# 05 Gestione della configurazione con Git

**Immagine che contiene testo, schermata, Carattere, diagramma

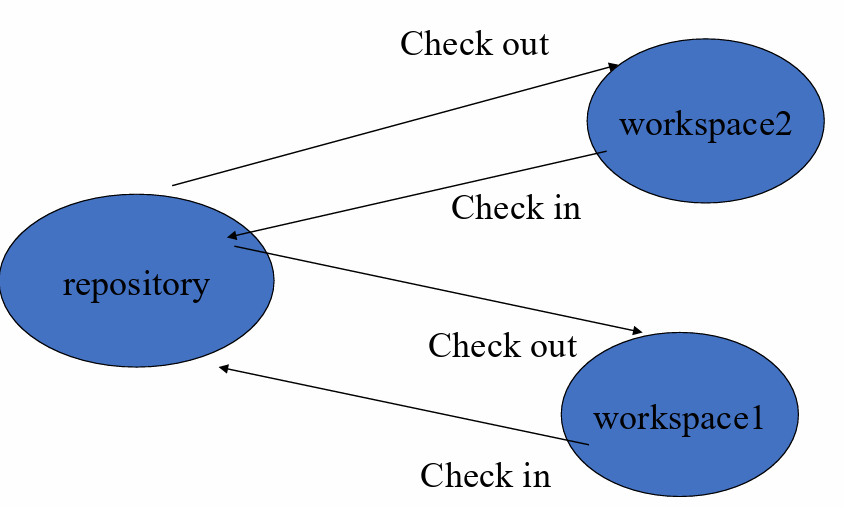
Descrizione generata automaticamenteConfiguration item**: oggetti soggetti a configurazione

* Documento dei requisiti
* Codice sorgente, etc

**Configuration**: insieme di configuration items, correlati e coerenti tra di loro

**Repository**: spazio in cui si trovano questi items.

Ogni item ha le varie **versioni** in modo di poter tornare a versioni precedenti in caso di errori (tranne se .ignore → configurazioni vscode, etc).

è possibile **cambiare il controllo** → persona X da oggi non può più fare determinate azioni.

**Check-in** → configuration item è stato modificato (variazione o cambio di accesso)

**Check-out** → configuration item ha avuto un accesso da un utente

**LOCK**: non esistono modifiche concorrenti e non si può lavorare in contemporanea sullo stesso file

*→ altra persona non può modificare finché non faccio lock*

**Copy-modify-merge**: utenti possono fare check-out in parallelo e poi si svolge merge

* Tutto ok se si lavora su righe diverse
* Ma se nel momento del merge, ci si accorge che due persone hanno modificato la stessa linea bisogna gestire → reject del commit

Molto importante inserire delle regole tra i componenti del team per quanto riguarda il **periodo di tempo** dopo il quale va fatto commit o quale evento lo scatena.

**LOCAL-CMS** → git init→ repository locale, solo sul nostro pc, nessuno vede ciò che sto facendo

**CENTRALIZED-CMS** → repository centrale che è l’unico punto di accesso e si fanno delle copie locali su cui lavorare. Si lavora su client ma il tutto è su server.

Immagine che contiene testo, schermata, Carattere, diagramma

Descrizione generata automaticamente**CMS DISTRIBUITO**: repository remoto ma facendo check-out si fa una copia locale di tutto e facendo check-out si ributta dentro (git). Versioni anche locali.

## Modello di salvataggio:

* Deltas → si salvano solo le differenze a partire da un punto di partenza
  + Per una versione X, devo partire da inizio e applicare tutte le modifiche
* Full-copies → copie di tutte le versioni modificate.

## GIT

Git commit 🡪 da staging area a locale → bisogna dare un commento sensato ad ogni modifica

Git push → da locale a remoto

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamenteImmagine che contiene testo, Carattere, schermata, bianco

Descrizione generata automaticamenteFILE:

Immagine che contiene testo, Carattere, bianco, schermata

Descrizione generata automaticamente

Immagine che contiene testo, Carattere, schermata, bianco

Descrizione generata automaticamente

Git pull --rebase → cerca di mettere tutto insieme, senza rendere il merge esplicito.

Miscellanea di comandi:

Immagine che contiene testo, schermata, Carattere, algebra

Descrizione generata automaticamente

→nome che vedranno, mail del poli

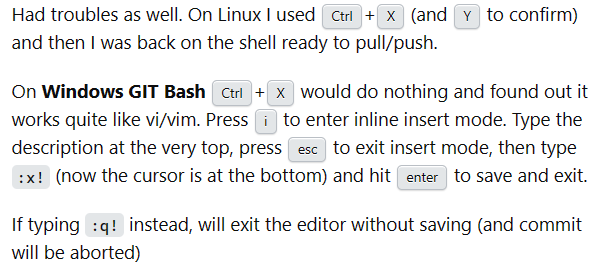
→ crea cartella

→ aggiungo a repository remoto

Immagine che contiene testo, Carattere, schermata, bianco

Descrizione generata automaticamenteImmagine che contiene testo, schermata, Carattere, algebra

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

Immagine che contiene testo, Carattere, schermata, bianco

Descrizione generata automaticamentegit ad

Immagine che contiene testo, schermata, diagramma

Descrizione generata automaticamente

Immagine che contiene testo, Carattere, bianco

Descrizione generata automaticamentese non fai git chechout, continui a lavorare su master

# Riassunto

## Fasi Principali

* Sviluppo: Definizione dei requisiti, progettazione, implementazione.
* Distribuzione
* Operazioni
* Manutenzione
* Ritiro

## Motivazioni

* Necessità di lavorare su documenti condivisi per lungo tempo.
* Gestione di accessi concorrenti e modifiche.
* Tracciamento delle modifiche e delle versioni.

## Concetti Chiave della Gestione della Configurazione

* Configuration Item (CI): Unità sotto gestione della configurazione, può essere composta da uno o più file.
* Versioning: Capacità di memorizzare tutte le versioni passate di un CI.
* Change Control: Controllo degli accessi e delle modifiche sui CI.
* Check-in/Check-out: Operazioni formali per accedere e modificare un CI.

## Modelli di Controllo delle Modifiche

* Lock-Modify-Unlock: Un utente blocca un CI, impedendo accessi concorrenti.
* Copy-Modify-Merge: Permette modifiche parallele.