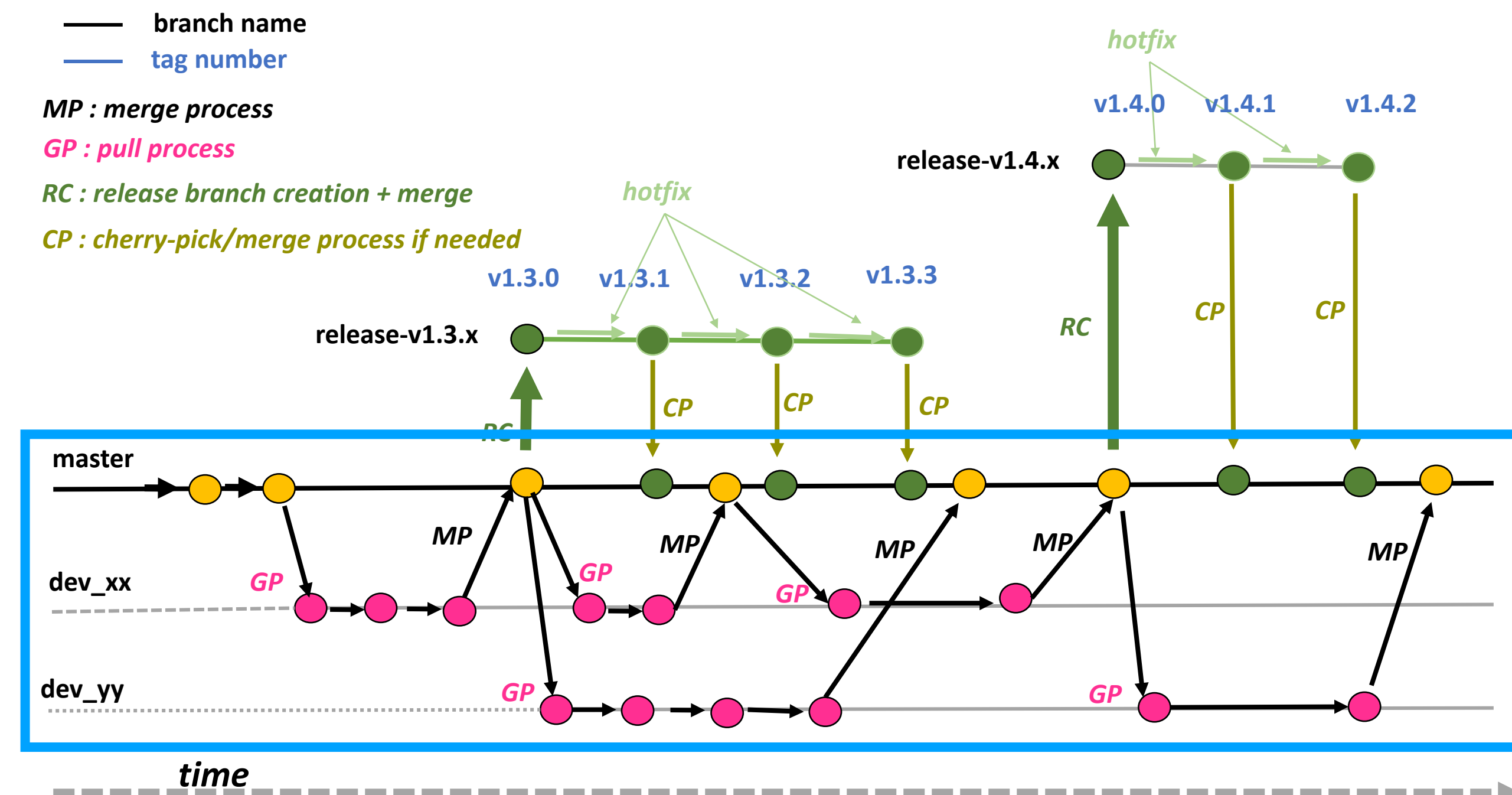
May branch off from : master ( RC + hot fix)

Must merge back into : master (CP)



