Versionskontrolle: Subversion und Git

Ein Vortrag von
Sascha Schulz, sascha@s10z.de

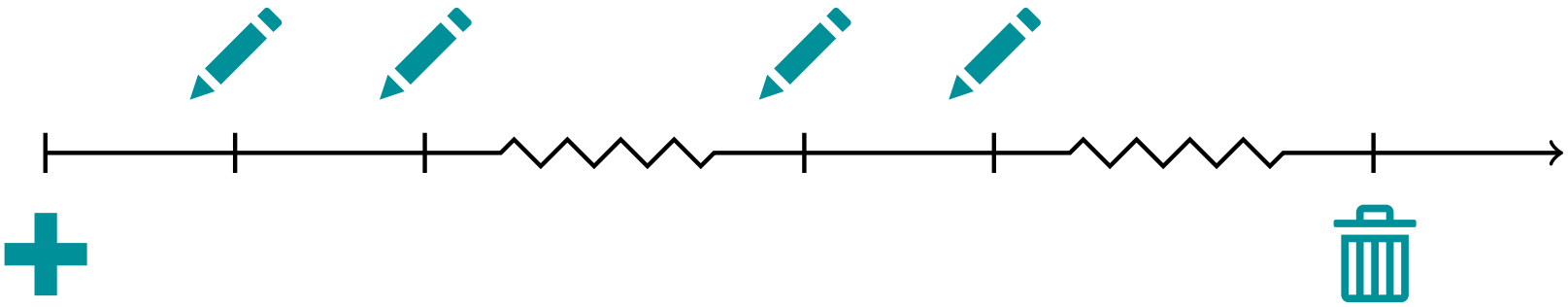
Universität Hamburg
Modul: Seminar Effiziente Programmierung
November 2016

Ablauf

1. Motivation: Warum versionieren?
2. Entwicklung und Verbreitung der verschiedenen Systeme
3. Kernunterschied SVN/git: Zentrale versus dezentrale VCS
4. Entwicklung einer intuitiven Vorstellung der Modelle
5. Fazit
6. *Optional:* Git Hands-on

Motivation: Warum versionieren?

Lebenszyklus eines Artefakts



1. Wie sah das Artefakt zum Zeitpunkt x aus?
2. Von wem stammt diese Änderung?
3. Warum wurde diese Änderung gemacht?

Manuelle Versionierung

/

Meine Ausarbeitung.odt

Manuelle Versionierung

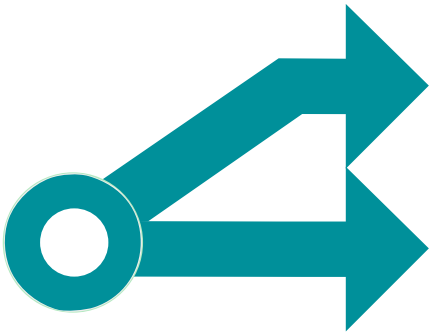
/

Meine Ausarbeitung.odt

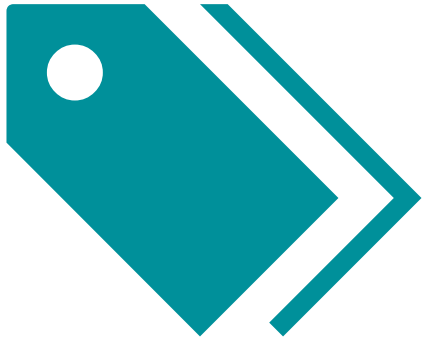
Meine Ausarbeitung v2.odt

Meine Ausarbeitung v3.odt

Kern-Features: Einstiegspunkte für den Anwender



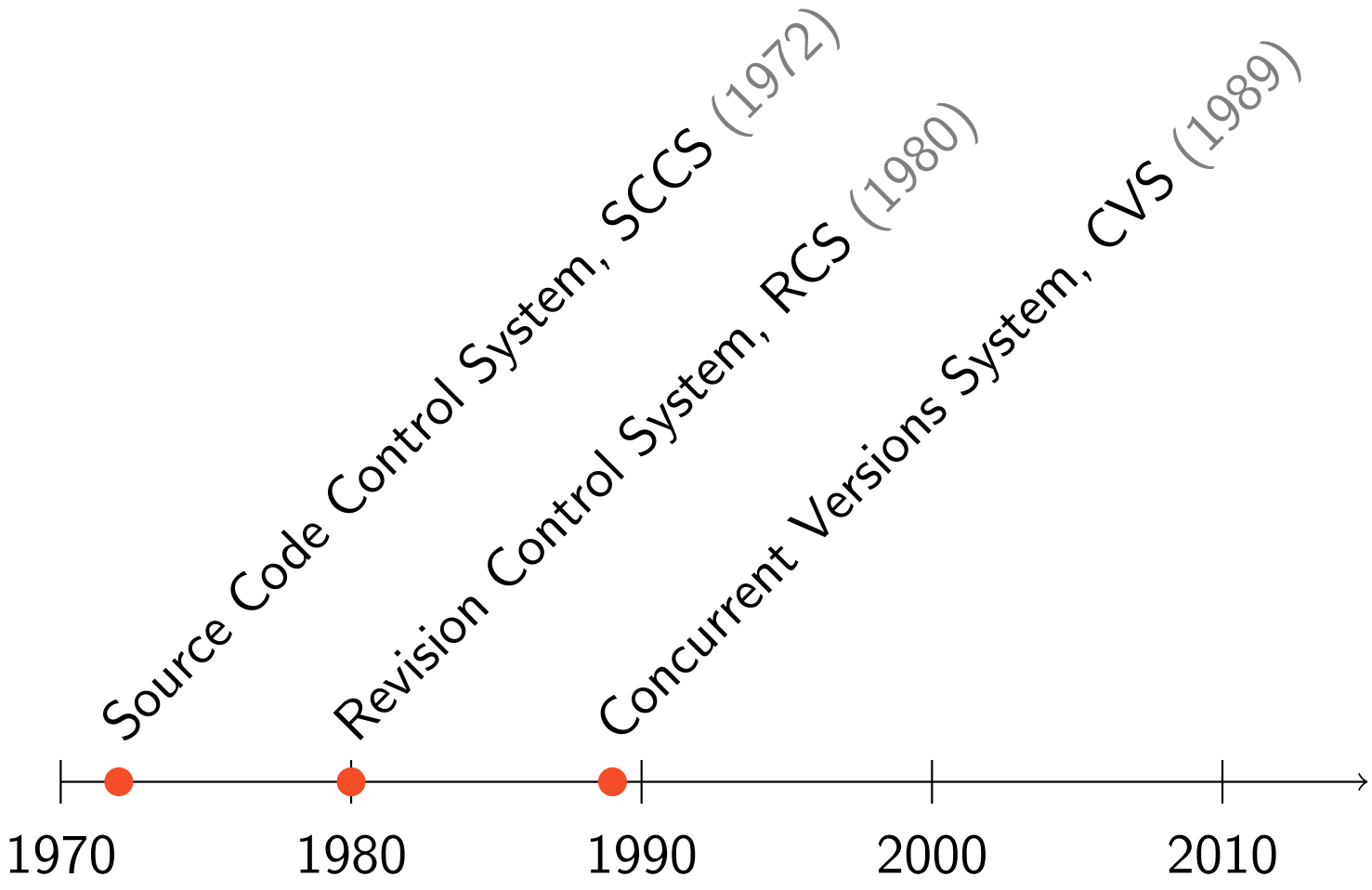
Branches



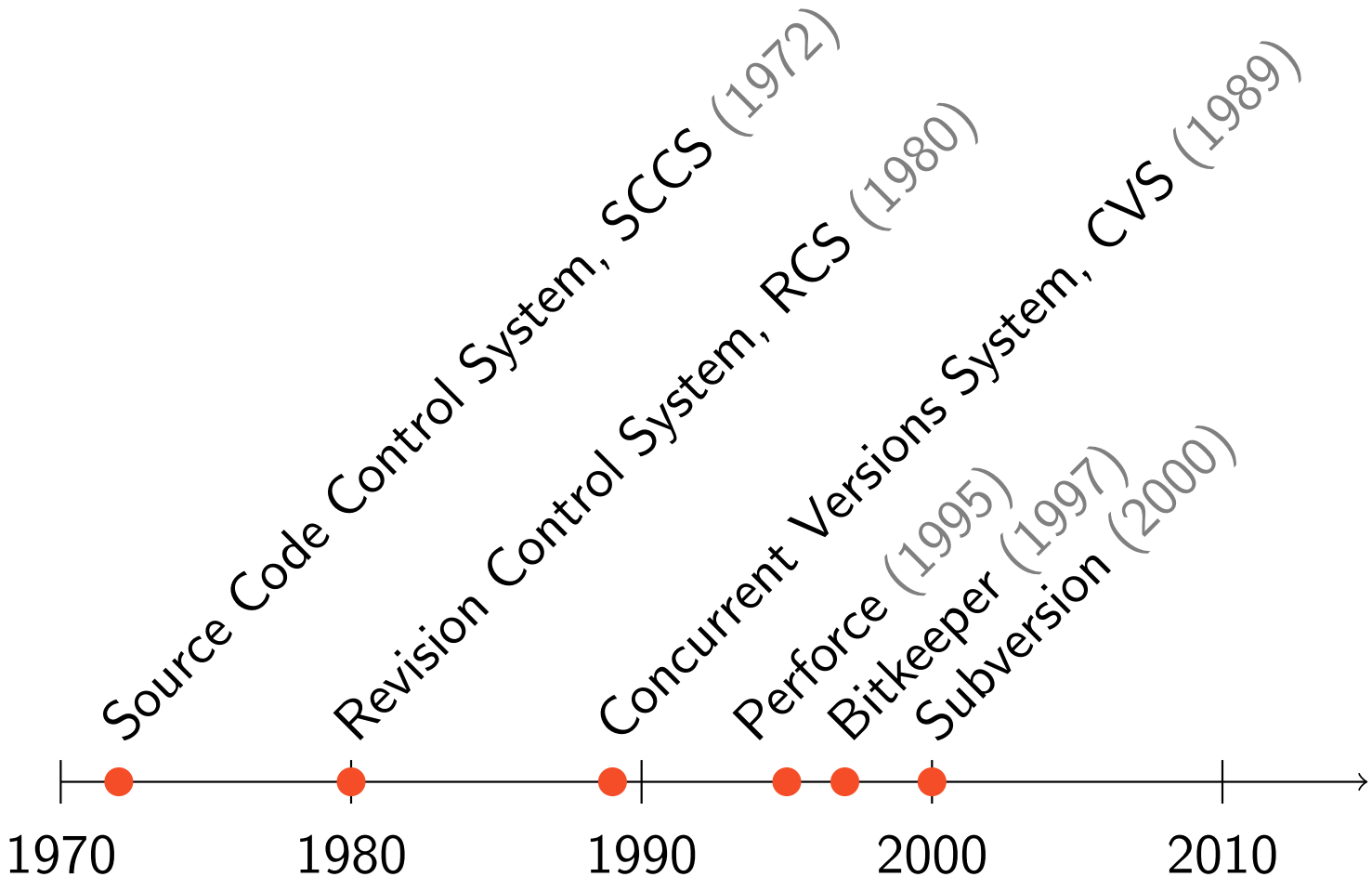
Tags

Entwicklung und Verbreitung der verschiedenen Versions-Kontroll-Systeme (*Version Control Systems, VCSs*)

