Sistemi di gestione del codice sorgente

2013

Sviluppo collaborativo

Organizzazione di 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 diramazioni (branch) di sviluppo

Sistemi di controllo di versione

Centralizzati

Sistemi di controllo di versione

- Locali
- Centralizzati
  - SVN
  - CVS

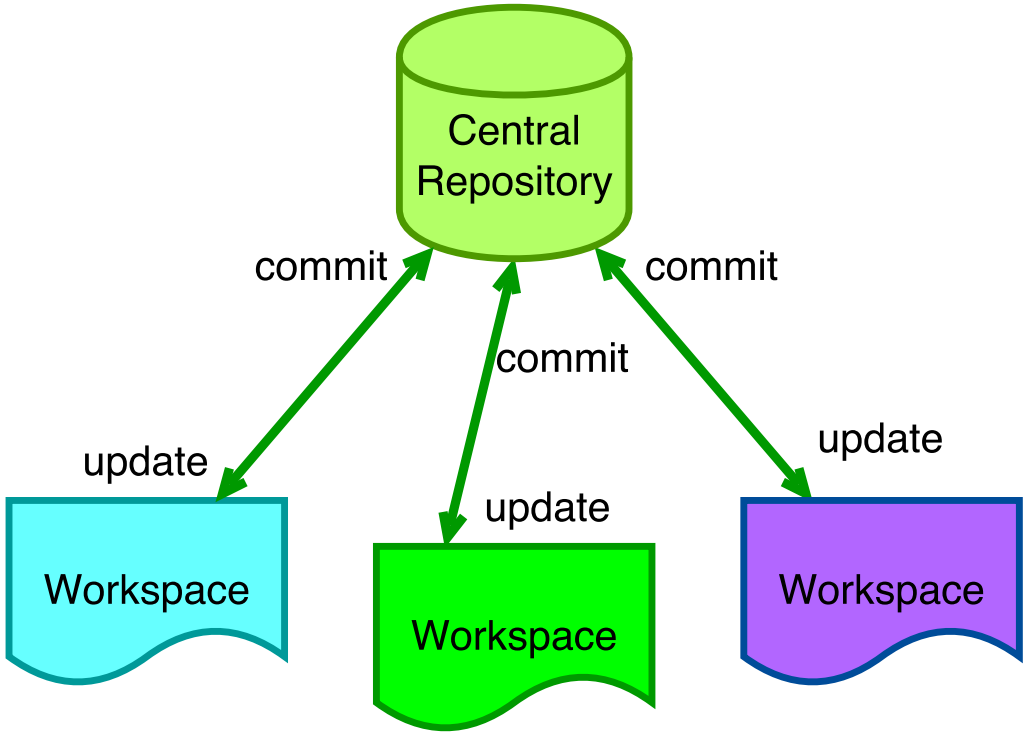


Figura : Sistemi di controllo di versione centralizzati

Sistemi di controllo di versione

Distribuiti

Sistemi di controllo di versione

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

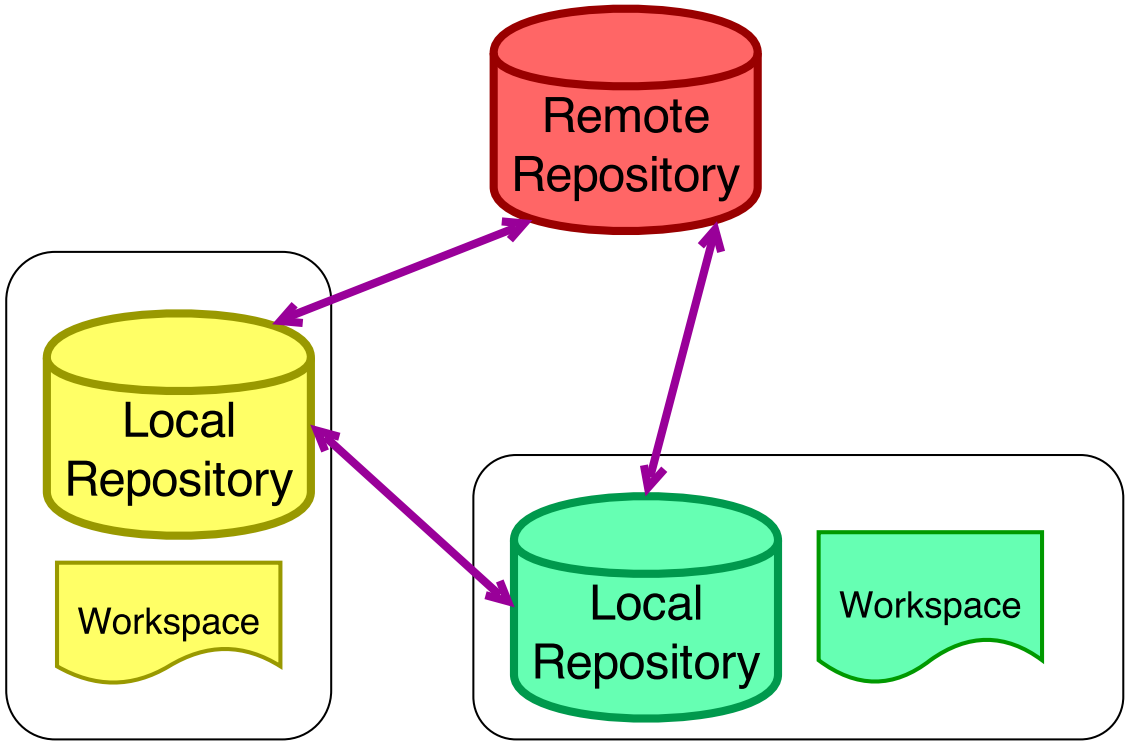


Figura : Sistemi di controllo di versione distribuiti

