



UNIVERSITÀ DEGLI STUDI
DI MILANO

SVILUPPO SOFTWARE IN GRUPPI DI LAVORO COMPLESSO

prof. Carlo Bellettini

prof. Mattia Monga

a.a. 2020-21

Per fare domande:

<https://homes.di.unimi.it/bellettini/questions>

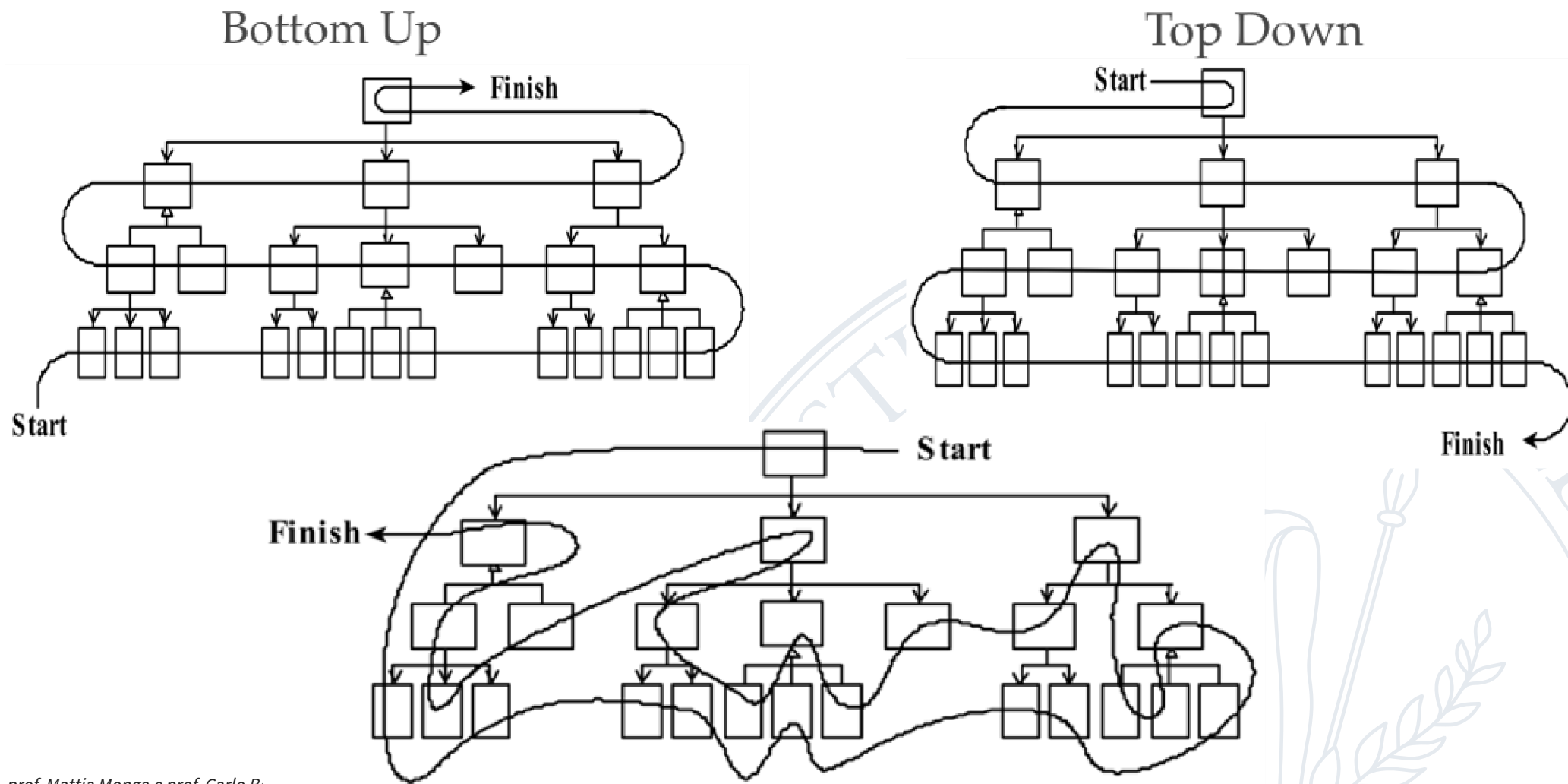
codice di oggi **20102020**



UNIVERSITÀ DEGLI STUDI
DI MILANO

7. Intro Continuous Integration

Integration



Unit vs Integration testing



Continuous Integration

Definizione:

