



VCS-GIT

1-Controllo di versione

Prof. Marcello Missiroli



DI CHE SI PARLA?

Quando un progetto software diventa complesso, anche la sua gestione è complicata.

Il controllo revisioni permette di gestire tutte le modifiche apportate ai documenti del progetto, archiviando tutte le versioni del progetto sin dal suo inizio

È uno strumento essenziale nello sviluppo moderno, e non usarlo può solo portare a conseguenze spiacevoli...

A large, bright yellow and orange nuclear mushroom cloud explosion dominates the center of the frame. The cloud has a thick, dark column rising from the ground, surrounded by smaller clouds. The background is a dark, orange-hued sky with swirling patterns. The foreground shows a desert landscape with mountains and a body of water. The overall tone is dramatic and ominous.

**No VCS?
AHI AHI AHI!**



1.

Il problema

Bigger is not always better



Programming in the small

- » Difficile recuperare le versioni più vecchie basandosi sulla data.
- » Difficile stabilire le differenze tra le versioni di uno stesso file.



Ieri



```
#include<stdio.h>  
>  
Int main()  
{  
    int a;  
}
```

File.c



Oggi

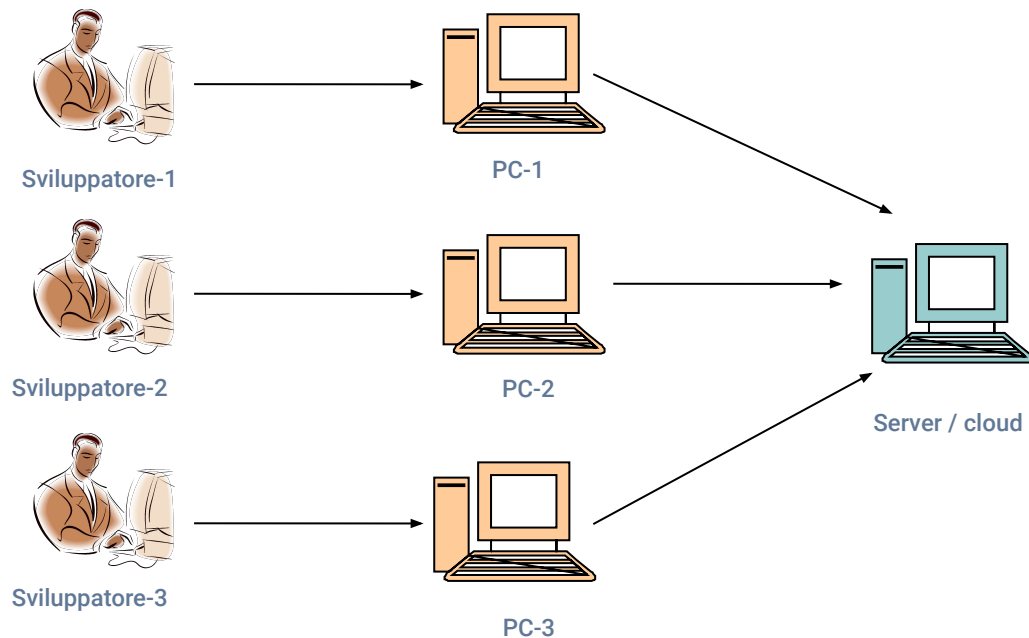


```
#include<stdio.h>  
>  
Int main()  
{  
    int a;  
    int b;  
}
```

File.c

Programming in the large/many

- » Non c'è modo di recuperare le versioni precedenti
- » Non c'è modo di vedere le differenze tra le versioni.
- » Processo di fusione versioni manuale, lento e di dubbio successo.
- » Sovrascritture accidentali



Version Control System

Detto anche Versioning, o Revision Control System o Source control management.

Revision control (also known as version control, source control or (source) code management (SCM)) is the management of changes to documents, programs, and other information stored as computer files.

In pratica, VCS ci dà...

- » Un posto dove memorizzare le varie versioni del codice (e tutti i documenti correlati), un backup multiplo e flessibile.
- » Possibilità di “tornare indietro” in caso di errore e tracciare le modifiche
- » Sviluppo parallelo facilitato, potendo lavorare su più versioni contemporaneamente



2.

Git

“You stupid git!”