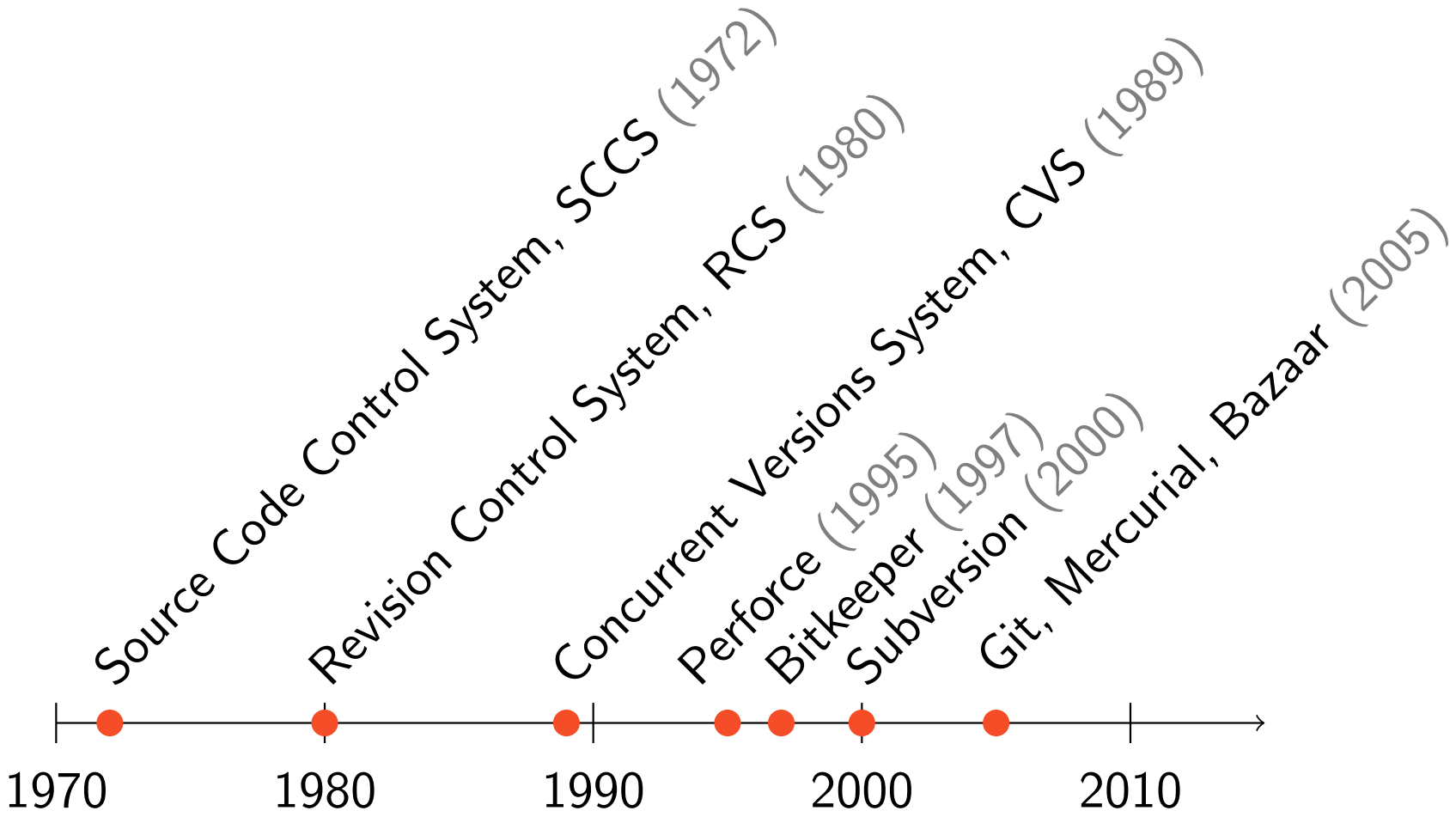
Entwicklungstart einschlägiger VCSs



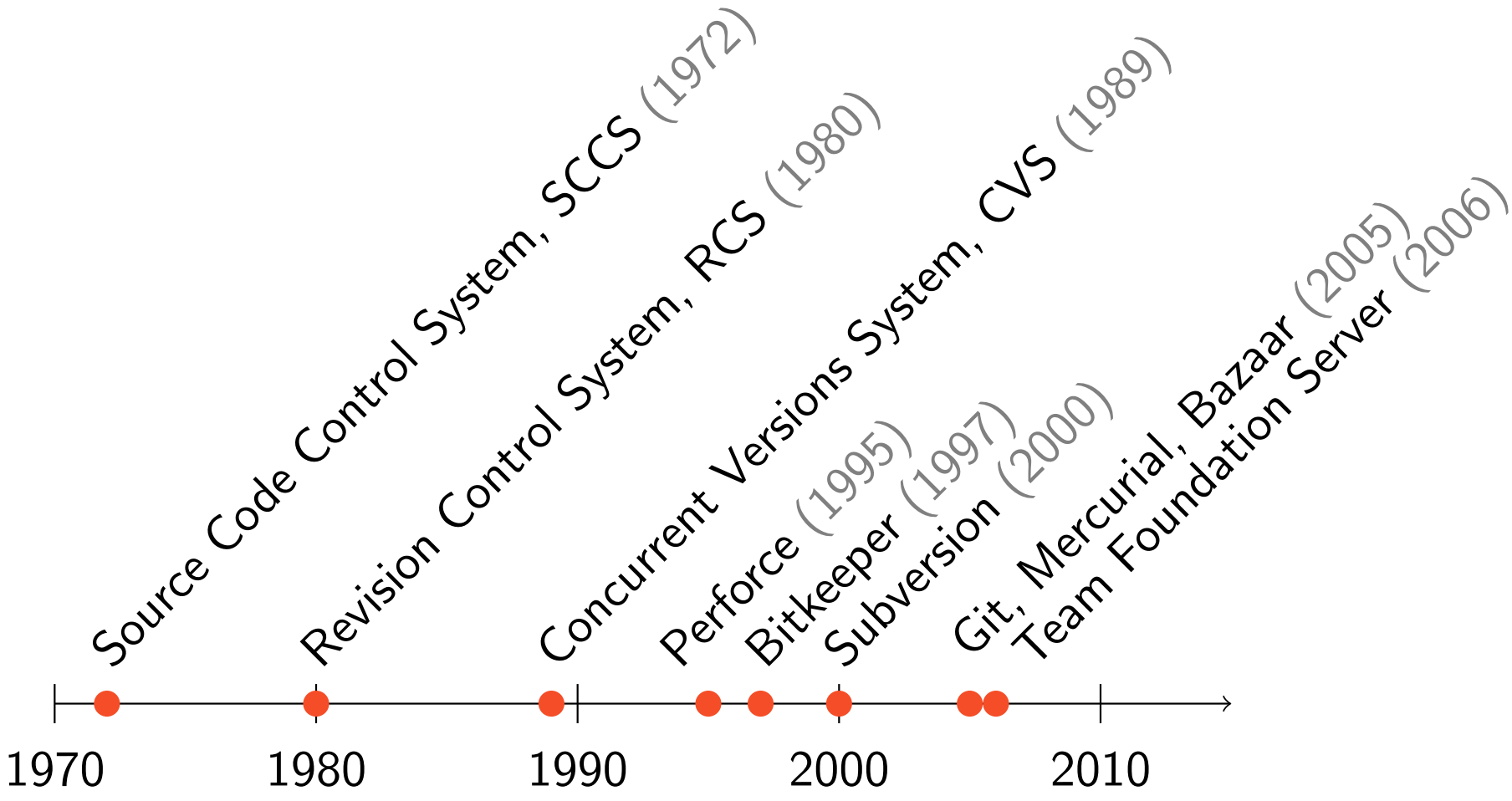
Entwicklungstart einschlägiger VCSs



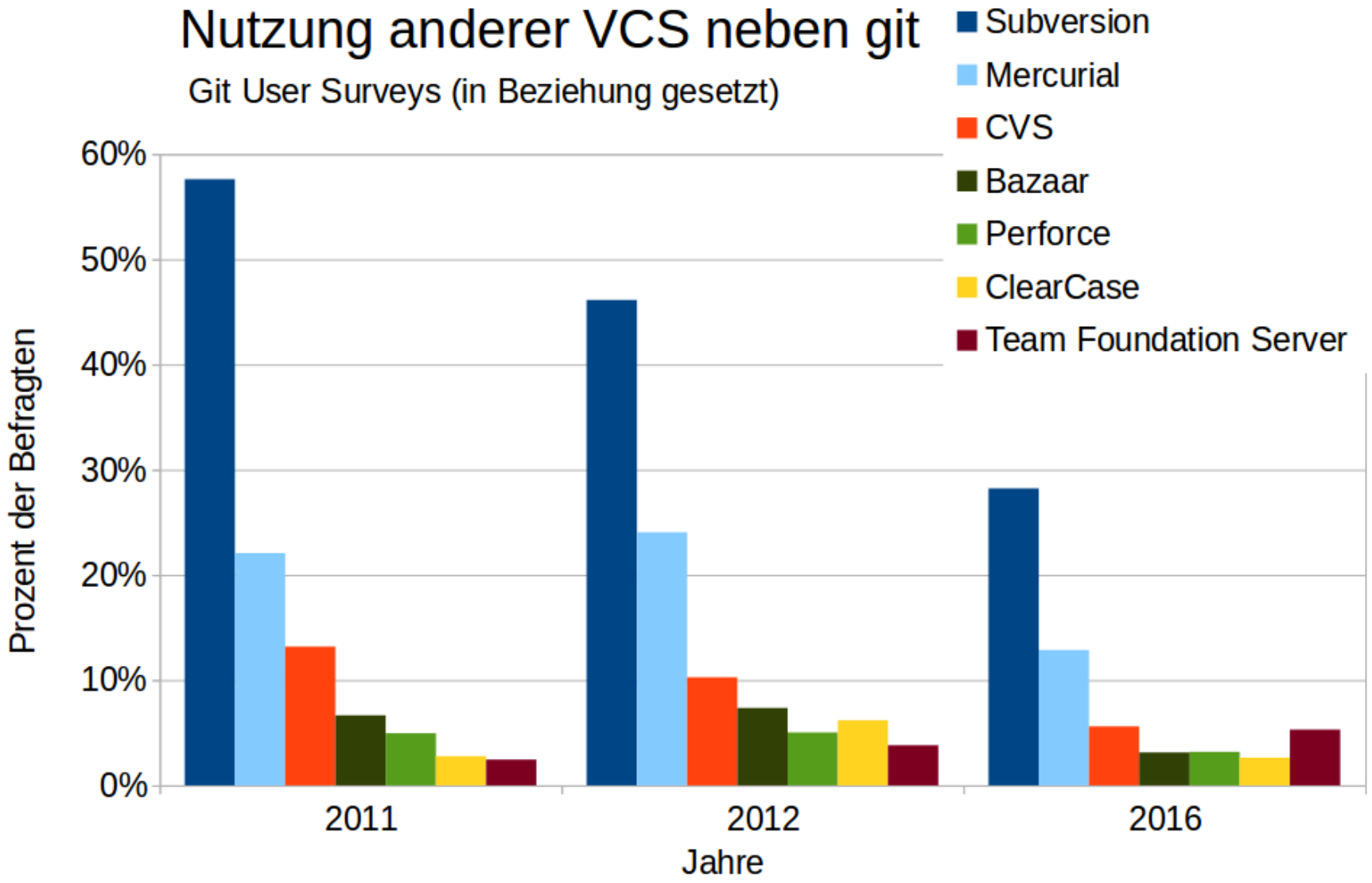
Entwicklungstart einschlägiger VCSs



Entwicklungstart einschlägiger VCSs



Entwicklung innerhalb der Git User Surveys

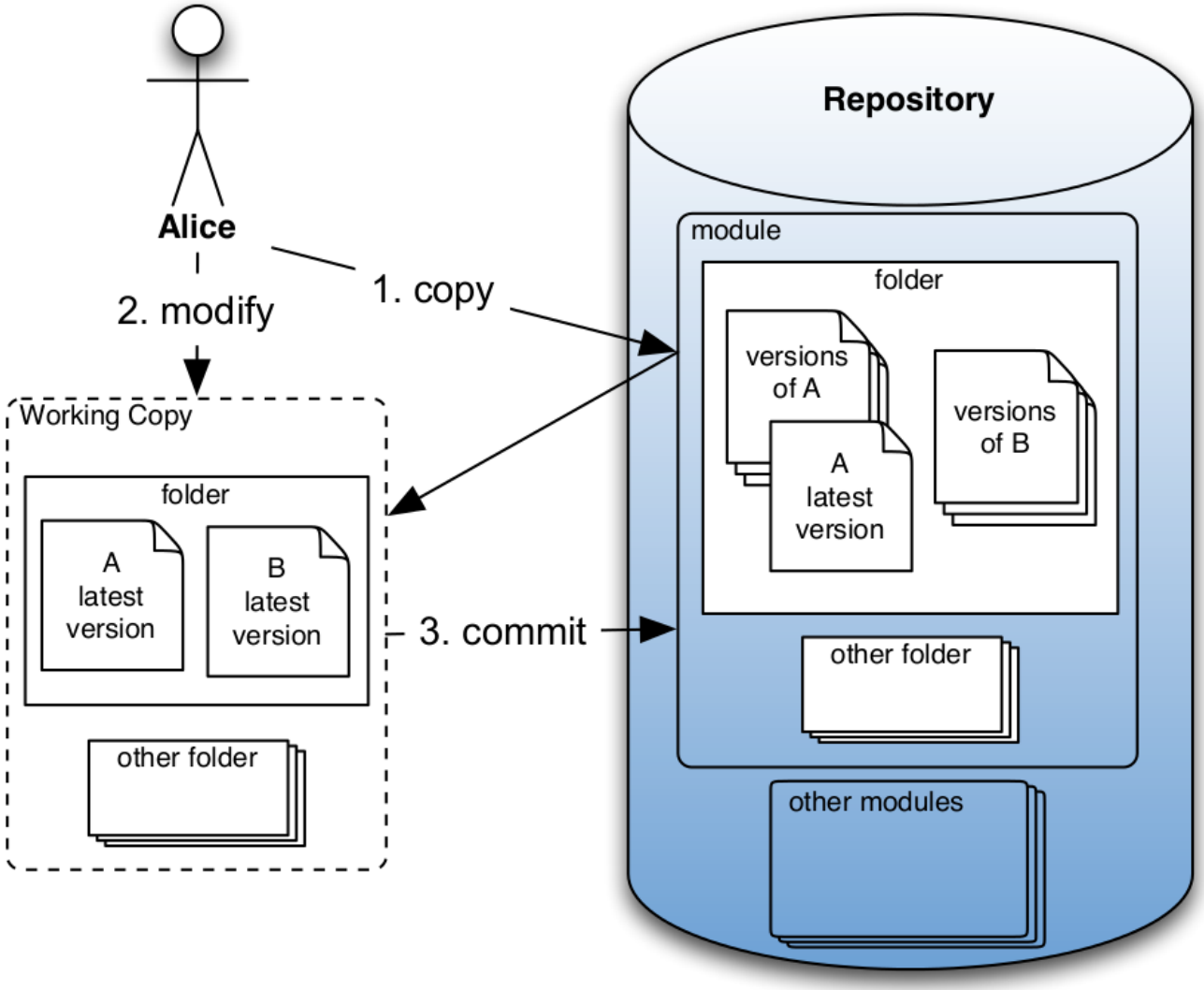


Mehrfachnennungen erlaubt. Nennungen im Bezug zur Basis aus Antworten+Enthaltungen gesetzt.

