La gestion de version

Christophe Demarey

Septembre 2014

SOMMAIRE

Motivations

- Ingénierie logicielle collaborative
  - plusieurs développeurs
  - implémentations parallèles
- Contrôle de l'évolution du logiciel
  - être capable de faire machine arrière si besoin
- Traçage des évolutions (navigation)
- Partage du code

Gestionnaire de version

- Acronymes communément utilisés
  - VCS: Version Control System
  - SCM
- Ne sert pas à
  - la sauvegarde
  - gérer les incidents (issues) et requêtes diverses
  - gérer un projet de développement (cf forges)

Que mettre dans VCS?

- Tous les sources nécessaires au projet
  - code source (.c .cpp .java .py …), y compris les tests
  - scripts de build (Makefile Cmakefile.txt pom.xml …)
  - Documentation (.txt .tex Readme …)
  - Ressources (images …)
  - Scripts divers (déploiement, BD .sql, …)

Ne pas stocker dans un VCS

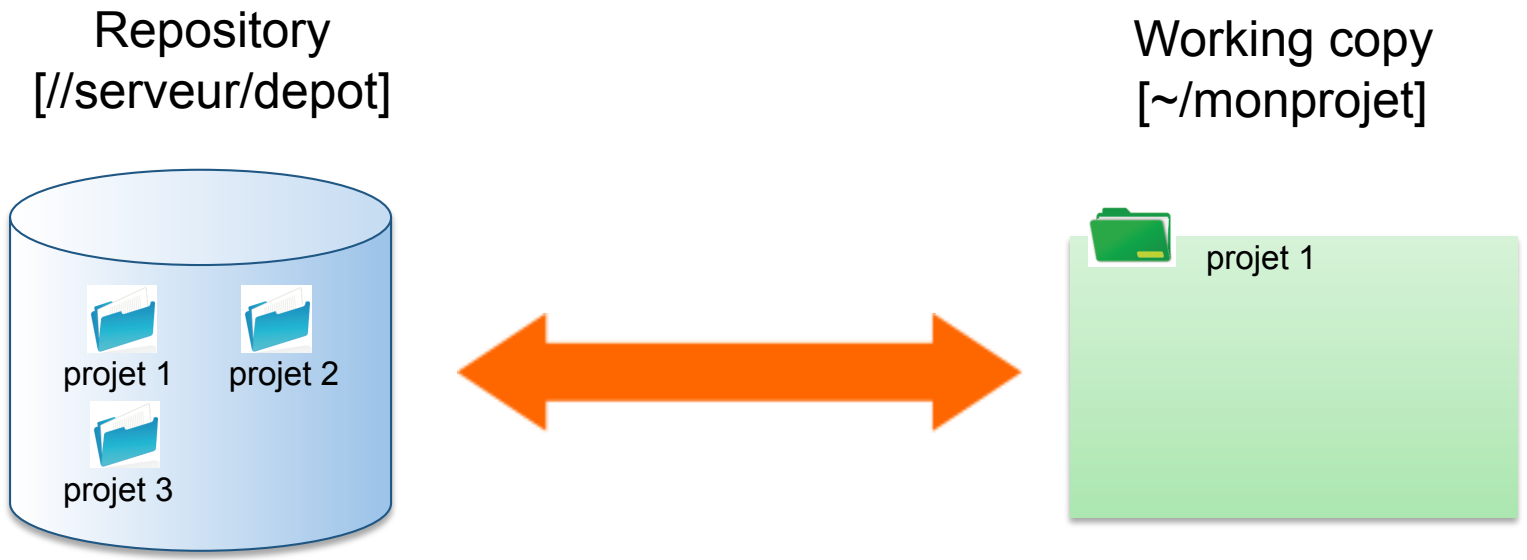
- Les fichiers générés
  - Résultat de compilation (.class .o .exe .jar …)
  - Divers
    - .ps .dvi .pdf avec latex
    - javadoc
    - etc.

