

VCS, DVCS, Git-flow

La gestione dei sorgenti e il controllo di versione

Tiziano Lattisi

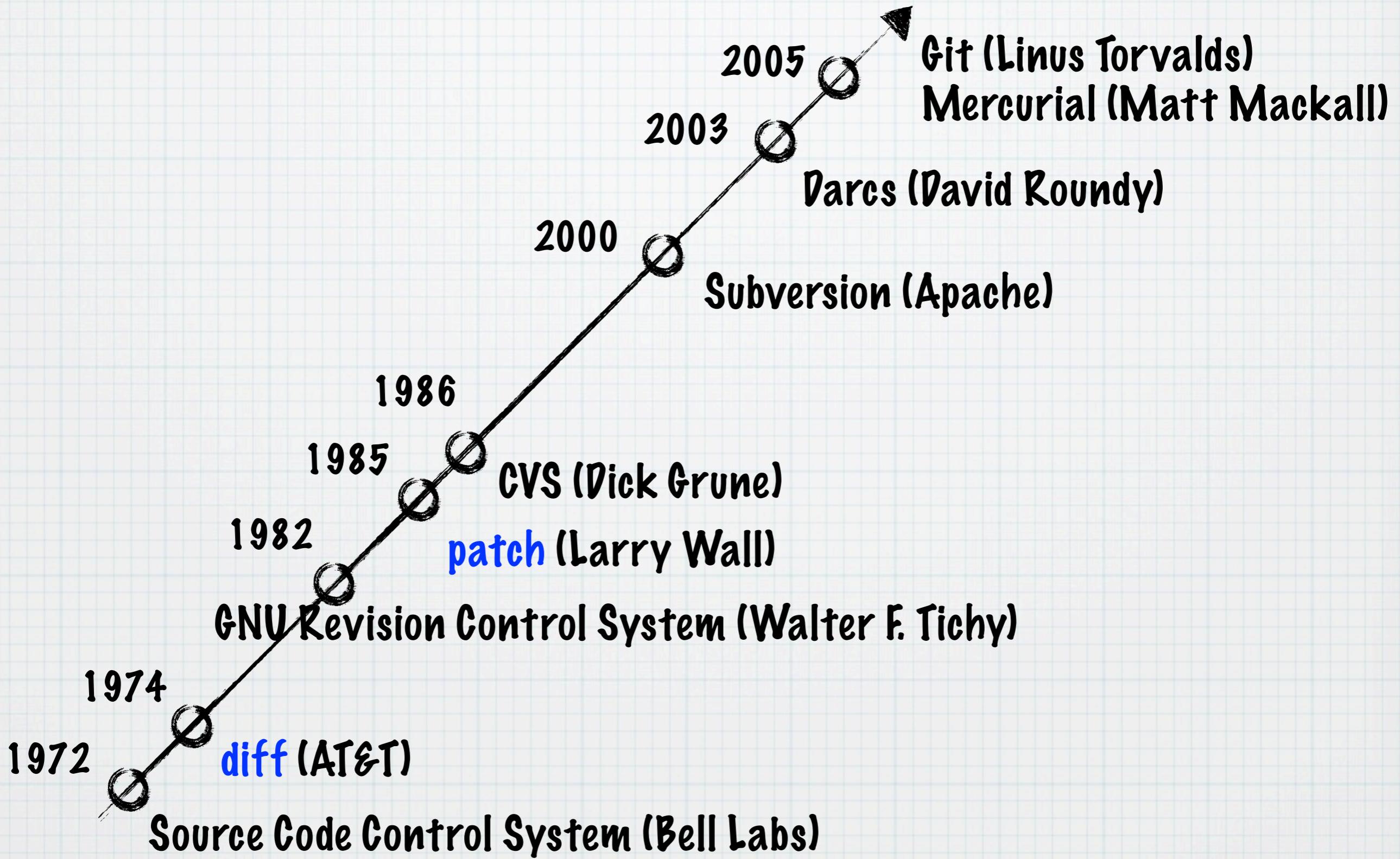
a cosa serve? (1)

- * Backup incrementale di modifiche
- * Gestione di versioni distinte di un software
(funzionalità introdotte a breve, medio, lungo termine; personalizzazioni)
- * “Time machine” (per passi di sviluppo)
- * Documentazione sorgenti più efficace dei commenti nel codice
- * Statistiche

a cosa serve? (2)

- * Lavoro contemporaneo sulla stessa risorsa
 - * lock dei file (profili di accesso, necessità di un responsabile...)
 - * strumenti di merge (automatici con risoluzione conflitti)
- * Risoluzione bug (che potrebbero essere di versione)

Storia



diff & patch

```
$ echo "prima riga" > test.txt
$ cp test.txt test2.txt
$ echo "seconda riga" >> test2.txt
$ diff -u test.txt test2.txt > test.diff
$ diff test.txt test2.txt
1a2
> seconda riga
$ patch << test.diff
>
$ patch < test.diff
patching file test.txt
$ diff test.txt test2.txt
$ cat test.diff
--- test.txt 2013-06-19 12:23:19.000000000 +0200
+++ test2.txt 2013-06-19 12:23:33.000000000 +0200
@@ -1 +1,2 @@
    prima riga
+seconda riga
```

definizioni

- * hunk (o change): una modifica atomica ad un file
- * commit (o record): un insieme di hunk che porta lo sviluppo da A a B
- * branch: un ramo di sviluppo che porta ad un obiettivo specifico (ad esempio una feature)

cantieri aperti

- * produzione (la versione in uso)
- * la correzione di un baco
- * il test per la futura versione di produzione
- * lo sviluppo di una o più funzionalità (alcune per la prossima versione in test, altre per milestone più lontane...)

flow

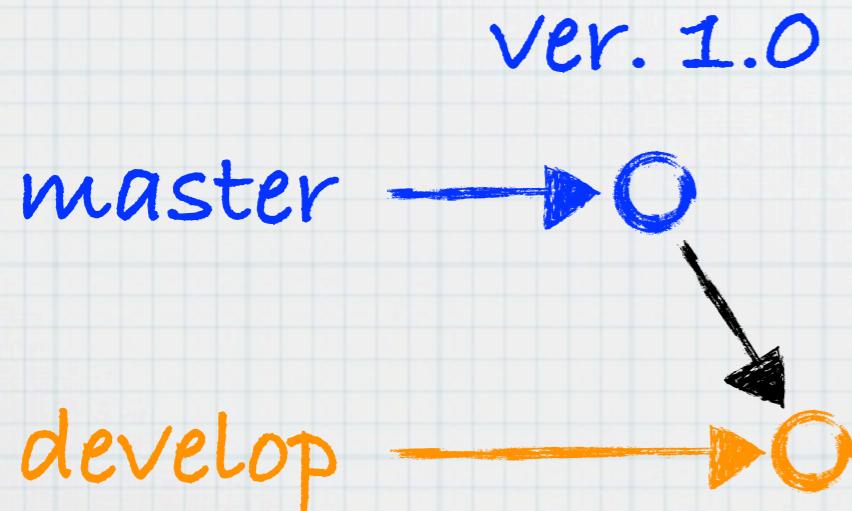
- * master: branch allineato con la versione in produzione
- * develop: branch allineato con la versione in sviluppo