Kernunterschied SVN/git

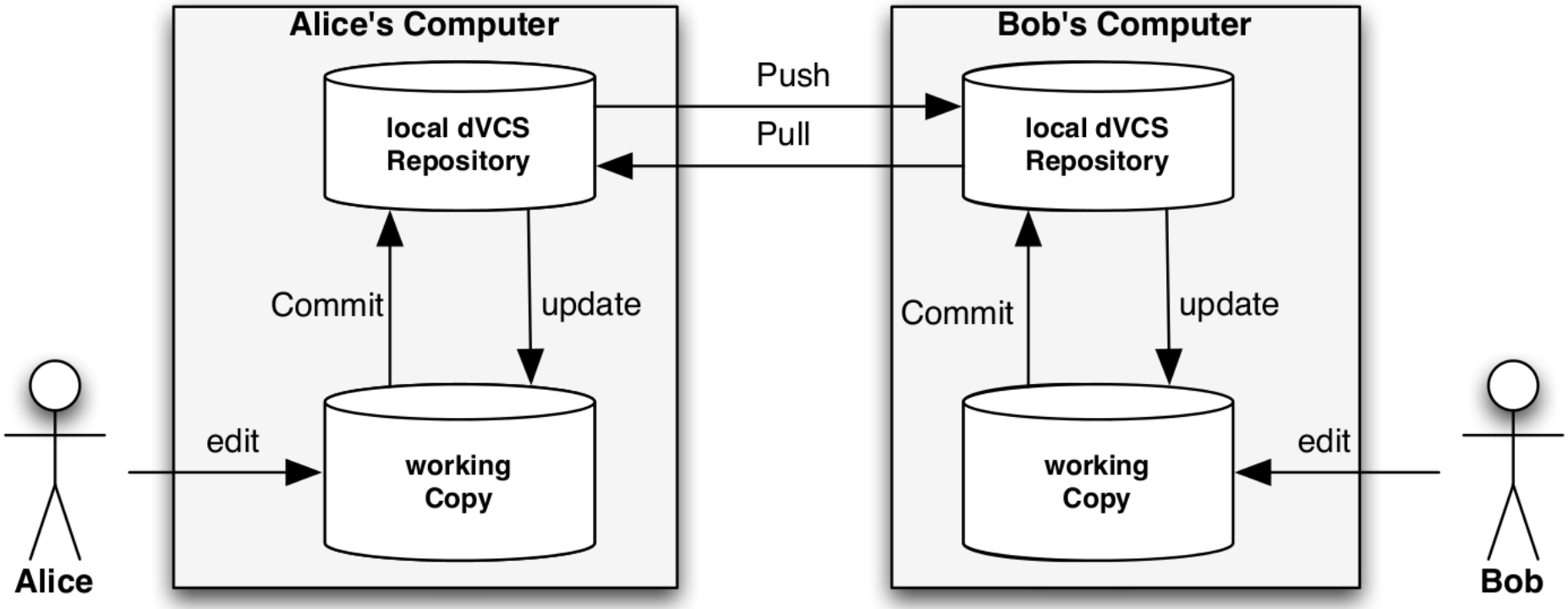
zentrales Versions-Kontroll-System (cVCS)
trifft auf
dezentrales Versions-Kontroll-System (dVCS)

Abstrakter Arbeitsablauf im cVCS



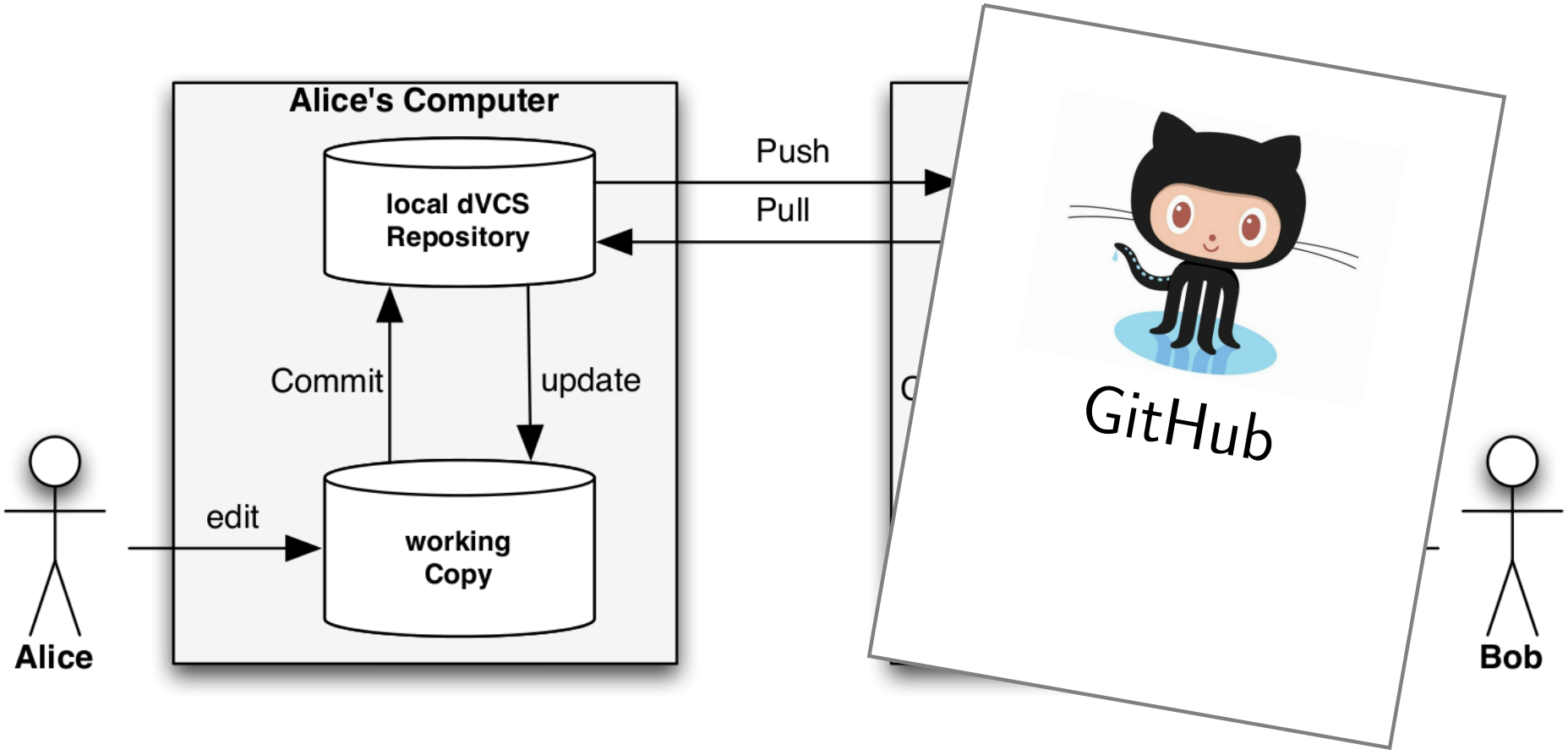
Quelle: Mukherjee2010

Abstrakter Arbeitsablauf im dVCS



Quelle: Mukherjee2010

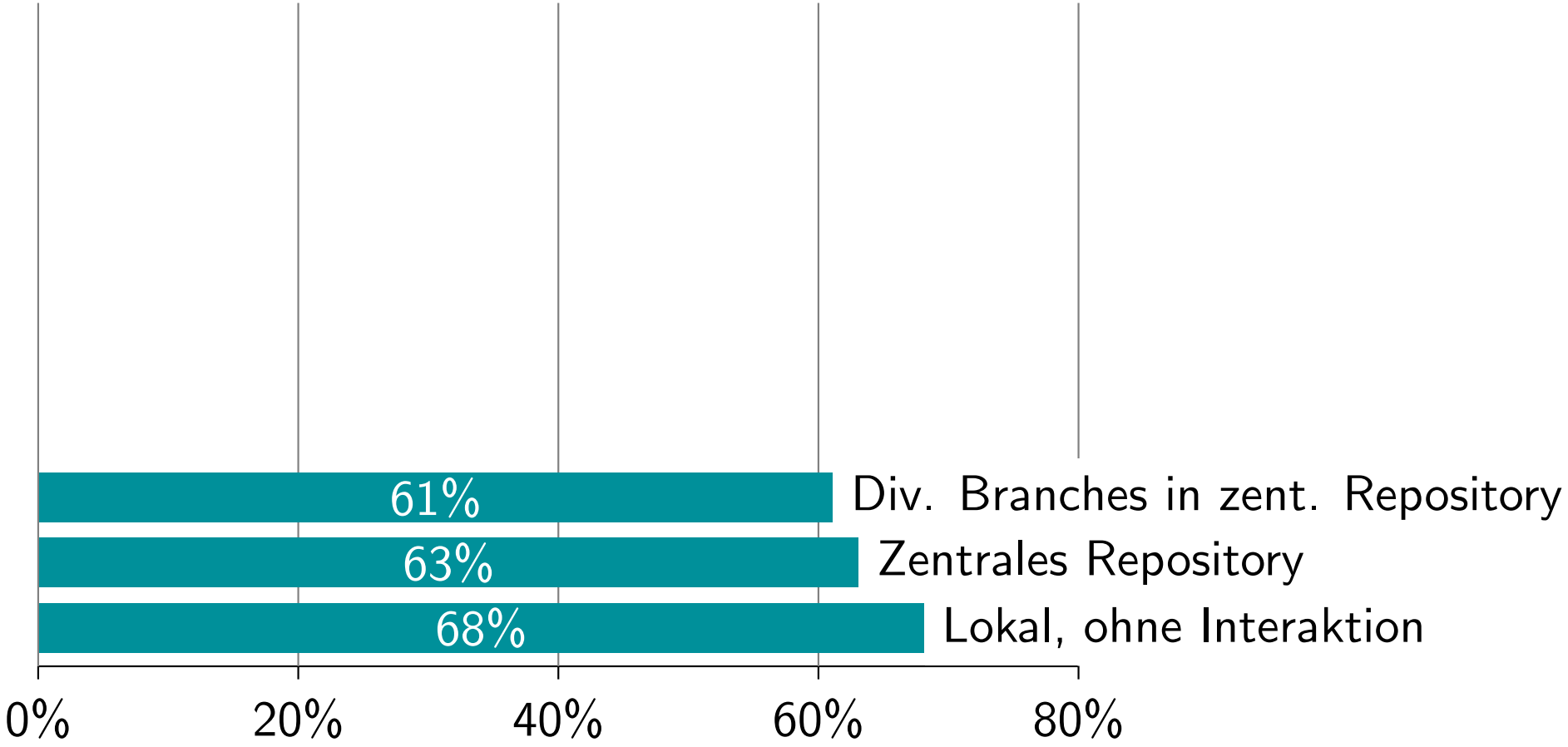
Abstrakter Arbeitsablauf im dVCS



Quelle: Mukherjee2010

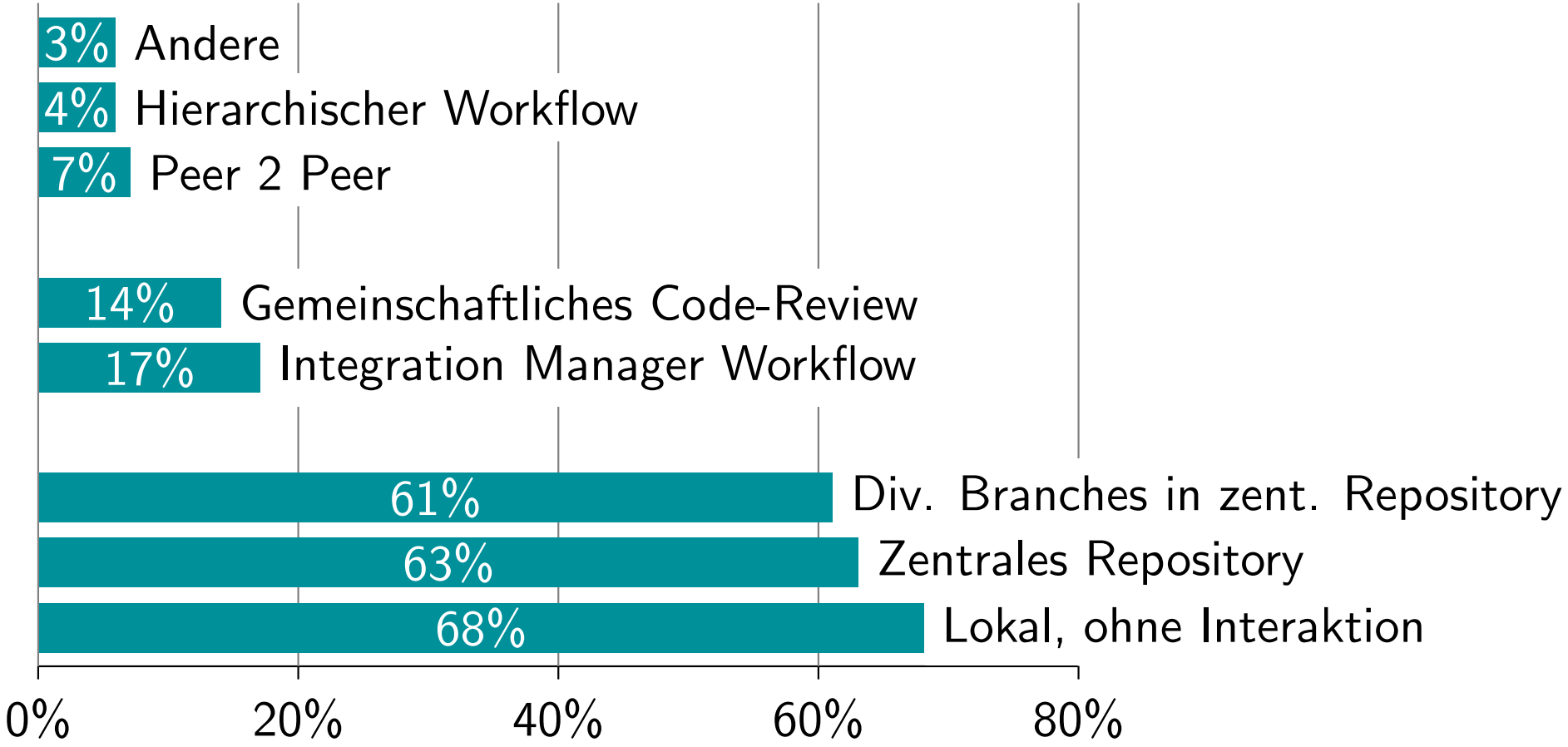
Nutzung der dezentralen Möglichkeiten?