1

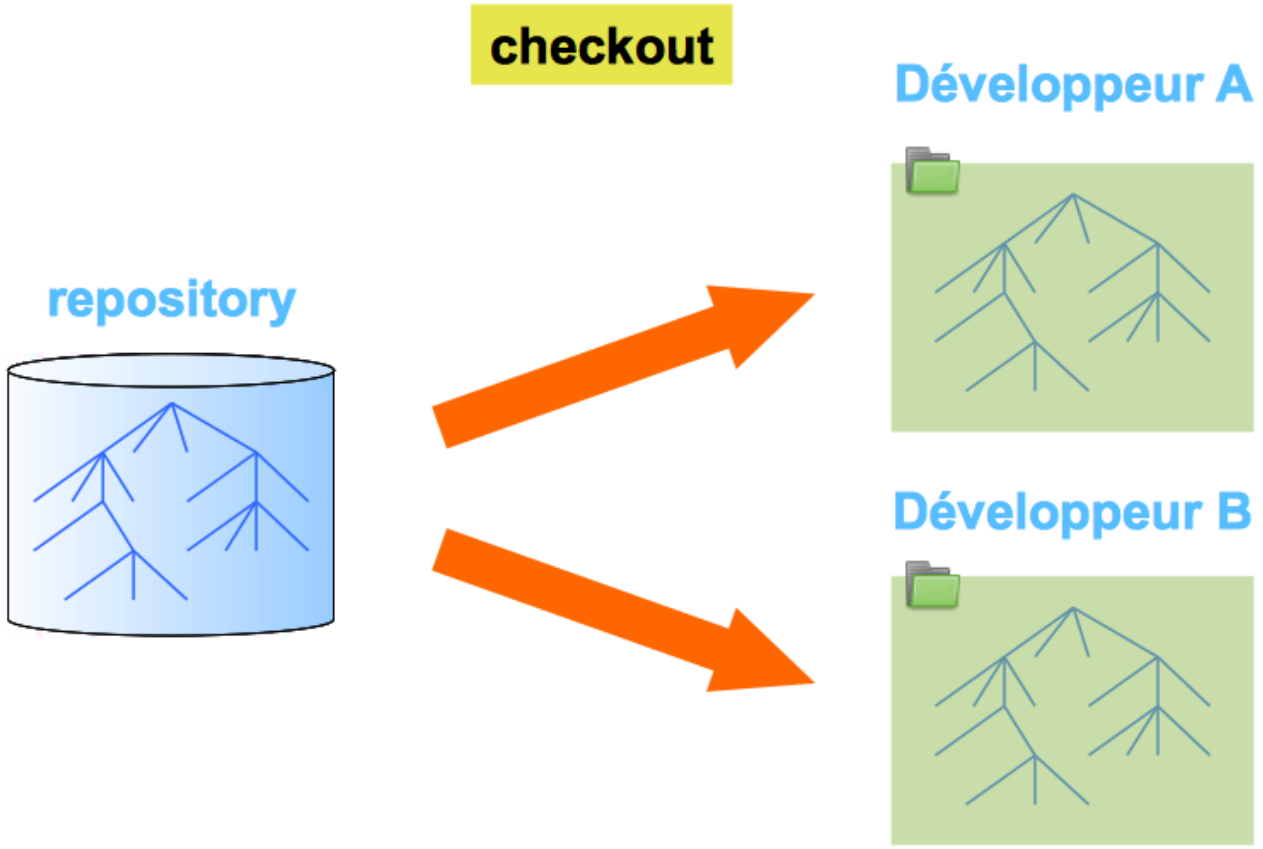
Notions de base

Sous-titre facultatif

Repository / working copy



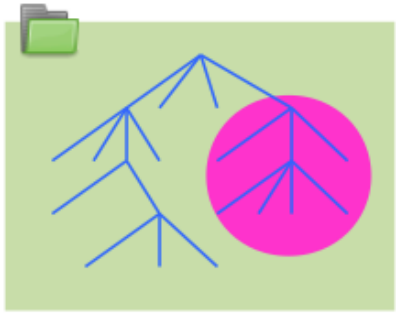
Récupération d'un projet



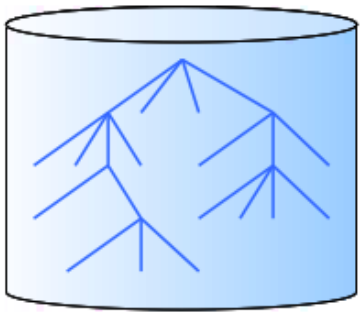
Mise à jour