# Je développe une nouvelle fonctionnalité YY [feature yy]

- je crée une branche dev\_year\_YY a partir de master  
\$ git checkout -b dev\_yy master  
Switched to a new branch « dev\_yy »
- j'effectue mes devs avec commits non pushé sur ma branche dev\_YY
- j'effectue les tests de CI : compilation, execution, repro parallèle et restartabilité en local ([wiki](#) : DEVELOPMENT GUIDE)
- quand ces tests passent, je rebase mes commits et effectue un commit global propre ([wiki](#) : DEVELOPMENT GUIDE)
- je bascule sur ma branche master locale  
\$ git checkout master  
Switched to branch 'master'
- je merge la branche dev\_year\_YY dans master, en mode no-fast forward  
\$ git merge --no-ff dev\_YY  
Updating ea1b82a..05e9557  
(Summary of changes)



Je développe une nouvelle fonctionnalité YY [feature yy]

- je résous les conflits éventuels
- je bascule sur la branche master
- 2 solutions :
  - je merge ma branche dev\_YY dans master  
`$ git push origin master`
  - je fais une merge request ( review + validation tierce)

**croco**

Project information

Repository

Issues 34

**Merge requests 1**

CI/CD

Security and Compliance

Deployments

Packages and registries

 L'opération de maintenance du week-end de Gitlab Community Edition. Une grosse mise à jour. Merci de votre patience.

