

# Introduzione al Configuration & Source Management

---

Università di Modena e Reggio Emilia

*Prof. Nicola Bicchocchi (nicola.bicchocchi@unimore.it)*



# Aziende ICT (grandi dimensioni)

---

- Scala multinazionale
- Oltre 200 persone coinvolte per arrivare ad un numero indefinito
- Diverse dislocate su tutto il pianeta
- Diverse lingue utilizzate
- Orari flessibili
- Modalità di lavoro diversificate
- Struttura gerarchica ben definita



# Configuration Management

---

- Configuration Management (ITILv3): «The Process responsible for maintaining information about Configuration Items required to deliver an IT Service, including their Relationships.»
- Citato come parte fondamentale del manifesto Agile
- Insieme di processi ideati per la gestione e il controllo degli oggetti di sistemi complessi



# Un'analogia culinaria

---

- **Source Code Management:** verificare la presenza di tutti gli “ingredienti” corretti e in giusta quantità
- **Build Engineering:** mischiare gli ingredienti e creare la “torta”
- **Environment Configuration:** verificare che la “vetrina” in cui si espone la torta creata sia pronta all’uso
- **Change Control:** verificare e decidere quando la torta è pronta per essere esposta al pubblico
- **Release engineering:** mettere la torta in vetrina in modo che le persone possano vederla (ma non comprarla)
- **Deployment:** effettiva consegna della torta al cliente finale



# Source Management

---

- Disciplina base del Configuration Management
- Impatta sulla **qualità del prodotto** e sulla **produttività del team** di lavoro
- Spesso trascurata



# Obiettivi

---

- Creare una “cassaforte” di tutti i sorgenti: **nessun sorgente deve mai perdersi**
- Migliorare la produttività dei team di sviluppo (e.g., gestire più di una linea di sviluppo)
- **Tracciabilità**: sapere sempre chi ha cambiato cosa, quando e, se necessario, essere in grado di compiere il rollback della modifica





# Concetti preliminari

---

**Baseline:** identificare l'esatta versione di ogni sorgente contenuto in una specifica release del software. Macchina del tempo virtuale, permette di portarsi a un determinato istante temporale.





# Concetti preliminari

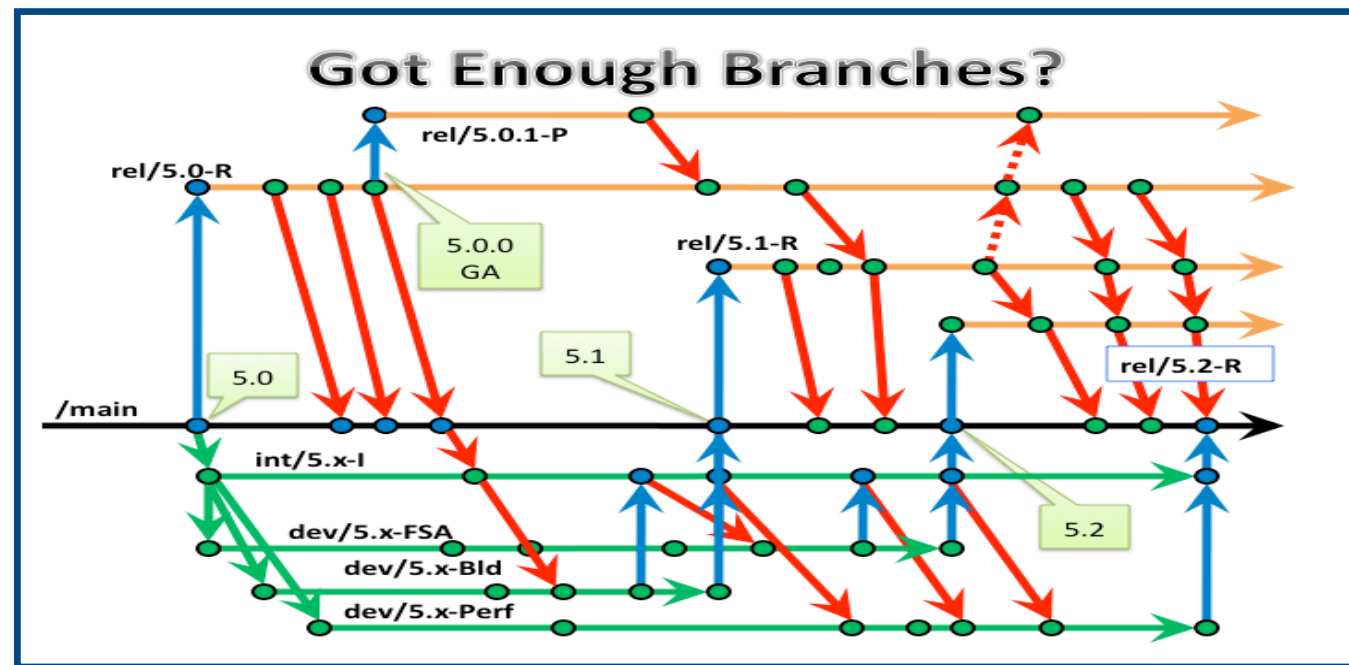
---

**Trunk:** linea base di produzione di un prodotto software. Modifiche importanti al trunk spesso si riflettono in una nuova release del software (e.g., iOS7 → iOS8)