développement

Développeur A

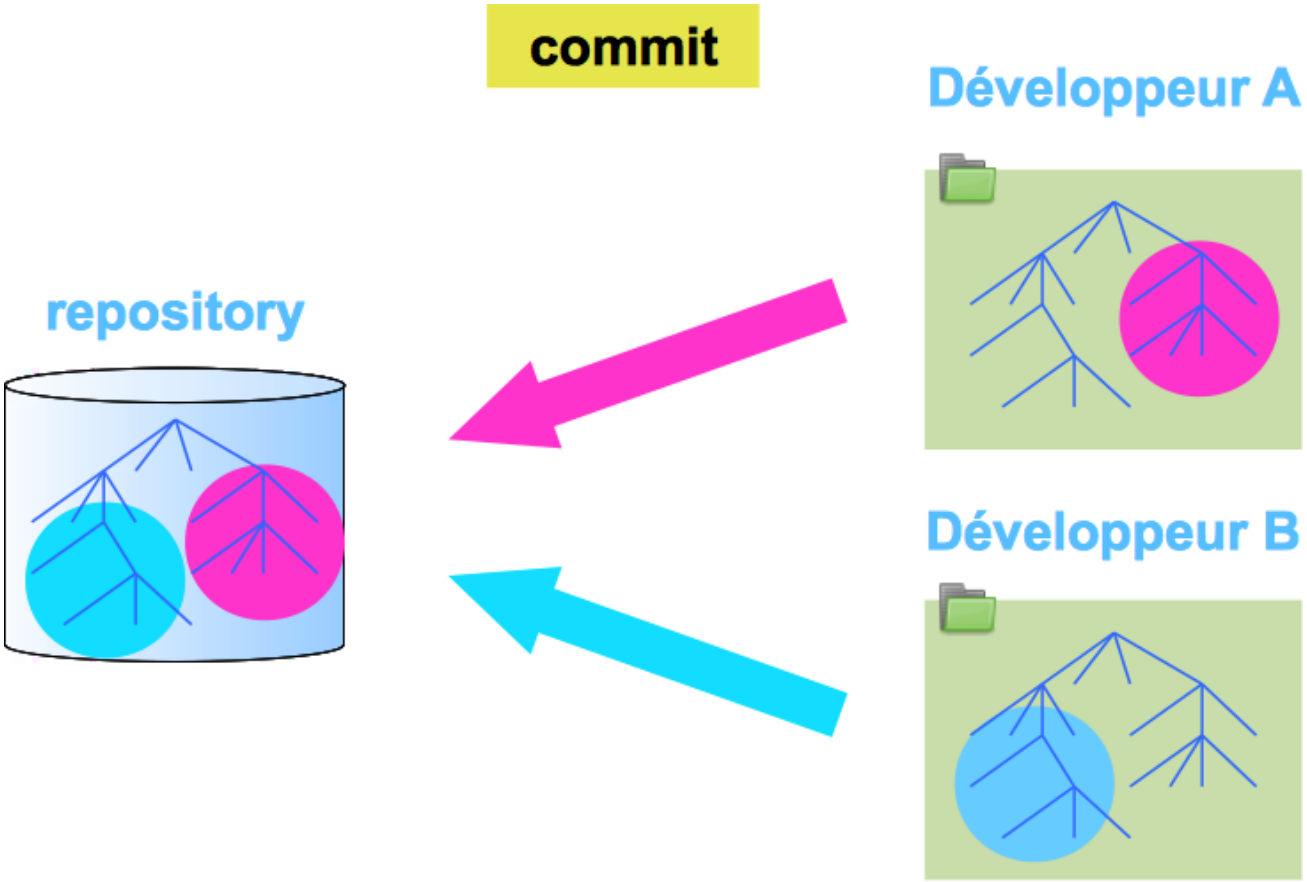


repository

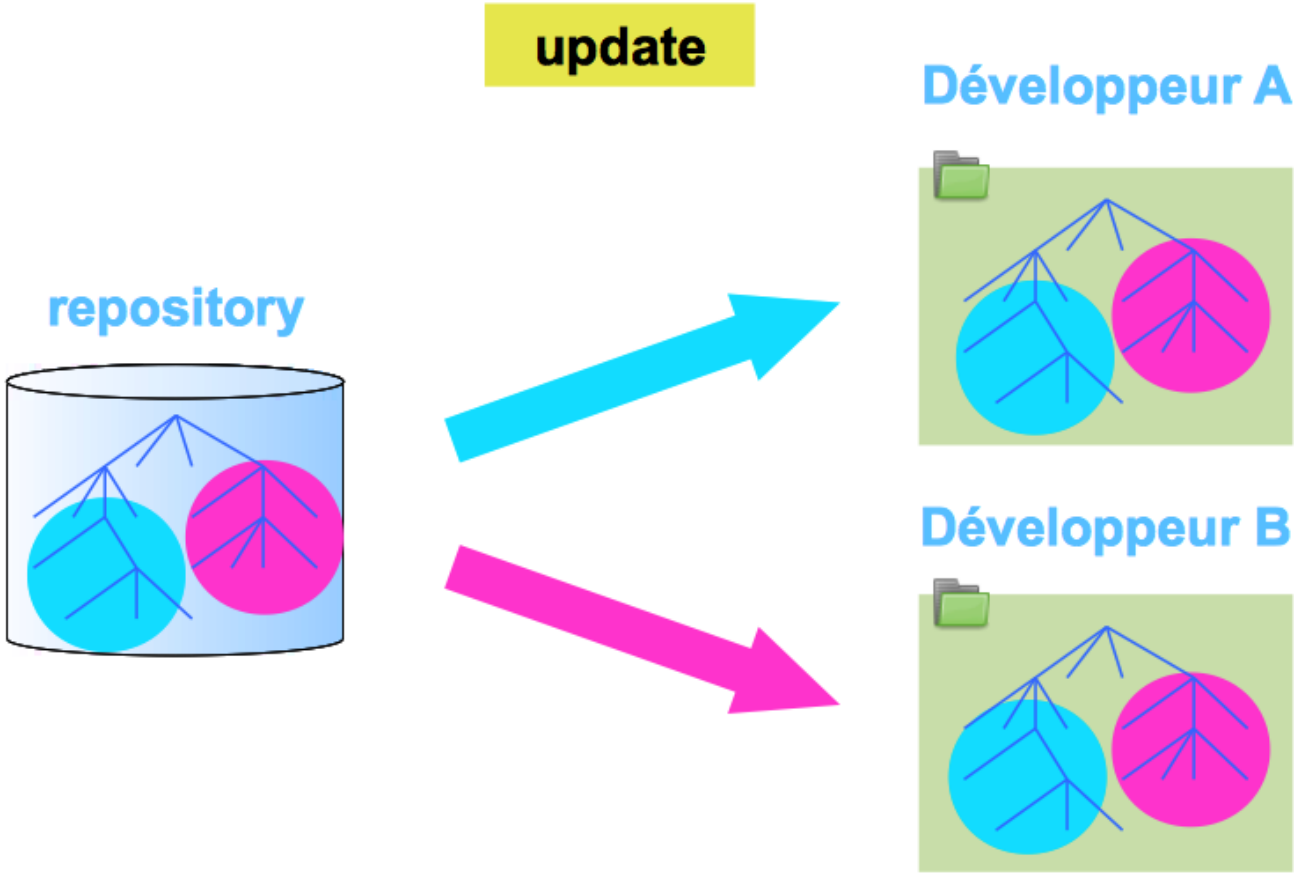


Développeur B

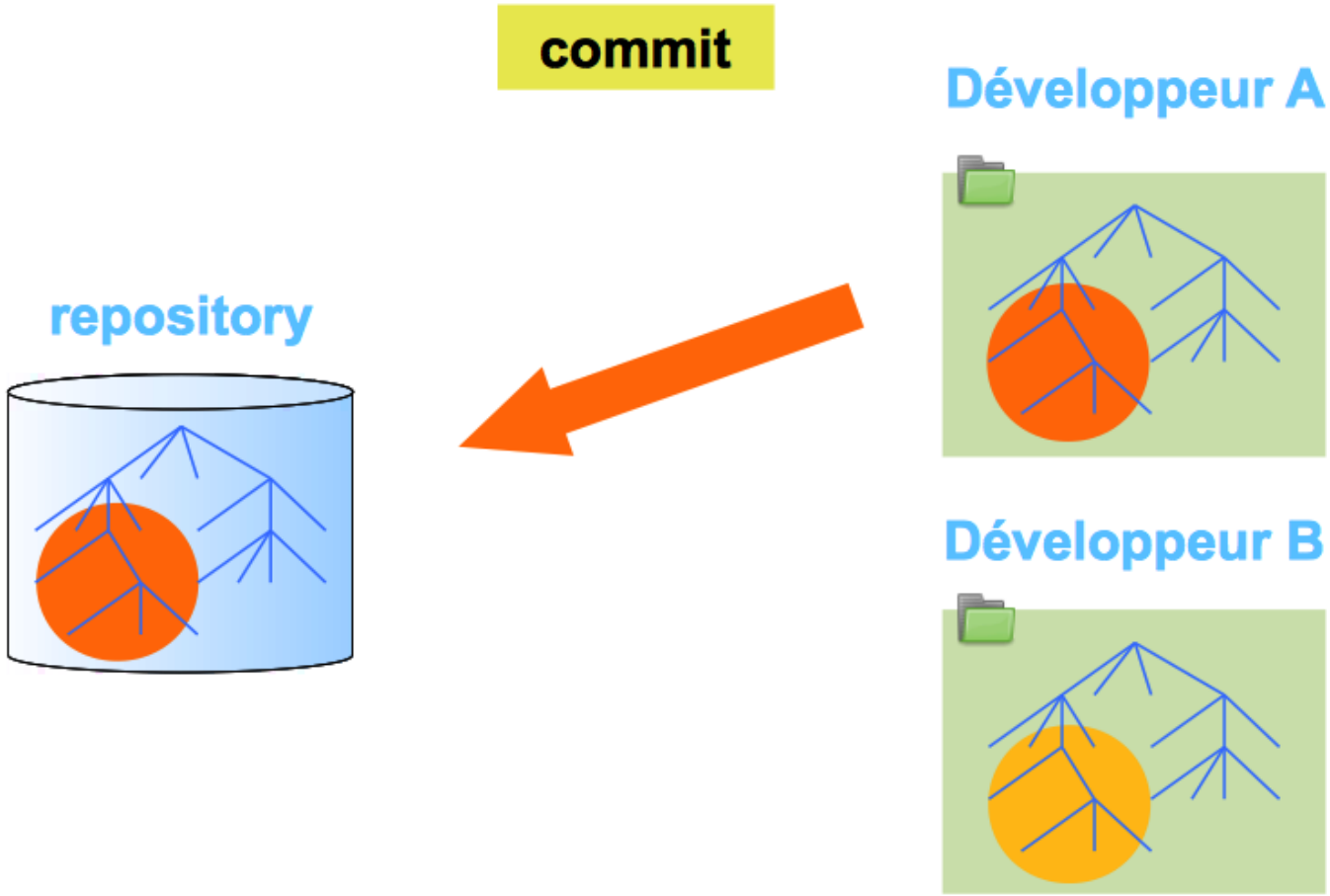
Mise à jour



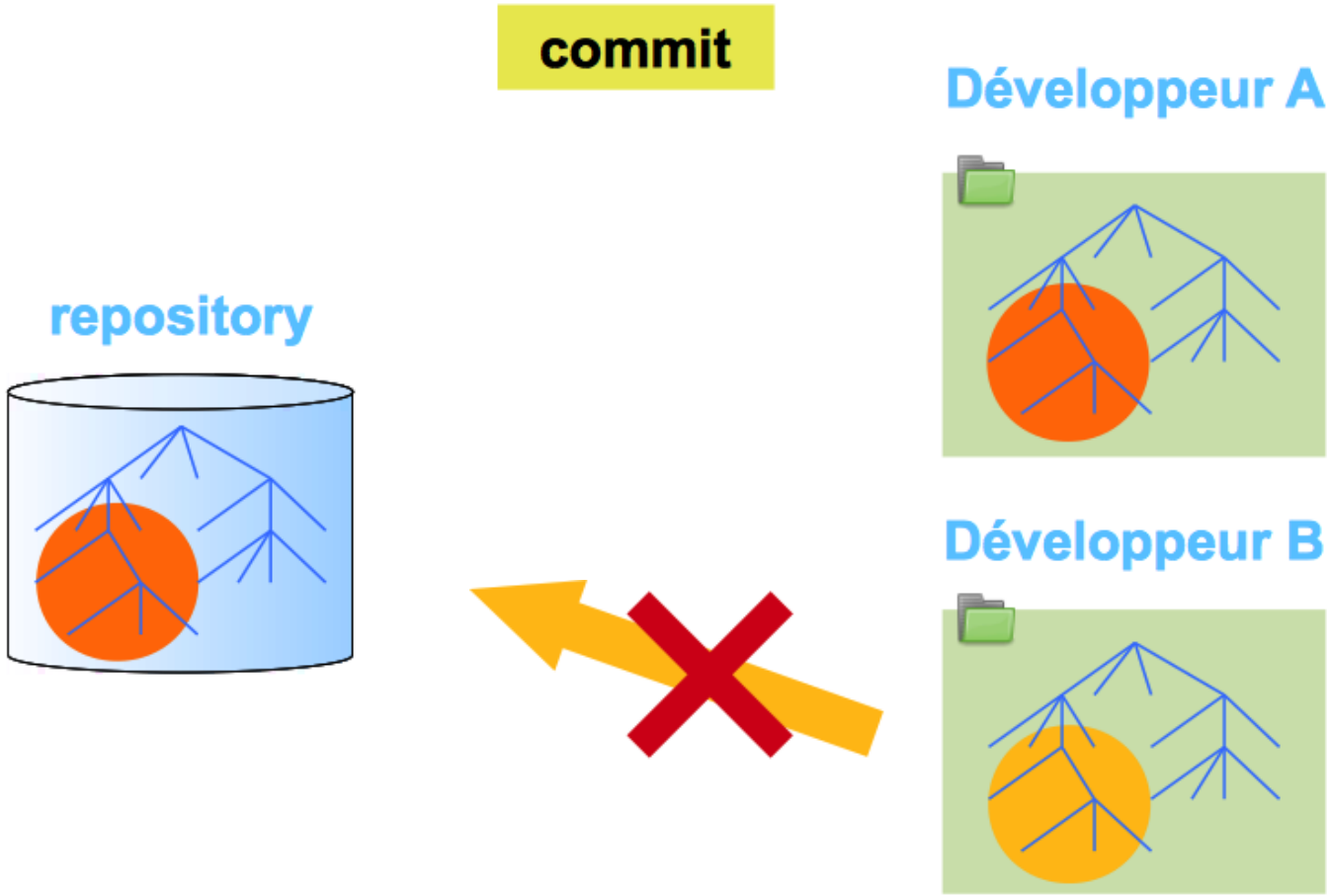