ver. 1.0

master → O

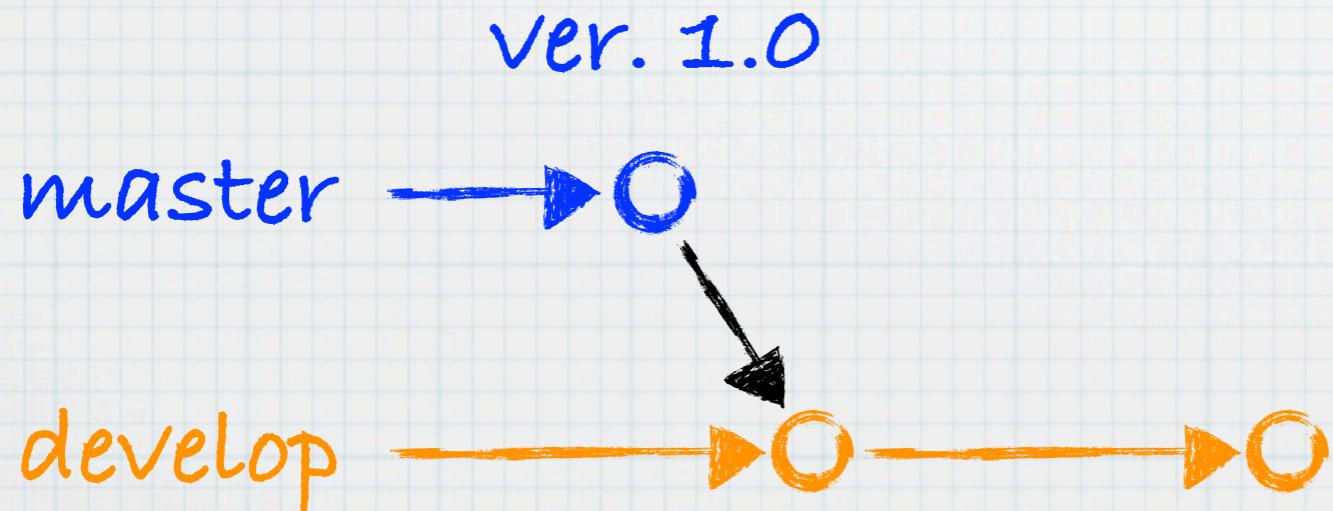
flow

- * master: branch allineato con la versione in produzione
- * develop: branch allineato con la versione in sviluppo



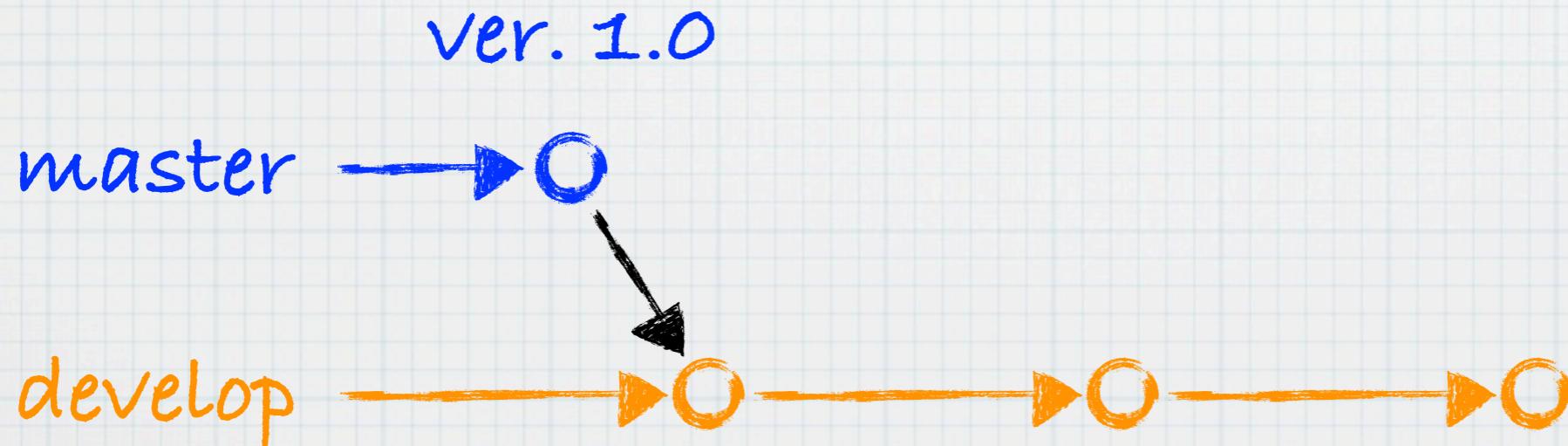
flow

- * master: branch allineato con la versione in produzione
- * develop: branch allineato con la versione in sviluppo



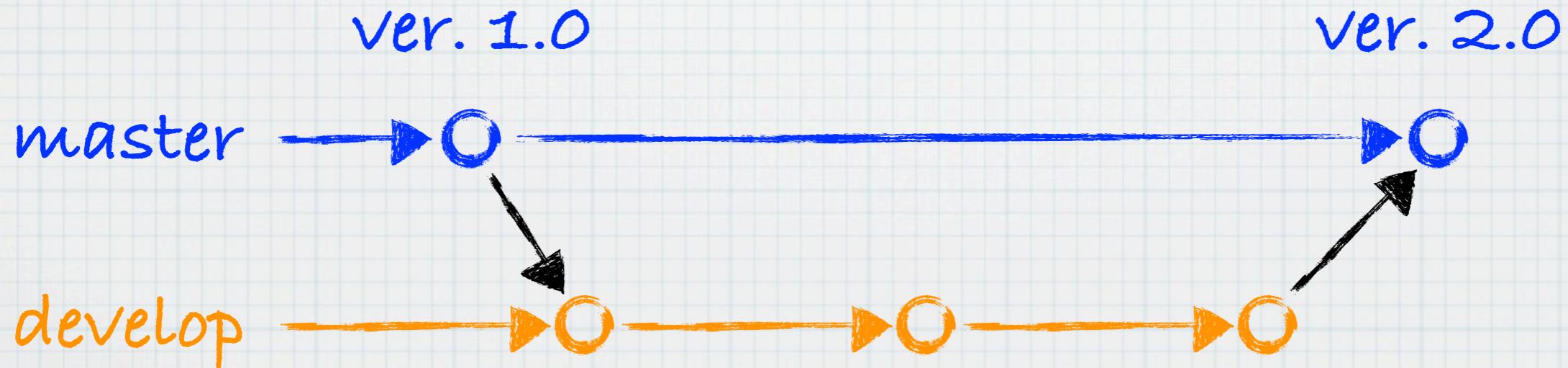
flow

- * master: branch allineato con la versione in produzione
- * develop: branch allineato con la versione in sviluppo



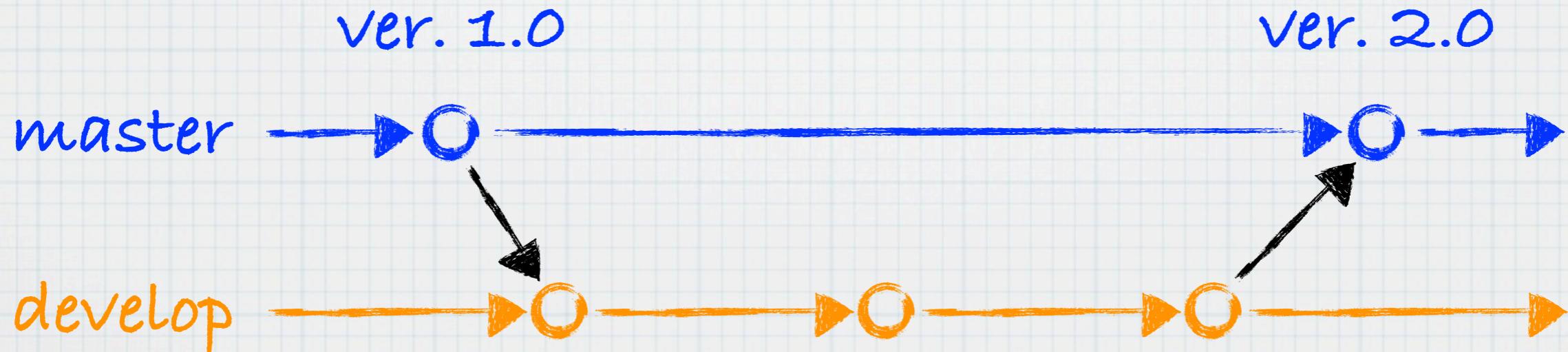
flow

- * master: branch allineato con la versione in produzione
- * develop: branch allineato con la versione in sviluppo



flow

- * master: branch allineato con la versione in produzione
- * develop: branch allineato con la versione in sviluppo