# Concetti preliminari

**Branch:** linea di sviluppo parallela al trunk, di cui ne prende le caratteristiche ad un determinato istante temporale. Si usa il branching per modifiche minori e personalizzazioni del prodotto (e.g., versioni Windows, Unix e Mac dello stesso applicativo). Va usato con cautela!



# Concetti preliminari

---

**Merging:** operazione opposta al branching. Consente la fusione di un branch con il trunk in un determinato istante temporale. Di solito questa operazione comporta l'uscita di una nuova release.



# Concetti preliminari

---

**Workspace:** spazio privato e isolato in cui lavorare. Sostanzialmente è un clone del repository globale trasportato in una directory privata.



# Concetti preliminari

---

**Check in / Check out:** Operazioni di caricamento / scaricamento di sorgenti da e verso un sistema di Source Management.



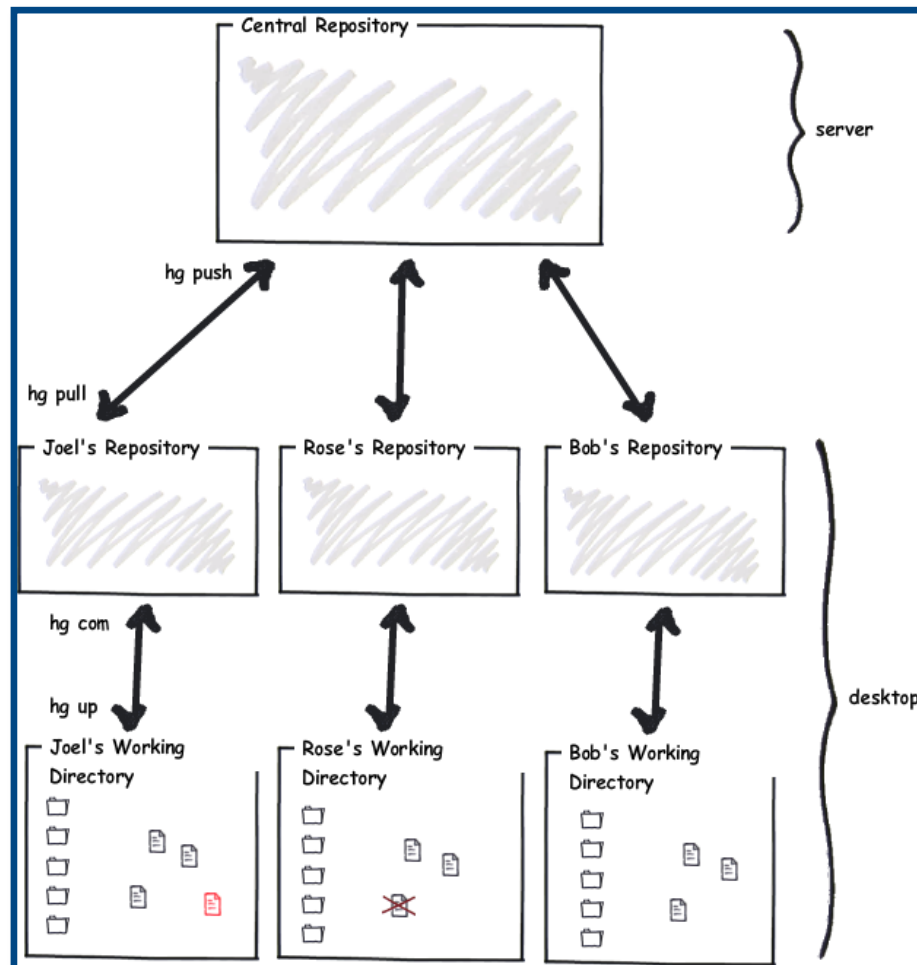
# Mercurial (hg)

---

- Software di Source Management
- Distribuito
- Open Source
- Tiene traccia della versione di ogni singolo file
- Può fare merge di versioni: il programmatore può lavorare sul singolo sorgente e fare in seguito il merge con le modifiche dei colleghi