 croco-ocean >  **croco**

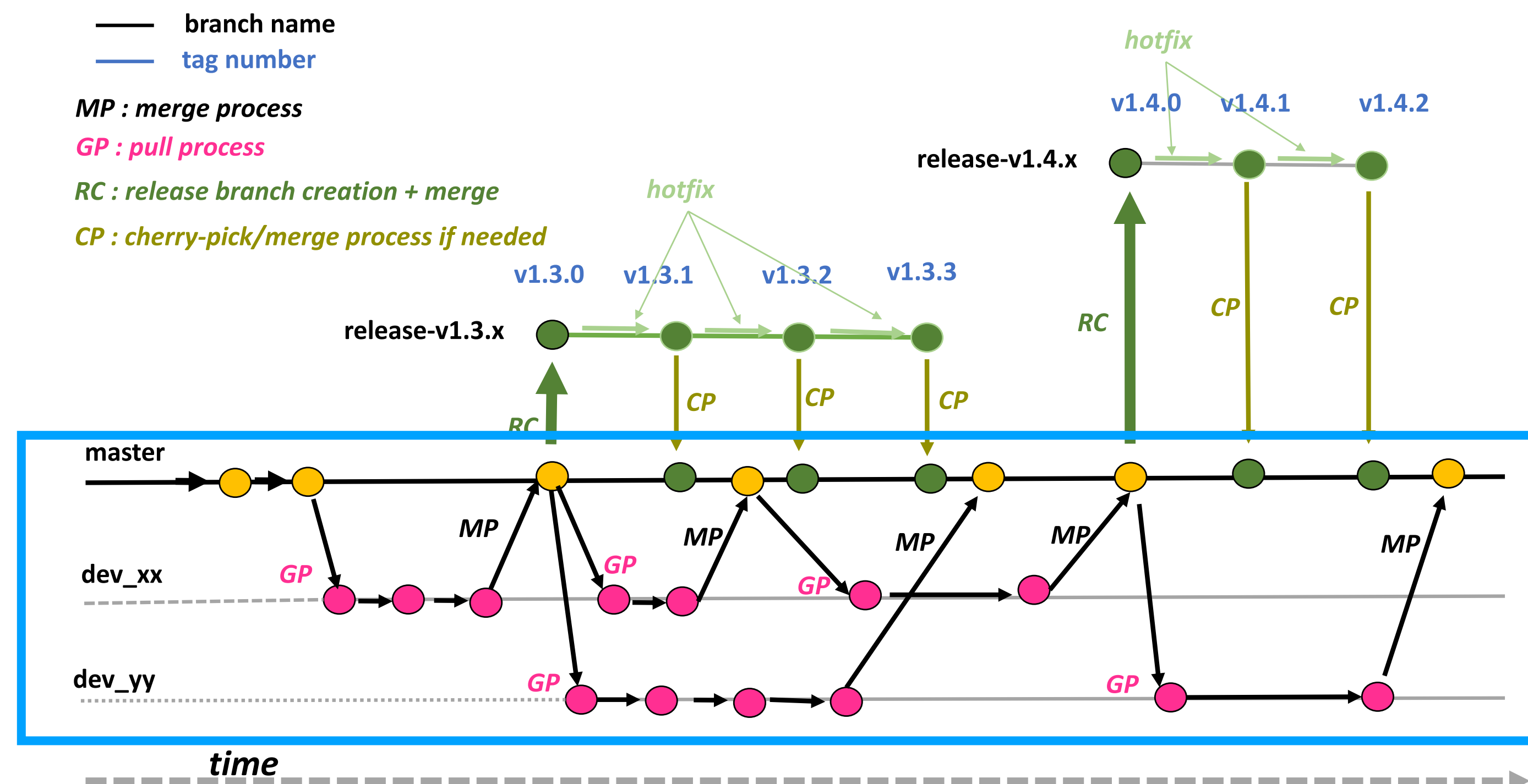
**croco** 

Project ID: 2954  [Leave project](#)