RCS

RCS is created by Walter Tichy

1982**BITKEEPER + SVN**

BitKeeper is created by Bitmover Inc and SVN is created by Collabnet Inc

2000**1972****SCSS**

SCSS is created by Marc Rochkind

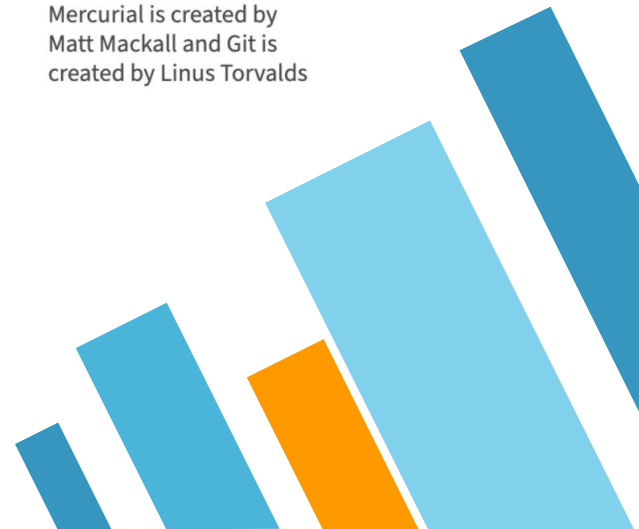
1986**CVS**

CVS is created by Dick Grune

2005**MERCURIAL + GIT**

Mercurial is created by Matt Mackall and Git is created by Linus Torvalds

RCS timeline



Confronto



SCCS & RCS ('82)



Subversion (2000)



CVS ('86)



GIT (2005)

SVN

Central-Repo-to-Working-Copy
Collaboration



GIT

Repo-to-Repo
Collaboration



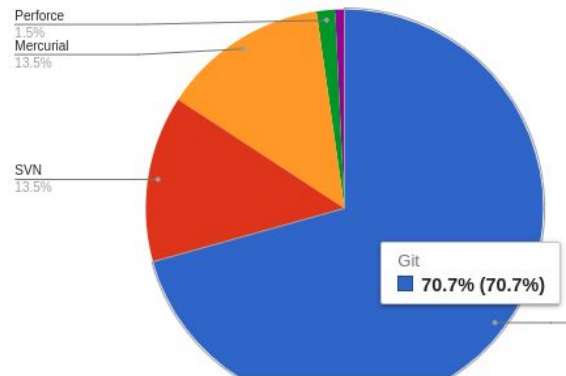
Gitstoria


Git è figlio di Linux. Per molto tempo lo sviluppo del kernel Linux le modifiche al software venivano passate sotto forma file compressi. Nel 2002, si iniziò ad utilizzare un sistema DVCS proprietario chiamato BitKeeper.

Nel 2005 fu revocato l'uso gratuito di BitKeeper con grandi polemiche. Ciò indusse Linus Torvalds sviluppare uno strumento proprio. Dopo 15 giorni di sviluppo, Git fu pronto e da allora si è fortemente evoluto e maturato.


Git è veloce ed efficiente anche per grandi progetti. Ha un sistema eccellente di ramificazioni per lo sviluppo non lineare. Oggi si stima lo usino almeno l'85% dei professionisti.

