Controllo di versione

2015

Sviluppo collaborativo

Organizzazione del processo di sviluppo

- Per realizzare un buon prodotto software non è sufficiente scrivere il codice, ma è altresì importante la gestione del processo di sviluppo.
- Organizzare un processo software vuol dire occuparsi di diverse problematiche:
  - Comunicazione tra i partecipanti
  - Gestione delle attività
  - **Gestione del codice sorgente**
    - Controllo di versione
    - Semplificazione della collaborazione
    - Gestione di diverse varianti (branch) del progetto

Sistemi di controllo di versione

Centralizzati

Sistemi di controllo di versione

- Locali
- Centralizzati
  - SVN
  - CVS

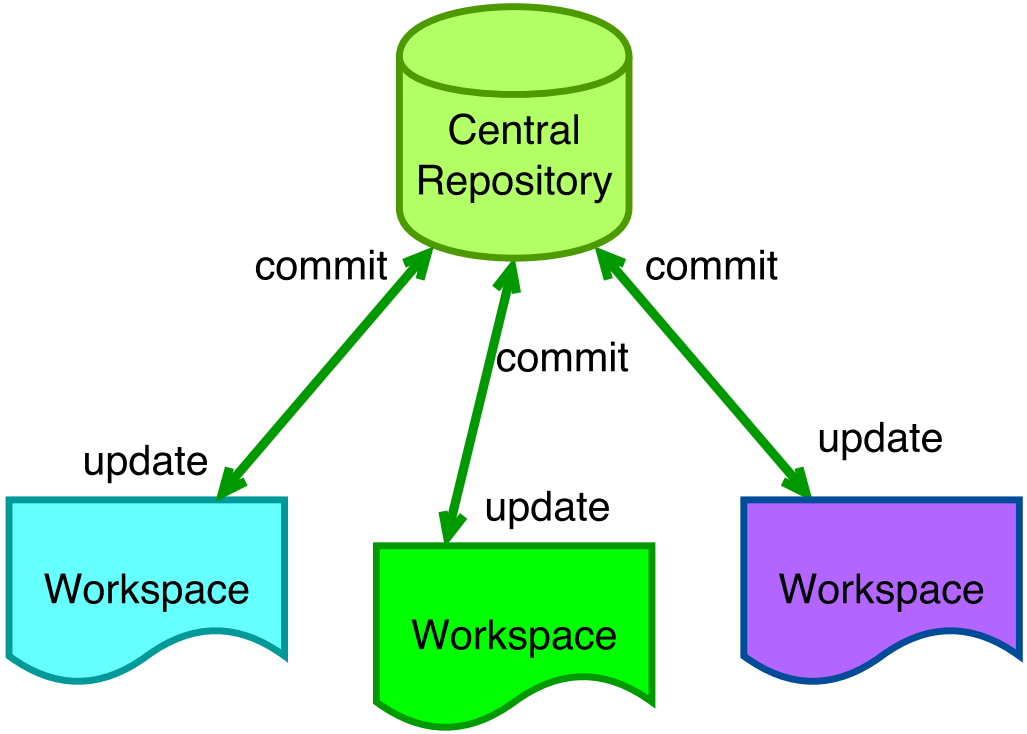


Figura: Sistemi di controllo di versione centralizzati

Sistemi di controllo di versione

Distribuiti

Sistemi di controllo di versione

- Distribuiti
  - **Git**
  - Mercurial
  - Bazaar

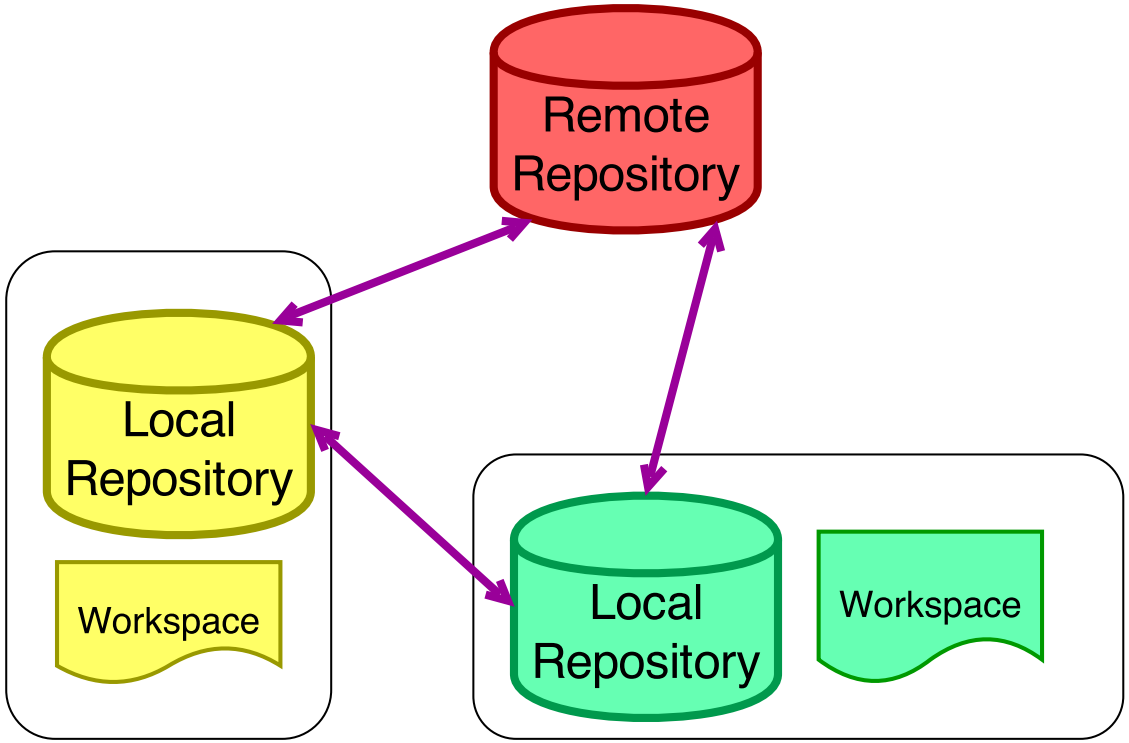


Figura: Sistemi di controllo di versione distribuiti

Git

Creato da **Linus Torvalds** per fornire supporto al processo di sviluppo del *kernel Linux*

Caratteristiche

- Sviluppo distribuito
- Orientato allo sviluppo non lineare
- Flessibilità ed efficienza, soprattutto per team numerosi

