

Upravljanje verzijama softvera

Bojan Tomić
tomicb@fon.rs

Šta je upravljanje verzijama softvera?

- Upravljanje različitim verzijama programa, biblioteka, dokumenata, sajtova itd.
- Deo oblasti upravljanja konfiguracijom softvera (Source Configuration Management - SCM)
- Version control, revision control, source control
- Sistem za upravljanje verzijama softvera - sinonimi
 - Version control system (VCS)
 - Revision control system (RCS)
 - Source code manager (SCM)

Šta je upravljanje verzijama softvera?

- Nije samo backup
- Mogućnost istovremenog rada više programera na jednom projektu tj. da istovremeno menjaju iste fajlove i posle ih sinhronizuju
- Mogućnost da se vrati starija verzija softvera i-ili uporedi sa novijom (npr. zbog nekog bug-a)
- Mogućnost odvajanja grana razvoja i istovremeni razvoj više verzija (topic, bug fix, release support)
- Mogućnost spajanja više grana u jednu i uključivanje u glavni tok razvoja

Neki popularni besplatni VCS

- Centralizovani (klijent-server) VCS
 - CVS
 - Subversion
- Distribuirani VCS (DVCS)
 - Mercurial
 - Git
- Poređenje velikog broja VCS se može naći ovde:
http://en.wikipedia.org/wiki/Comparison_of_revision_control_software

CVS

- CVS (Concurrent Versions System ili Concurrent Versioning System)
- Non-GNU Savannah, The GNU Project
- Prva verzija još 1986. godine
- Besplatan, otvorenog koda, sajt:
<http://savannah.nongnu.org/projects/cvs>
- Literatura: „Version Management with CVS“, Per Cederqvist et al. (besplatno se može preuzeti sa CVS sajta)

Subversion

- Subversion (SVN)
- Apache Software foundation
- Besplatan, otvorenog koda, sajt:
<http://subversion.apache.org/>
- Literatura: „Version Control with Subversion“, Ben Collins-Sussman, Brian W. Fitzpatrick, C. Michael Pilato
(besplatno se može preuzeti sa SVN sajta)
- Hostovanje: Sourceforge

Mercurial

- Napisan u Python jeziku
- Besplatan, otvorenog koda, sajt:
<http://mercurial.selenic.com/>
- Literatura: „Mercurial: The Definitive Guide“ Bryan O'Sullivan (besplatno se može preuzeti sa Mercurial sajta)
- Hostovanje Bitbucket

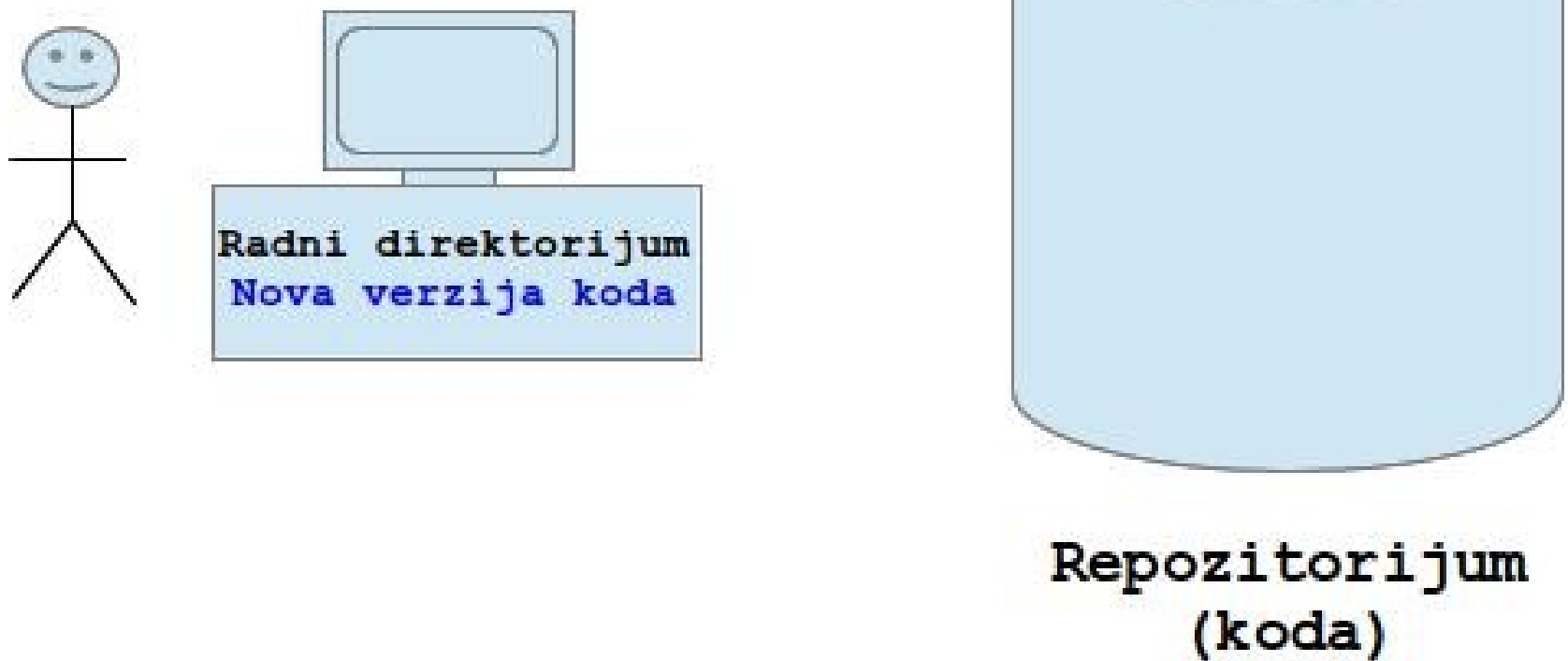
Git

- Autor Linus Torvalds – tvorac Linux operativnog sistema
- Besplatan, otvorenog koda, sajt <http://git-scm.com/>
- Literatura:
 - „Pro Git“ Scott Chacon (besplatno se može preuzeti sa Git sajta)
 - „Version control with Git“ Jon Loeliger, Matthew McCullough
 - Kratak git tutorijel <http://rogerdudler.github.io/git-guide/>
- Hostovanje Github, BitBucket

VCS osnove

- Radni direktorijum, radna kopija (working directory, working copy) - ona verzija koda na kojoj programer trenutno radi i koju menja (na primer sav kod u src folderu u Eclipse projektu).
- Repozitorijum (repository) – deo VCS u kojem se skladište sve verzije koda za dati projekat (sve revizije).
- Revizija (revision, commit) – jedna verzija koda koja je sačuvana u repozitorijumu.