Web Search Interest Share, top-5 VCS, 2016

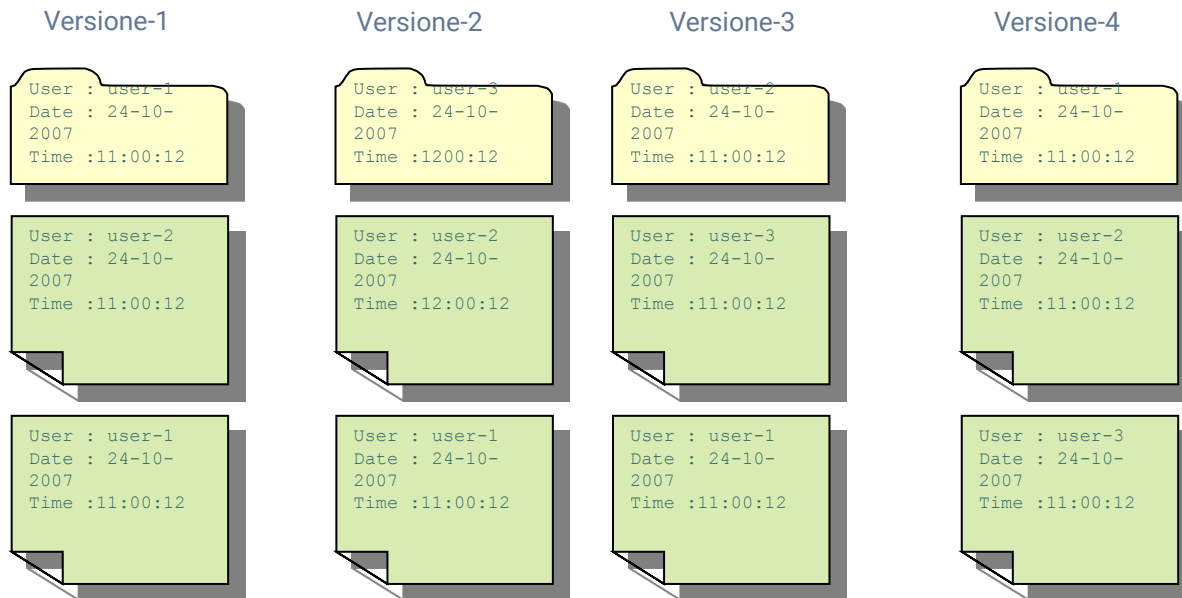




“The problem with Git jokes?
Everybody has their own
version!”
– *Anonimo*



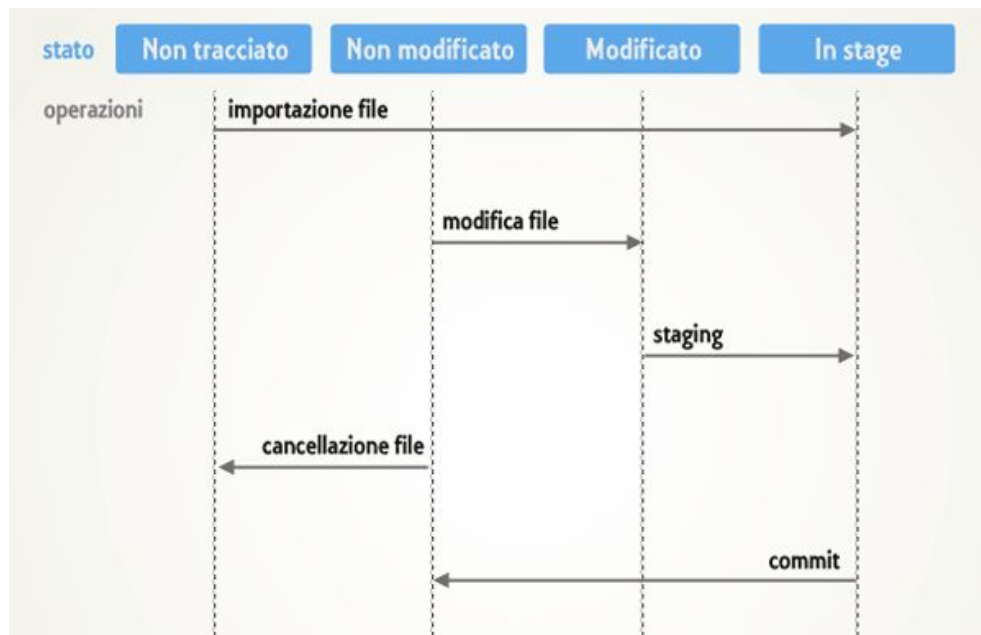
Concetto base: il repository




Glossario minimo


- » **Repository** (deposito): un insieme di oggetti (file) e i riferimenti necessari - in pratica una “foto” del progetto. Può essere online o offline
- » **Clone**: l'atto di duplicare localmente un repository
- » **Commit** (lett. 'impegno'): Invio delle modifiche effettuate al repository
- » **Checkout**: Ripristino di una particolare versione del progetto