Allineamento frequente (es. "molte volte al giorno") dagli ambienti di lavoro degli sviluppatori verso l'ambiente condiviso (mainline)

- Non è più una fase... viene affogata nello sviluppo normale...
 - finita una parte (una feature, una patch, ...) si integra
 - l'integrazione diventa un NONEVENT

Capisaldi (Martin Fowler 2006)

- Maintain a Single Source Repository
- Automate the Build
- Make Your Build Self-Testing
- Everyone Commits To the Mainline Every Day
- Every Commit Should Build the Mainline on an Integration Machine
- Fix Broken Builds Immediately
- Keep the Build Fast
- Test in a Clone of the Production Environment
- Make it Easy for Anyone to Get the Latest Executable
- Everyone can see what's happening
- Automate Deployment



Capisaldi (Martin Fowler 2006)

- **Maintain a Single Source Repository**
- Automate the Build
- Make Your Build Self-Testing
- Everyone Commits To the Mainline Every Day
- Every Commit Should Build the Mainline on an Integration Machine
- Fix Broken Builds Immediately
- Keep the Build Fast
- Test in a Clone of the Production Environment
- Make it Easy for Anyone to Get the Latest Executable
- Everyone can see what's happening
- Automate Deployment



Single Source Repository

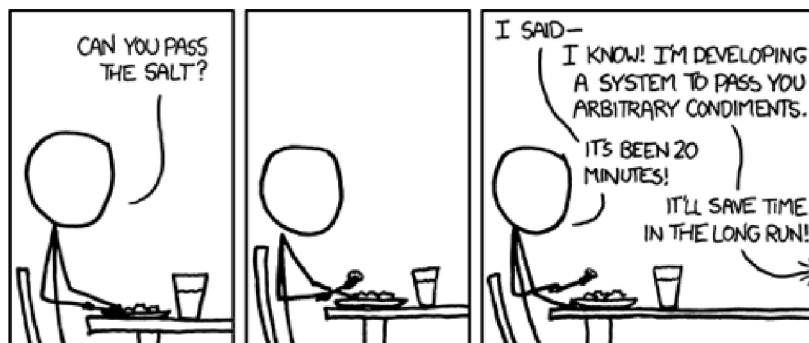


Capisaldi (Martin Fowler 2006)

- Maintain a Single Source Repository
- Automate the Build
- Make Your Build Self-Testing
- Everyone Commits To the Mainline Every Day
- Every Commit Should Build the Mainline on an Integration Machine
- Fix Broken Builds Immediately
- Keep the Build Fast
- Test in a Clone of the Production Environment
- Make it Easy for Anyone to Get the Latest Executable
- Everyone can see what's happening
- Automate Deployment

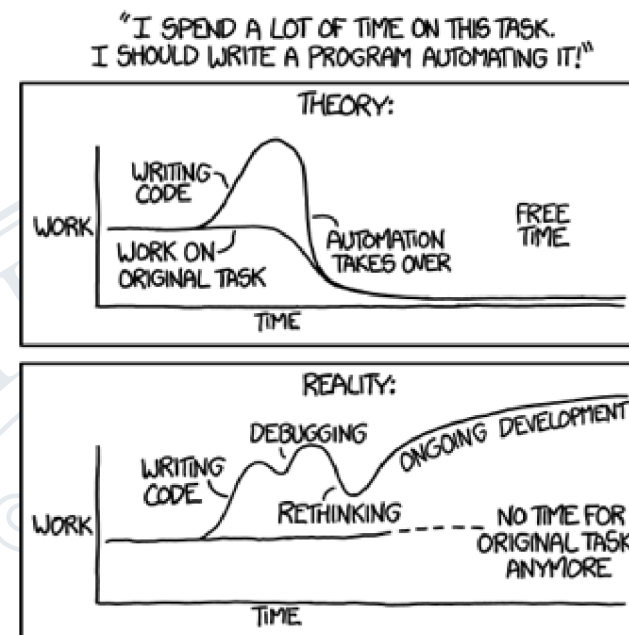


Automate the build



HOW LONG CAN YOU WORK ON MAKING A ROUTINE TASK MORE EFFICIENT BEFORE YOU'RE SPENDING MORE TIME THAN YOU SAVE? (ACROSS FIVE YEARS)

