Revisionskontrollsystem GIT

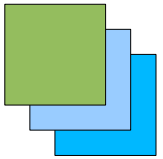
Diplom Informatiker Gregor Rebel

Versionskontrollsysteme

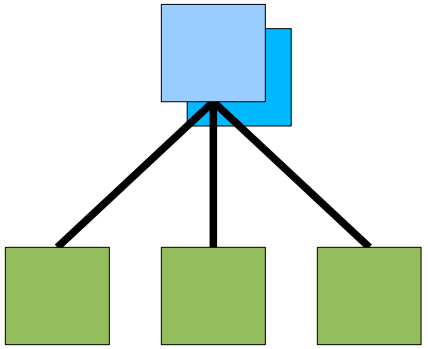
- Erfassung von Änderungen an Dateien
  - **Protokollierung** von Änderungen
  - **Wiederherstellung** alter Zustände
  - **Archivierung** der gesamten Historie
  - **Koordinierung** des gemeinsamen Zugriffs
  - **Verzweigung** in mehrere Branches

Funktionsweisen

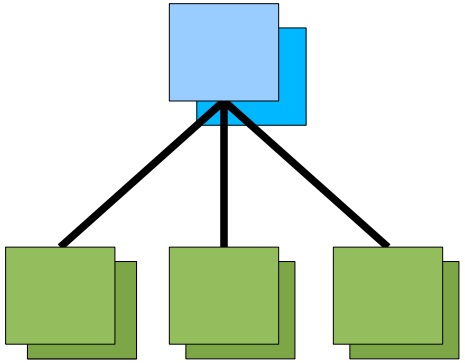
- Lokale Versionsverwaltung



- Zentrale Versionsverwaltung

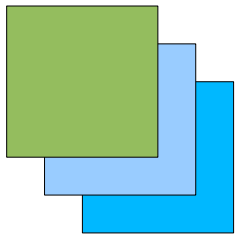


- Verteilte Versionsverwaltung



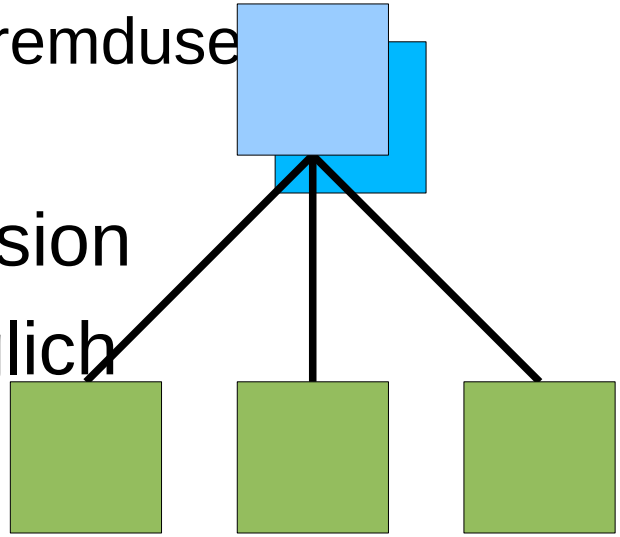
Lokale Versionsverwaltung

- Oft nur einzelne Datei versioniert
- Änderungen in Datei selbst gespeichert
- Werkzeuge: SCCS, RCS
- Beispiele
  - Technische Zeichnung mit Änderungsindex