Verwendete Git-Workflows laut Git User Survey 2016.
(*Mehrfachnennungen erlaubt*)



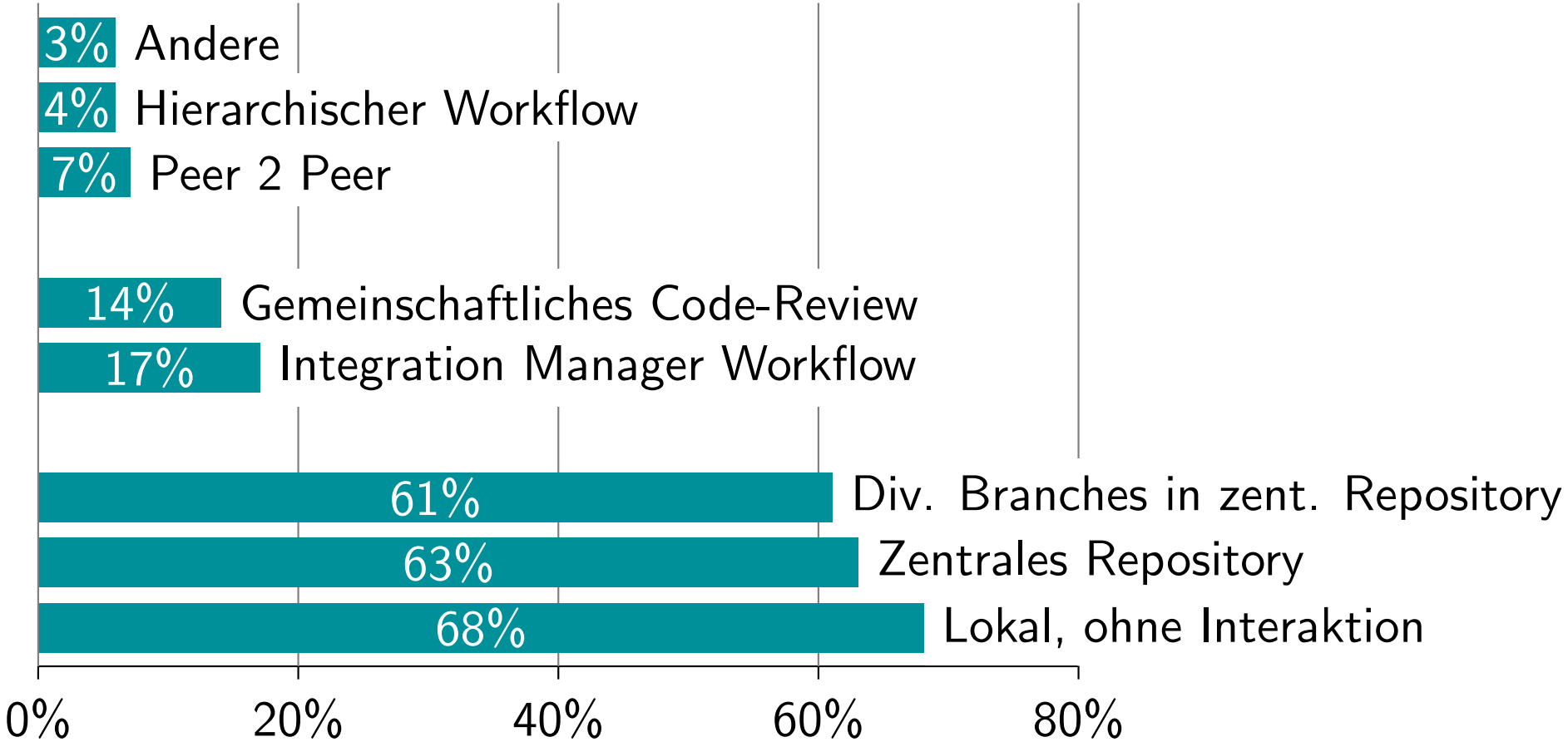
Nutzung der dezentralen Möglichkeiten?

Verwendete Git-Workflows laut Git User Survey 2016.
(*Mehrfachnennungen erlaubt*)



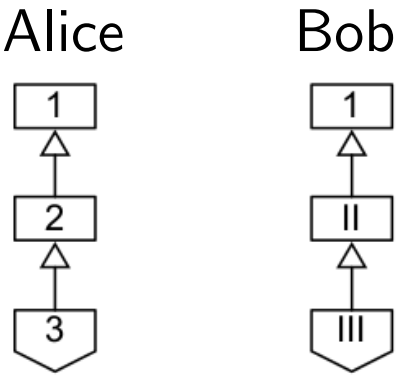
Nutzung der dezentralen Möglichkeiten?

Verwendete Git-Workflows laut Git User Survey 2016.
(*Mehrfachnennungen erlaubt*)



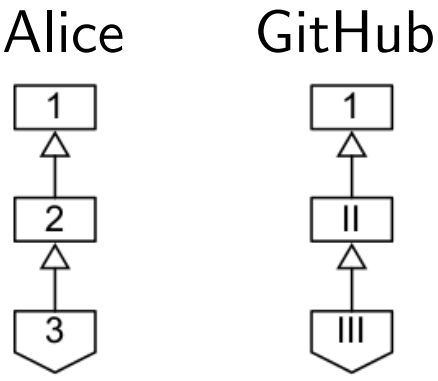
Bemerkenswert: **37%** der Befragten verwenden u.a. Repositories mit nur einem Branch