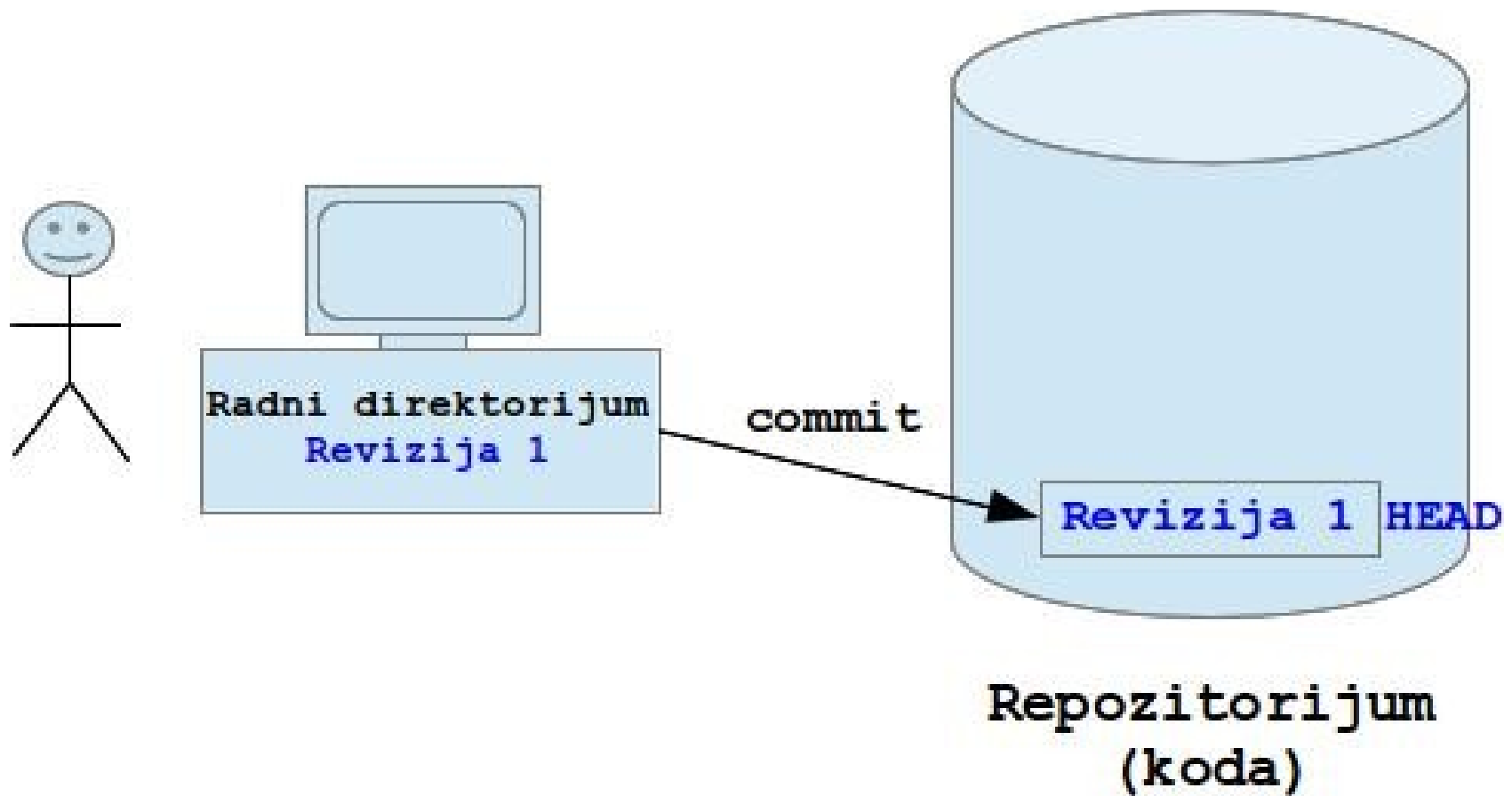
VCS osnove



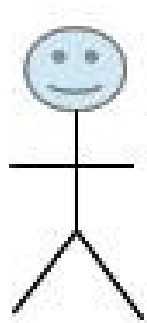
VCS osnove

- Commit operacija – skladištenje verzije koda iz radnog direktorijuma u repozitorijum. Ta verzija koda postaje nova revizija i dobija ID (redni broj ili SHA1 broj, simbolički naziv itd.)
- Commit operacija je jedini način da se sačuva nova revizija u repozitorijumu.
- Dobro vreme za commit: na kraju radnog dana, posle otklanjanja nekog bug-a, itd.
- HEAD revizija – verzija koda koja je poslednja sačuvana u repozitorijumu („najnovija“).

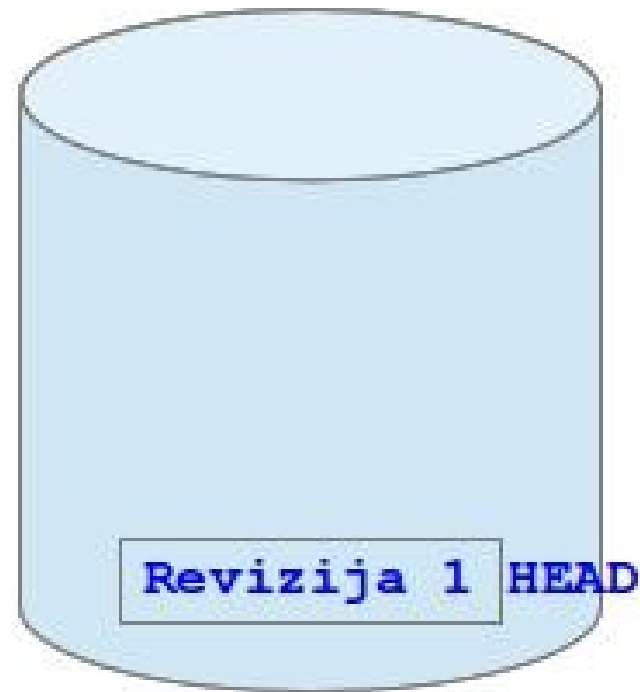
VCS osnove



VCS osnove

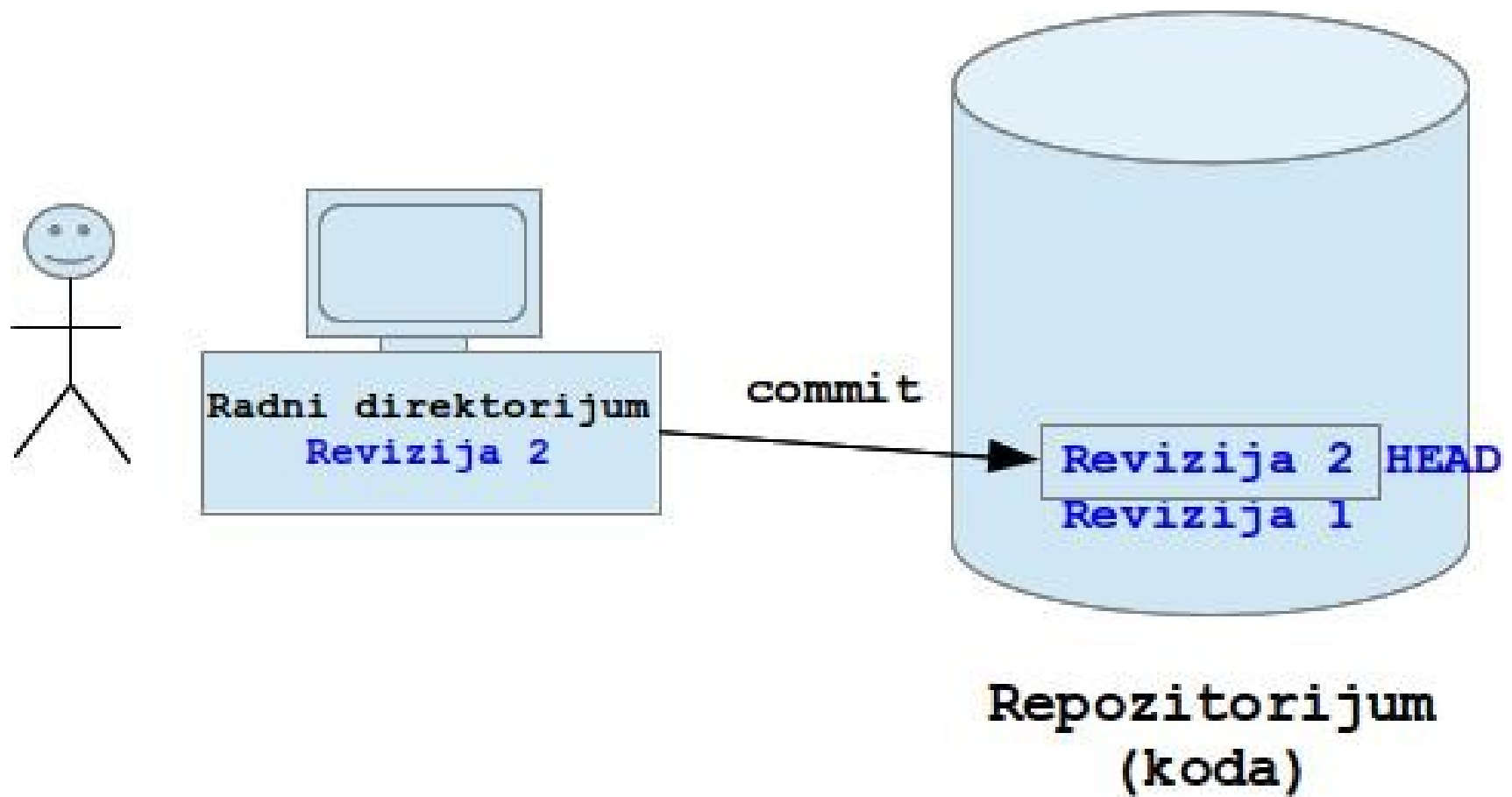


Radni direktorijum
Revizija 1*
(izmenjena)



Repozitorijum
(koda)