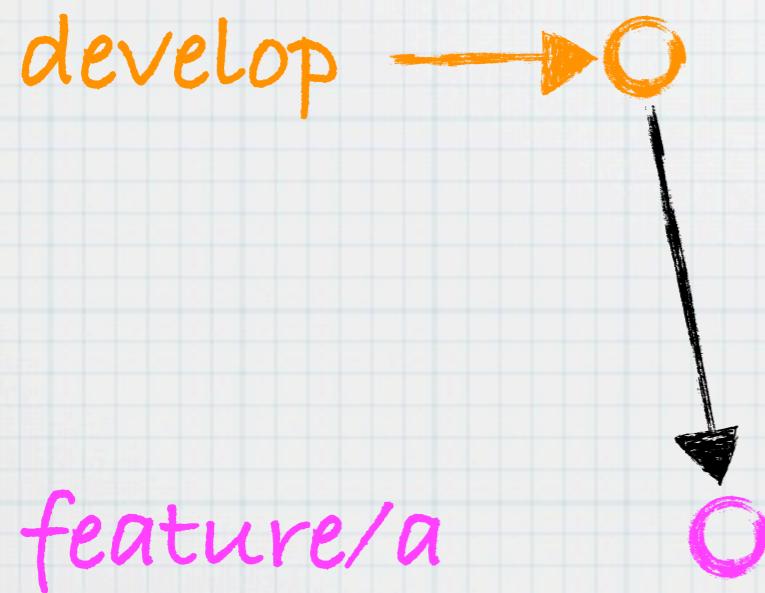
feature branch

- * branch di sviluppo di una nuova funzionalità
- * ramifica da develop e si innesta su develop
- * a fine sviluppo viene cancellato (merge o rebase)

develop →○

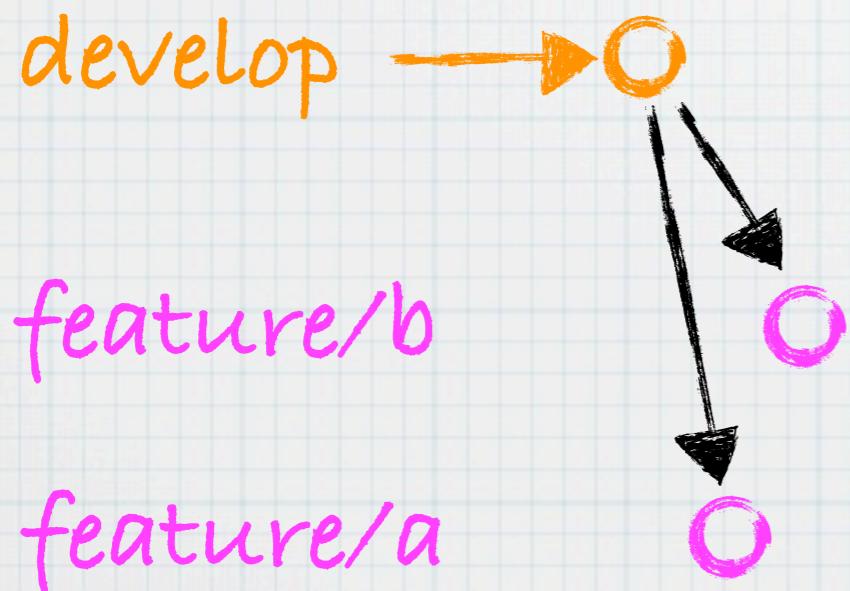
feature branch

- * branch di sviluppo di una nuova funzionalità
- * ramifica da develop e si innesta su develop
- * a fine sviluppo viene cancellato (merge o rebase)



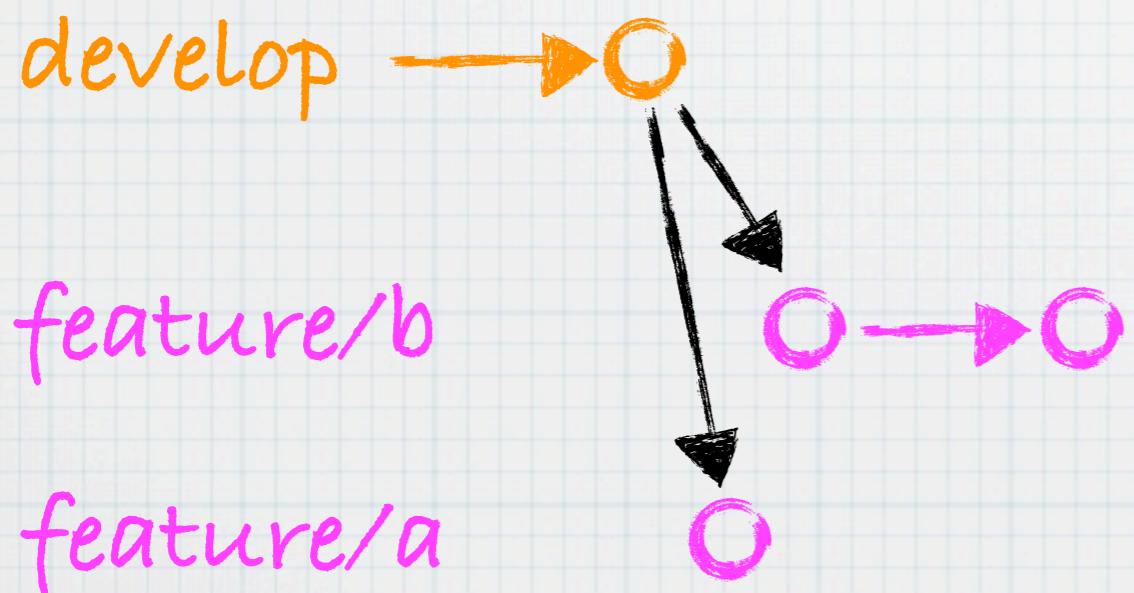
feature branch

- * branch di sviluppo di una nuova funzionalità
- * ramifica da develop e si innesta su develop
- * a fine sviluppo viene cancellato (merge o rebase)



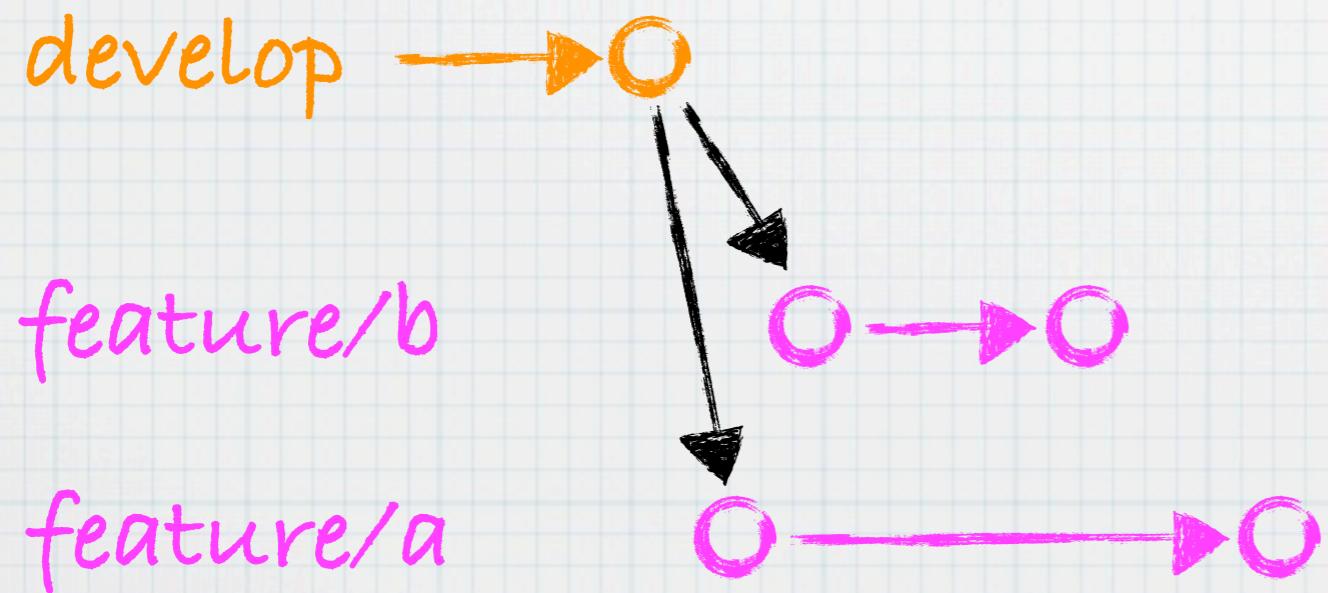
feature branch

- * branch di sviluppo di una nuova funzionalità
- * ramifica da develop e si innesta su develop
- * a fine sviluppo viene cancellato (merge o rebase)