 **5,839** Commits  **20** Branches  **8** Tags

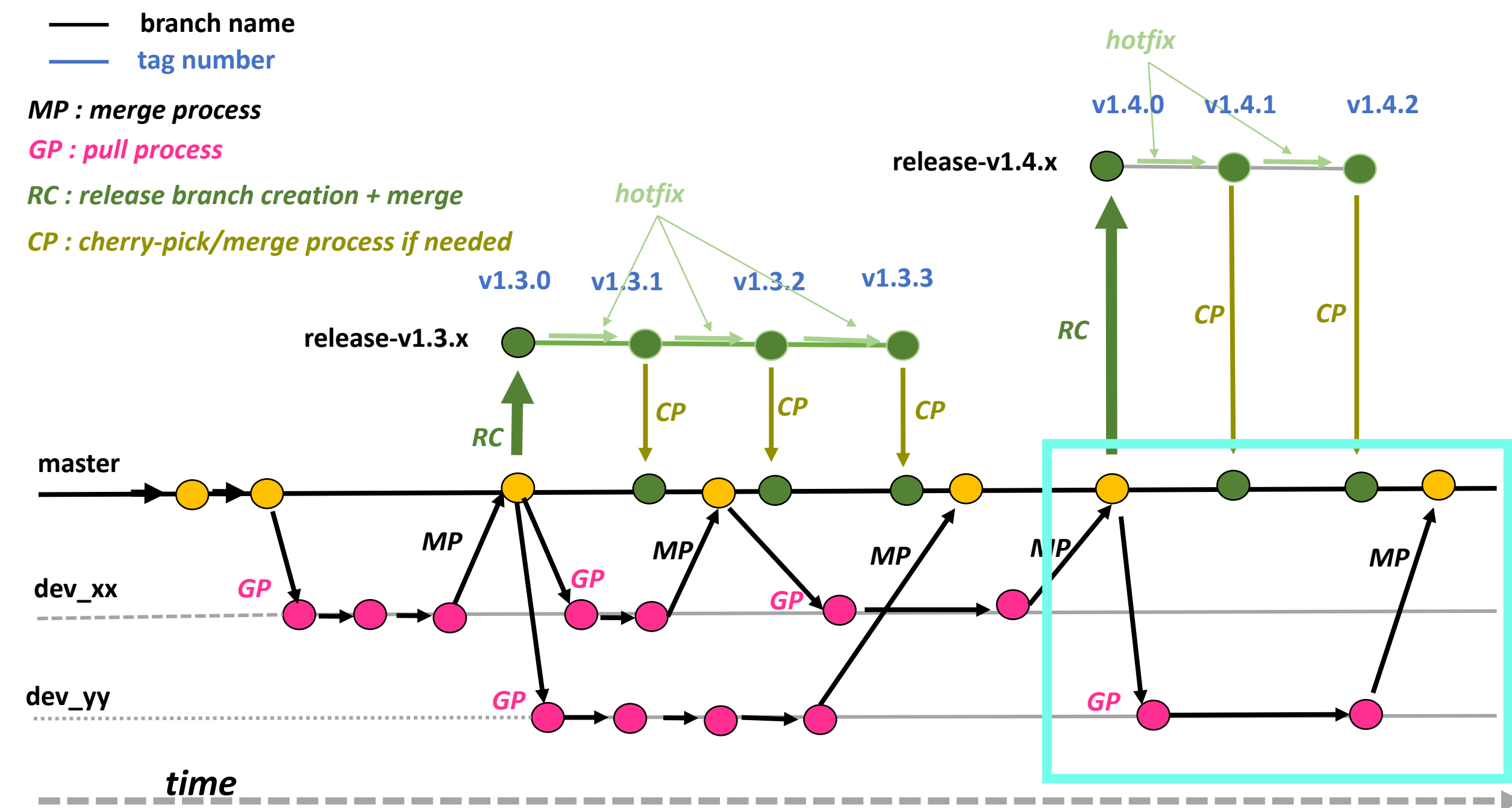
Topics: [ocean](#)