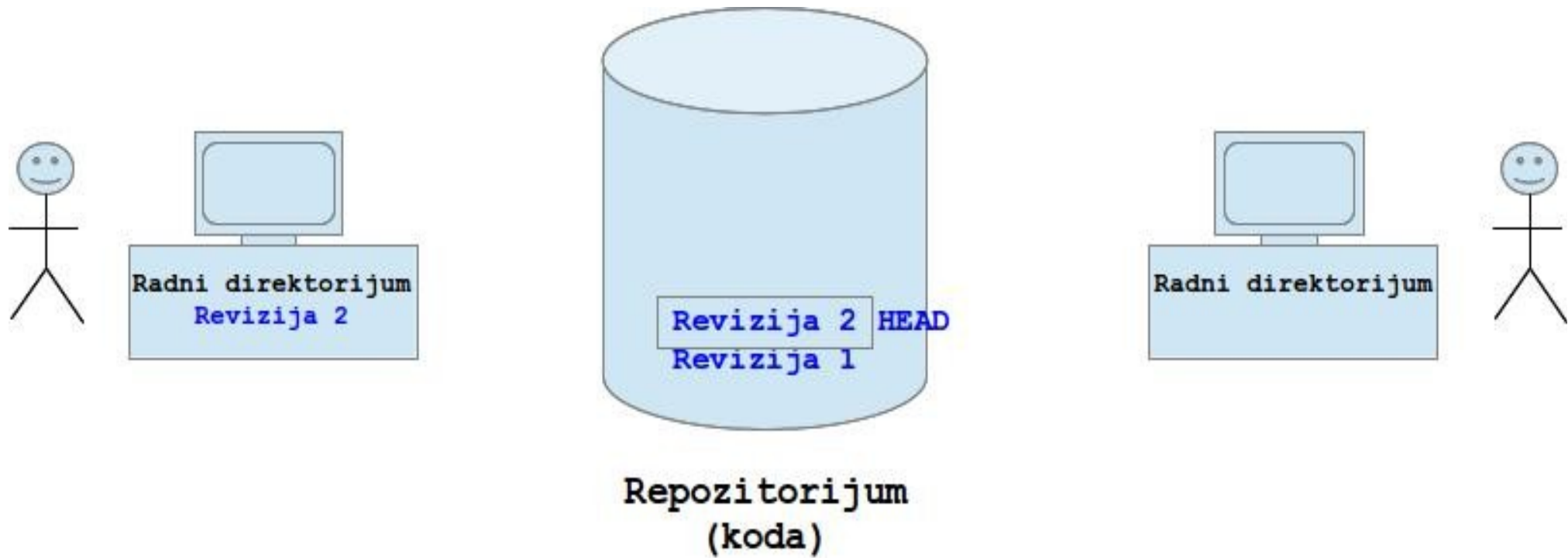
VCS osnove



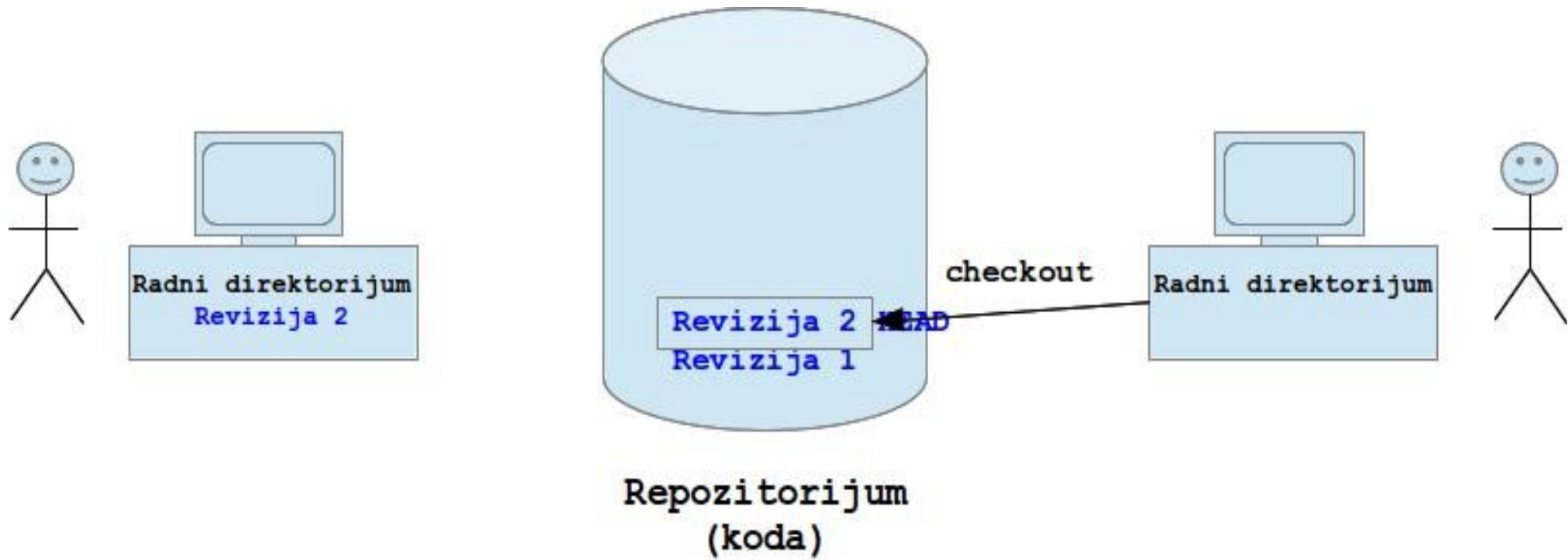
VCS osnove

- Checkout operacija – kad se neko nov pridruži projektu (timu) i želi da počne sa radom, preuzima trenutnu (ili željenu) reviziju pozivanjem ove komande.
- Kopira se celokupan sadržaj revizije iz repozitorijuma u radni direktorijum i programer je spreman za rad.
- Opciono, može se preuzeti i neka starija revizija ili grana – ako je potrebno rešiti neki bug, nastaviti razvoj neke starije verzije programa itd.