# Mercurial (hg)



# Comandi base

```
Command Prompt

c:\hginit> hg
Mercurial Distributed SCM

basic commands:

add          add the specified files on the next commit
annotate    show changeset information by line for each file
clone        make a copy of an existing repository
commit       commit the specified files or all outstanding changes
diff         diff repository (or selected files)
export       dump the header and diffs for one or more changesets
forget       forget the specified files on the next commit
init         create a new repository in the given directory
log          show revision history of entire repository or files
merge        merge working directory with another revision
pull         pull changes from the specified source
push         push changes to the specified destination
remove       remove the specified files on the next commit
serve        export the repository via HTTP
status       show changed files in the working directory
summary      summarize working directory state
update       update working directory

use "hg help" for the full list of commands or "hg -v" for details
```





# hg init

---

- Crea fisicamente il repository sul file system (directory .hg)
  - Contiene metadati (versioni, timestamp, ...)
  - Non modificare la cartella in nessun modo

# hg clone

---

- Inizializza un repository locale scaricando la versione corrente di un repository remoto
  - E.g., `$ hg clone http://selenic.com/hg repo`



# hg add

---

- Aggiunge uno o più files al repository locale
  - E.g., `$ hg add Application.java`



# hg remove

---

- Rimuove uno o più files dal repository locale
  - E.g., `$ hg remove Application.java`



# hg commit

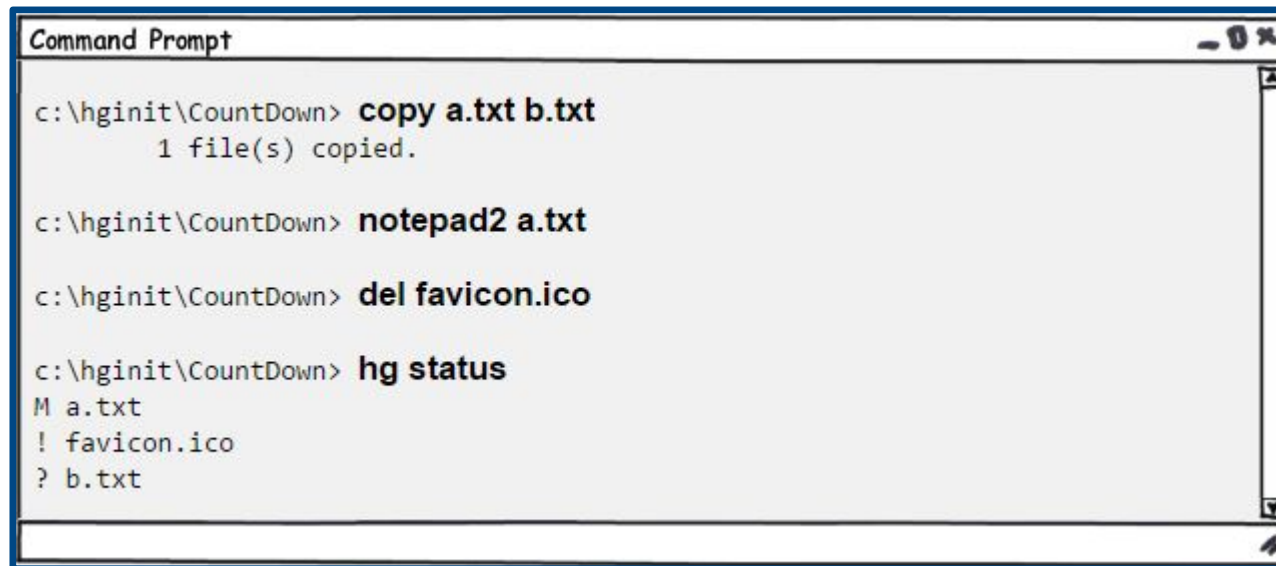
---

- “Fotografa” lo stato del workspace in un determinato istante temporale.
  - Commit viene di solito utilizzato per salvare il proprio lavoro in stati “accettabili” (e.g., compila, non ci sono problemi di dipendenze)
  - E.g., `$ hg commit`