- Eventuellement j'efface ma branche de dev  
\$ git branch -d dev\_year\_YY  
Deleted branch dev\_year\_YY (was 05e9557).



# Je continue de developper sur ma branche dev\_YY [feature yy]

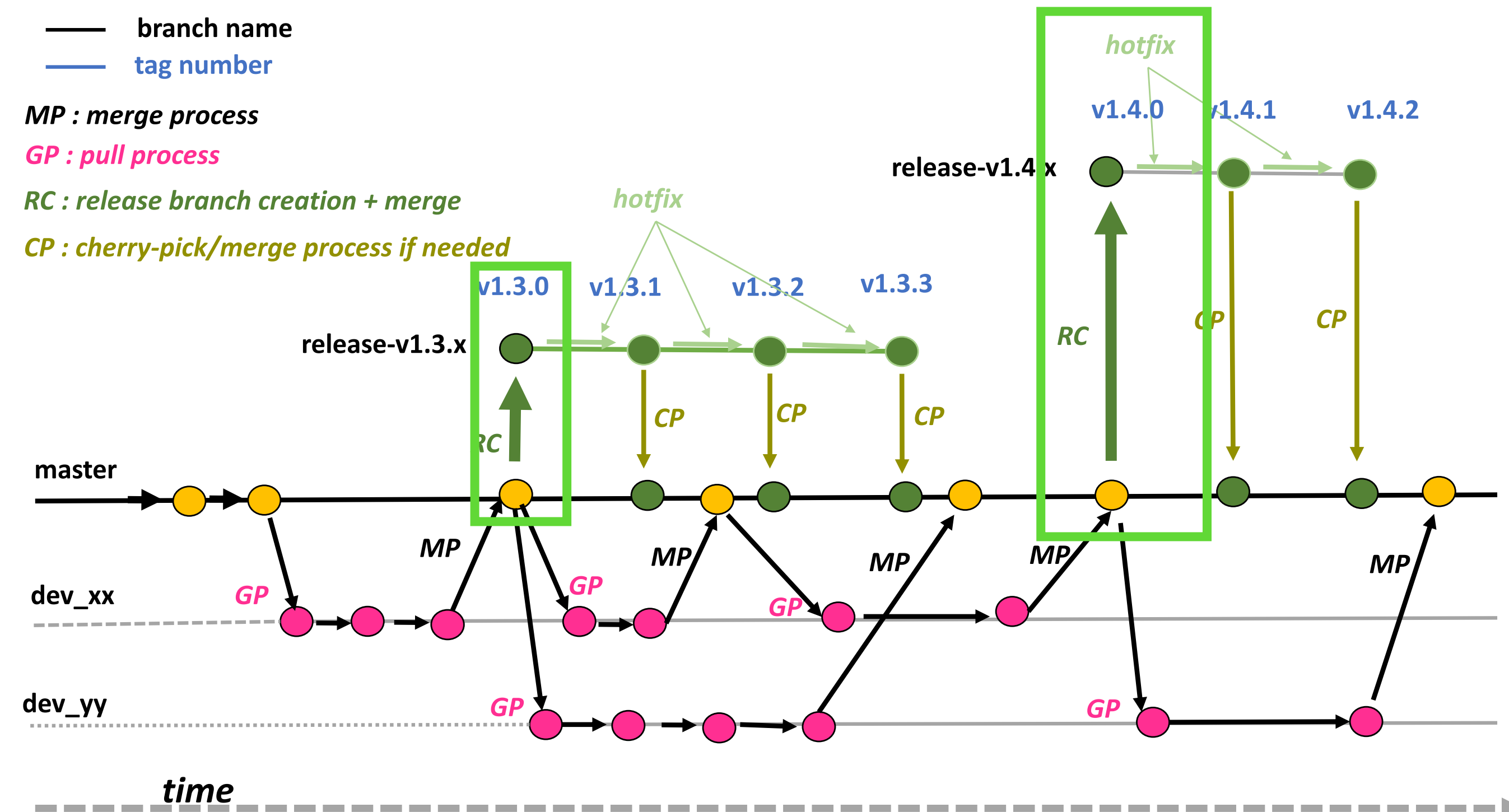
- je synchronise ma branche dev\_yy avec master (GP)  
\$ git checkout dev\_YY  
\$ git merge no-ff master
- j'effectue mes devs avec commits non pushés sur ma branche dev\_YY (cf slide 6)
- j'effectue les tests de CI : compilation, execution, repro parallèle et restartabilité en local (cf slide 6)
- quand ces tests passent, je rebase mes commits et effectue un commit global propre (squash commit) (cf slide 6)