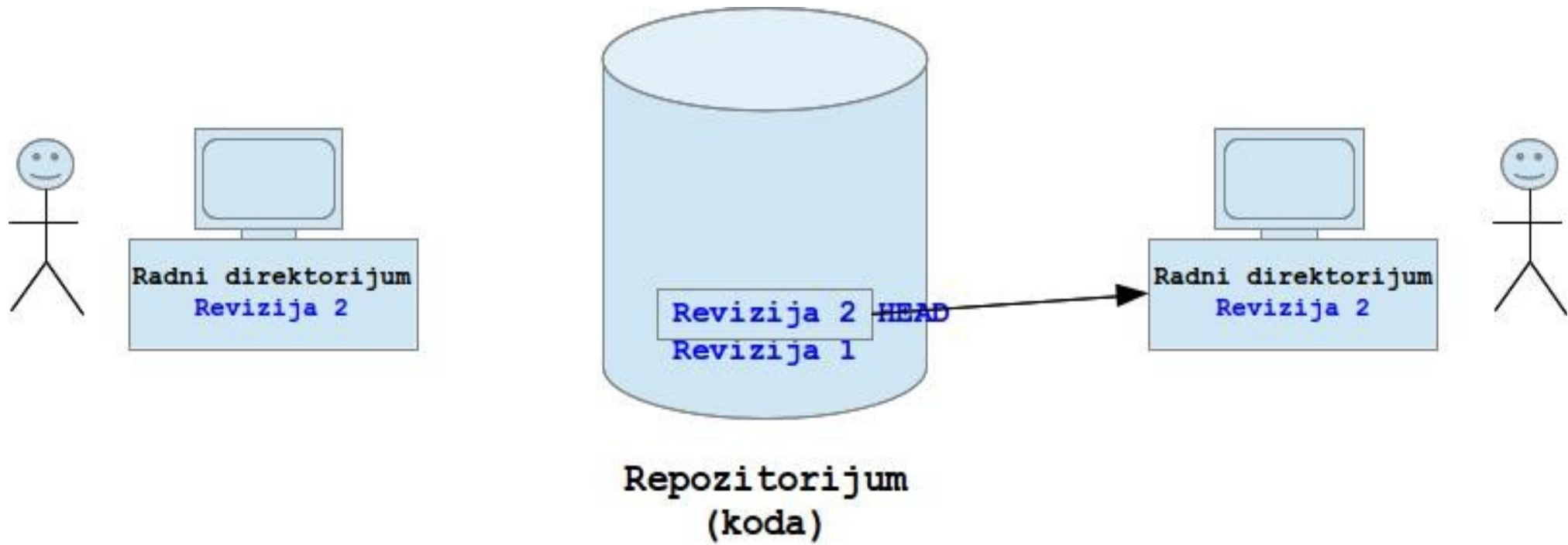
VCS osnove



VCS osnove



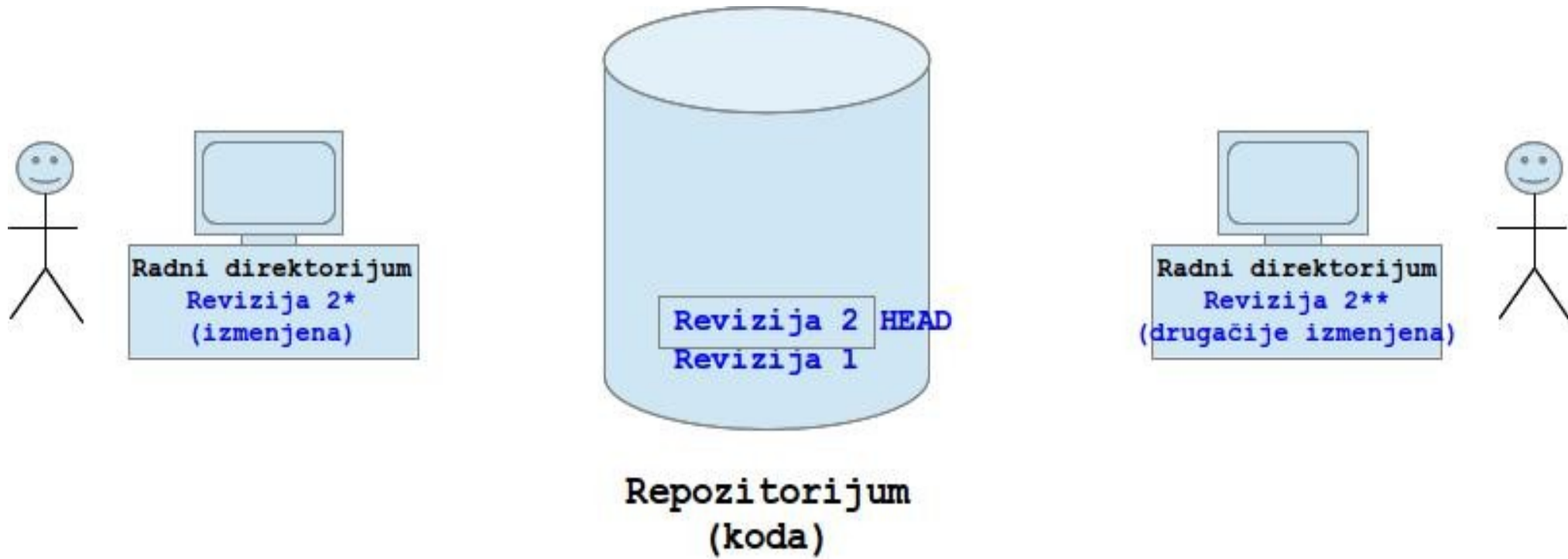
VCS osnove



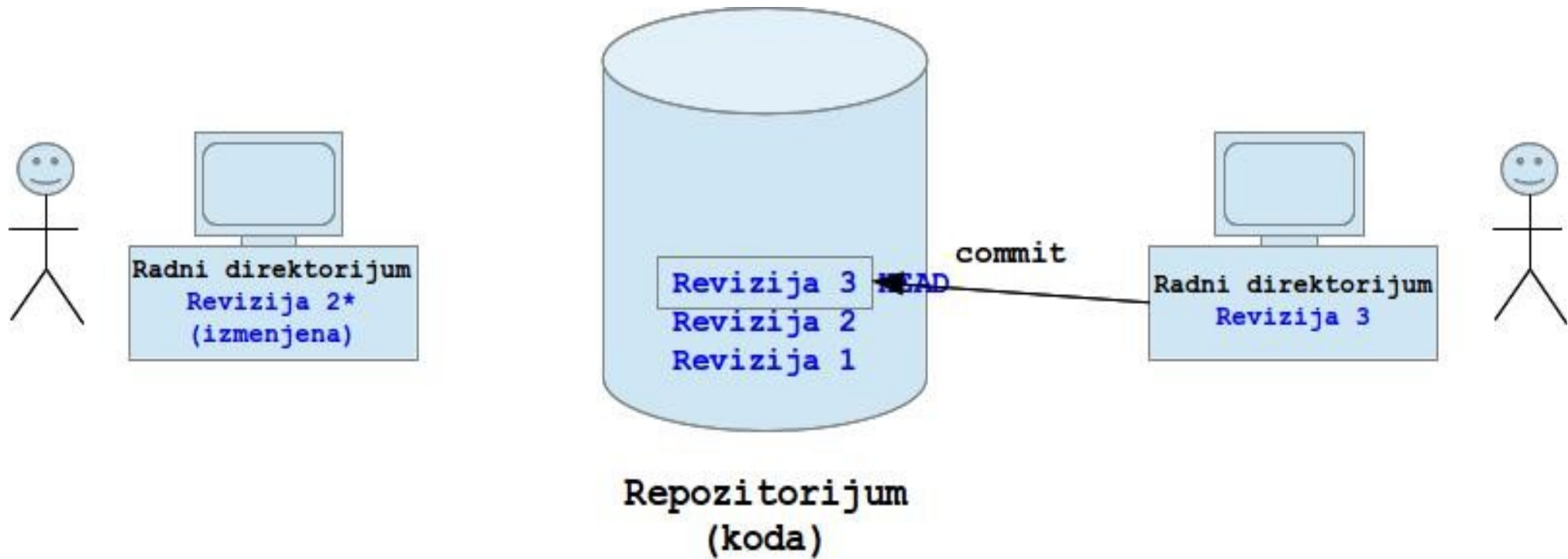
VCS osnove

- Update operacija – preuzimanje promena tj. nove revizije koju je neko drugi poslao u repozitorijum i spajanje sa sadržajem radnog direktorijuma.
- Obavezna pre izvršenja svake commit operacije ako je u međuvremenu došlo do promena u repozitorijumu – ovako nema gubitka podataka.
- Pri spajanju, mogući su konflikti – ako dva programera istovremeno promene isti fajl.
- Rešavanje konflikata preko modela konkurentnosti: file lock ili merge.