Git

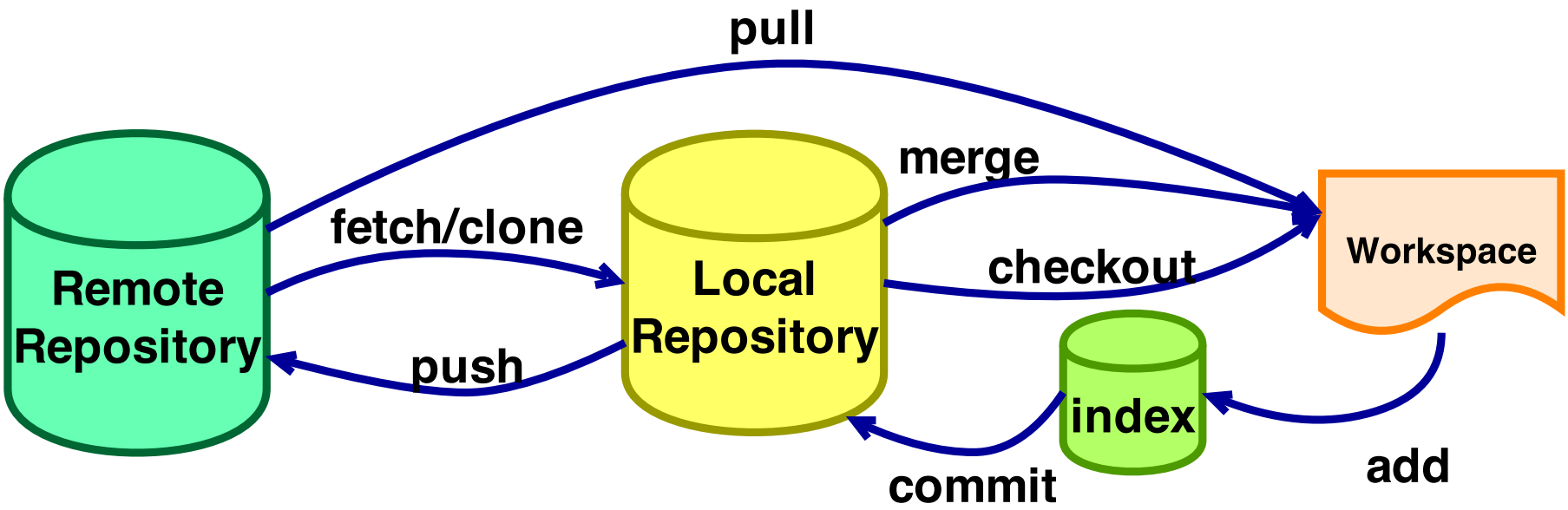
installazione

- INSTALLAZIONE
  - **Debian** `apt-get install git`
  - **Arch** `pacman -S git`
  - **Fedora** `yum install git`
  - **Gentoo** `emerge -ask -verbose dev-vcs/git`
  - **Windows** `http://msysgit.github.io/`

EGit

Client Git per *Eclipse IDE* é basato su libreria *JGit*.

Configurazione tipica



Workspace, Index, Repository

Working directory

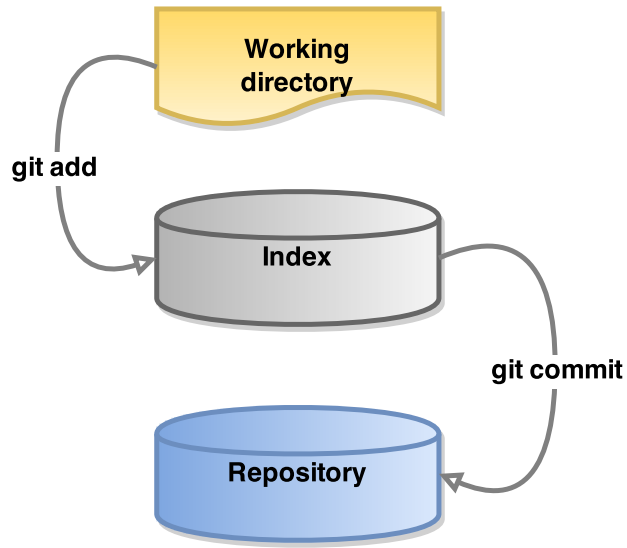
directory con i file

Index(Staging Area)