Abstrakter Arbeitsablauf im dVCS



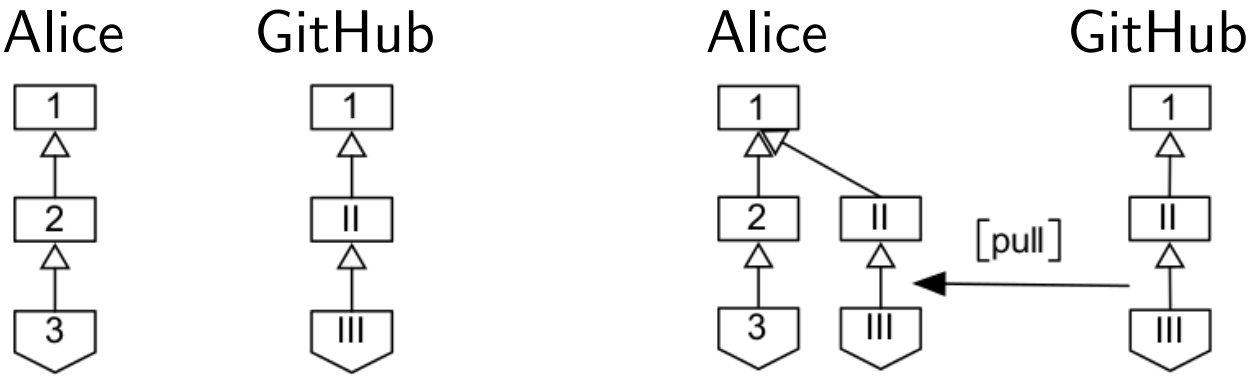
Quelle: Mukherjee2010

Abstrakter Arbeitsablauf im dVCS



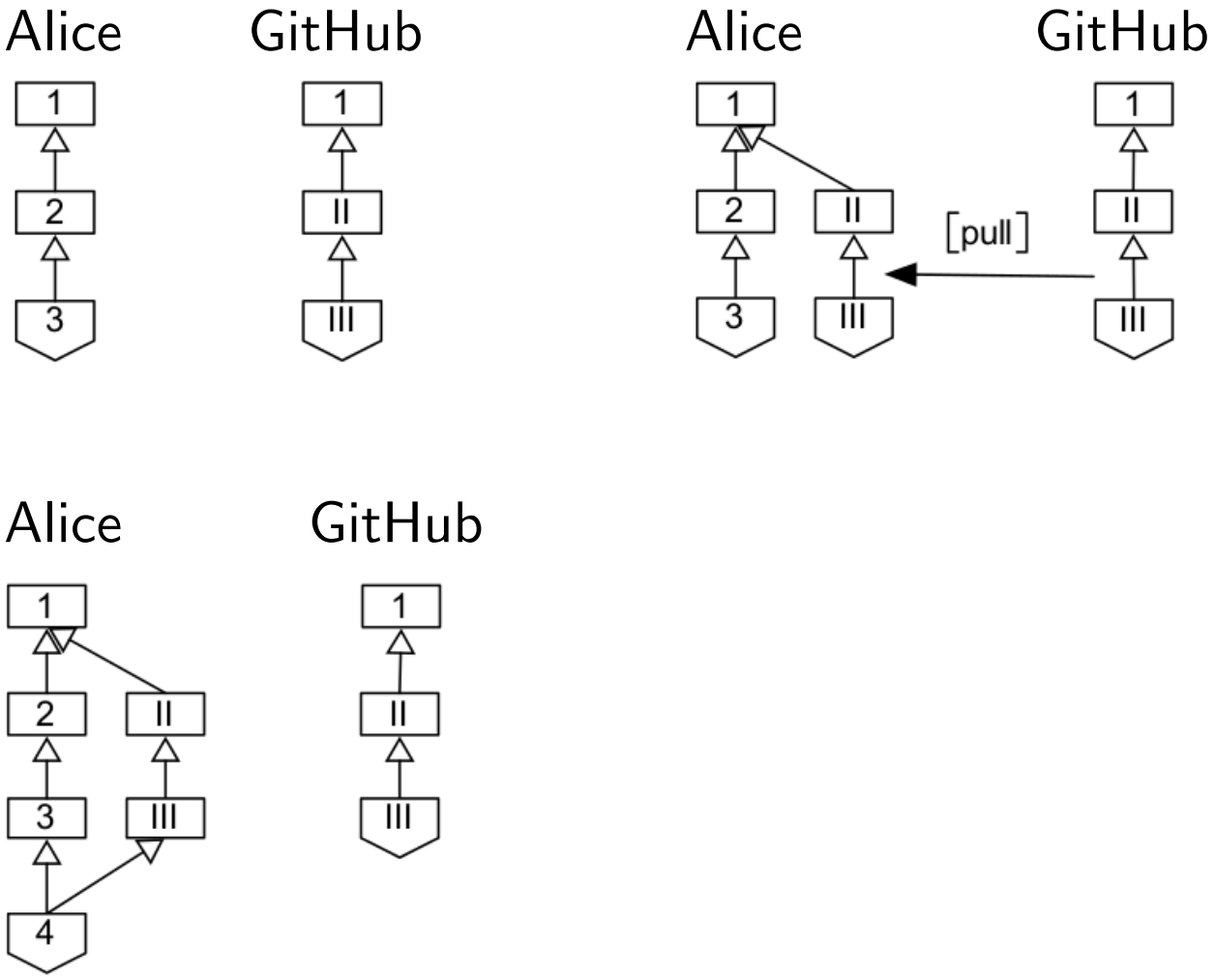
Quelle: Mukherjee2010

Abstrakter Arbeitsablauf im dVCS



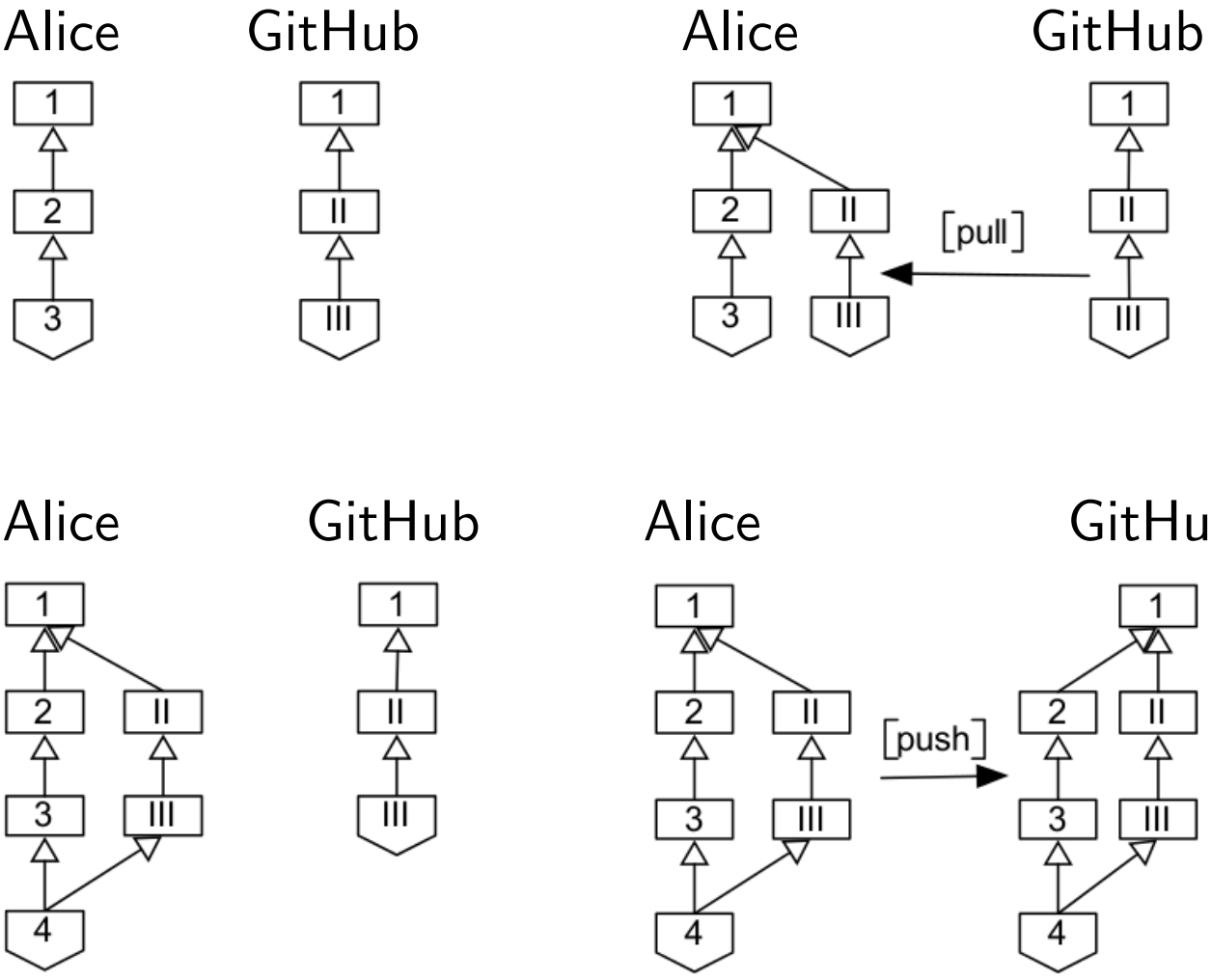
Quelle: Mukherjee2010

Abstrakter Arbeitsablauf im dVCS



Quelle: Mukherjee2010

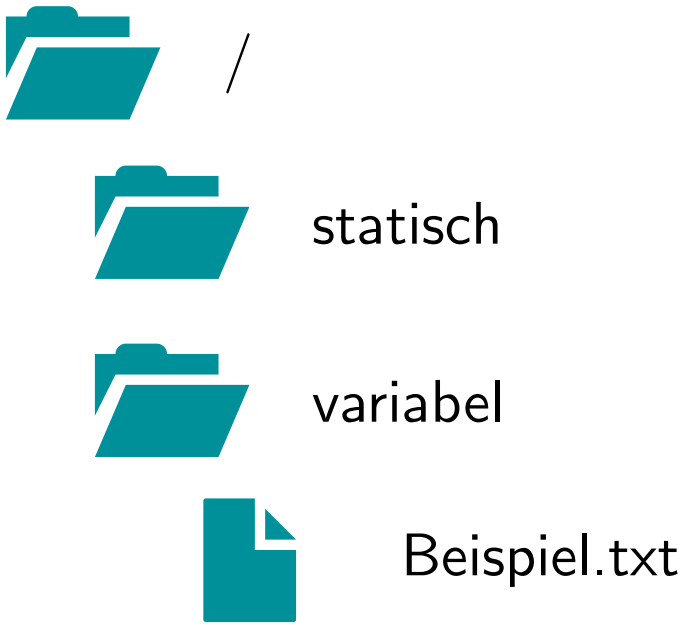
Abstrakter Arbeitsablauf im dVCS



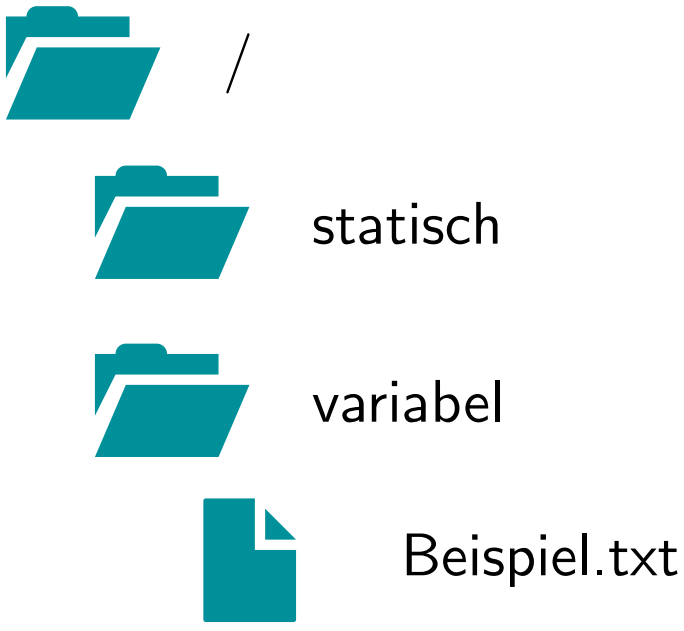
Quelle: Mukherjee2010

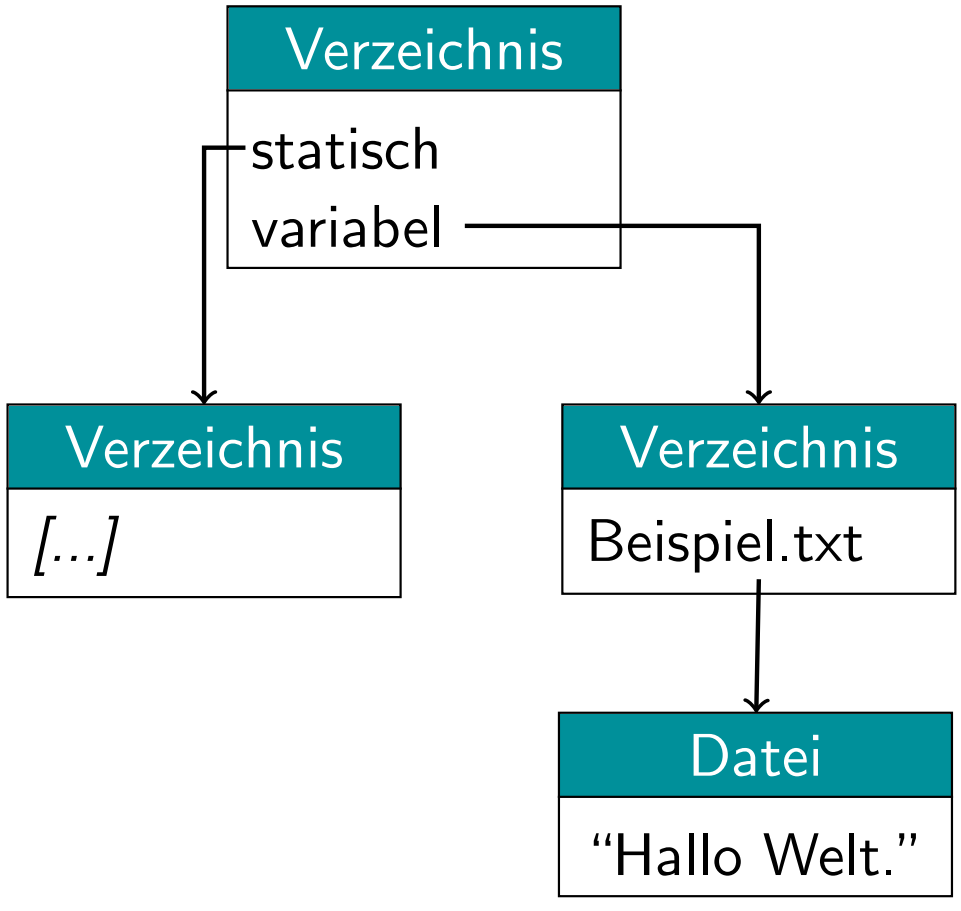
Entwicklung einer intuitiven Vorstellung der Modelle