# hg status

---

- Mostra le differenze fra il workspace e l'ultimo commit



```
Command Prompt

c:\hginit\CountDown> copy a.txt b.txt
1 file(s) copied.

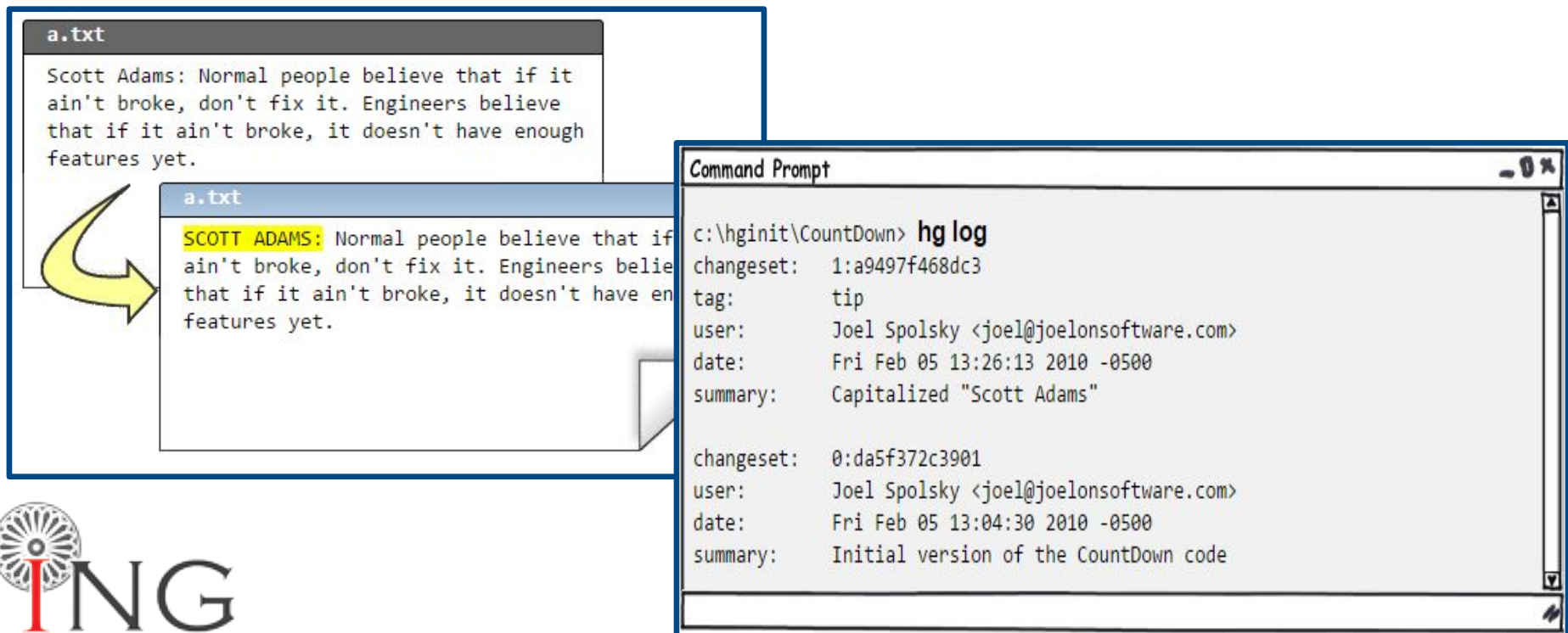
c:\hginit\CountDown> notepad2 a.txt

c:\hginit\CountDown> del favicon.ico

c:\hginit\CountDown> hg status
M a.txt
! favicon.ico
? b.txt
```

# hg log

- Mostra il log delle operazioni compiute sul repository
  - E.g., `$ hg log`



The diagram illustrates the output of the `hg log` command. On the left, a file named `a.txt` is shown with its content: "Scott Adams: Normal people believe that if it ain't broke, don't fix it. Engineers believe that if it ain't broke, it doesn't have enough features yet." A yellow arrow points from this file to a commit entry in the `hg log` output. The `Command Prompt` window shows the command `hg log` being executed in the directory `c:\hginit\CountDown`. The output lists two commits, each with its changeset, tag, user, date, and summary.

```
Command Prompt
c:\hginit\CountDown> hg log

changeset: 1:a9497f468dc3
tag:       tip
user:      Joel Spolsky <joel@joelonsoftware.com>
date:      Fri Feb 05 13:26:13 2010 -0500
summary:   Capitalized "Scott Adams"

changeset: 0:da5f372c3901
user:      Joel Spolsky <joel@joelonsoftware.com>
date:      Fri Feb 05 13:04:30 2010 -0500
summary:   Initial version of the Countdown code
```

**ING**

# hg push/pull

---

- Sincronizzano lo stato del repository locale con lo stato del repository globale
  - **push** invia le modifiche locali al server
  - **pull** riceve le modifiche globali dal server



# hg outgoing/incoming

---

- Mostrano tutte le modifiche da sincronizzare con il repository globale, senza effettuarle
  - **outgoing** le modifiche locali da inviare al repository globale (fake push)
  - **incoming** le modifiche globali da ricevere in locale (fake pull)