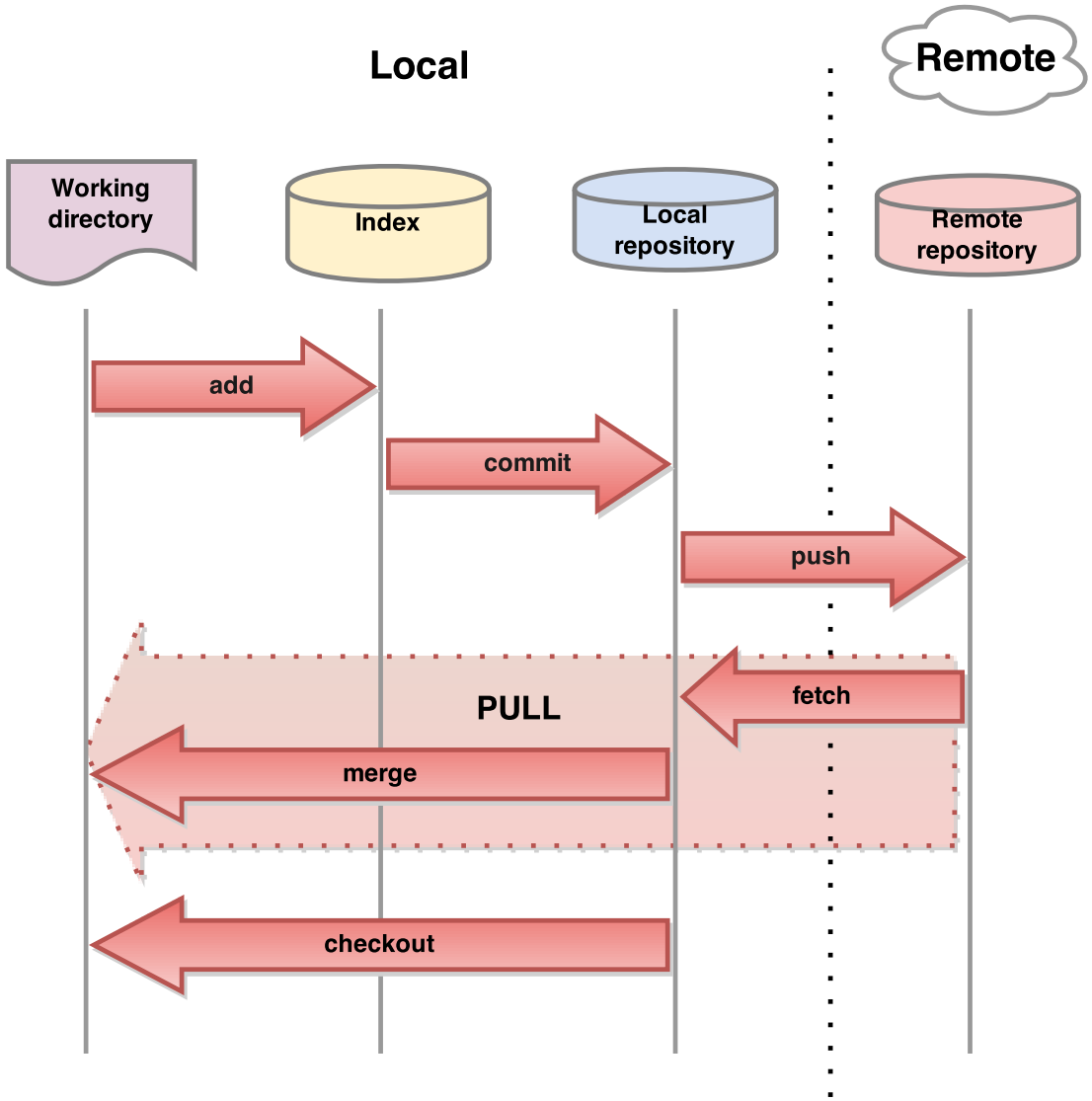
spazio per creare il `commit`
successivo prima di registrarlo
nel repository