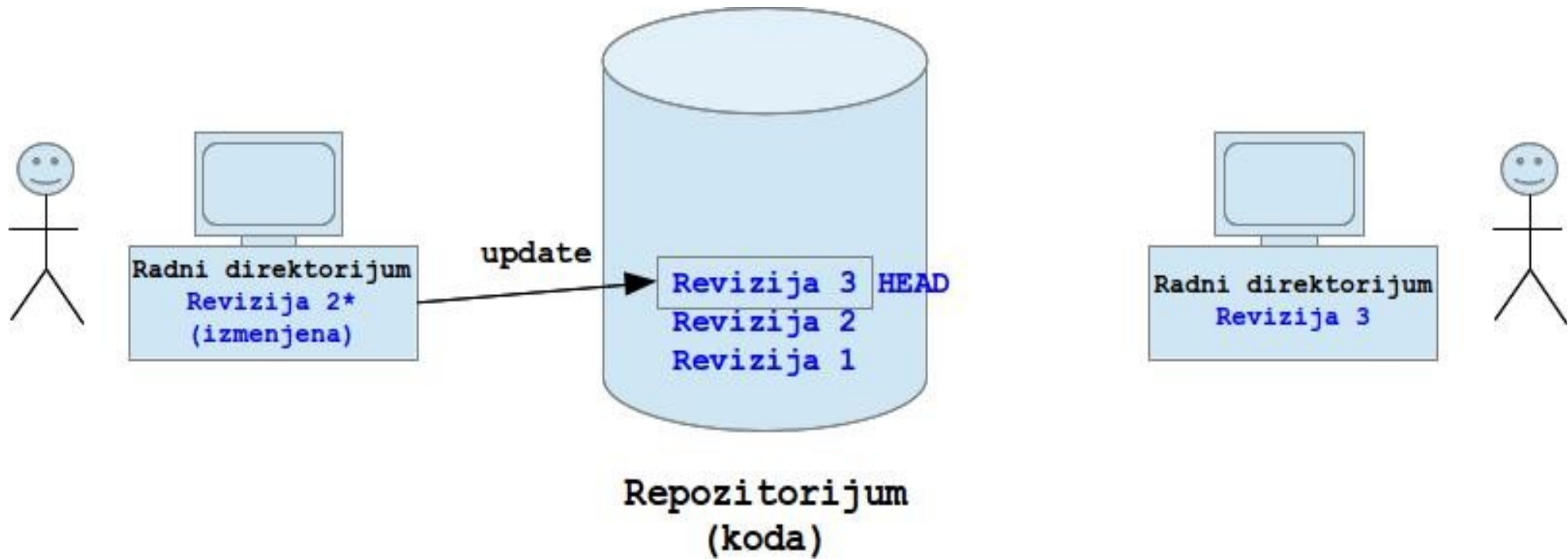
VCS osnove



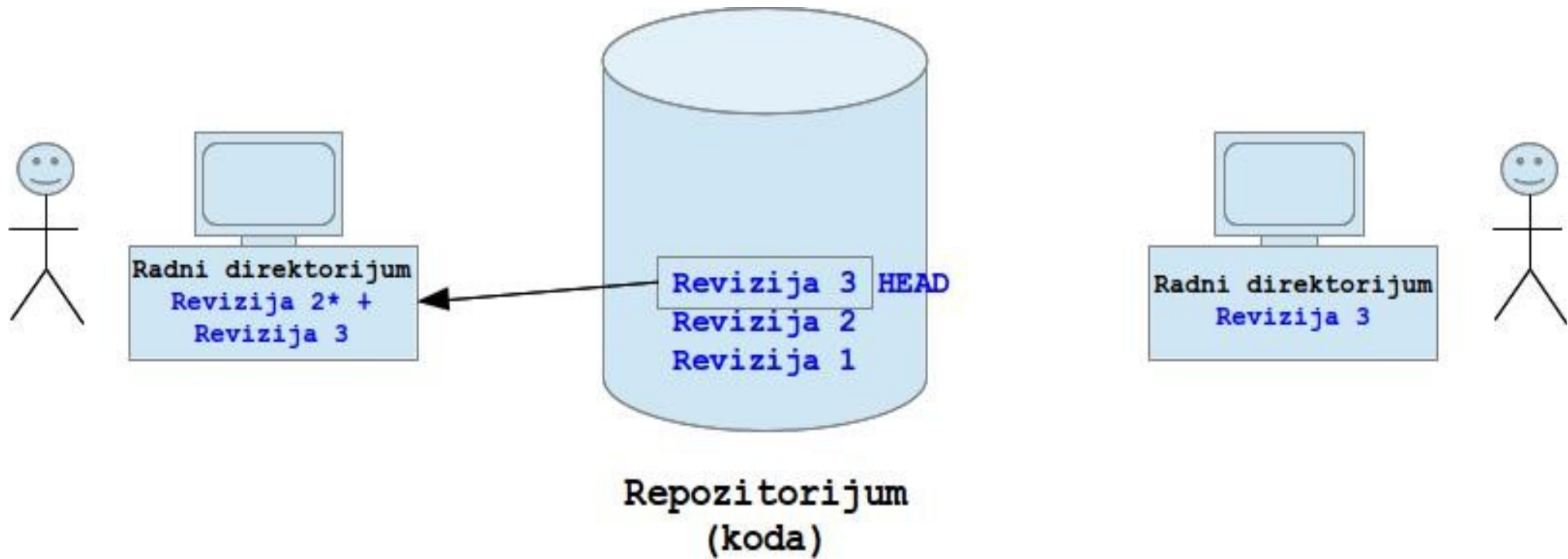
VCS osnove



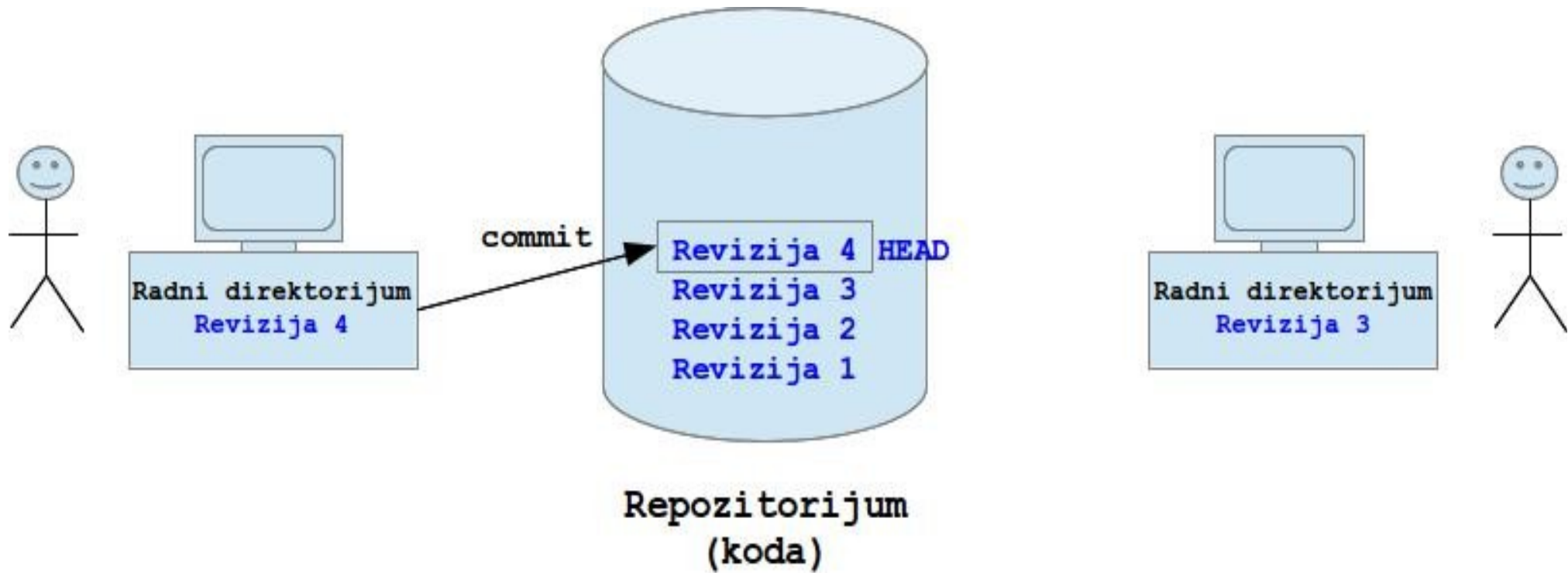
VCS osnove



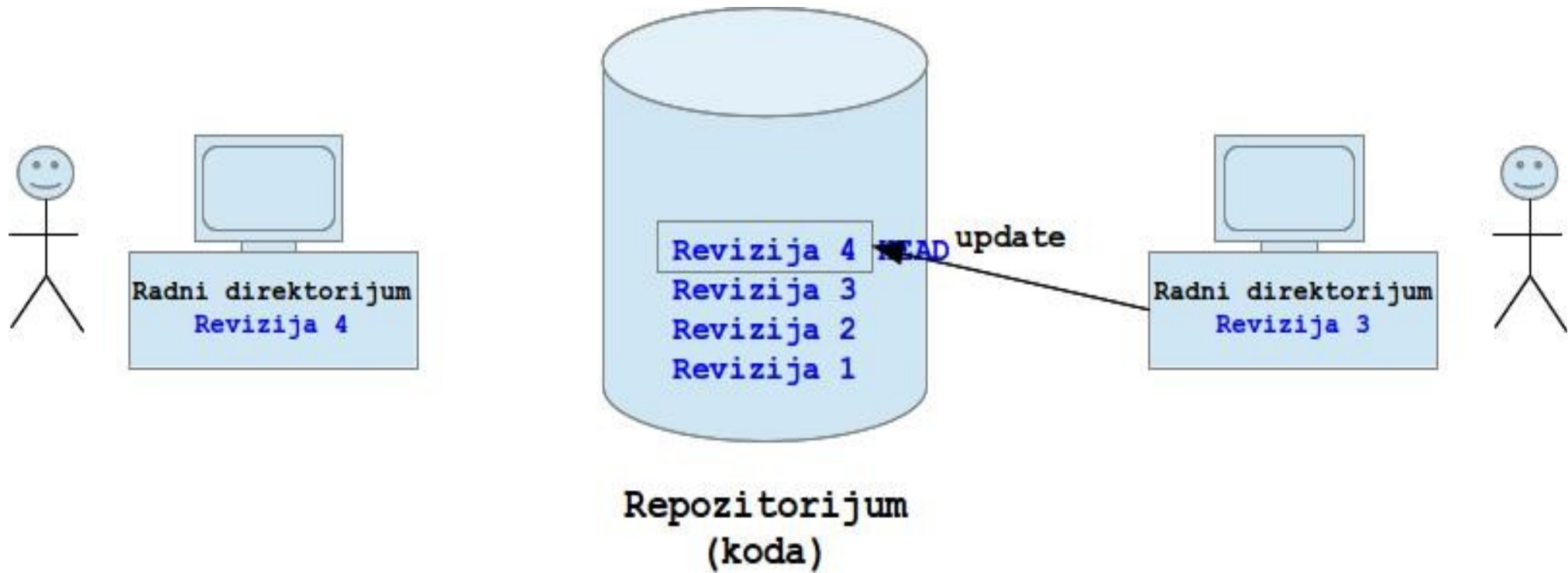
VCS osnove



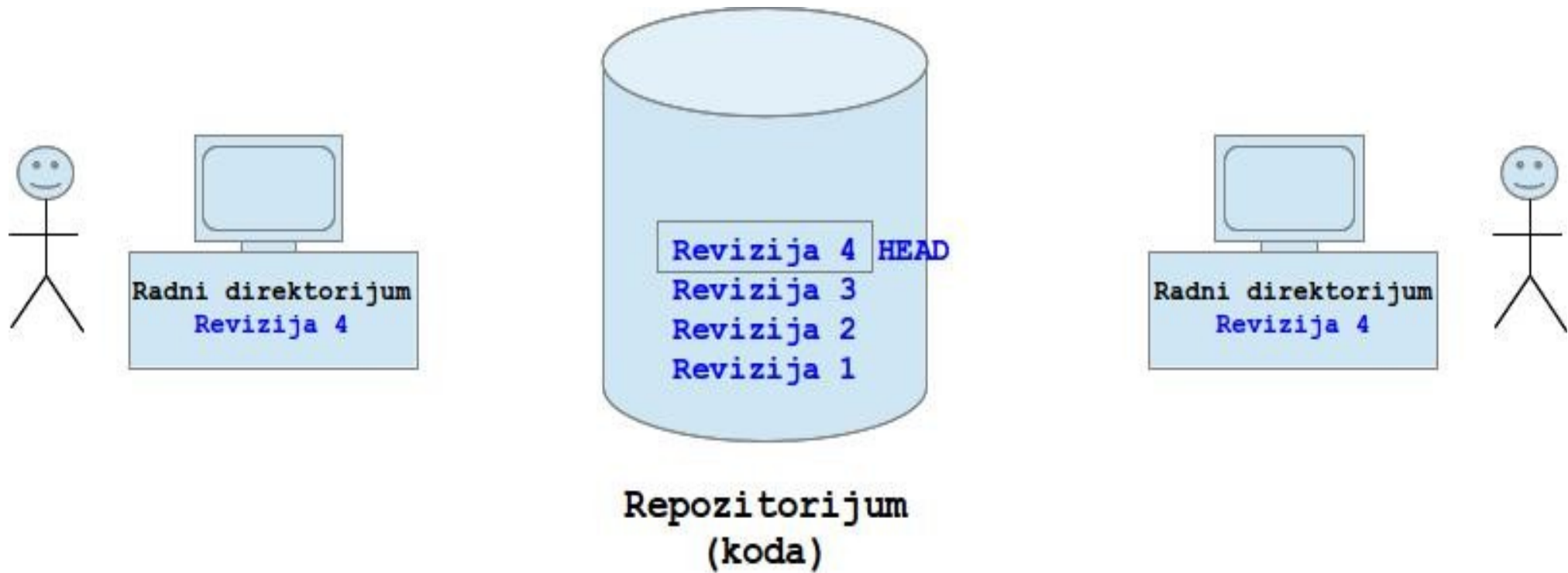
VCS osnove



VCS osnove



VCS osnove