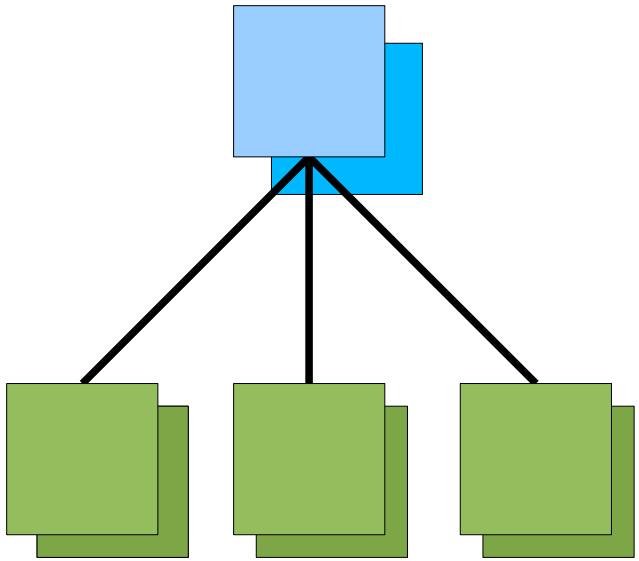
Zentrale Versionsverwaltung

- Als Client-Server System aufgebaut
- Ein zentraler Server speichert komplette Historie
  - optional: Read-Only Zugriff für Fremduse
  - optional: Web-Interface
- Entwickler jeweils nur eine Version
- Zugriff auch per Netzwerk möglich
- Werkzeuge
  - cvs, svn (OpenSource)
  - AlienBrain, Perforce, Team Foundation, ...



Verteilte Versionsverwaltung

- Jeder Entwickler hat komplette Historie
- Zusätzlich zentraler Server möglich
  - Änderungen zwischen Entwicklern austauschen
  - optional: Read-Only Zugriff für Fremduser
  - optional: Web-Interface
- Werkzeuge
  - git, GNU arch, Mercurial, Bazaar (OpenSource)
  - BitKeeper



Wie Git das?

- Übersicht
- Installation
- Dezentrale Verwaltung
- Verzweigte Entwicklung
- Datenaustausch
- Markierte Revisionen
- Interoperabilität
- Web-Interface

Git - Übersicht

- Freie Software
- Verteilte Versionsverwaltung von Dateien
- Entwickelt von Linus Torvalds
  → Verwaltung der Linux Kernel Quelltexte
- Betriebsysteme
  - Linux, Solaris, Mac OS X, FreeBSD, …
  - Windows (Cygwin, Msysgit, TortoiseGit-Shell)

- Aktuell eingesetzt in vielen OpenSource Projekten
  - Amarok, Android, BusyBox, Debian, DragonFly BSD, Eclipse, Erlang, Fedora, Git, Gnome, KDE, Qt, Ruby, …
- Kurzanleitung: → halbleiterbauelemente.de
- Webseite: → git-scm.com

Git - Installation

- Debian/ Ubuntu
  - sudo apt-get install git
- openSuSE
  - sudo zypper install git
- Windows
  - http://de.wikipedia.org/wiki/Cygwin
  - http://de.wikipedia.org/wiki/TortoiseGit

Git - Dezentrale Verwaltung

- Jeder Benutzer hat eigenes Repository

> git pull

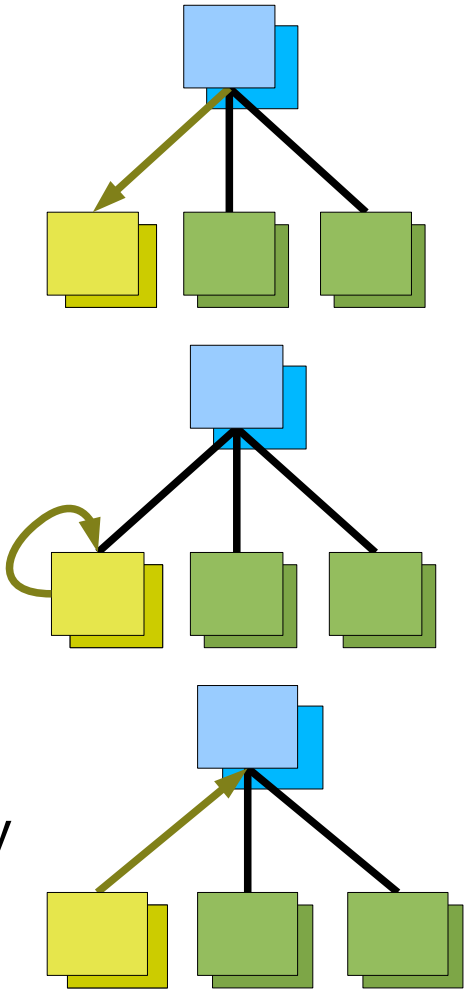
  - Holt neuesten Stand vom Zentralrepository
  - Kann Merge ausführen!