Datenstruktur zur Abbildung des Dateisystems

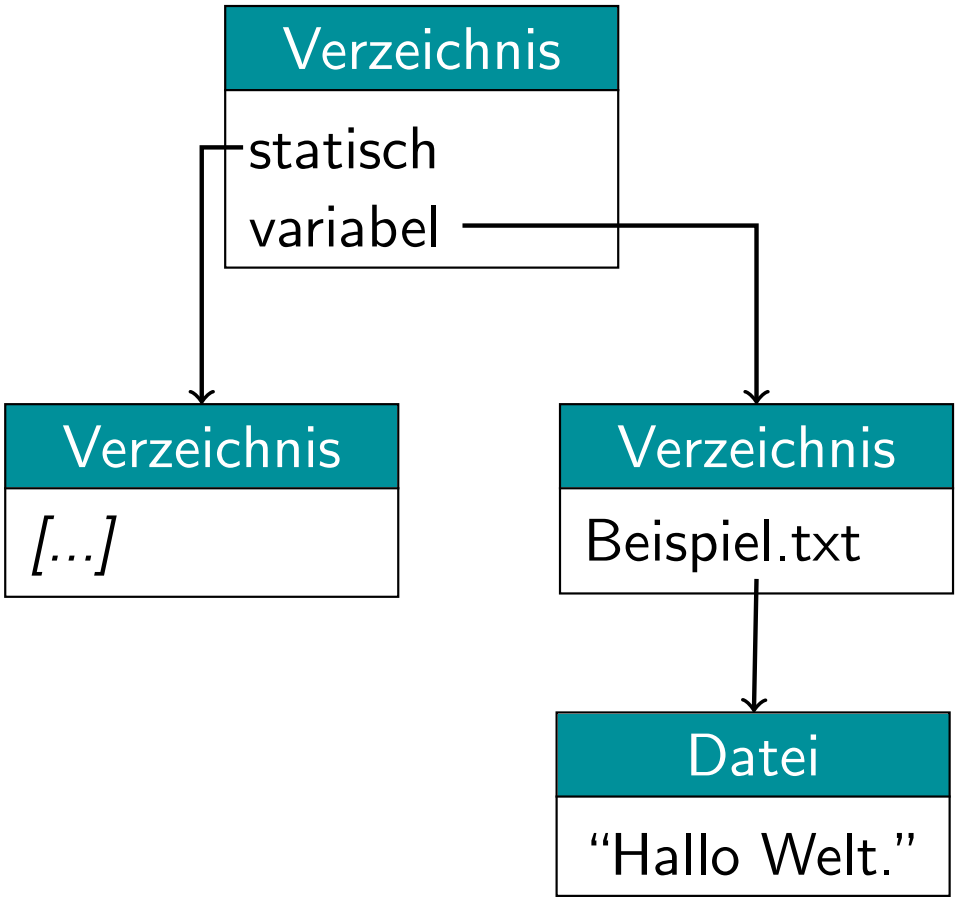


Datenstruktur zur Abbildung des Dateisystems

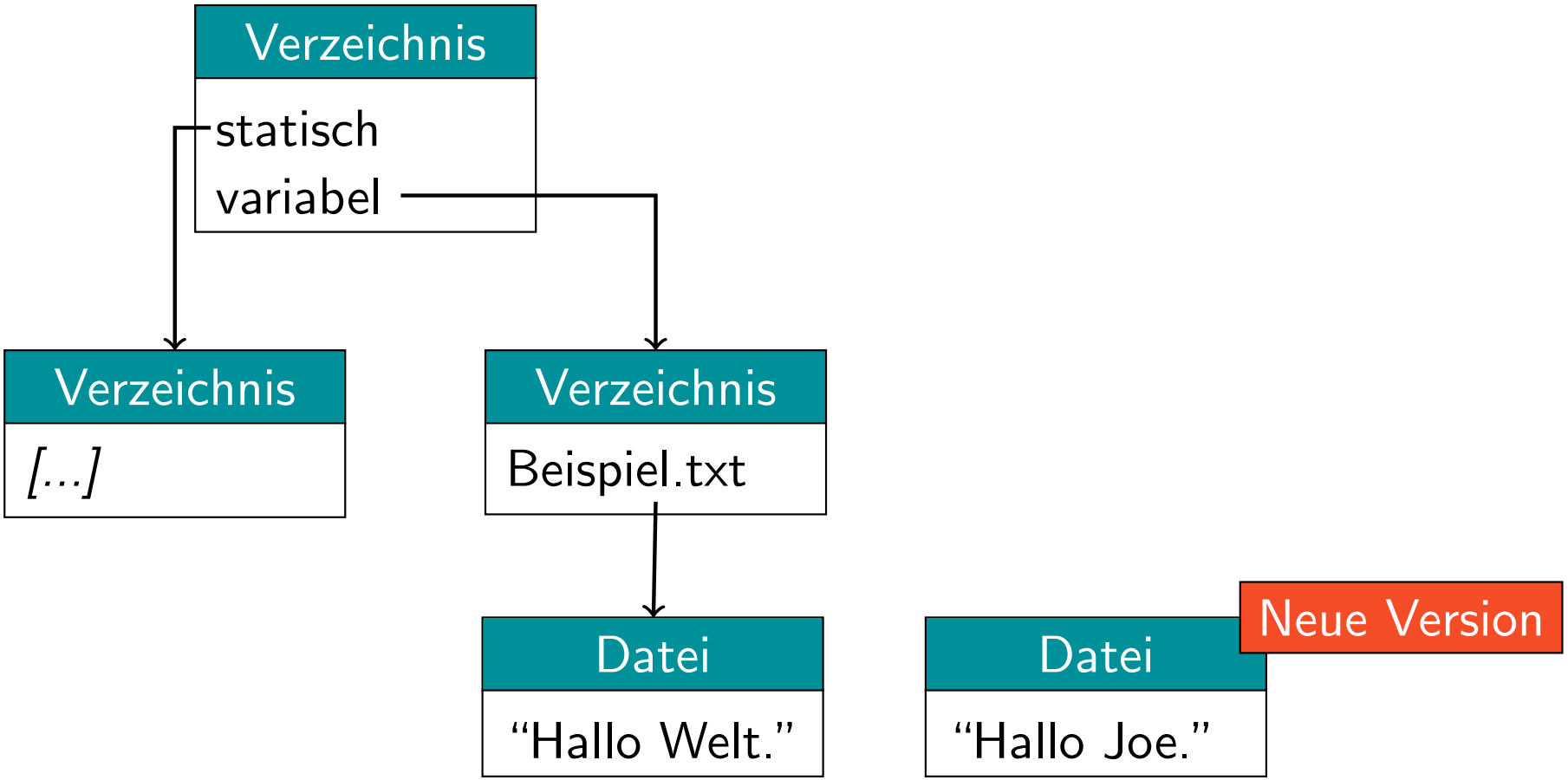




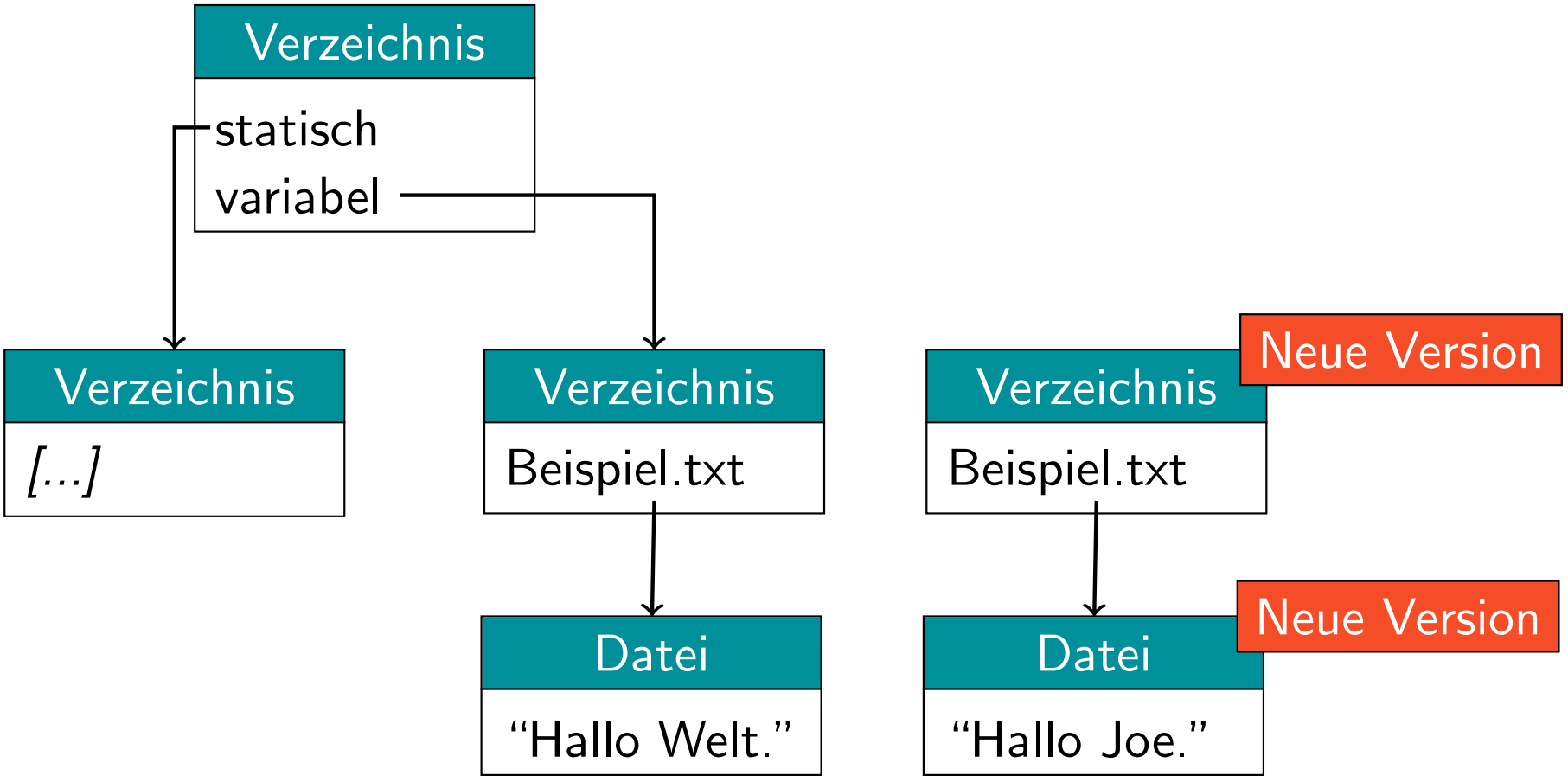
Auswirkung der Bearbeitung einer Datei



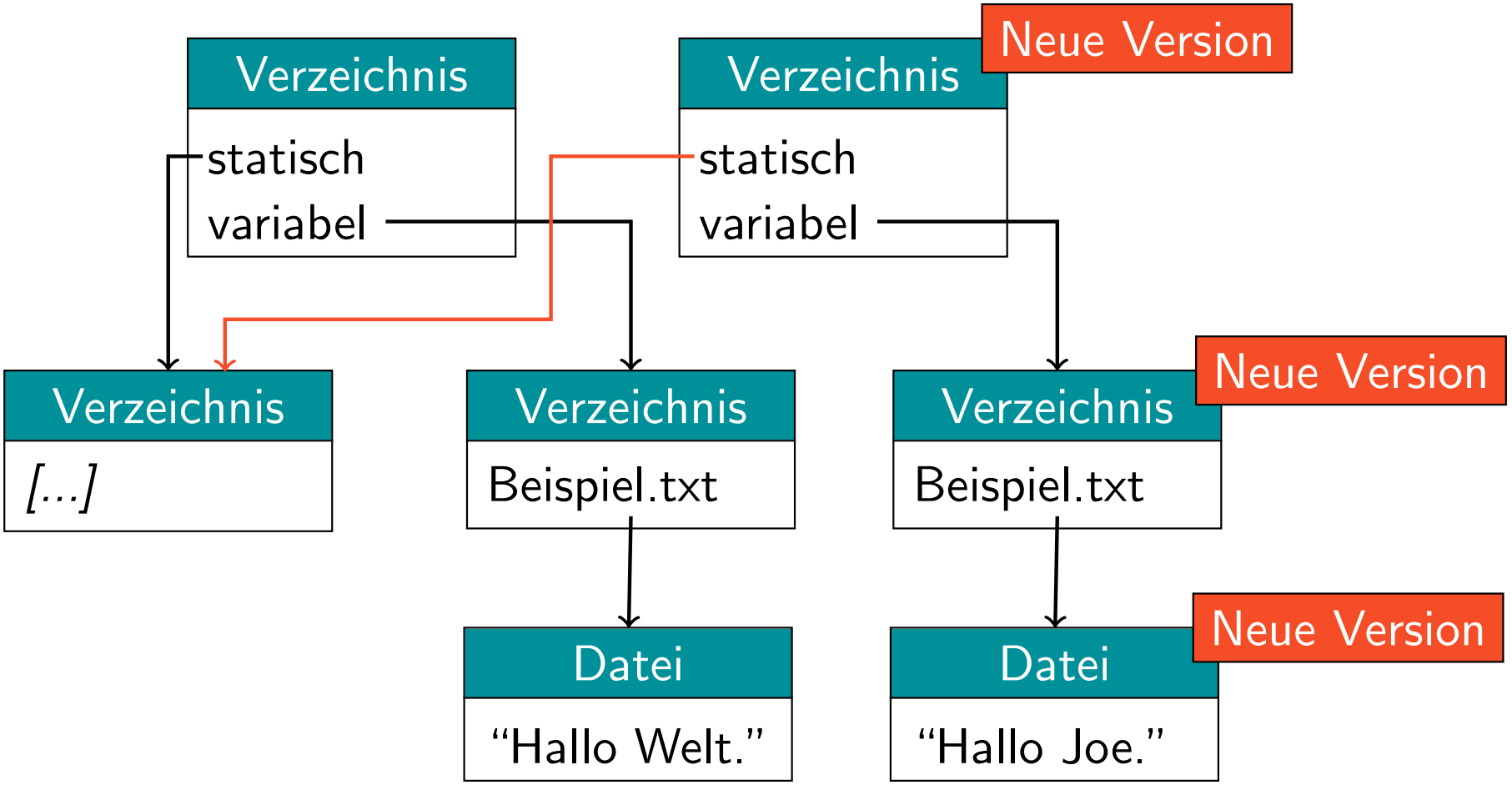