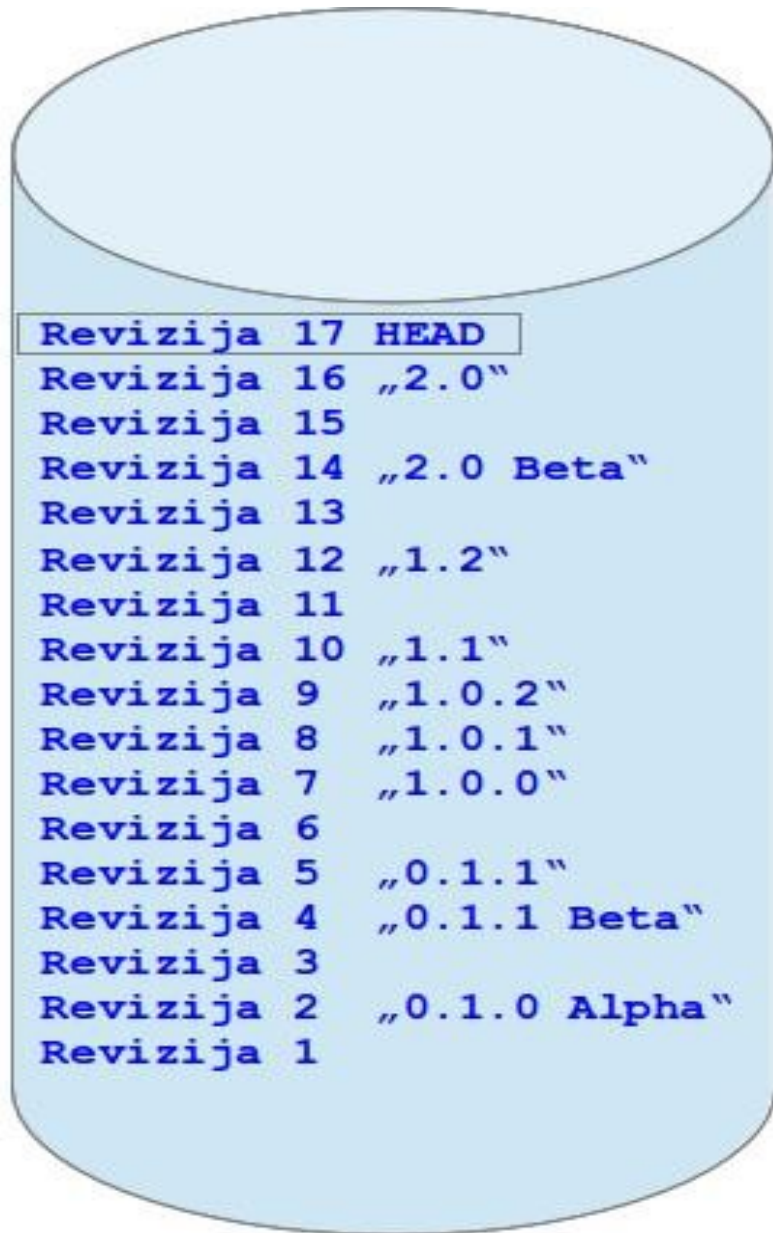
VCS osnove

- Diff operacija – automatsko označavanje razlika između dve revizije: novi fajlovi-folderi, obrisani fajlovi-folderi, preimenovani fajlovi-folderi, tačno koje su izmene u okviru fajlova red po red itd.
- Dve revizije koje se žele uporediti se unose kao parametri u diff operaciju.