Stati dei file su git





Sequenza di lavoro tipica



Working Copy



Staging Area

Commit

Working Copy



Staging Area

Commit

Working Copy

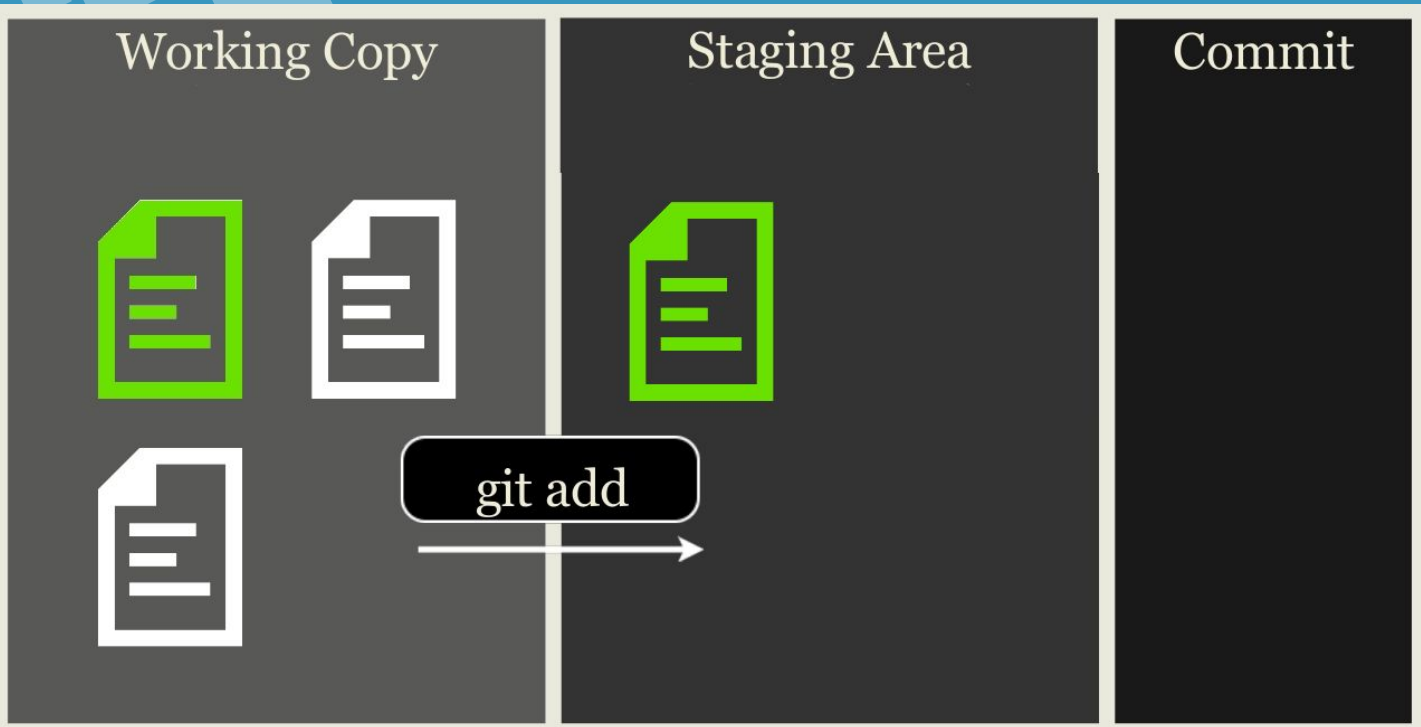


git add



Staging Area

Commit



Working Copy



Staging Area



Commit

Working Copy



Staging Area



Commit



git commit



Working Copy



Staging Area



Commit



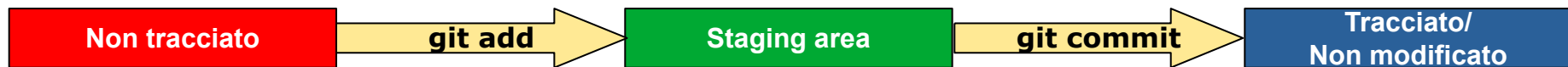
Working Copy



Staging Area

Commit

Riassunto



Quello che abbiamo visto è un esempio tipico del ciclo di lavoro base di git. Ora proveremo ad applicarlo davvero con il nostro computer.



3.

Social coding



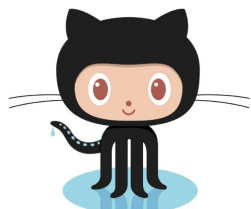
Le origini

L'arrivo di Internet 2.0, la necessità di condividere il codice con la comunità, il bisogno di accedere al codice sempre, comunque e da dovunque ha portato alla nascita dei primi siti di condivisione codice, come ad esempio Sourceforge (1999).

Alcuni di questi siti, grazie anche alla potenzialità di networking di Git, si sono evoluti in vere e proprie comunità di social networking




Github



Git**H**ub

Fondato nel 2008, si tratta del sito sicuramente più famoso e con maggior numero di utenti. Fu comprato nel 2018 da Microsoft causando sconcerto e preoccupazione soprattutto nella comunità FLOSS.