Auswirkung der Bearbeitung einer Datei



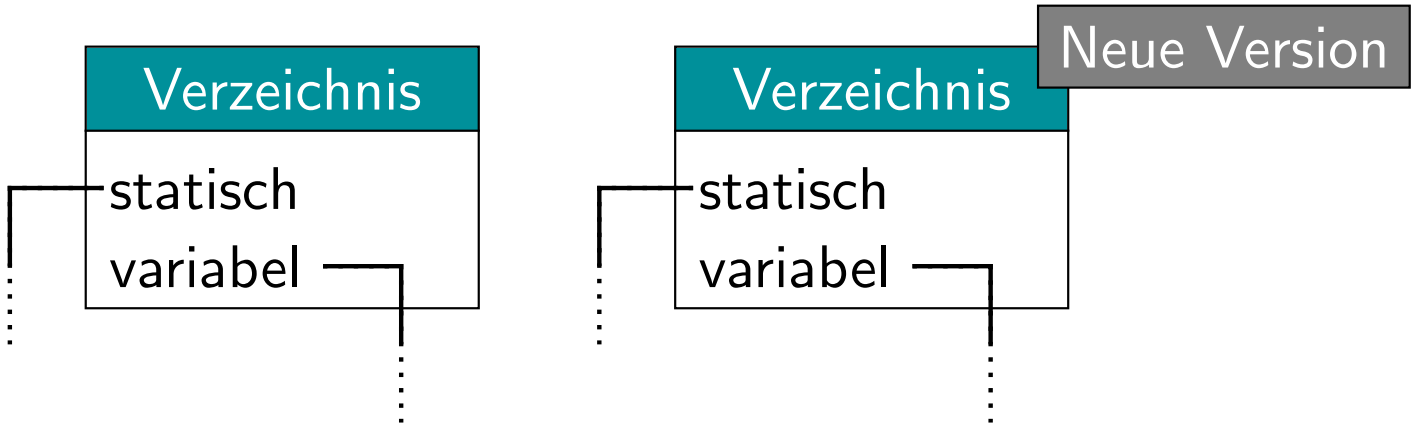
Auswirkung der Bearbeitung einer Datei



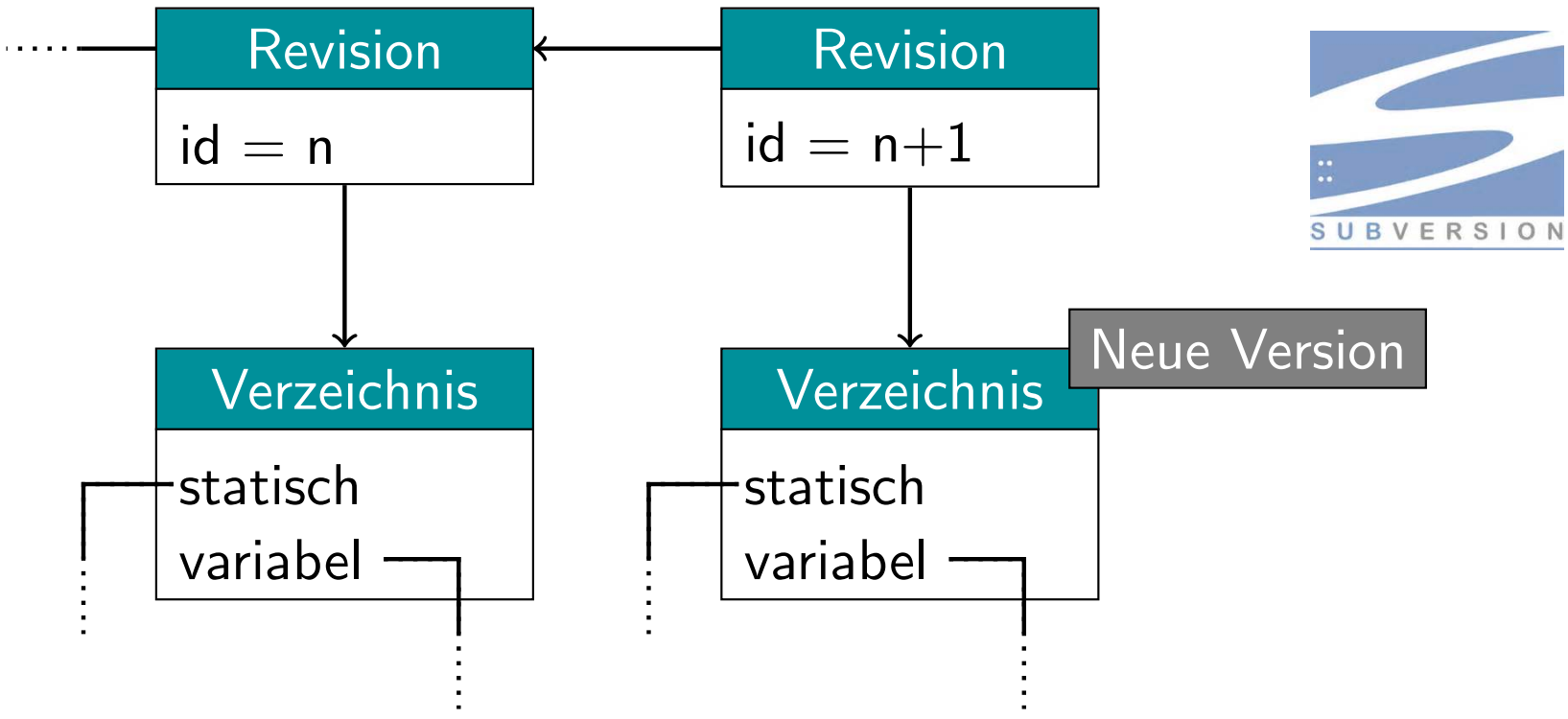
Auswirkung der Bearbeitung einer Datei



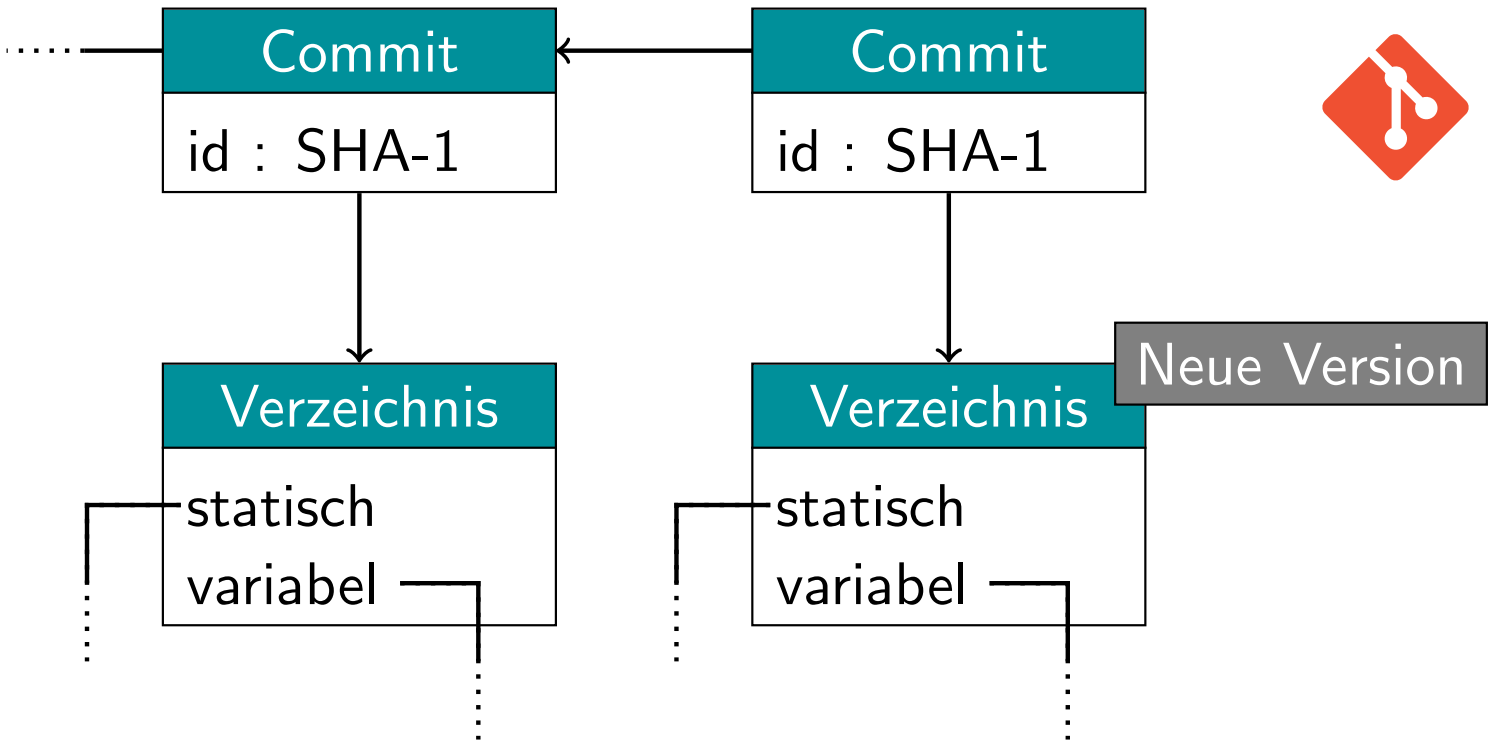
Anbindung der Datenstruktur an das Versionsmodell