		HOW OFTEN YOU DO THE TASK					
		50/DAY	5/DAY	DAILY	WEEKLY	MONTHLY	YEARLY
1 SECOND		1 DAY	2 HOURS	30 MINUTES	4 MINUTES	1 MINUTE	5 SECONDS
5 SECONDS		5 DAYS	12 HOURS	2 HOURS	21 MINUTES	5 MINUTES	25 SECONDS
30 SECONDS		4 WEEKS	3 DAYS	12 HOURS	2 HOURS	30 MINUTES	2 MINUTES
1 MINUTE		8 WEEKS	6 DAYS	1 DAY	4 HOURS	1 HOUR	5 MINUTES
5 MINUTES		9 MONTHS	4 WEEKS	6 DAYS	21 HOURS	5 HOURS	25 MINUTES
30 MINUTES			6 MONTHS	5 WEEKS	5 DAYS	1 DAY	2 HOURS
1 HOUR			10 MONTHS	2 MONTHS	10 DAYS	2 DAYS	5 HOURS
6 HOURS					2 MONTHS	2 WEEKS	1 DAY
1 DAY						8 WEEKS	5 DAYS



Capisaldi (Martin Fowler 2006)

- Maintain a Single Source Repository
- Automate the Build
- **Make Your Build Self-Testing**
- Everyone Commits To the Mainline Every Day
- Every Commit Should Build the Mainline on an Integration Machine
- Fix Broken Builds Immediately
- Keep the Build Fast
- Test in a Clone of the Production Environment
- Make it Easy for Anyone to Get the Latest Executable
- Everyone can see what's happening
- Automate Deployment



Capisaldi (Martin Fowler 2006)

- Maintain a Single Source Repository
- Automate the Build
- Make Your Build Self-Testing
- **Everyone Commits To the Mainline Every Day**
- Every Commit Should Build the Mainline on an Integration Machine
- Fix Broken Builds Immediately
- Keep the Build Fast
- Test in a Clone of the Production Environment
- Make it Easy for Anyone to Get the Latest Executable
- Everyone can see what's happening
- Automate Deployment



Capisaldi (Martin Fowler 2006)

- Maintain a Single Source Repository
- Automate the Build
- Make Your Build Self-Testing
- Everyone Commits To the Mainline Every Day
- Every Commit Should Build the Mainline on Integration Machine
- Fix Broken Builds Immediately
- Keep the Build Fast
- Test in a Clone of the Production Environment
- Make it Easy for Anyone to Get the Latest Executable
- Everyone can see what's happening
- Automate Deployment

Capisaldi (Martin Fowler 2006)

- Maintain a Single Source Repository
- Automate the Build
- Make Your Build Self-Testing
- Everyone Commits To the Mainline Every Day
- Every Commit Should Build the Mainline on an Integration Machine
- **Fix Broken Builds Immediately**
- Keep the Build Fast
- Test in a Clone of the Production Environment
- Make it Easy for Anyone to Get the Latest Executable
- Everyone can see what's happening
- Automate Deployment