Ciononostante, il sito è florido e ben congegnato. È tuttora la scelta principale per i progetti open source, grazie alla sua lunga storia e alle funzionalità di ricerca interna.





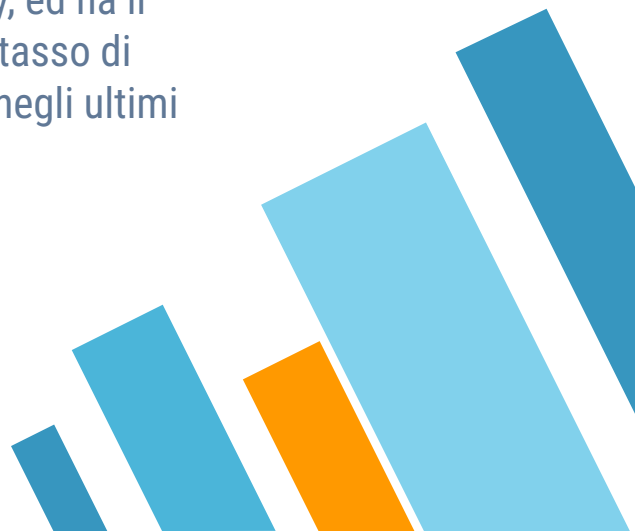
Gitlab



GitLab

Lanciato nel 2014 dalla due sviluppatori ucraini, il sistema è cresciuto considerevolmente negli anni e si distingue dai prodotti simili per essere open source ed essere ottimizzato per il modello di sviluppo DevOps.

È favorito dalle aziende con problemi di privacy, ed ha il maggior tasso di crescita negli ultimi anni.



Bitbucket



Il servizio è di proprietà di Atlassian, un grosso calibro nella gestione automatica dei progetti, noto in particolare per la suite Jira. Per questo motivo, è particolarmente diffuso nelle aziende medio-grandi.

Suo punto di forza è l'integrazione con altri servizi come JIRA Software, HipChat, Confluence e Bamboo.

Su che basi scegliere?

Github ha il miglior supporto della community, essendo il più anziano e pullula di progetti interessanti. Pende l'ombra di Microsoft.

Gitlab permette di lavorare installando un server privato e si concentra sulla catena di produzione

BitBucket è sicuramente la scelta più “corporate”.

E quindi?

Per i nostri interessi, tutte queste aziende offrono la possibilità di utilizzare gratuitamente i loro servizi, e regalano non meno di 500Gb per ogni repository. Le differenze s'iniziano a vedere a partire dai piani a pagamento. Qui faremo riferimento soprattutto a **Gitlab**, per via del suo impegno a favore del software open source.

È molto probabile che li userete tutti e tre.



“I’m an egotistical bastard,
and I name all my products
after myself. First Linux, then
Git.”

Linus Torvalds.

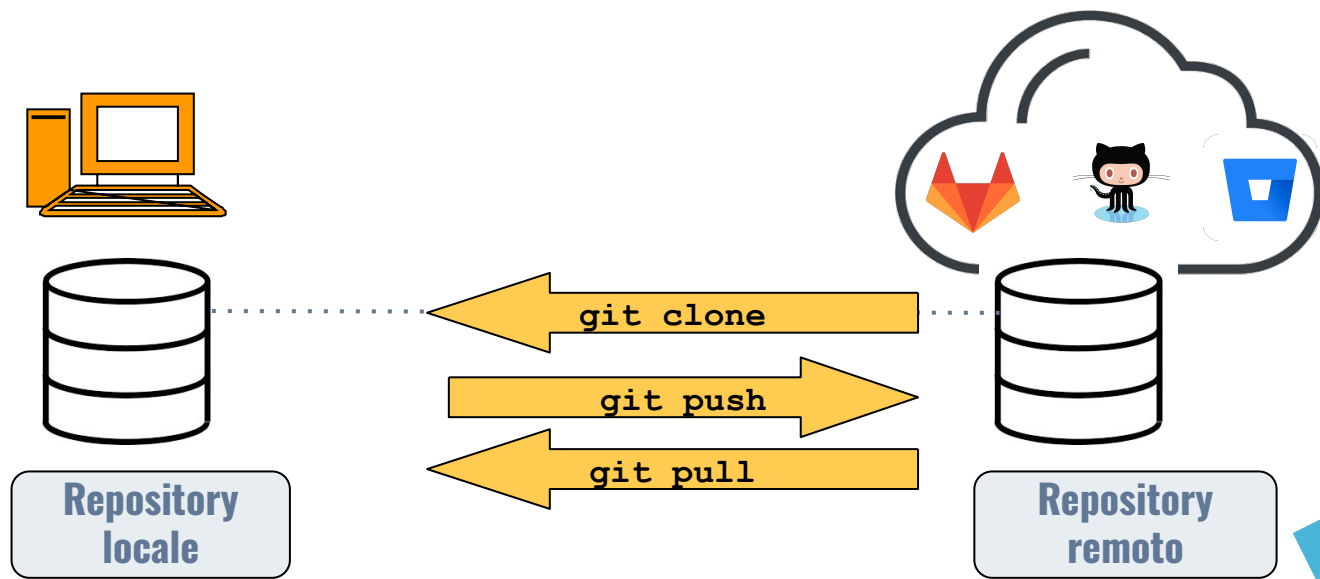
Git: a foolish or
worthless
person.