Repository

database su file che conserva
i vari commit



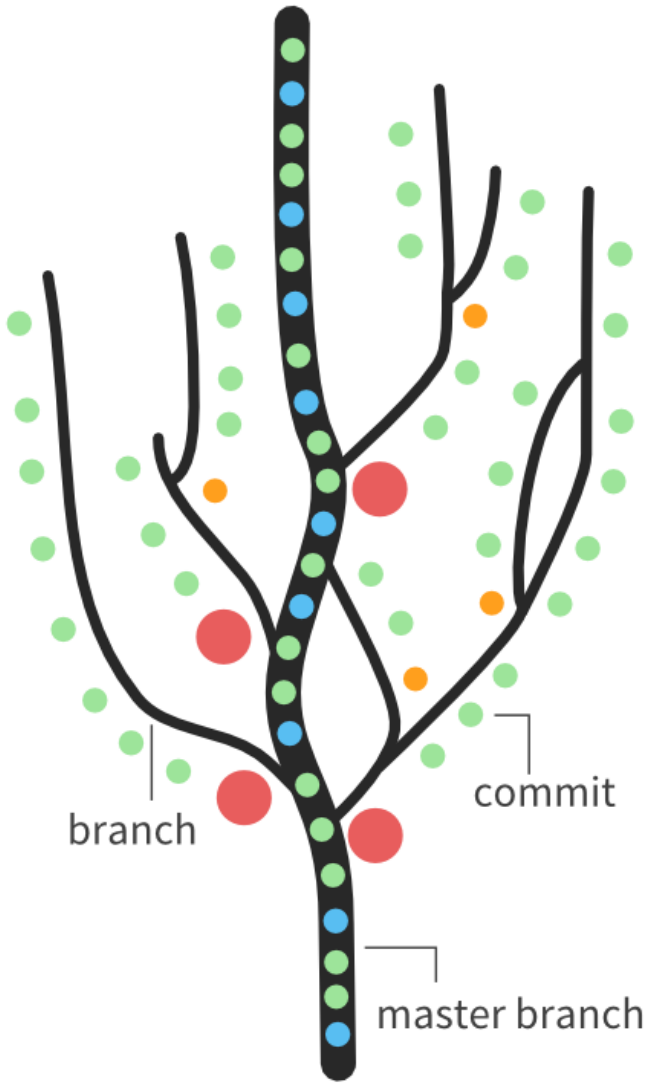
Esempio d'utilizzo



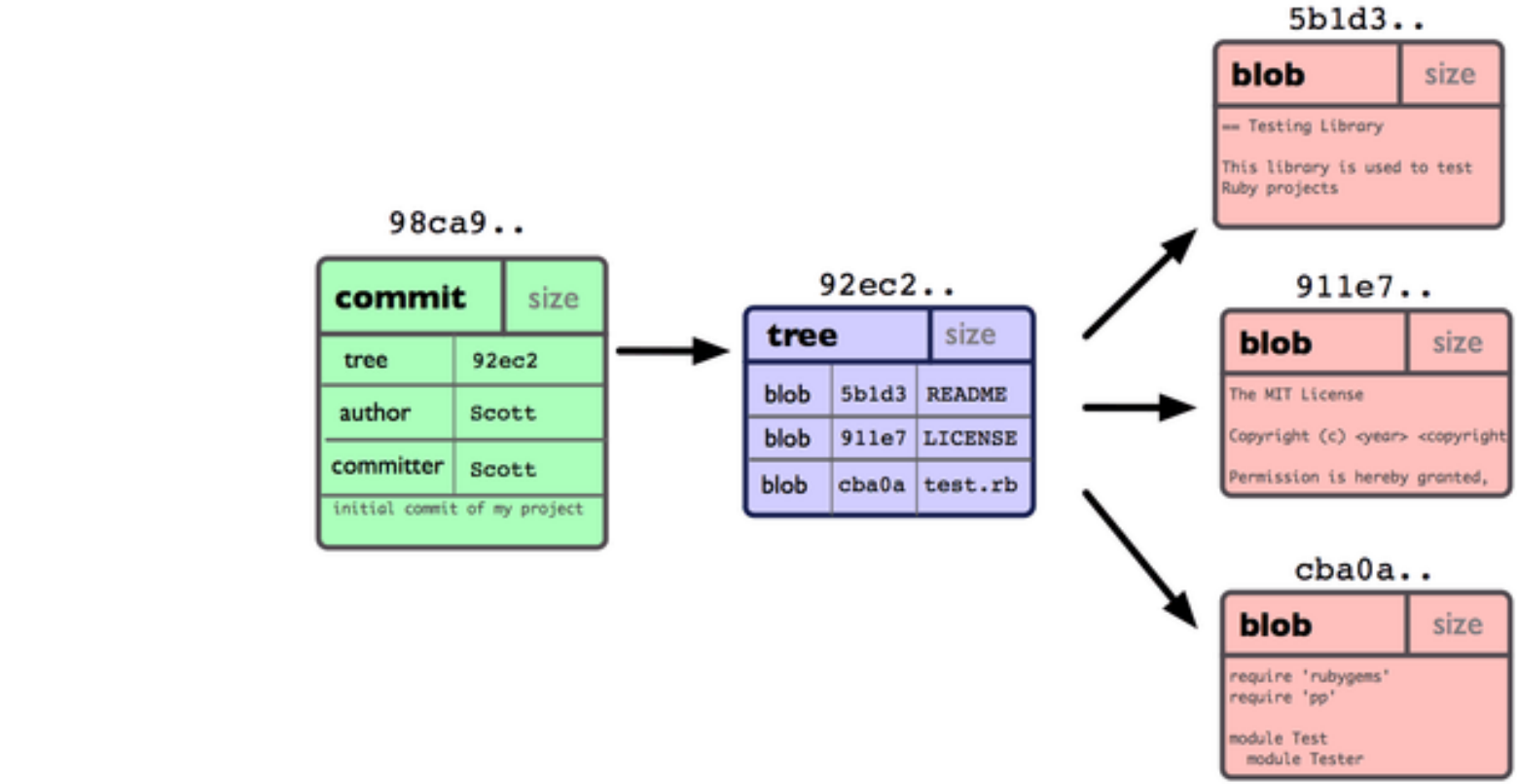
Branch

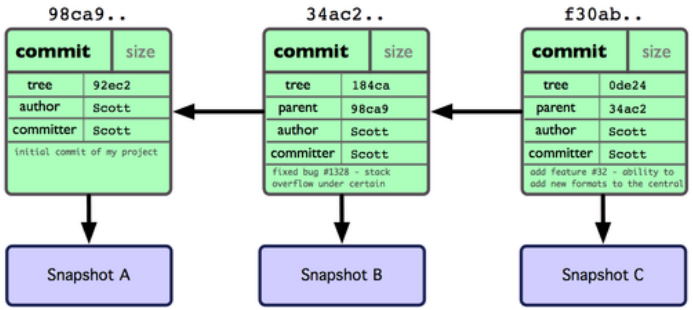
“killer feature“

- Permette di lavorare su divese versioni “in parallelo”
- Funzionalitá piú potente di Git che lo differenzia dagli altri SCM.
- Git permette di avere molteplici branch completamente indipendenti.
- Le operazioni di creazione, unione e cancellazione sono veloci e facili da eseguire.
  - branch che contiene tutte le versioni stabili
  - branch per sviluppare una nuova funzionalitá
  - branch per testare
- Non é obbligatorio condividere tutti i branch con il repository remoto.
- E’ possibile associare ogni branch locale ai branch dei diversi repository remoti.