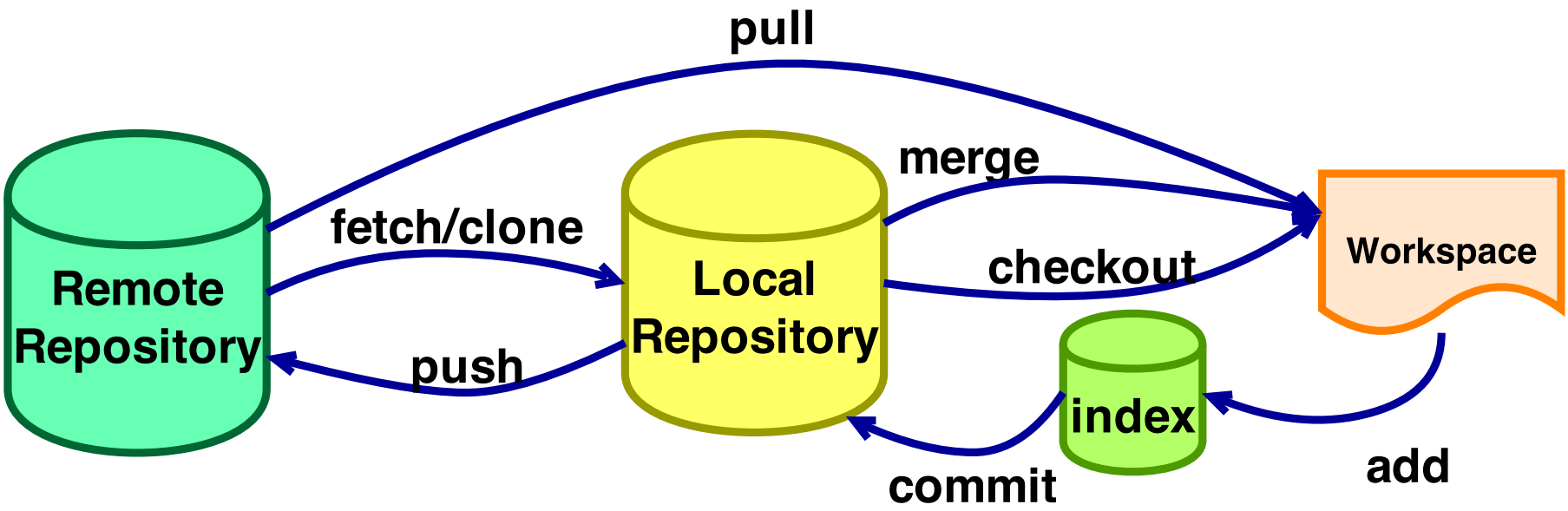
Git

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

Collaborazione tramite repository remoto



Repository Git sul server del corso

- Repository per l'esercitazione di ogni gruppo
  ssh://islt2013@infolab.ingce.unibo.it:80/Es1/group<numero>
- Repository comune per l'esercitazione accessibile in lettura
  ssh://islt2013@infolab.ingce.unibo.it:80/Es1/finalISLT2013
- Repository di gruppo
  ssh://islt2013@infolab.ingce.unibo.it:80/group<numero>
- Repository comune accessibile in lettura
  ssh://islt2013@infolab.ingce.unibo.it:80/finalISLT2013.git

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