> git commit

  - sichert aktuellen Stand lokal
  - kommentiert aktuellen Stand
  - erzeugt einzigartige Commit-ID (Hexzahl)

> git push

  - kopiert neuesten Stand auf Zentralrepository



Git - Verzweigte Entwicklung 1/3

- Verzweigung in der Entwicklung

```
> git checkout -b B1
```

- Verzweigung ab einem älteren Zustand

```
> git checkout -b B2 2
```

- Verzweigungspunkt:
  Branchname/ Commit-ID/ Tag
- Alle verfügbaren Branches auflisten

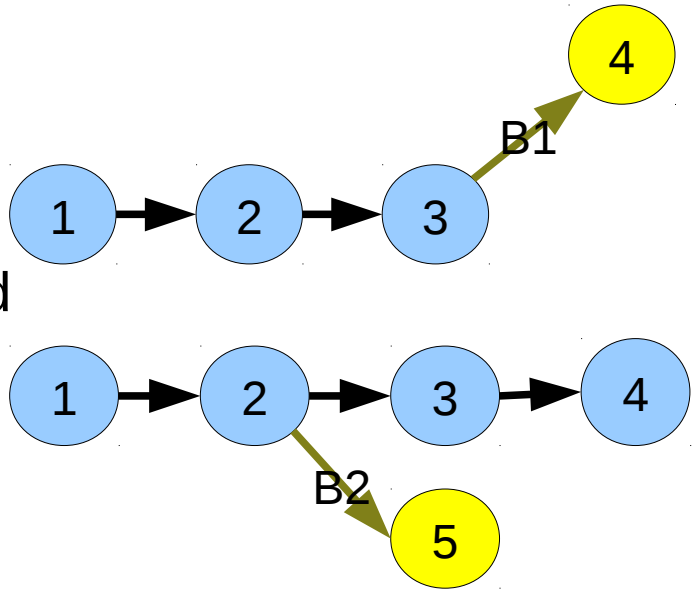
```
> git branch -r
```

- Auf anderen Branch wechseln

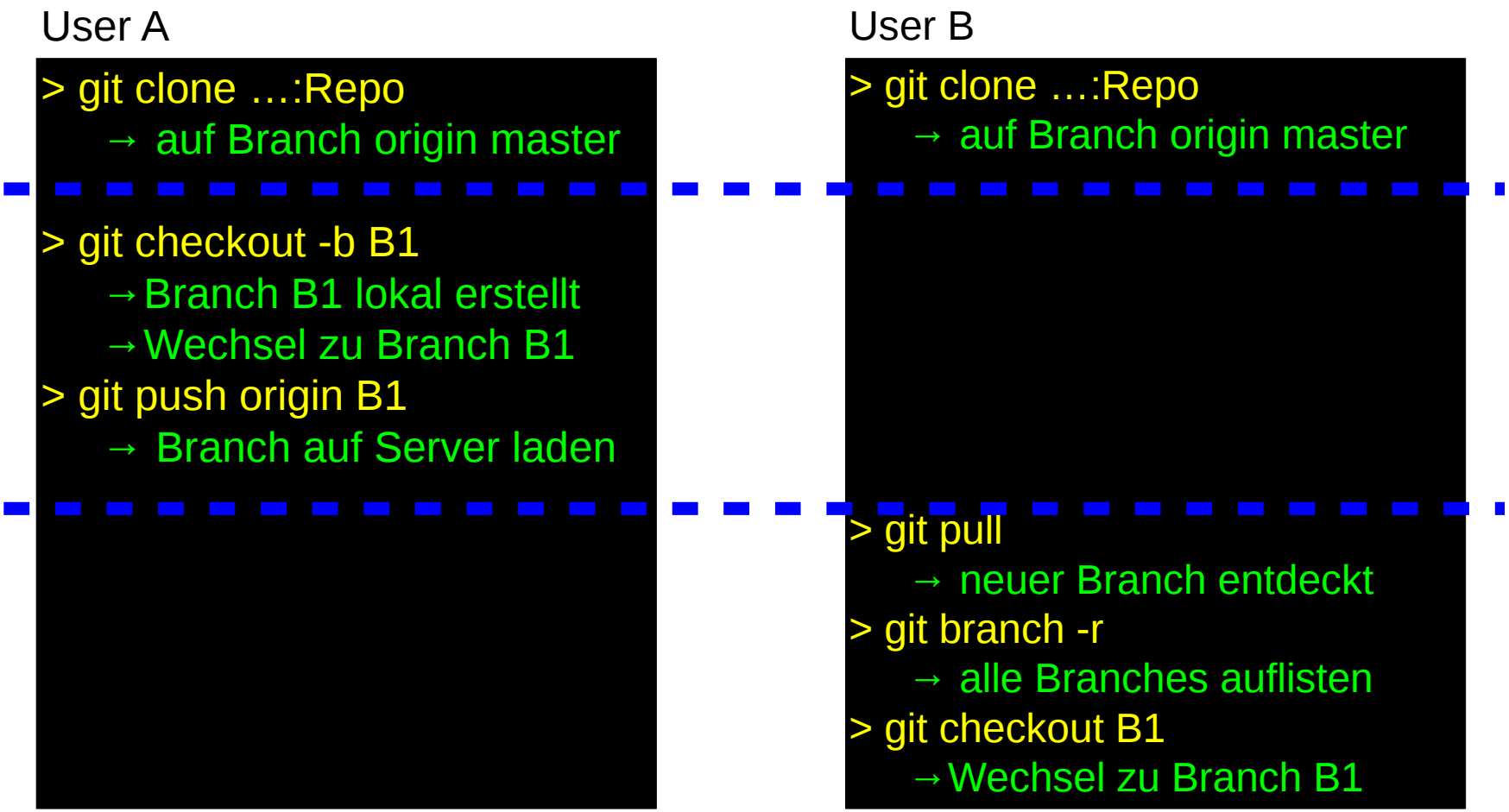
```
> git checkout BRANCH_NAME
```

- Branch zum Server hochladen

```
> git push origin BRANCH_NAME
```



Git - Verzweigte Entwicklung 2/3



Git - Verzweigte Entwicklung 3/3

- Implizite Verzweigung
  - User A

```
> git commit -a -m „C1“
```

  - User B

```
> git commit -a -m „C2“
> git push
```

    → C1 fehlt → Impliziter Branch

  - User B

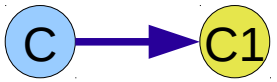
```
> git pull
> git commit -a
```

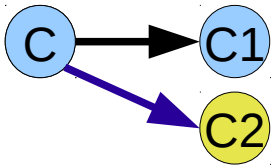
    → automatischer Merge von C1

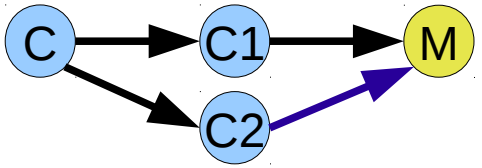
  - User A

```
> git pull
```

    → A jetzt auf Stand M







Git – Datenaustausch 1/2

- International durch verschiedene Protokolle
  - Git-Protokoll über TCP Port 9418
  - SSH (verschlüsselt)
  - HTTP
  - HTTPS
  - FTP
  - rsync