Mise à jour



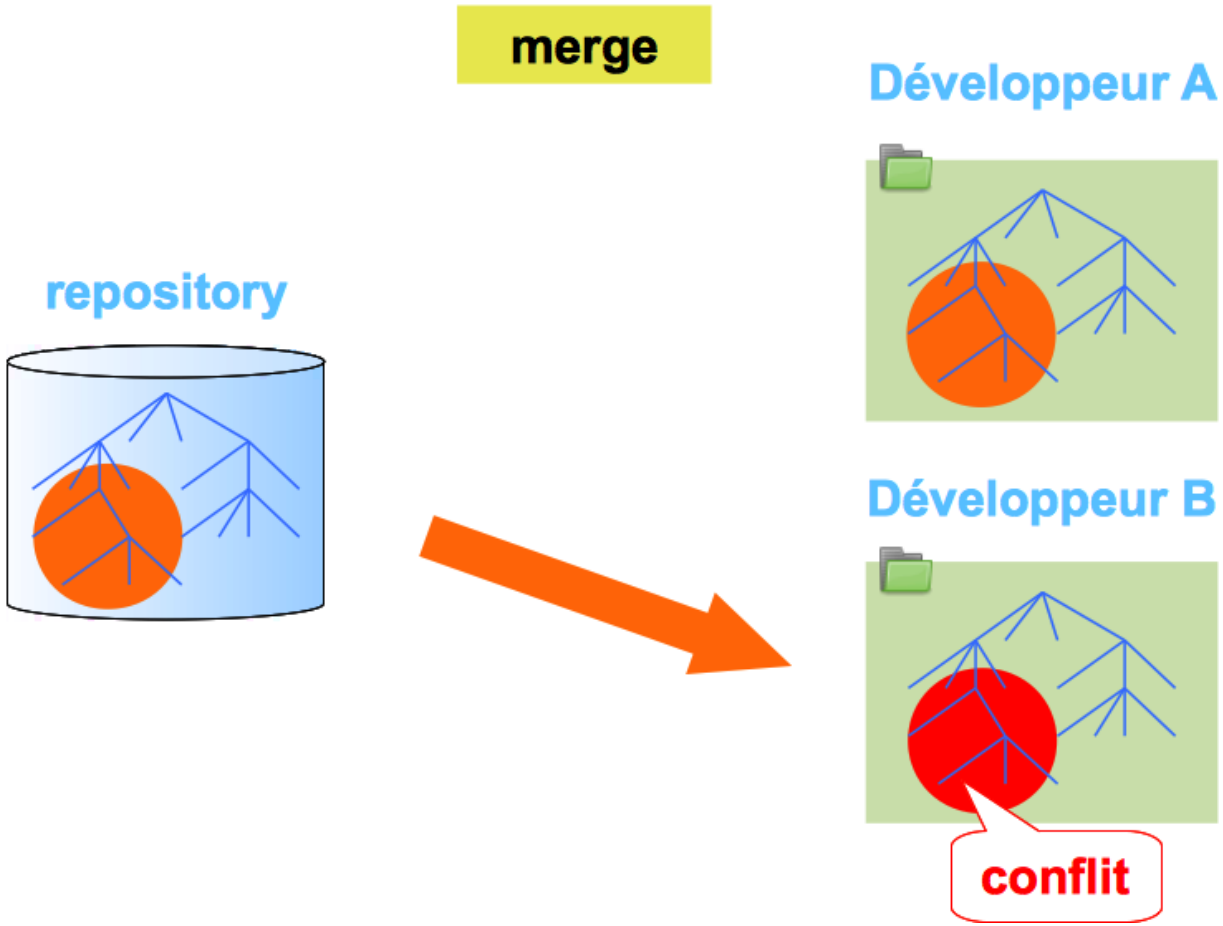
Conflit



Conflit



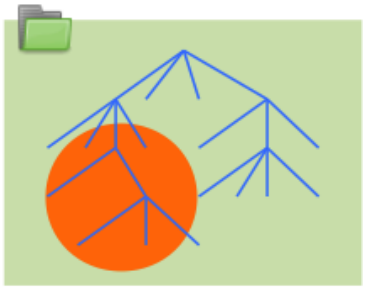
Conflit



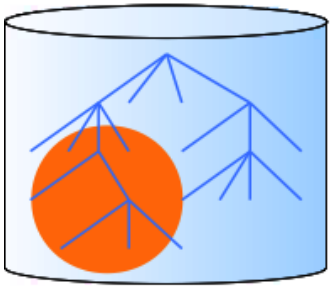
Conflit

diff

Développeur A



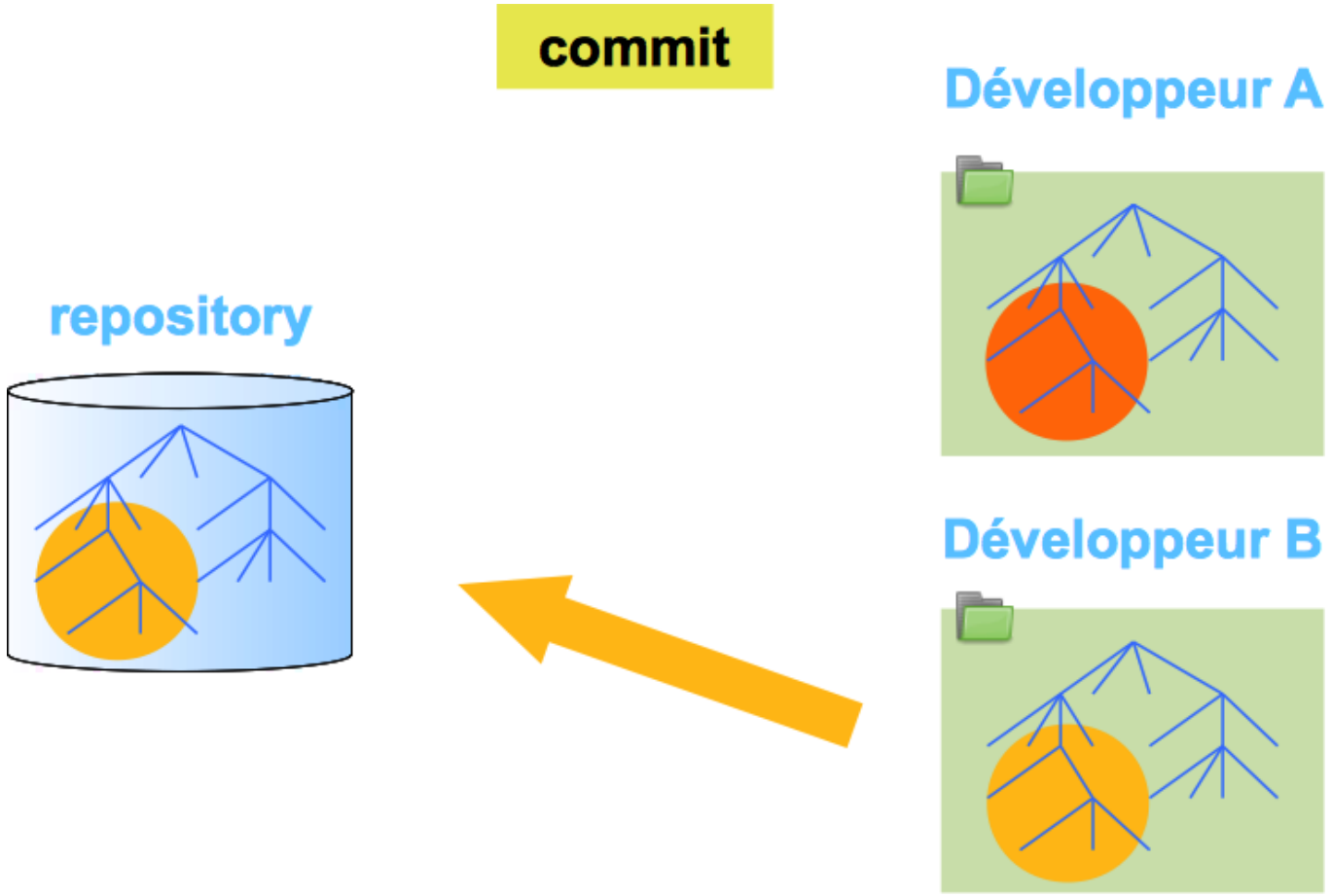