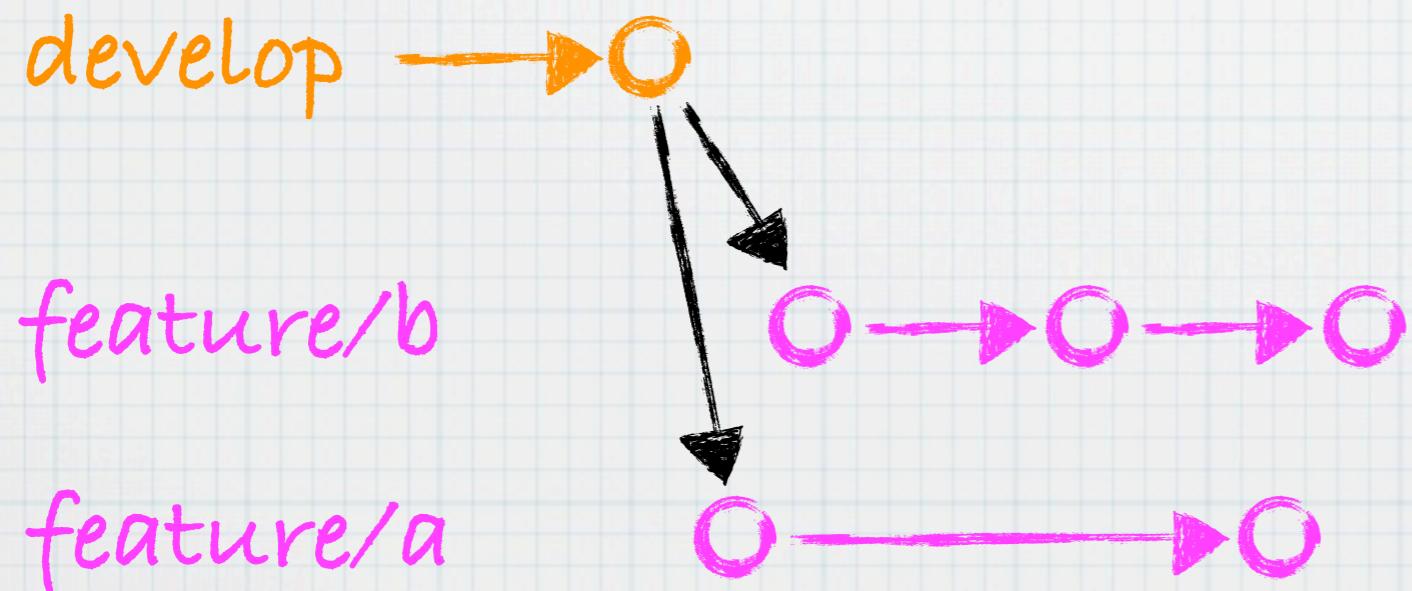
feature branch

- * branch di sviluppo di una nuova funzionalità
- * ramifica da develop e si innesta su develop
- * a fine sviluppo viene cancellato (merge o rebase)



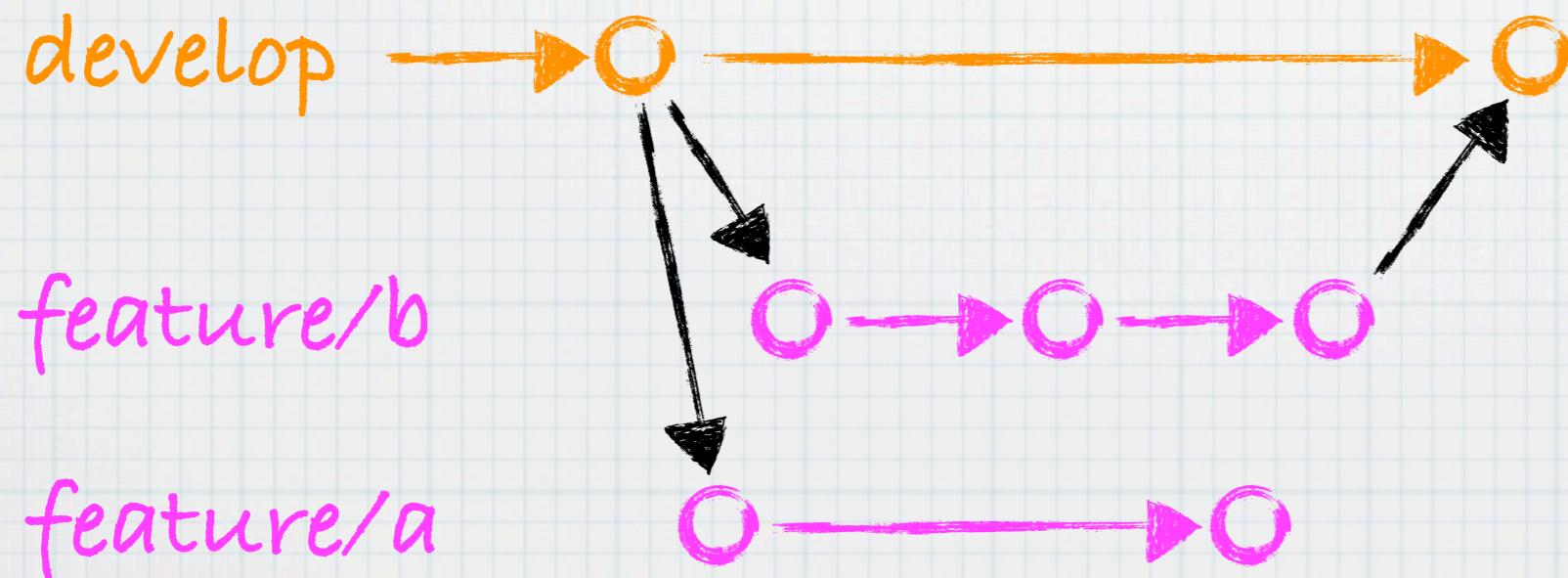
feature branch

- * branch di sviluppo di una nuova funzionalità
- * ramifica da develop e si innesta su develop
- * a fine sviluppo viene cancellato (merge o rebase)