4.

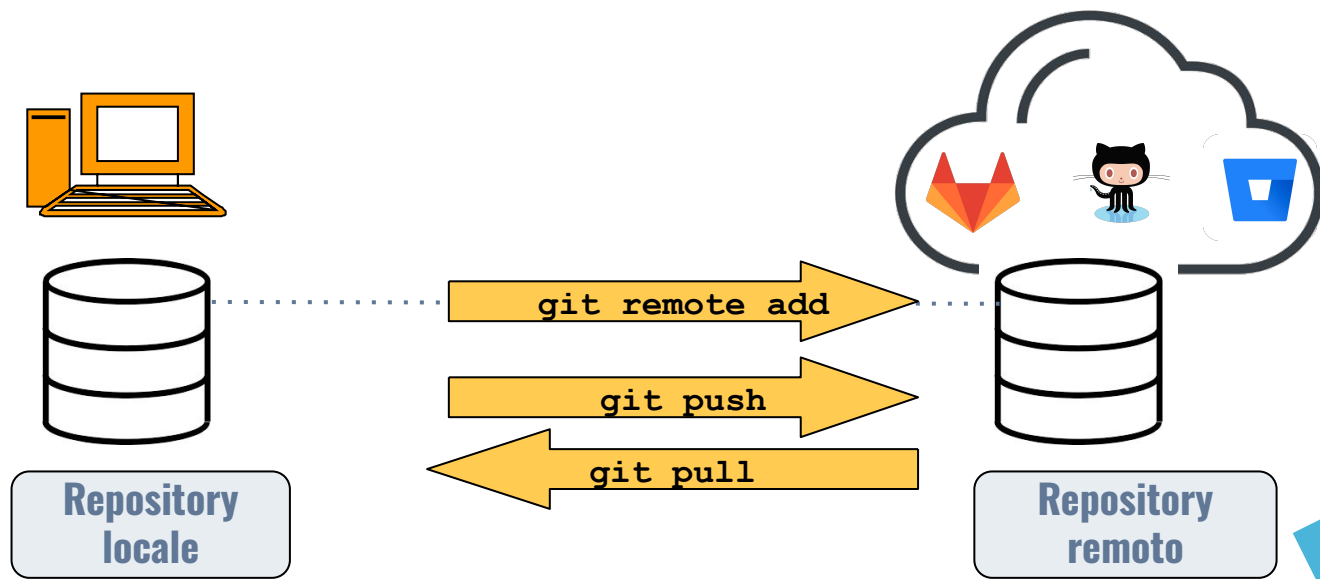
Git remotes

Backup senza sforzo

Schema di base



Schema alternativo



“Git gets easier once you get
the basic idea that branches
are homeomorphic
endofunctors mapping
submanifolds of a Hilbert
space.”

Isaac Wolkerstorfer





GRAZIE!

Ora si va sul pratico



CREDITS

Ringraziamenti a :

- » Presentation template by SlidesCarnival
- » Photographs by Unsplash
- » Anil Gupta (www.guptaanil.com)
- » Pete Nicholls (github.com/Aupajo)
- » Armando Fox

***Questo documento è distribuito con
licenza CreativeCommon BY-SA 3.0***