repository



Développeur B

conflit

Conflit



Marquage de version

tag

- souvent utilisé pour marquer le repository lors d'événements importants comme la release du logiciel
- permet de retrouver facilement une version spécifique du logiciel
  - utiliser / distribuer une release donnée
  - reproduire des bugs pour une release donnée

my_project-v1.3

Tag / Branches

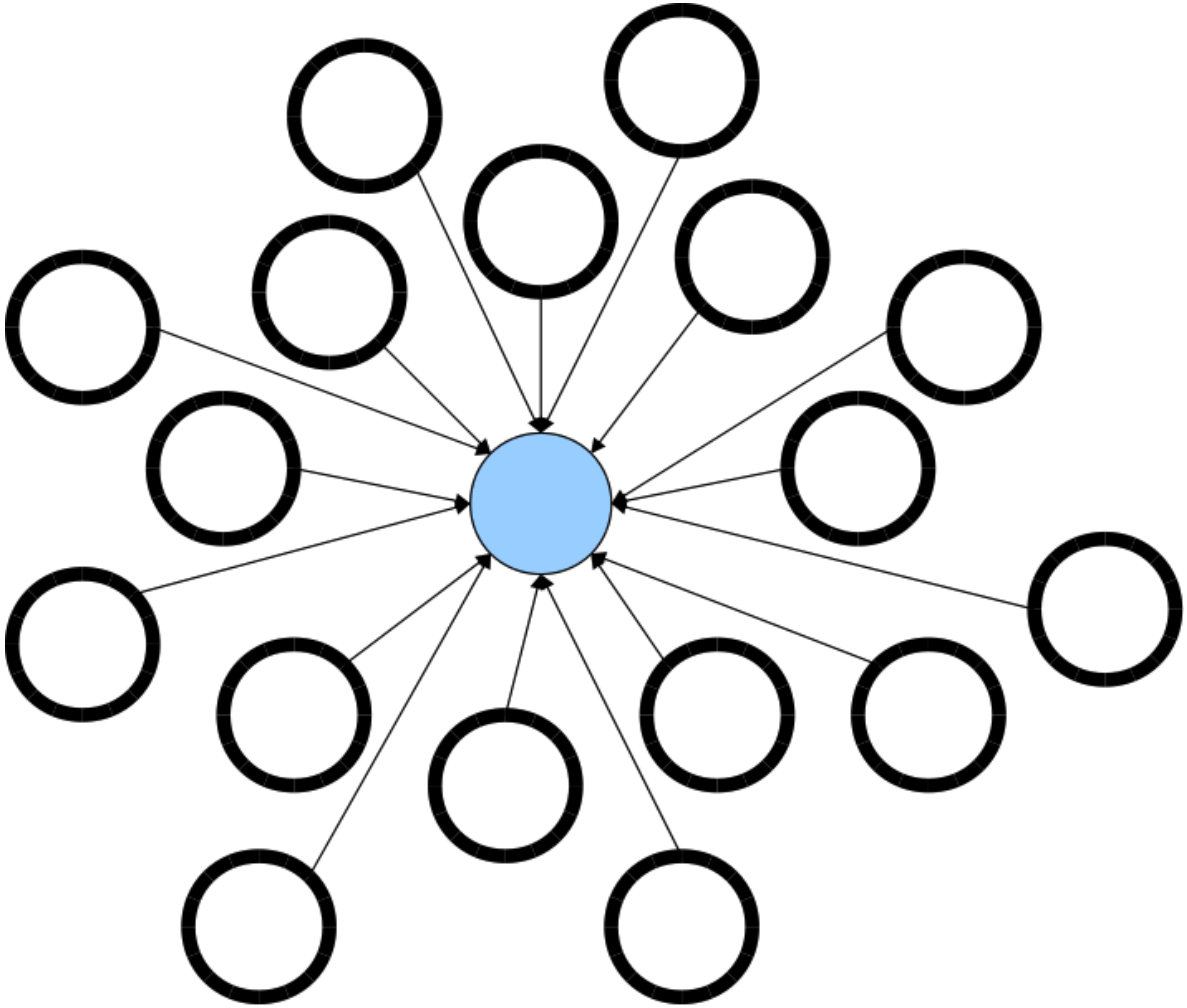
Git for beginners handout (slides 11-22)