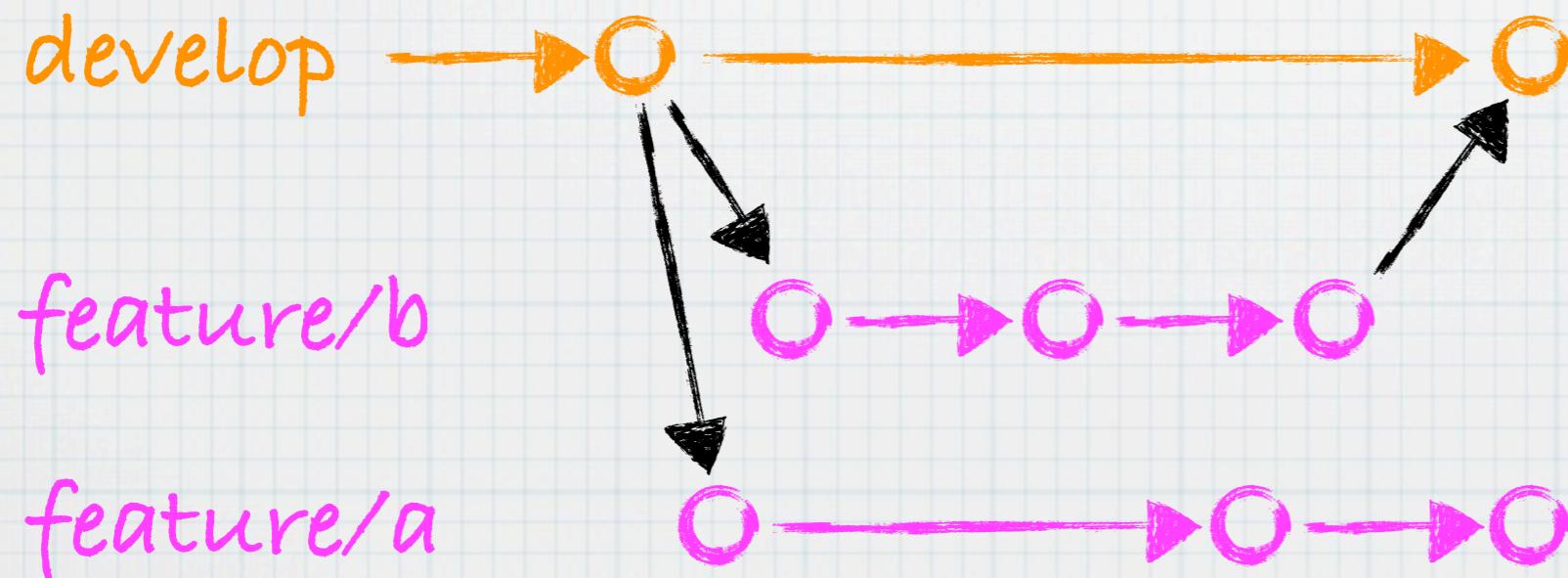
feature branch

- * branch di sviluppo di una nuova funzionalità
- * ramifica da develop e si innesta su develop
- * a fine sviluppo viene cancellato (merge o rebase)



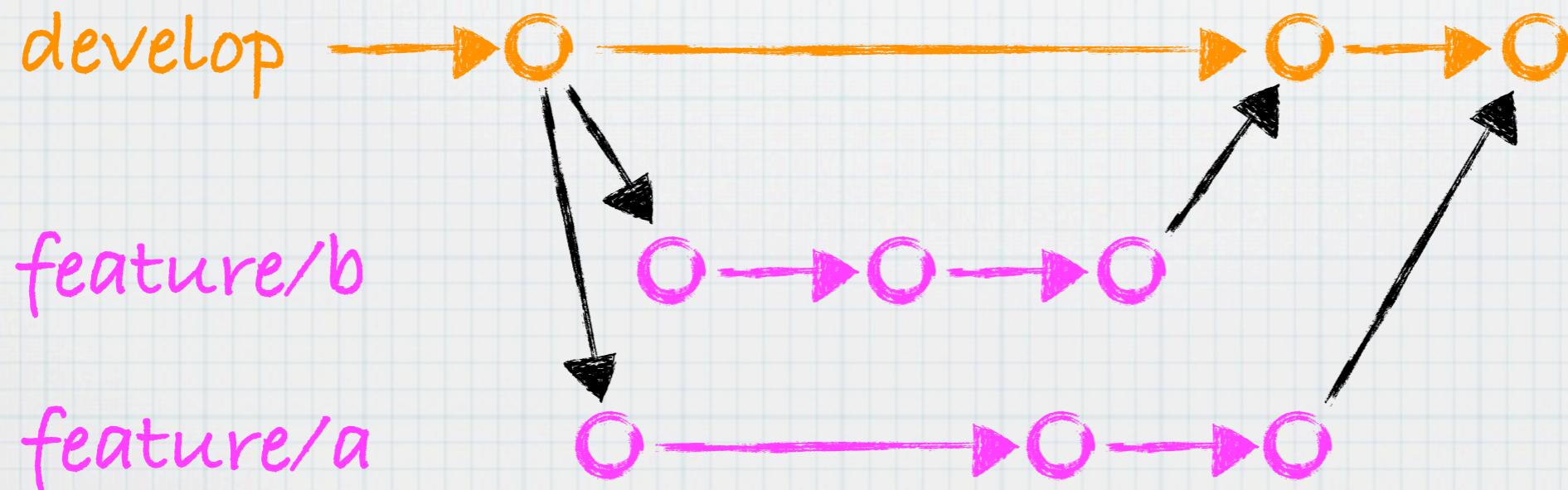
feature branch

- * branch di sviluppo di una nuova funzionalità
- * ramifica da develop e si innesta su develop
- * a fine sviluppo viene cancellato (merge o rebase)



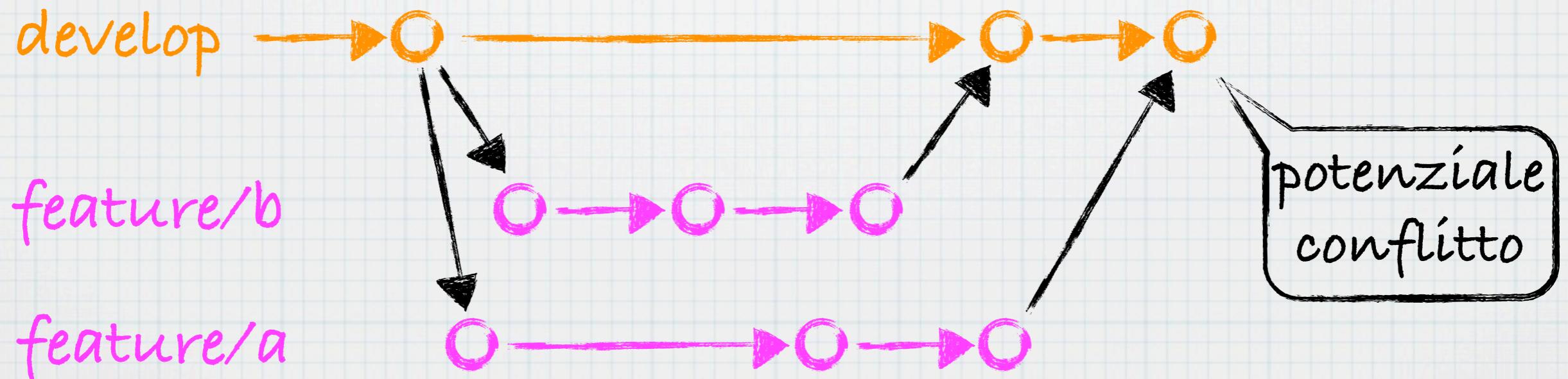
feature branch

- * branch di sviluppo di una nuova funzionalità
- * ramifica da develop e si innesta su develop
- * a fine sviluppo viene cancellato (merge o rebase)



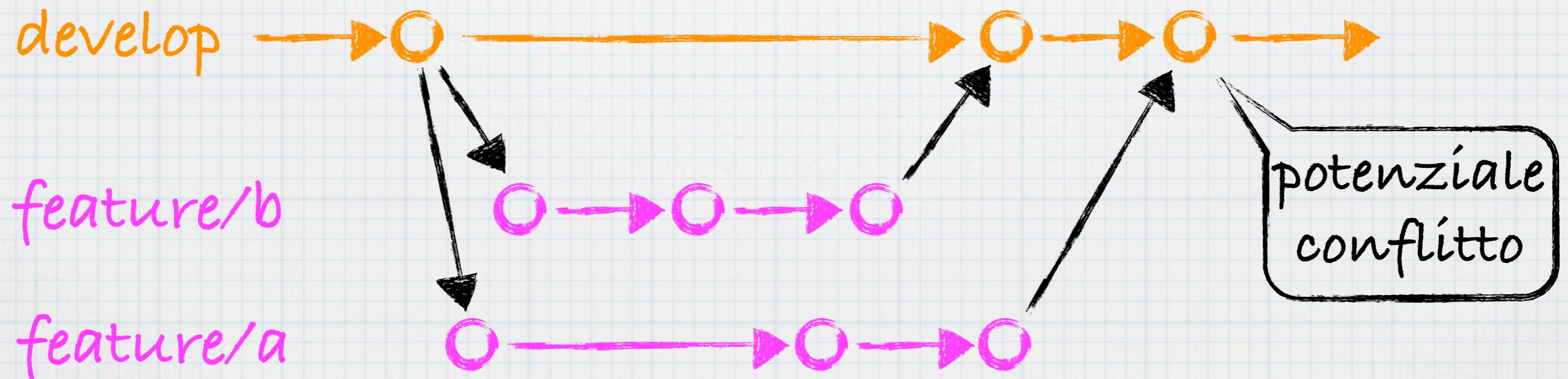
feature branch

- * branch di sviluppo di una nuova funzionalità
- * ramifica da develop e si innesta su develop
- * a fine sviluppo viene cancellato (merge o rebase)