  - EMail (Patches)

Git – Datenaustausch 2/2

- Verschlüsselte Kommunikation per SSH
  - Benutzer erzeugt RSA Schlüsselpaar

```
> ssh-keygen -t rsa -C VORNAME.NACHNAME
```

~/.ssh/id_rsa ← Private Key

~/.ssh/id_rsa.pub ← Public Key

- Publickey auf Server kopieren
- Bei Anmeldung wird Private Key benötigt

  > git clone gitosis@hlb-labor.de:REPOSITORY

  - Server verschlüsselt Rechenaufgabe mit Public Key
  - Client entschlüsselt mit Private Key und sendet Ergebnis
  - Server akzeptiert Client

Git - Markierte Revisionen

> git tag MARKIERUNG

- Definiert Markierung für aktuellen Stand
- Markierung beschreibt z.B. Releaseversion
- Markierung später für Branch nutzbar

> git tag

- listet alle Markierungen auf

> man git-tag

Git - Interoperabilität

- Git's das auch anders?

- Kompatibilitätsinterface
  - GNU Arch (git-archimport)
  - svn (git-svn)
  - cvs
    (git-cvsexportcommit, git-cvsimport git-cvsserver)
  - Darcs (darcs-fastconvert, darcs2git)
  - Quilt (git-quiltimport)
  - manpages

Git - Web-Interface

- Hübscher Zugang zu git Repository
- Quelloffene Software
  - gitweb ist Teil der GIT Veröffentlichung
    → git-scm.com
  - GitLab ist sehr umfangreich
    → gitlab.com
  - gitalist in PERL geschrieben
    → github.com/broquaint/Gitalist
- Existing Webhoster (one)
  - GitHub – Closed Source hoster
    → github.com

Fragen?