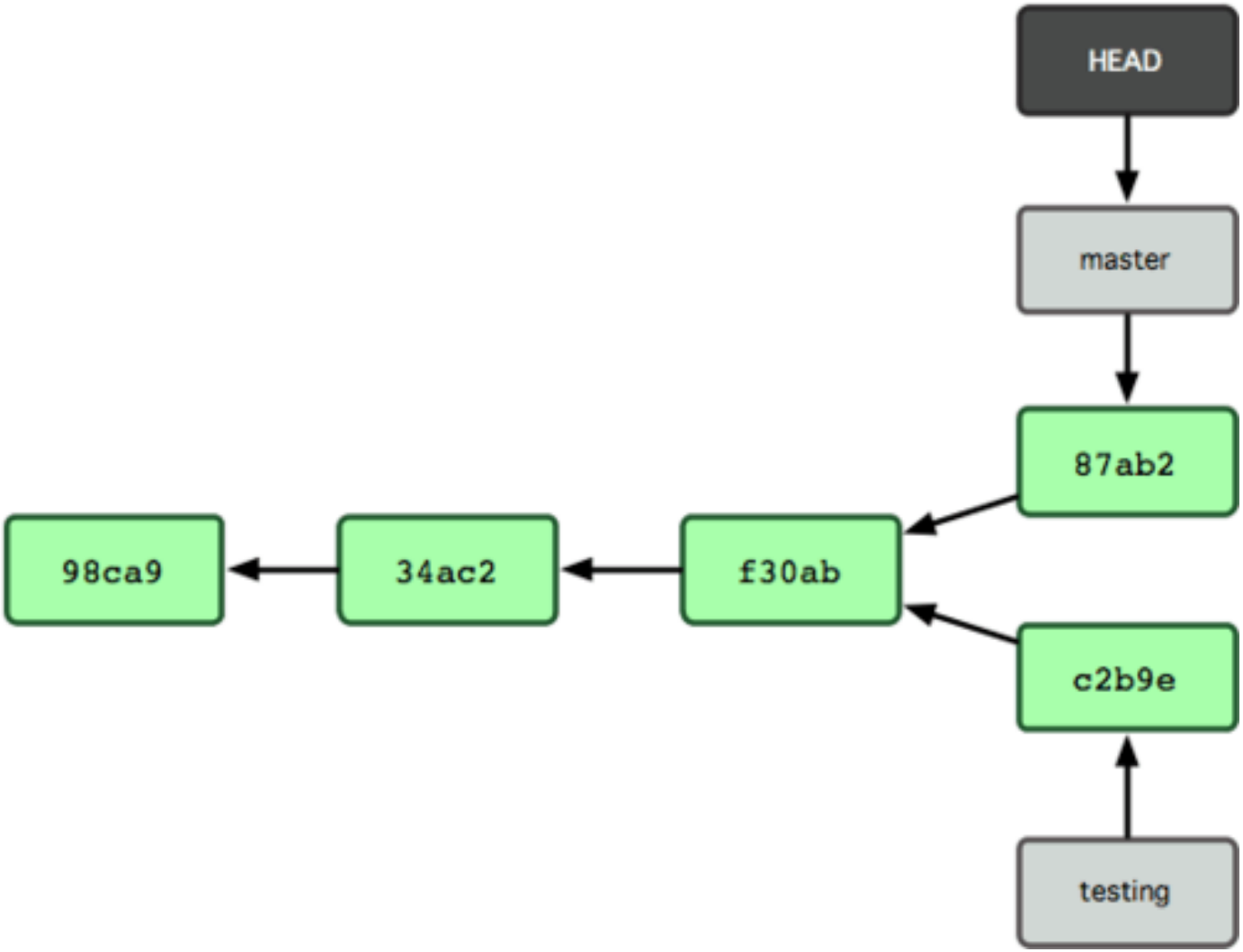


Modello di Dominio





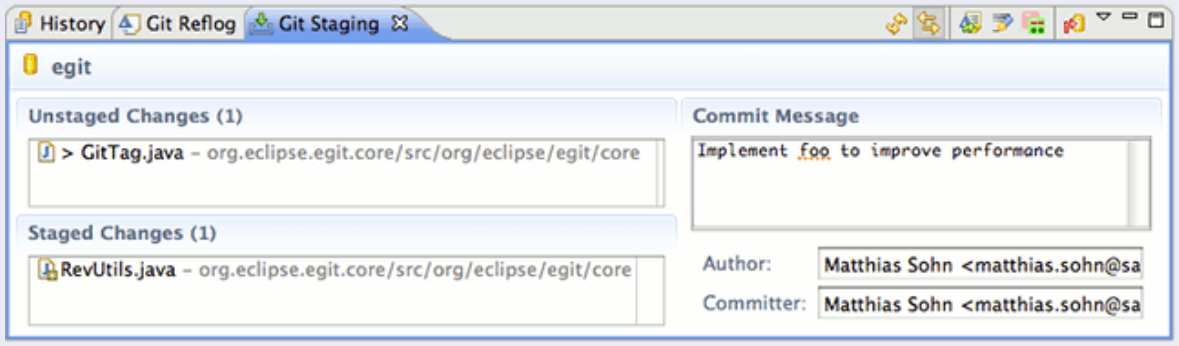
Branch



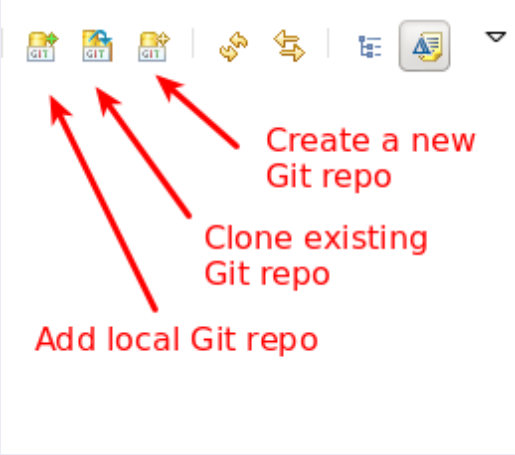
Eclipse Git View

`Window -> Show View -> Other...  -> Git ->`

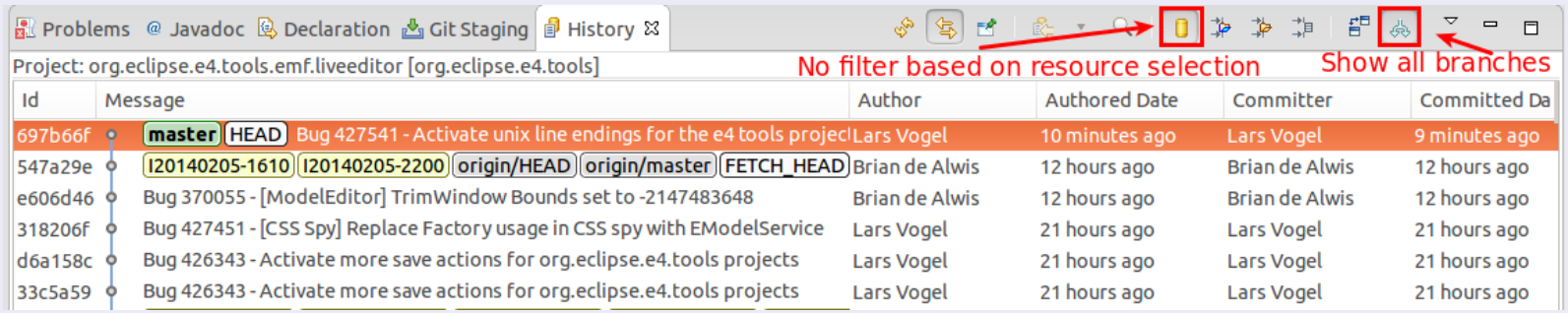
Git Staging



Git Repositories



Git History



Repository Git sul server del corso

- Gitolite
  « git hosting on a central server, with fine-grained access control and many more powerful features.»
  `https://github.com/sitaramc/gitolite`

- Accessibile con il protocollo SSH basato sulla crittografia asimmetrica, utilizzando una coppia di chiavi.

- Indirizzi
  `ssh://isslm2014@137.204.107.21:80/<repo>`
  `ssh://isslm2014@infolab.ingce.unibo.it:80/<repo>`

Best Practice

Raccomandazioni per i commit “puliti”

- Ogni commit deve avere un autore - Nome Cognome, email dell’università.
- Il testo dei commit deve contenere la descrizione breve dei cambiamenti.
- Il commit non deve contenere i file che non corrispondono ai cambiamenti, per esempio i file che vengono creati da strumenti esterni. I file “indesiderati” possono essere aggiunti in *.gitignore*.
- Il commit non deve aggiungere o rimuovere le righe vuote o gli spazi, ad esclusione dei casi che corrispondono alla logica di commit (refactoring, pulizia del codice).
- Lo stile dei cambiamenti deve essere mantenuto uniforme. Per esempio se per l’allineamento del codice vengono utilizzati i tab, il codice aggiuntivo deve seguire lo stesso modello.
- Il codice deve essere scritto in modo da minimizzare i conflitti dovuti a eventuali modifiche.

Demo Eclipse, EGit

Esercizio

Lavoro in gruppo attraverso server centrale

- Impostazione dell'ambiente di lavoro
- Importazione dei contenuti dal repository remoto
- Invio/recensione delle modifiche dal repository remoto del gruppo
- Risoluzione dei conflitti

Link utili

- http://git-scm.com/book
- http://wiki.eclipse.org/EGit/User_Guide
- http://pcottle.github.io/learnGitBranching/
- http://get-git.rtfd.org
- http://vogella.com/tutorials/Git/article.html
- http://vogella.com/tutorials/EclipseGit/article.html
- http://githowto.com/