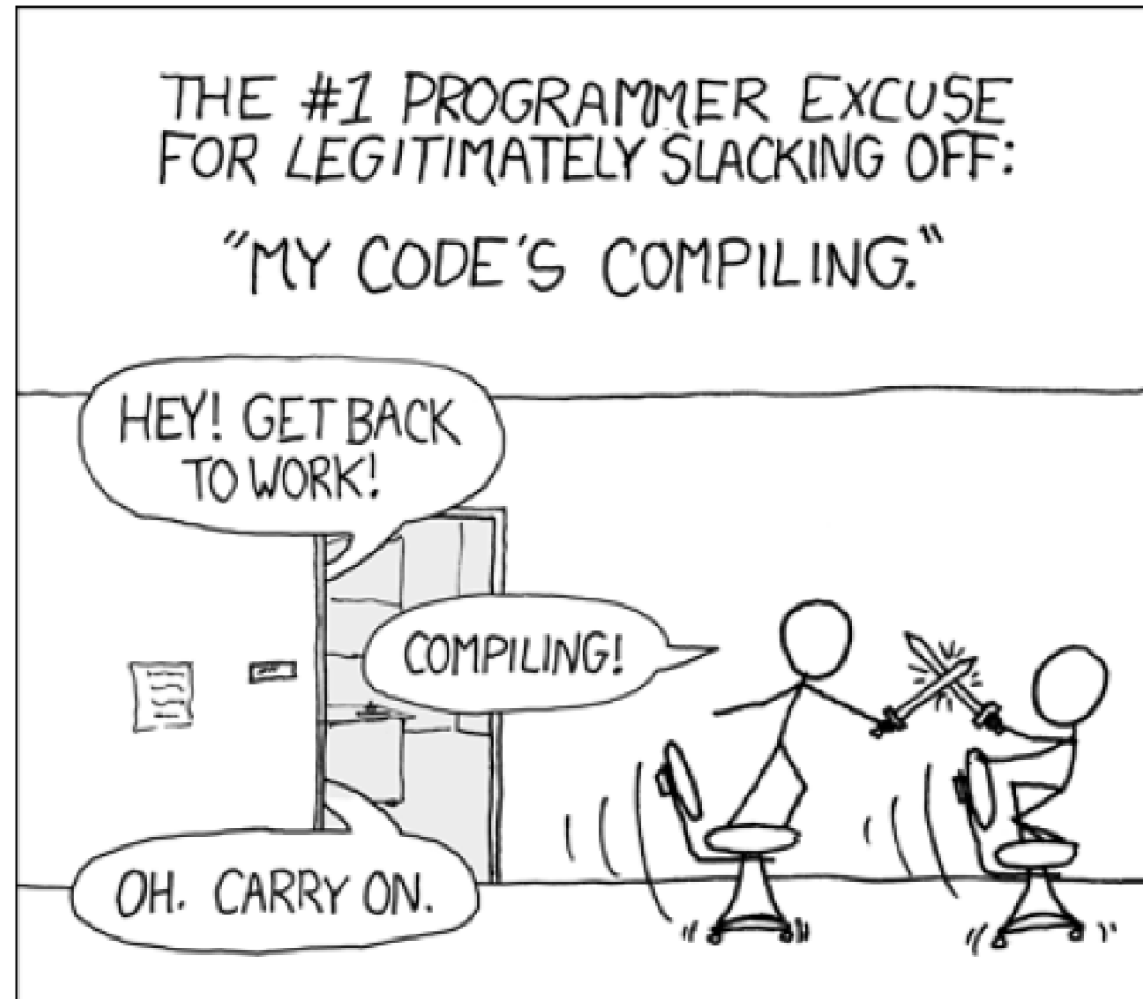


Capisaldi (Martin Fowler 2006)

- Maintain a Single Source Repository
- Automate the Build
- Make Your Build Self-Testing
- Everyone Commits To the Mainline Every Day
- Every Commit Should Build the Mainline on an Integration Machine
- Fix Broken Builds Immediately
- **Keep the Build Fast**
- Test in a Clone of the Production Environment
- Make it Easy for Anyone to Get the Latest Executable
- Everyone can see what's happening
- Automate Deployment



Keep the build fast



Capisaldi (Martin Fowler 2006)

- Maintain a Single Source Repository
- Automate the Build
- Make Your Build Self-Testing
- Everyone Commits To the Mainline Every Day
- Every Commit Should Build the Mainline on an Integration Machine
- Fix Broken Builds Immediately
- Keep the Build Fast
- **Test in a Clone of the Production Environment**
- Make it Easy for Anyone to Get the Latest Executable
- Everyone can see what's happening
- Automate Deployment



Capisaldi (Martin Fowler 2006)

- Maintain a Single Source Repository
- Automate the Build
- Make Your Build Self-Testing
- Everyone Commits To the Mainline Every Day
- Every Commit Should Build the Mainline on an Integration Machine
- Fix Broken Builds Immediately
- Keep the Build Fast
- Test in a Clone of the Production Environment
- **Make it Easy for Anyone to Get the Latest Executable**
- Everyone can see what's happening
- Automate Deployment



Capisaldi (Martin Fowler 2006)

- Maintain a Single Source Repository
- Automate the Build
- Make Your Build Self-Testing
- Everyone Commits To the Mainline Every Day
- Every Commit Should Build the Mainline on an Integration Machine
- Fix Broken Builds Immediately
- Keep the Build Fast
- Test in a Clone of the Production Environment
- Make it Easy for Anyone to Get the Latest Executable
- Everyone can see what's happening
- Automate Deployment



Capisaldi (Martin Fowler 2006)

- Maintain a Single Source Repository
- Automate the Build
- Make Your Build Self-Testing
- Everyone Commits To the Mainline Every Day
- Every Commit Should Build the Mainline on an Integration Machine
- Fix Broken Builds Immediately
- Keep the Build Fast
- Test in a Clone of the Production Environment
- Make it Easy for Anyone to Get the Latest Executable
- Everyone can see what's happening
- **Automate Deployment**



Source code management (versioning)

One of the features of version control systems is that they allow you to create **multiple branches**, to handle different streams of development. This is a useful, nay essential, feature - but it's **frequently overused and gets people into trouble**. Keep your use of branches to a minimum.

M. FOWLER

Linus presenta git a Google

Tech Talk: Linus Torvalds on git