VCS osnove

- Tag – određena revizija kojoj se daje simboličko ime jer predstavlja npr. objavljenu verziju programa (release). Na primer tag „version 2.3.0“
- Objavljena verzija programa (release) se, po konvenciji „taguje“ sa dva, tri ili četiri broja (major, minor, micro) i, eventualno rečima „alpha“, „beta“ itd. Na primer:
 - 0.1.0 Alpha
 - 1.0.0 Beta
 - 2.3.6

VCS osnove



Repozitorijum

- Major number 2.3.6
- Minor number 2.3.6
- Micro number 2.3.6
- Pre Alpha, Alpha, Beta, Beta 2, Final, RC (Release Candidate)

VCS osnove

- Grana (branch) – odvojena linija razvoja u okviru projekta koja ima svoje revizije.
- Zašto? Da se ne bi remetio glavni tok razvoja.
 - Odvajanje razvoja određene karakteristike programa.
 - Isprobavanje i eksperimentisanje sa programom.
 - Odvajanje linije za rešavanje nekog bug-a.
- Grane su obično privremenog karaktera.
- Posle nekog vremena, može se spojiti sa glavnom granom razvoja, nekom drugom granom ili obrisati.

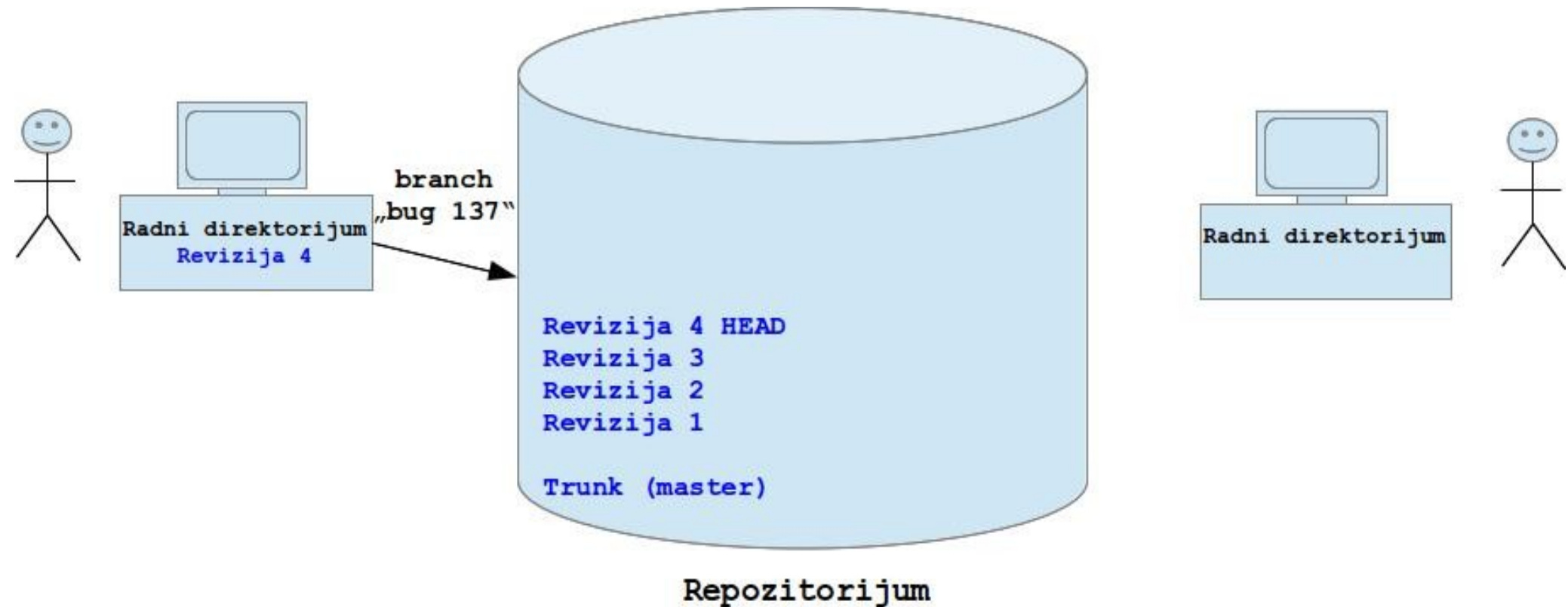
VCS osnove

- Svaki repozitorijum uglavnom ima:
 - Trunk (ili master) grana – glavna grana tj. glavna linija razvoja u koju se integrišu svi rezultati
 - Topic grane – grane na kojima se vrši odvojen razvoj određenih novih funkcionalnosti programa.
 - Bug grane – grane na kojima se rešav neki bug.
 - Tags – grane ili simboličke reference koje predstavljaju neku reviziju koja je objavljena.

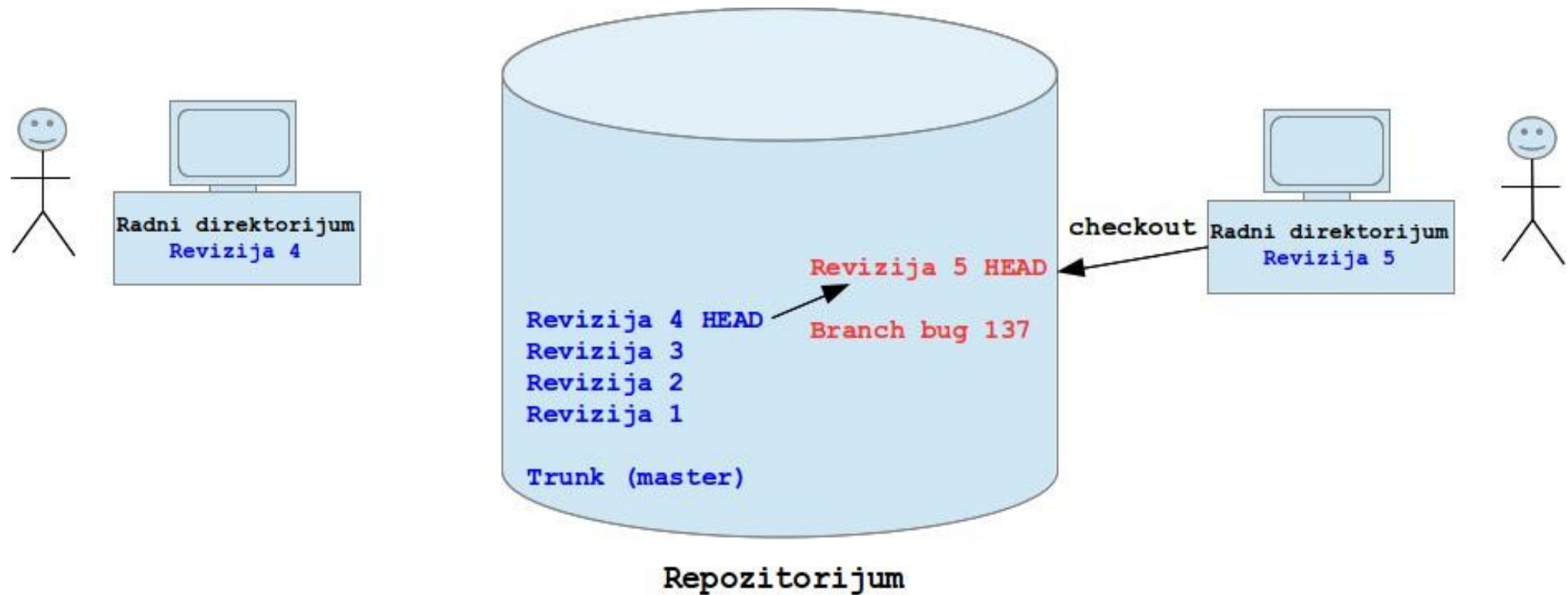
VCS osnove

- Merge operacija – spajanje (najčešće) poslednje revizije neke grane sa (najčešće) poslednjom revizijom glavne grane.
- Ovako se rezultat rada na grani integriše u glavni tok projekta – nova funkcionalnost, rešenje bug-a itd.
- Pri spajanju se mogu javiti konflikti koji se automatski detektuju, ali ručno razrešavaju.
- Posle ove operacije, grana se više ne razvija i ne dobija nove revizije.