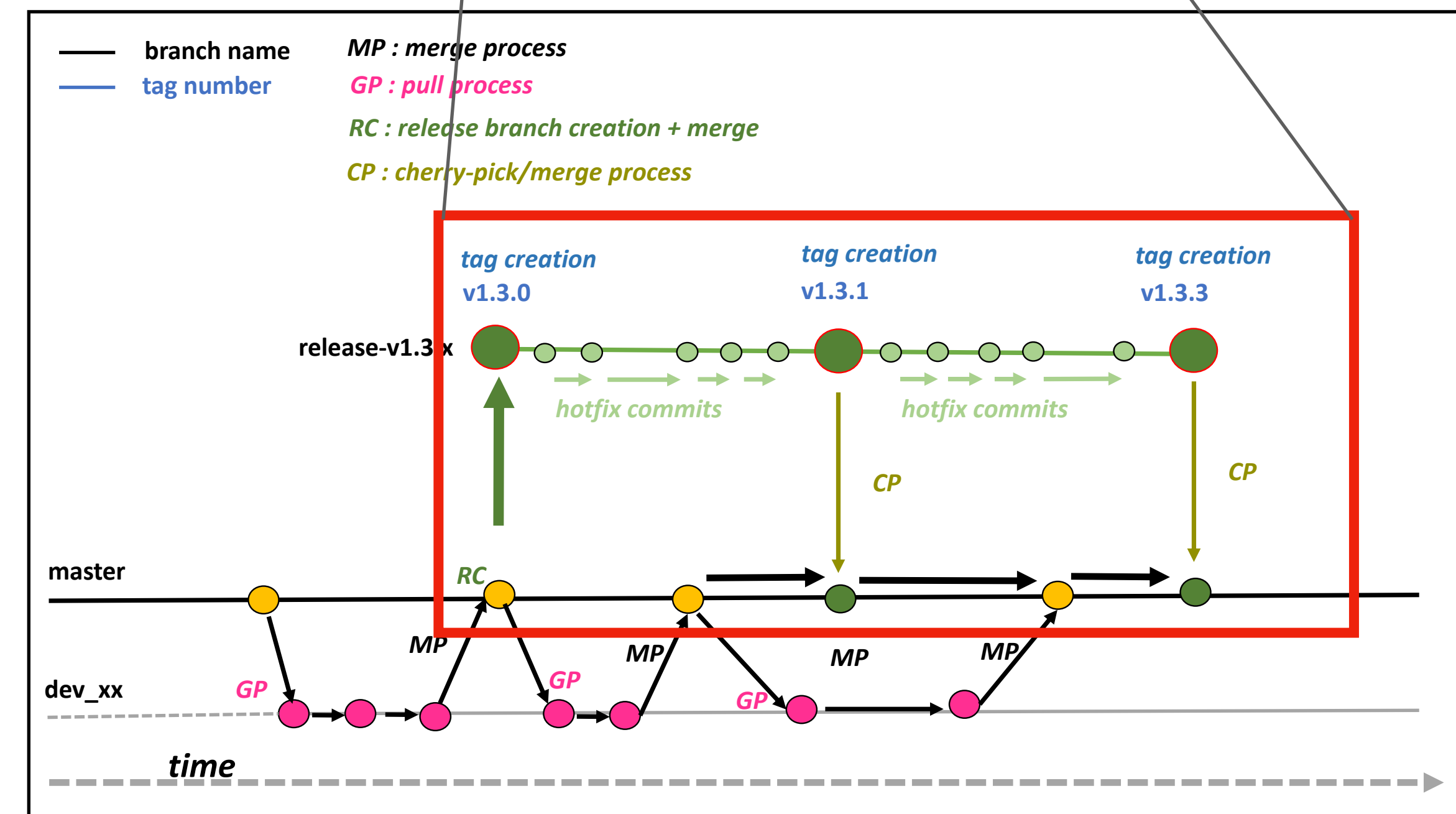
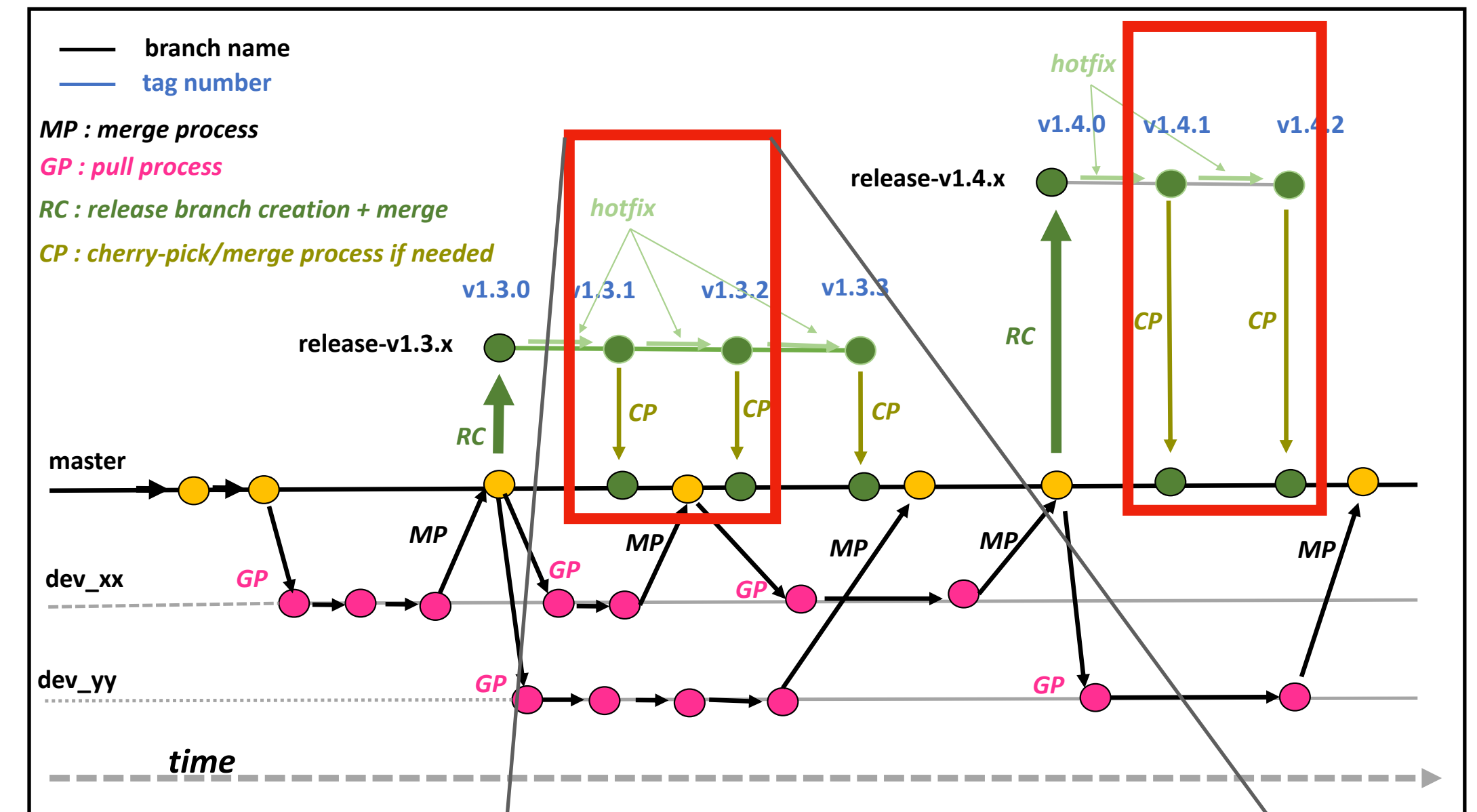
# Je prepare une release majeure (tag va.b.0)