Anbindung der Datenstruktur an das Versionsmodell



Anbindung der Datenstruktur an das Versionsmodell



Git identifiziert Commits durch Hashes

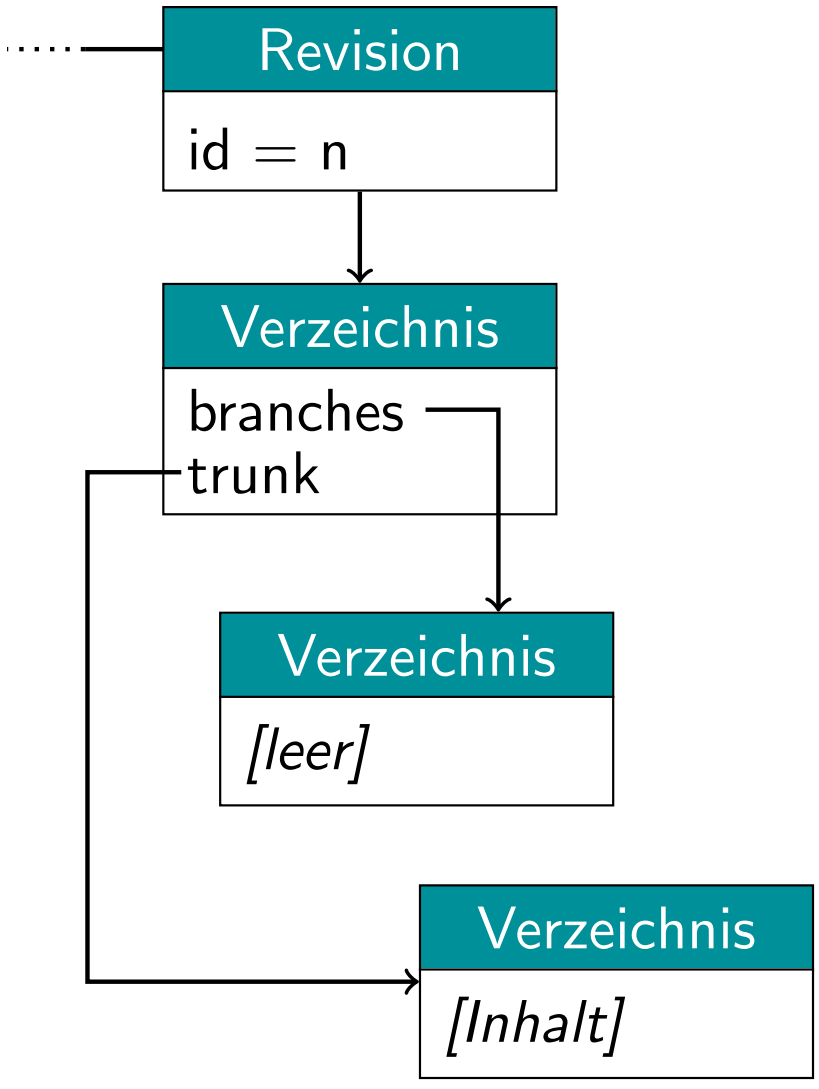
Hash-Alg. SHA-1

f246a1ffd416b9b1a939cdbd4f9c4a821474d14c

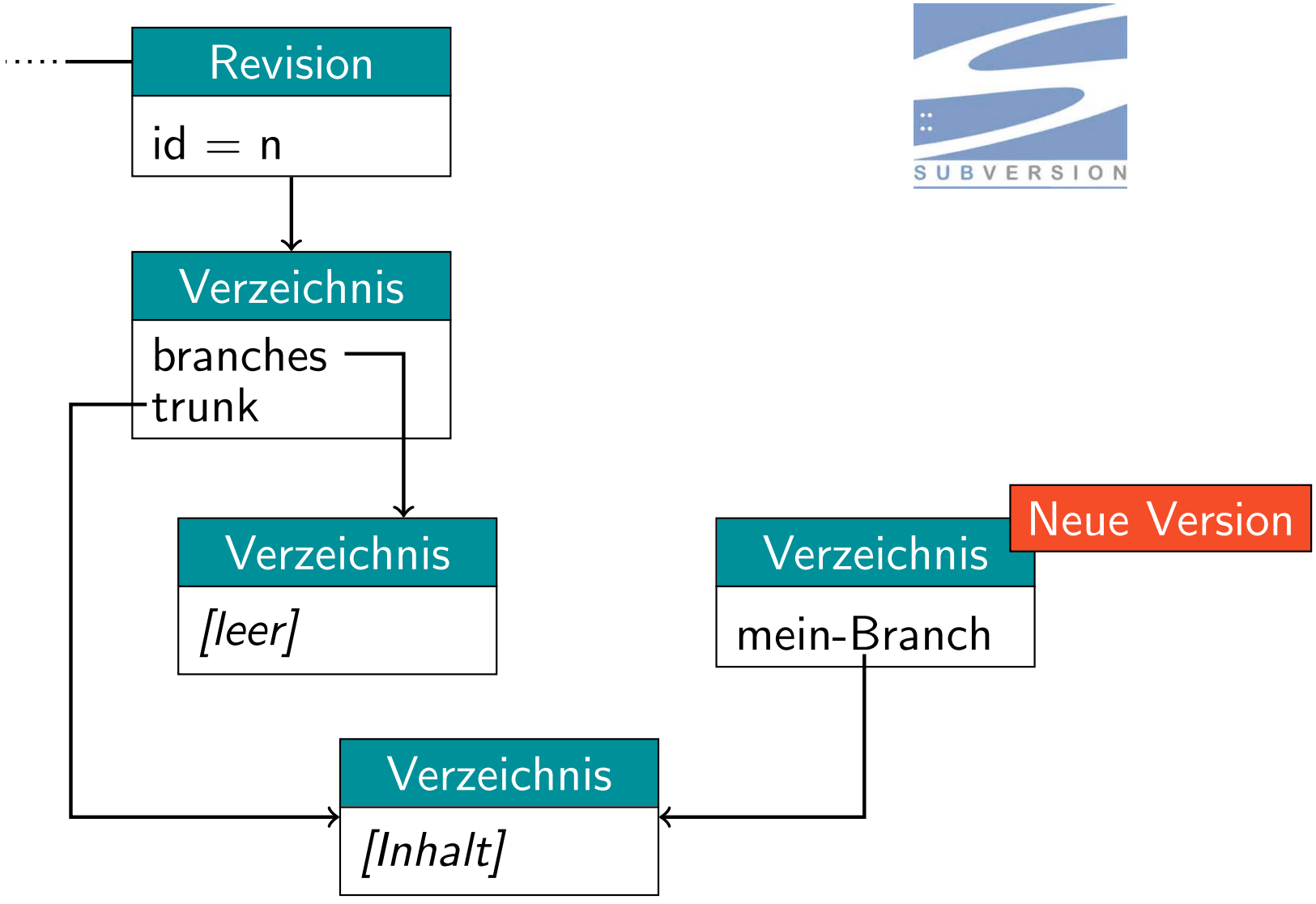
Dateisystem-Konventionen

Dateisystem-Konventionen

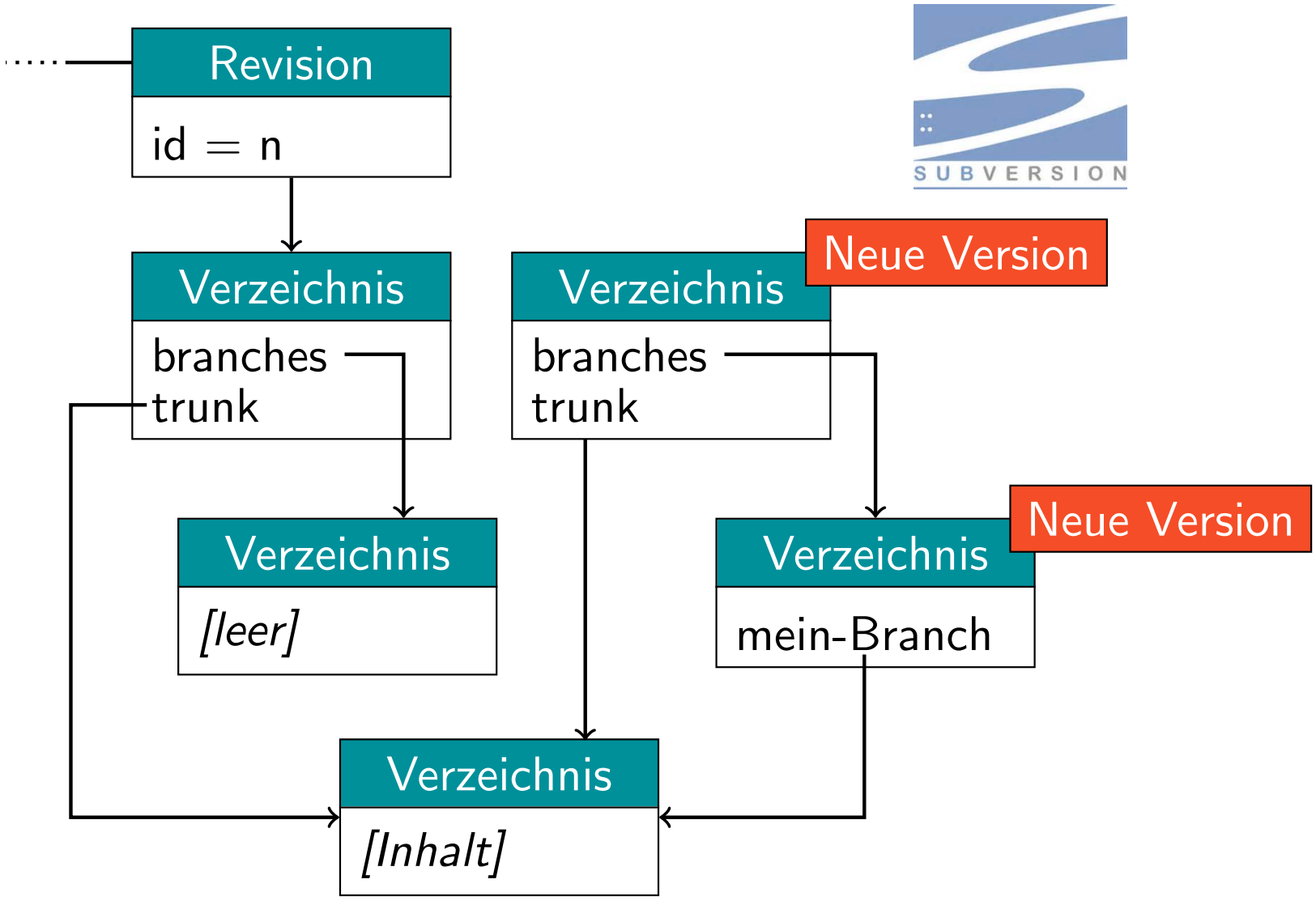
Hintergrund von Subversions Dateisystem-Konvention



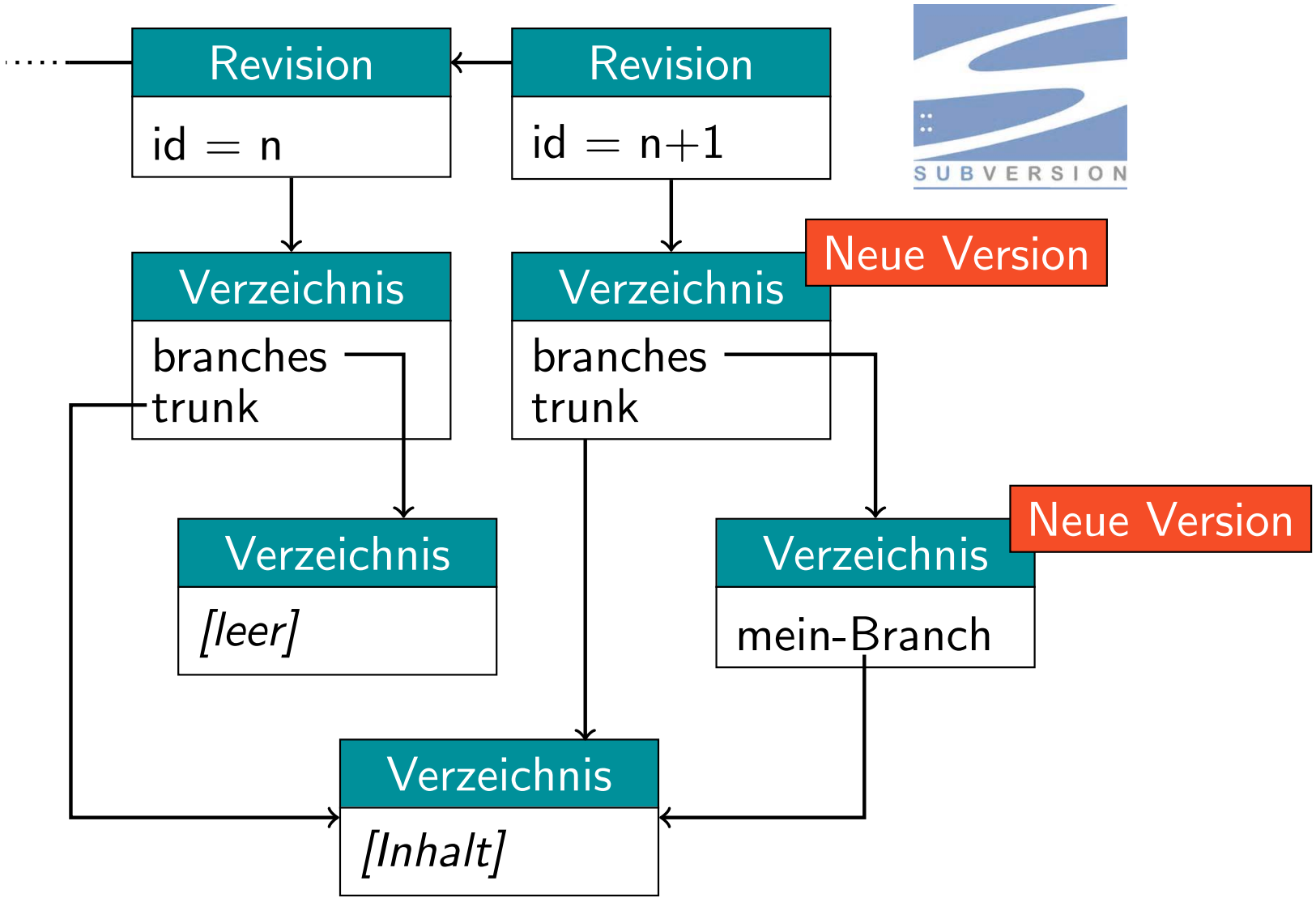
Hintergrund von Subversions Dateisystem-Konvention



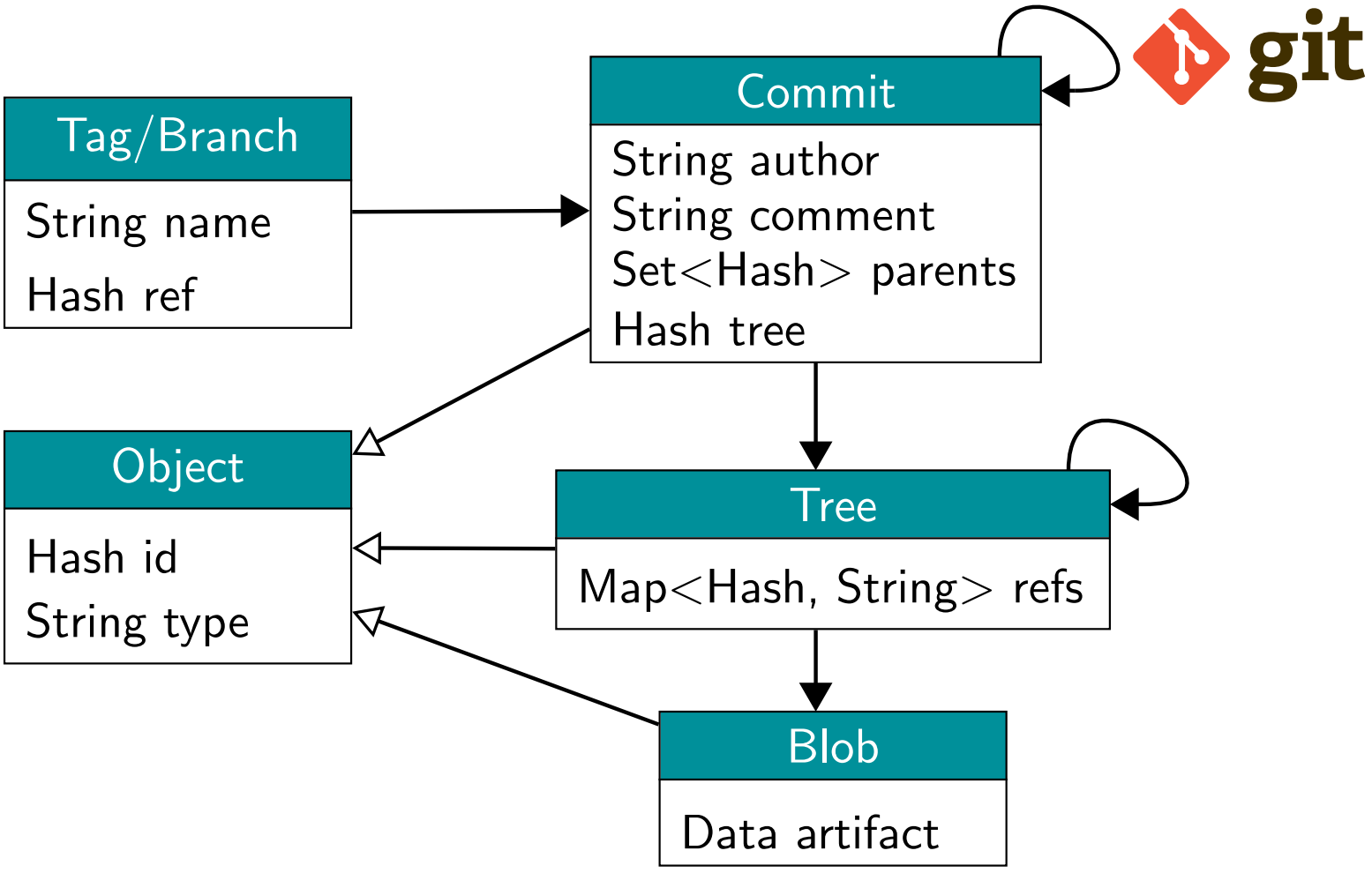
Hintergrund von Subversions Dateisystem-Konvention



Hintergrund von Subversions Dateisystem-Konvention



Hintergrund für Gits Flexibilität



Darstellungen auf Basis von Mukherjee2010

Fazit

Fazit: Allgemein

- Versionierung wird durch Tools unterstützt
- Vielzahl an verschiedenen VCSs,
  auch veraltete sind noch im Einsatz
- Primäre Entscheidung: Zentral oder dezentral?
- Mit dezentralen VCSs werden trotzdem gern
  zentrale Strukturen etabliert

Fazit: SVN versus Git

- Hohe Ähnlichkeit im Modell des Dateisystems
- Essentieller Unterschied im Modell der Versionen
- Gerichteter azyklischer Graph (DAG) statt Revisions-Kette ermöglicht höhere Flexibilität
- Die Möglichkeiten von Git werden oft nur wenig genutzt

Fazit: SVN versus Git

- Hohe Ähnlichkeit im Modell des Dateisystems
- Essentieller Unterschied im Modell der Versionen
- Gerichteter azyklischer Graph (DAG) statt Revisions-Kette ermöglicht höhere Flexibilität
- Die Möglichkeiten von Git werden oft nur wenig genutzt

- SVN bildet Entwicklungspfade im Dateisystem ab, Git virtualisiert diese (kein gleichzeitiger Zugriff)
- SVN erlaubt partielle Arbeitskopien, Git erfordert komplettes Spiegeln des Repositories
- SVN ermöglicht Zugriffskontrolle, Git ermöglicht lokales/offline Arbeiten

(Optional) Git Hands-on

Quellen

- Icons übernommen/abgeleitet von den Glyphicons (glyphicons.com) des Bootstrap-Projektes (getbootstrap.com).
- Darstellung der Datenstruktur basiert auf
  - ASCII-Art zur "Bubble-Up Method" aus den initialen SVN-Design-Dokumenten

    https://svn.apache.org/repos/asf/subversion/trunk/notes/subversion-design.html#server.fs.struct.bubble-up

    (aufgerufen am 2016-11-02)
  - "Abbildung 6-3" aus R. Preißel, und B. Stachmann, *Git: dezentrale Versionsverwaltung im Team, Grundlagen und Workflows*, 3. Auflage, 2015, dpunkt.verlag
- Mukherjee2010: P. Mukherjee, *A fully Decentralized, Peer-to-Peer Based Version Control System*, TU Darmstadt, Ph.D. Thesis, 2011, tuprints.ulb.tu-darmstadt.de/2488 (Stand 27.12.2015)