feature branch

- * branch di sviluppo di una nuova funzionalità
- * ramifica da develop e si innesta su develop
- * a fine sviluppo viene cancellato (merge o rebase)



hotfix branch

- * branch di fix per un bug in produzione
- * ramifica da master (produzione) e si innesta su master
- * a fine sviluppo viene cancellato (può essere necessario un merge su develop)

ver. 2.0
master → O

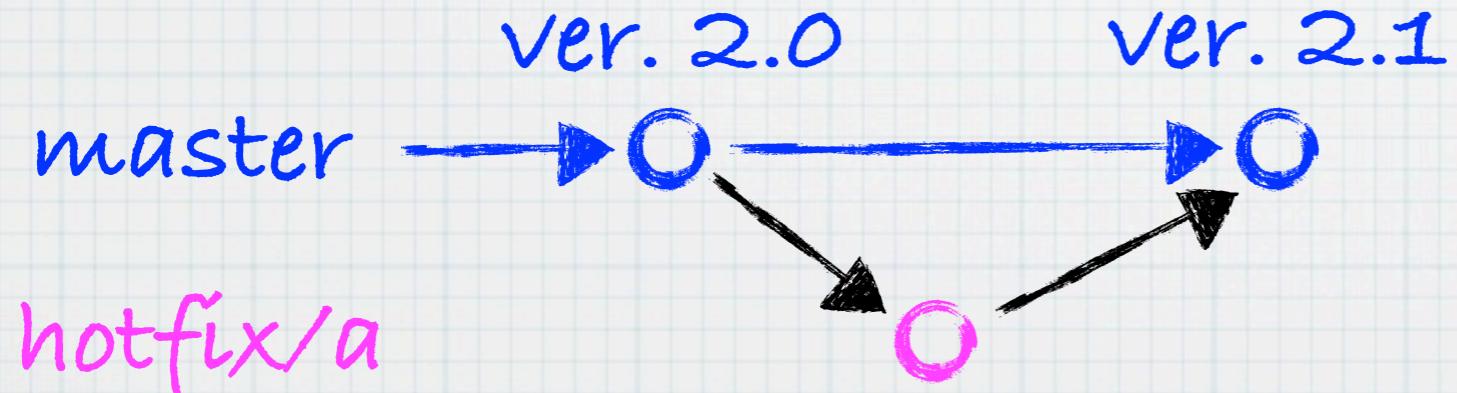
hotfix branch

- * branch di fix per un bug in produzione
- * ramifica da master (produzione) e si innesta su master
- * a fine sviluppo viene cancellato (può essere necessario un merge su develop)



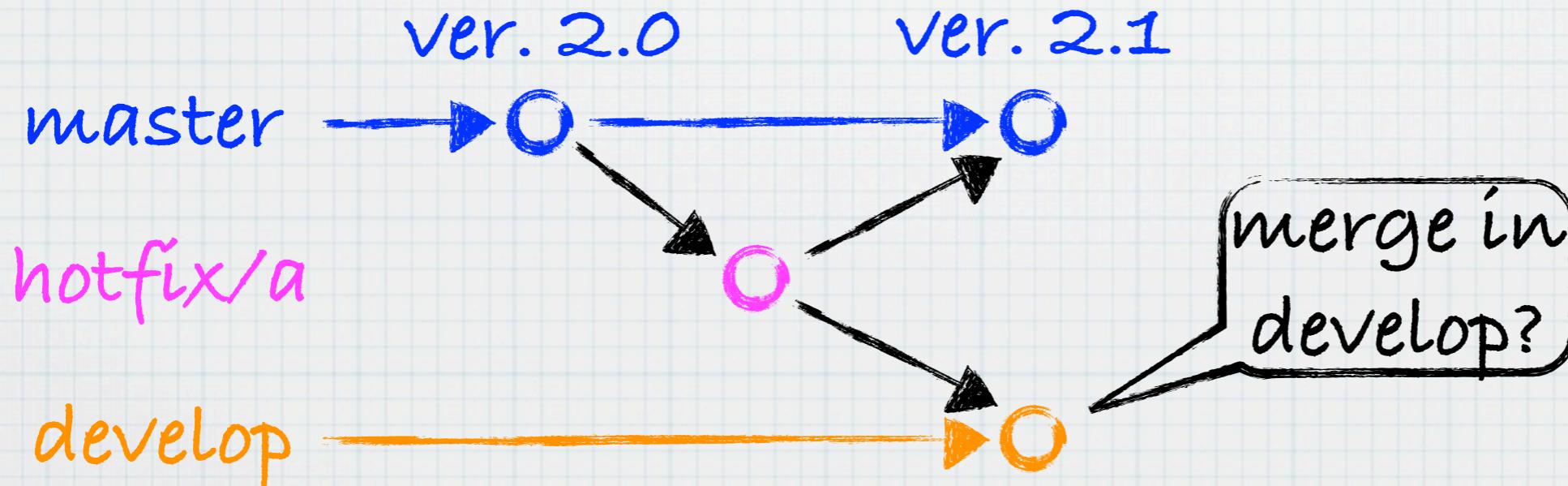
hotfix branch

- * branch di fix per un bug in produzione
- * ramifica da master (produzione) e si innesta su master
- * a fine sviluppo viene cancellato (può essere necessario un merge su develop)



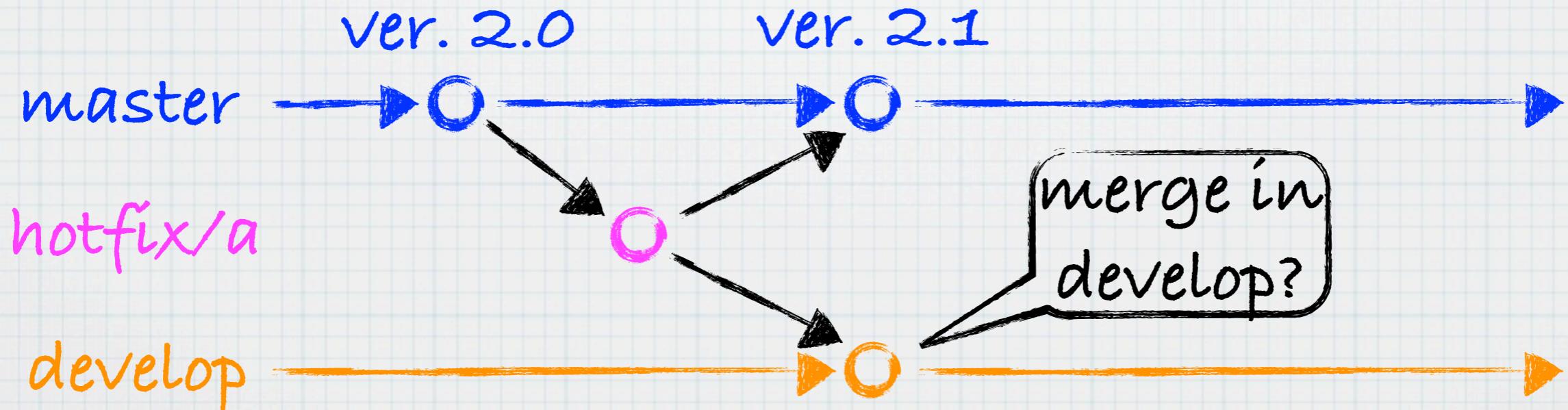
hotfix branch

- * branch di fix per un bug in produzione
- * ramifica da master (produzione) e si innesta su master
- * a fine sviluppo viene cancellato (può essere necessario un merge su develop)



hotfix branch

- * branch di fix per un bug in produzione
- * ramifica da master (produzione) e si innesta su master
- * a fine sviluppo viene cancellato (può essere necessario un merge su develop)