- je me place sur master
- je mets a jour ma branche master locale
- je mets à jour le fichier CHANGELOG.md via un commit pour le premier tag va.b.0
- je crée une branche release-va.b.x à partir de master (RC) (x ne bouge pas)
- je fais les ajustements et commits nécessaire, mais uniquement des bugfix ( pas de dev)
- les branches de release sont nommées release-va.b.x (le x ne bouge) : 2 digits
- je pose le premier tag est va.b.0
- je pousse la branche release-va.b.x sur le depot distant origin/release-va.b.x
- dans master et release-va.b.x, le CHANGELOG.md a été incrémenté



# Je détecte un bug dans le tag v1.3.0 de release-v1.3.x [hotfix]

- je crée une issue #xx sur le gitlab en détaillant le bug et le tag de la release touchée
- je me place sur la branche release-v1.3.x
- j'effectue mes corrections : commits (non-pushés) sur ma branche release-v1.3.x
- j'effectue les tests de CI : compilation, execution, repro parallèle et restartabilité en local
- quand ces tests passent, je rebase/squashe mes commits et effectue un commit global propre
- je pousse dans la branche distante origin/release-v1.3.x



# Je détecte un bug dans le tag v1.3.0 de release-v1.3.x [hotfix]

- je merge, si c'est nécessaire, ces commits dans le master, en mode -no-ff (non-fast-forward)

```
$ git checkout master
```