2

Centralisé / Décentralisé

Sous-titre facultatif

Gestionnaire de version centralisé



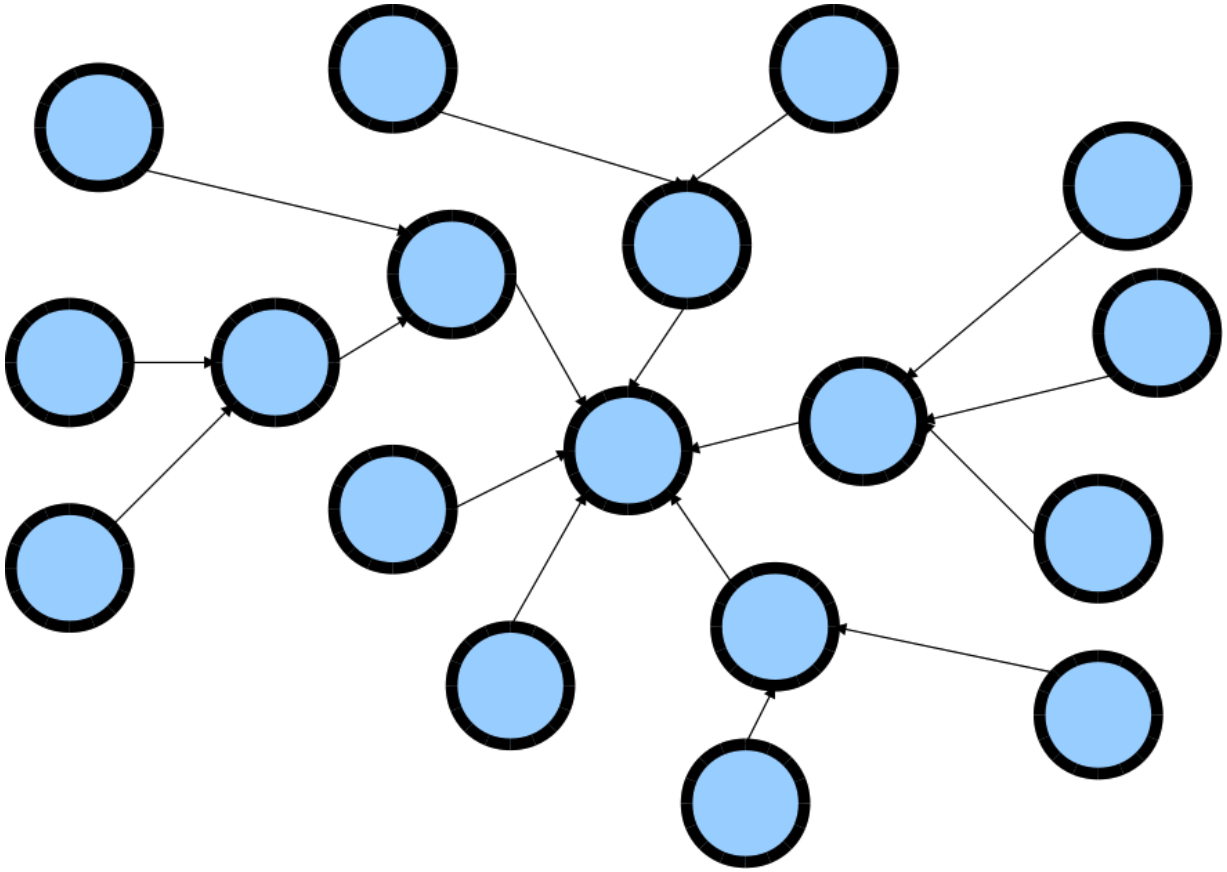
Gestionnaire de version centralisé

Un seul dépôt, n working copies

Ex: (CVS,) SVN

- \+ Simple à utiliser
- − Besoin d'être en mode connecté pour la plupart des commandes
- − Gestion des travaux concurrents (merge) difficile
- − Utilisateurs privilégiés (committeurs)

Gestionnaire de version décentralisé



Apport du décentralisé

- Chaque développeur peut avoir son propre dépôt
- Utilisation en mode déconnecté

  (commandes disponibles offline)

  ex : faire des commits localement
- Créer une branche sans avoir à demander l'autorisation