release branch

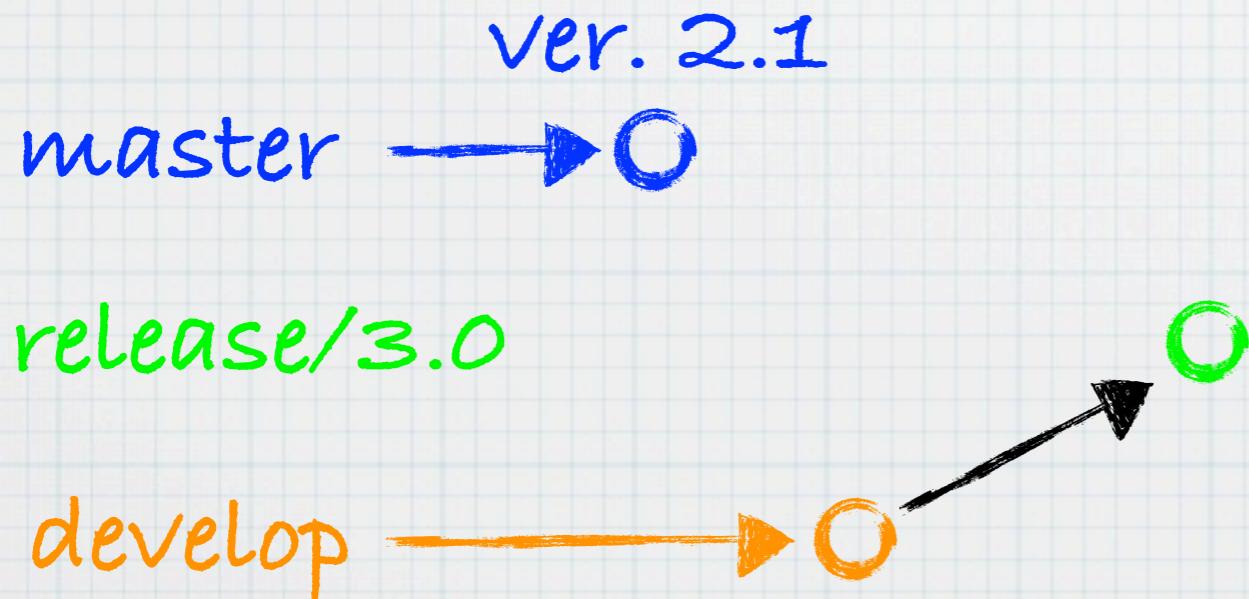
- * candidata in test per la prossima versione di produzione
- * ramifica da develop e si innesta su master
- * viene generalmente “taggata” con un numero di versione

ver. 2.1
master → O

develop → O

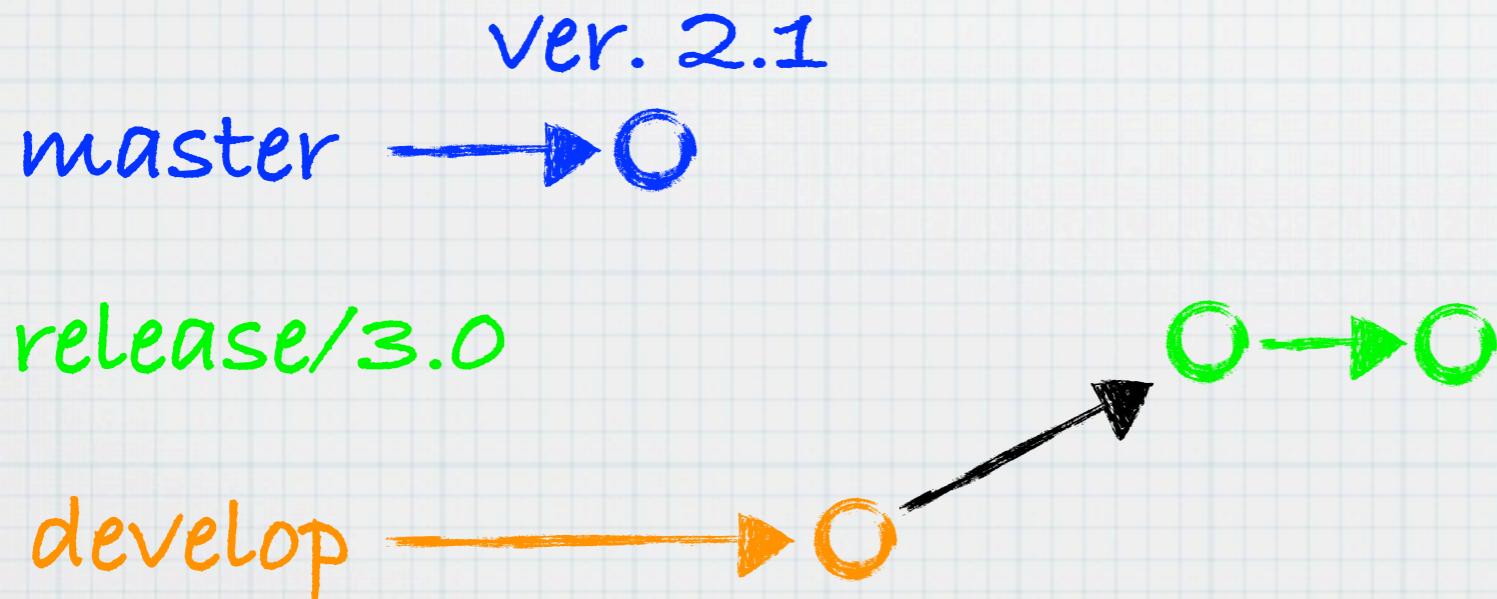
release branch

- * candidata in test per la prossima versione di produzione
- * ramifica da develop e si innesta su master
- * viene generalmente “taggata” con un numero di versione



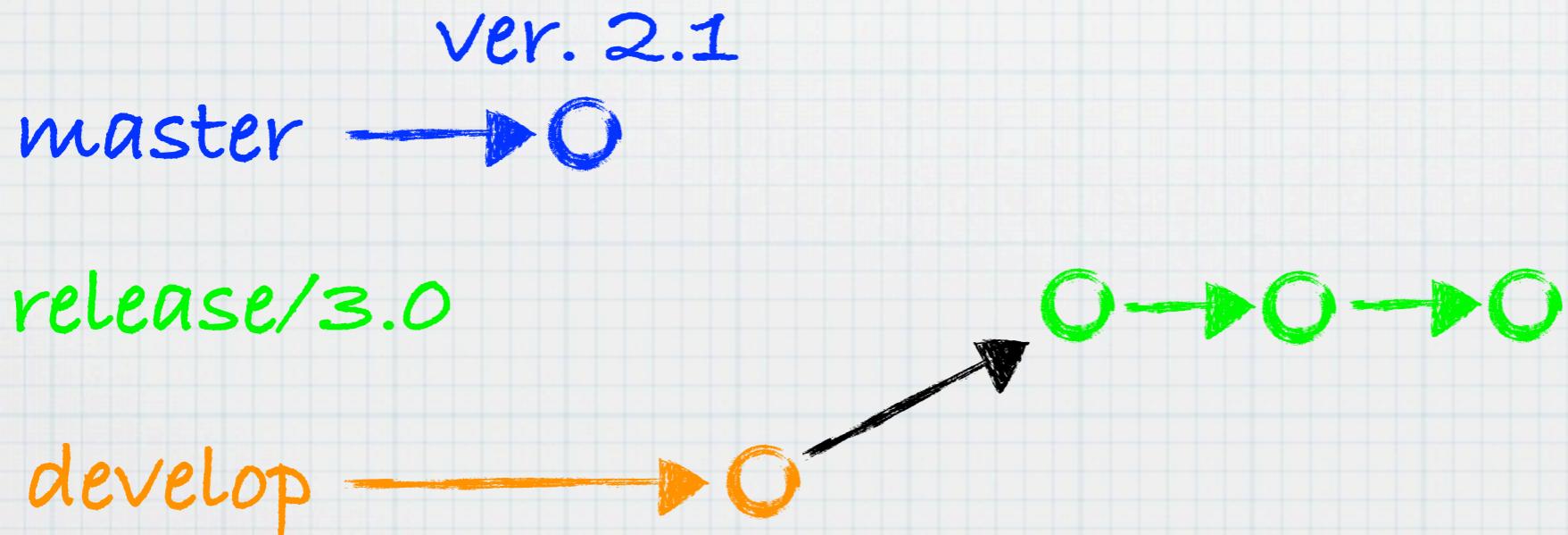
release branch

- * candidata in test per la prossima versione di produzione
- * ramifica da develop e si innesta su master
- * viene generalmente “taggata” con un numero di versione