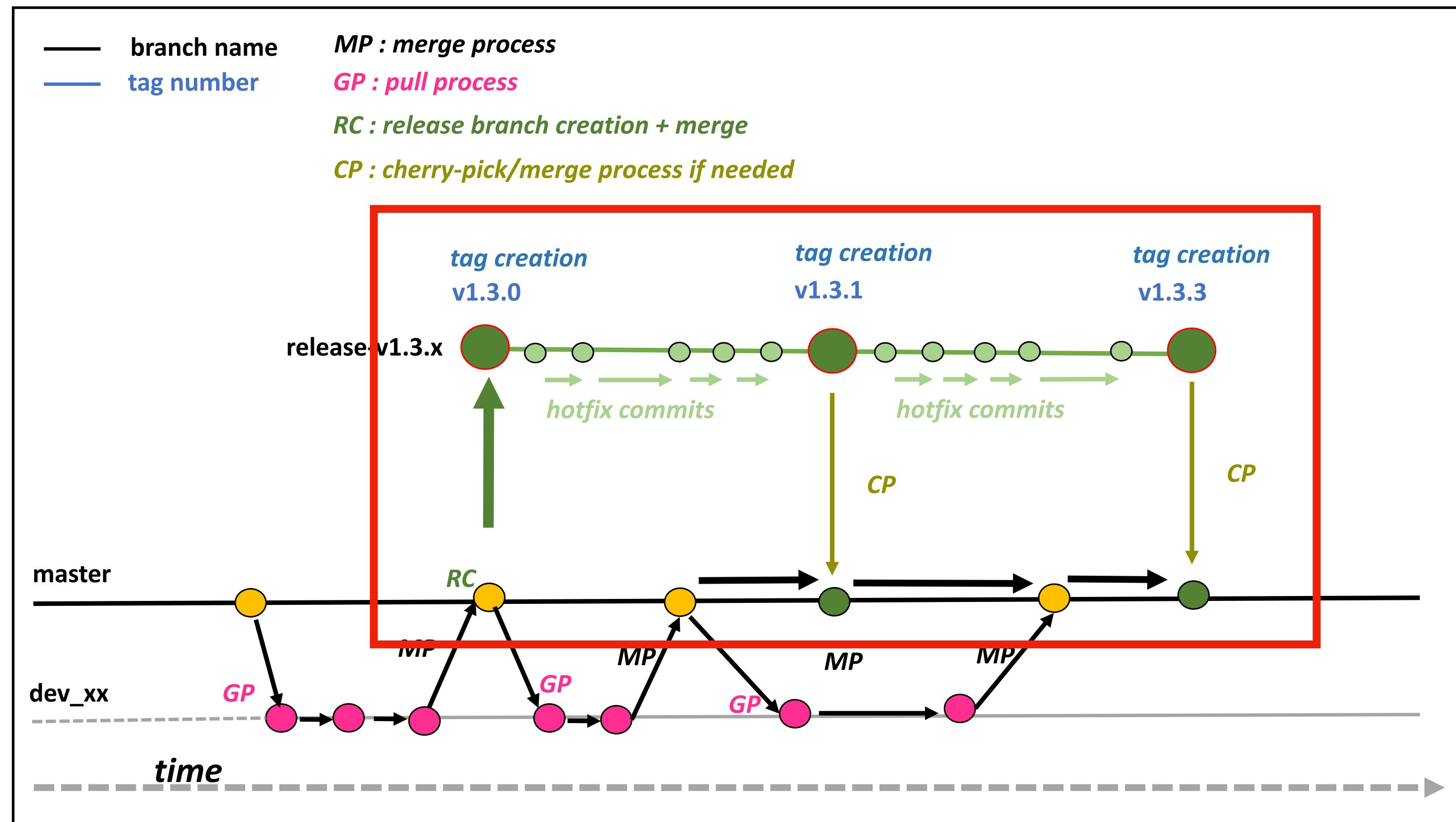
```
Switched to branch 'master'
```

```
$ git merge --no-ff release-v1.3.x
```

- je clos l'issue #xx :
  - en indiquant le numero du commit dans release-v1.3.x
  - en indiquant le numero du commit dans master

Si un nouveau tag doit être fait :

- je me place dans release-v1.3.x
- juste avant la création d'un nouveau tag, je mets à jour le fichier CHANGELOG.md via un commit
- je fais un nouveau tag (ex v1.3.0 -> v1.3.1) sur la branche release-v1.3.x
- je fais un « CP » dans le master (notamment avec le CHANGELOG.md updaté)



=> les hotfix amènent à la création d'un nouveau tag incrémenté : va.b.1, va.b.2 etc ...

# Distribution

---

- sous forme de tar.gz sur croco-ocean.org : => croco-release-v1.3.1
- tag **va.b.c**, de la branche **release-va.b.x** (le x est fixe, c'est la branche contenant la « famille » de tags va.b.c) sur gitlab.inria.fr