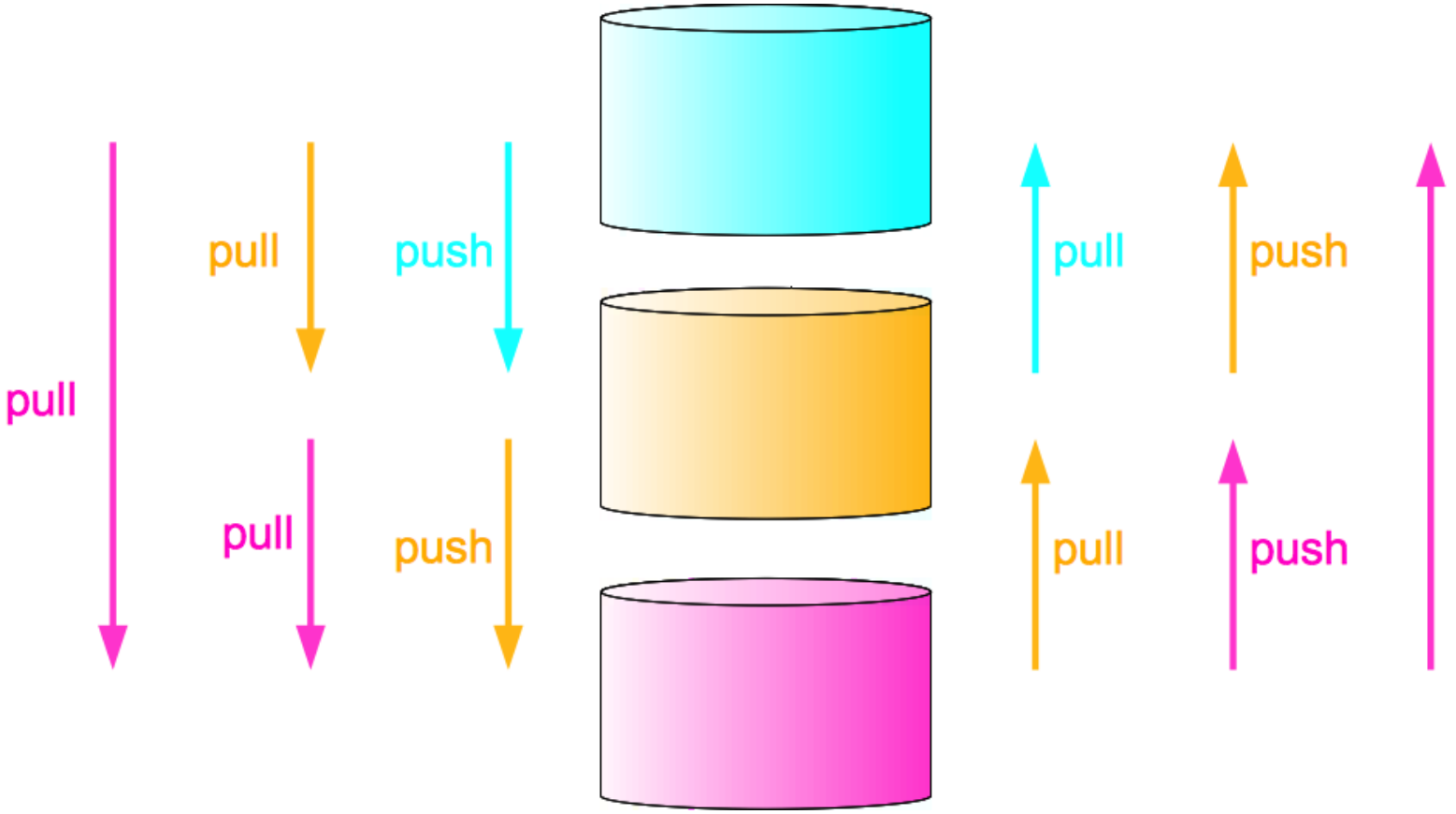
  ex : communauté Open Source

Inconvénients

Besoin de synchronisation entre dépôts

- pull = récupérer les modifications d'un dépôt distant vers son dépôt local
- push = apporter ses modifications à un dépôt distant

Echange entre dépôts distants



Algorithmes évolués de merge

- **star merge**

  se souvient des merges précédemment effectués pour ne pas appliquer 2 fois la même modification

- **cherrypick**

  sélection des changements qui doivent être appliqués d'une branche à une autre

  >> Git for beginners handout (slide 23)

3

Git

Sous-titre facultatif

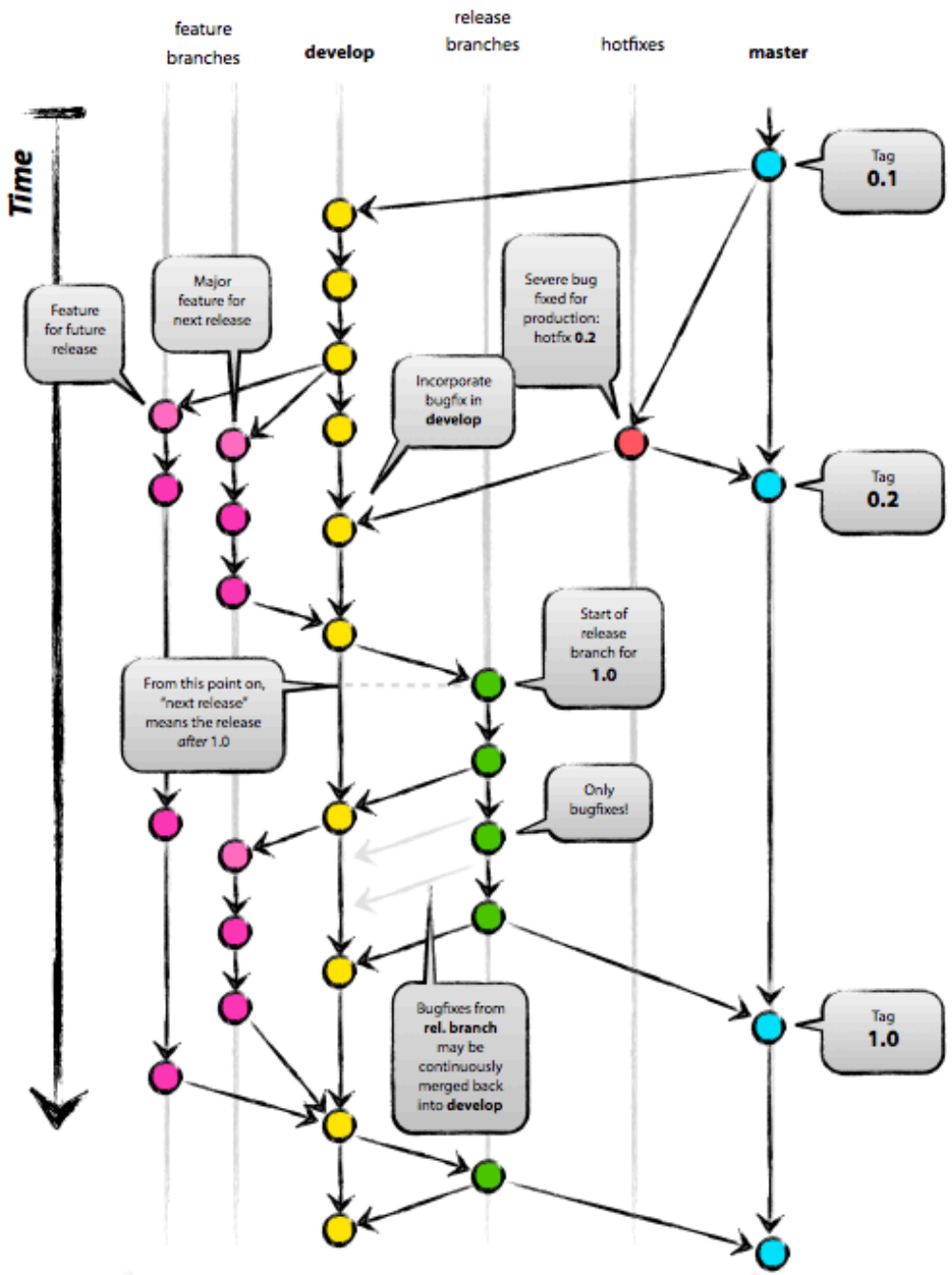
Git

Git for beginners handout (slides 27 - 101)

Git Flow

Guidelines to manage a software development process

http://nvie.com/posts/a-successful-git-branching-model/



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