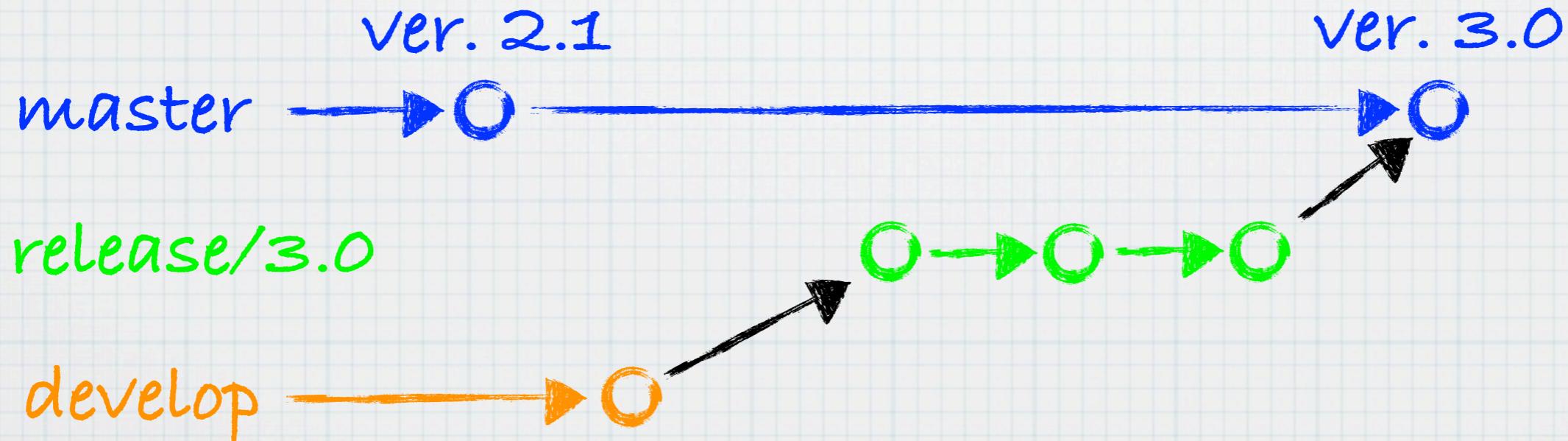
release branch

- * candidata in test per la prossima versione di produzione
- * ramifica da develop e si innesta su master
- * viene generalmente “taggata” con un numero di versione



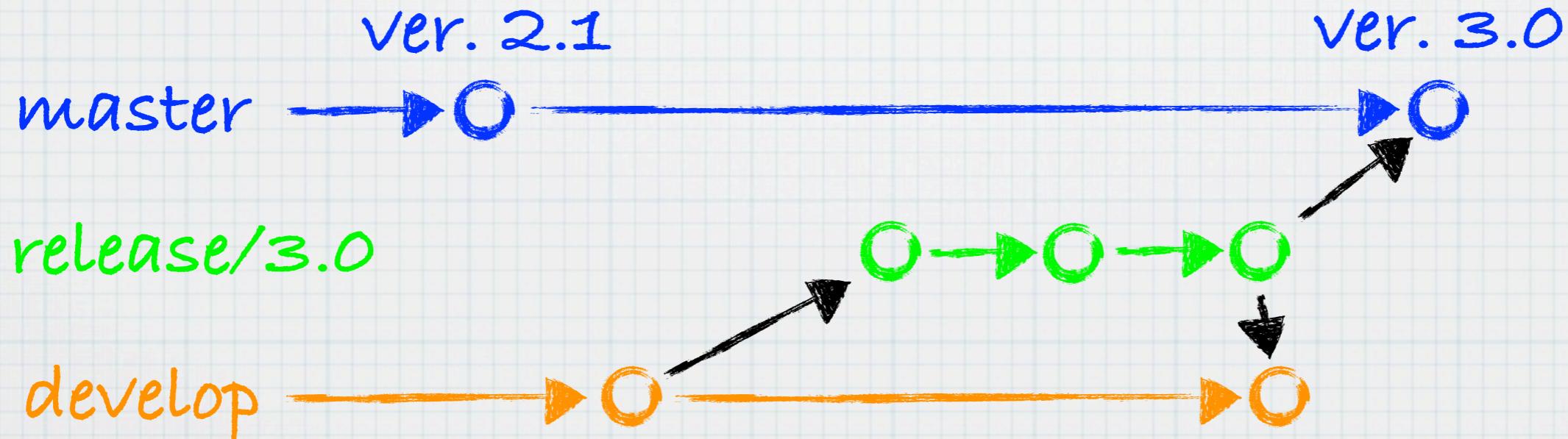
release branch

- * candidata in test per la prossima versione di produzione
- * ramifica da develop e si innesta su master
- * viene generalmente “taggata” con un numero di versione



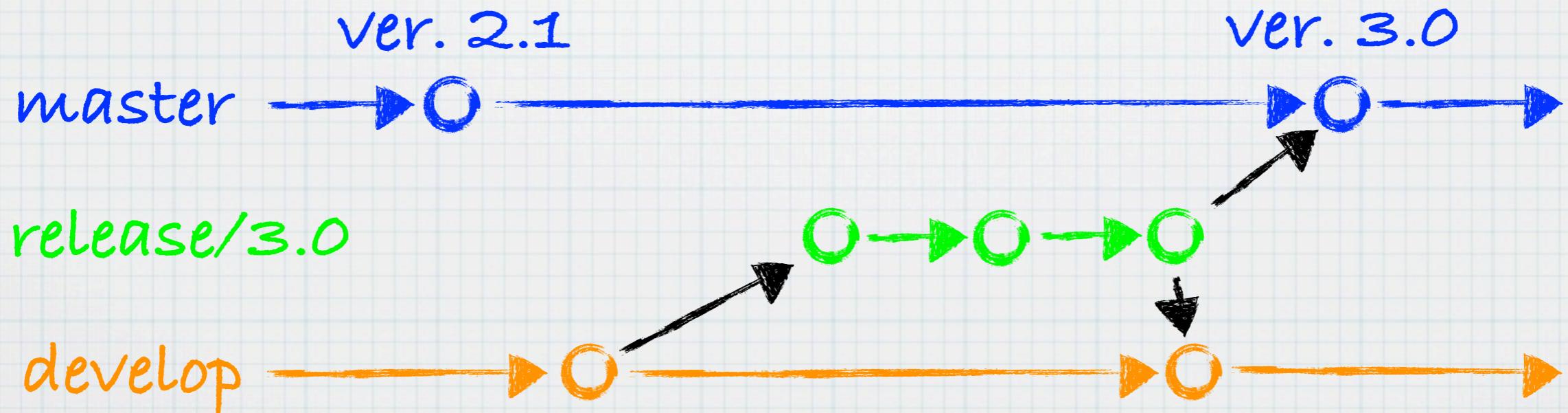
release branch

- * candidata in test per la prossima versione di produzione
- * ramifica da develop e si innesta su master
- * viene generalmente “taggata” con un numero di versione



release branch

- * candidata in test per la prossima versione di produzione
- * ramifica da develop e si innesta su master
- * viene generalmente “taggata” con un numero di versione



buona creanza

- * pull, (risolvi i conflitti), push
- * commit frequenti (ma “significativi”, potrebbero andare in changelog)
- * test prima di push
- * regole di stesura dei commenti (es. niente nomi di file, forma dichiarativa, etc)
- * definizione delle regole di flow:
 - * utilizzo dei branch
 - * tempi di vita dei branch
 - * responsabili dei branch principali

fine

Thank you.
Tiziano Lattisi

Me: <http://www.linkedin.com/in/tizianolattisi/>

Slides: <http://www.slideshare.net/lattisi/>

Spunti e approfondimenti

<http://leif.me/papers/Singer2012a.pdf>

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