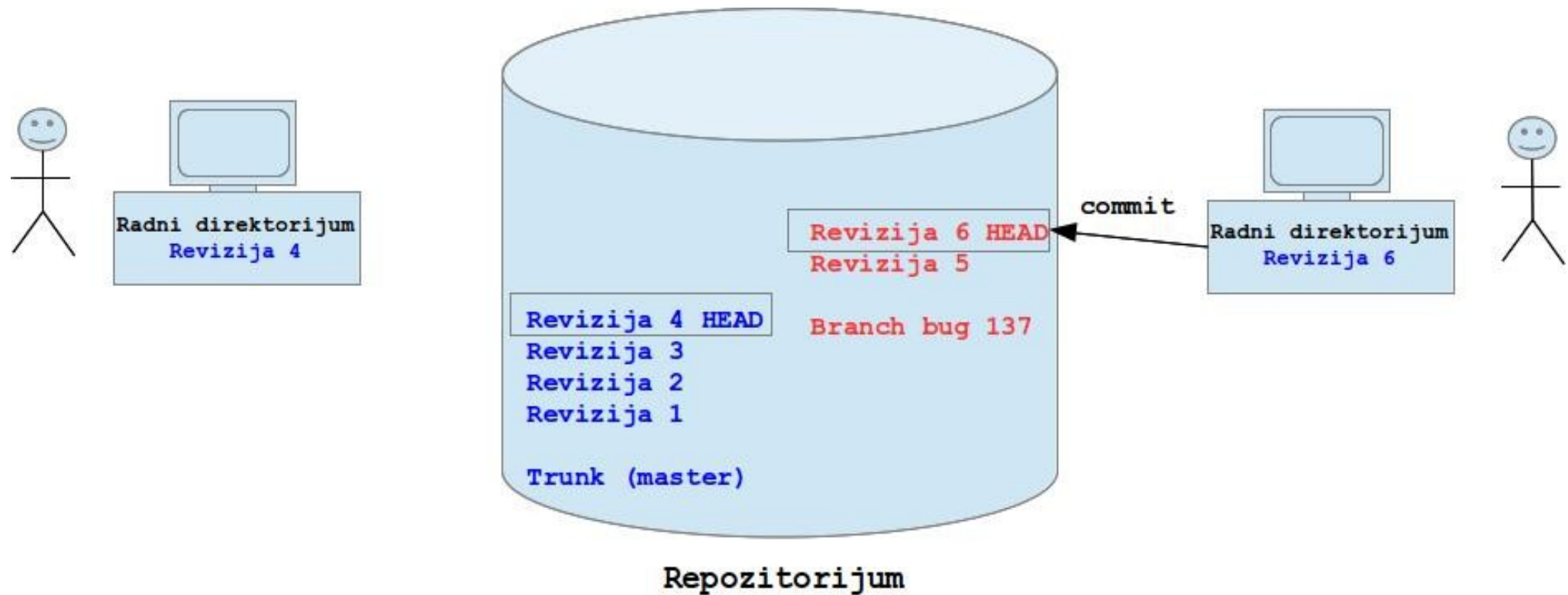
VCS osnove



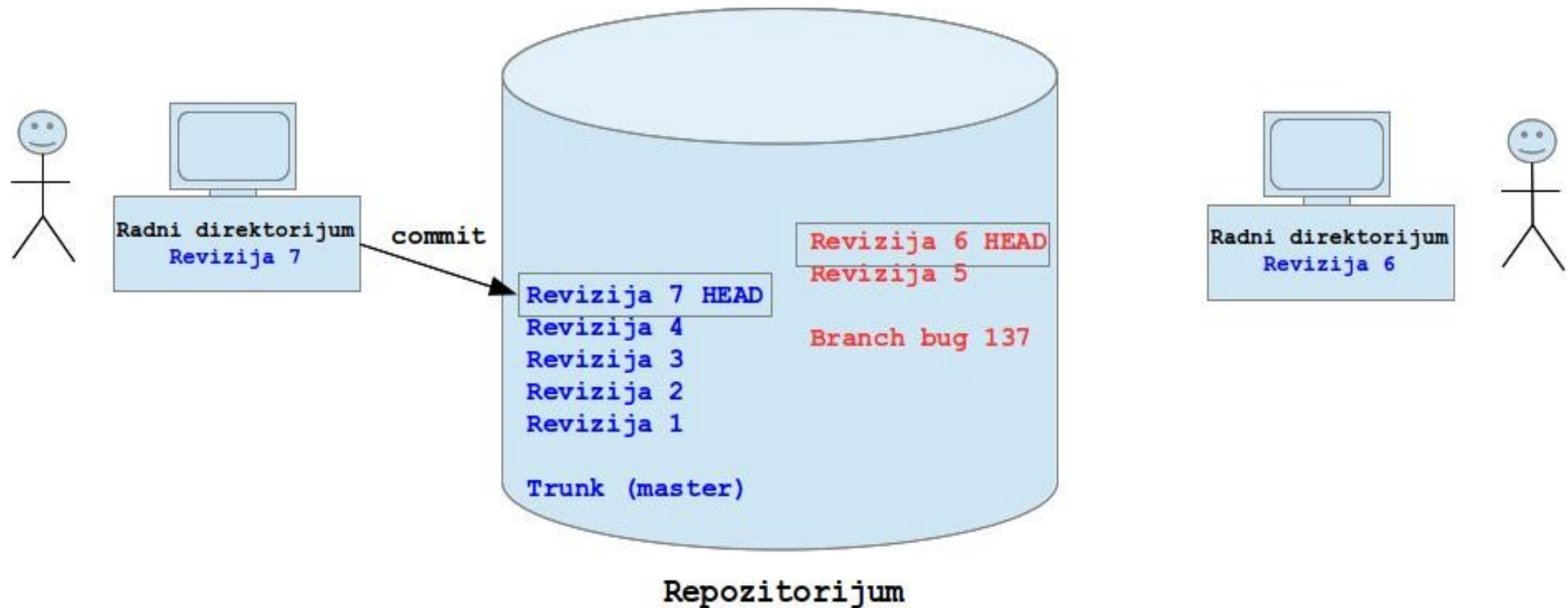
VCS osnove



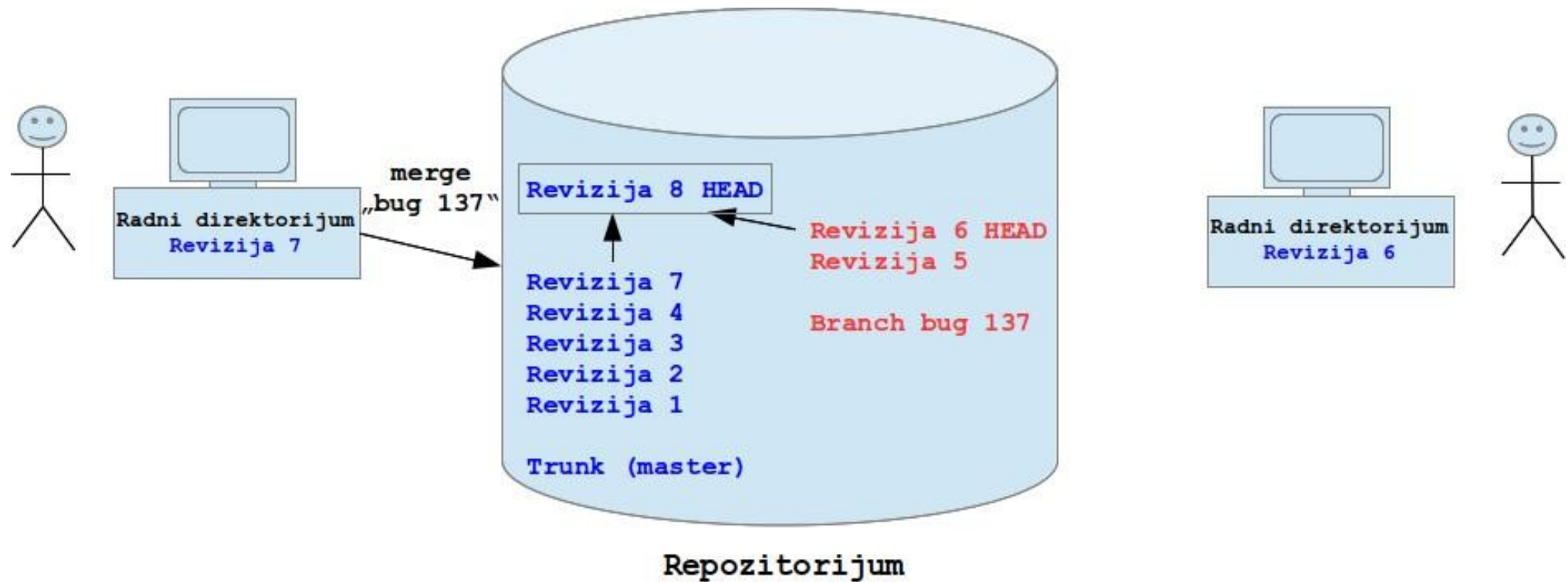
VCS osnove



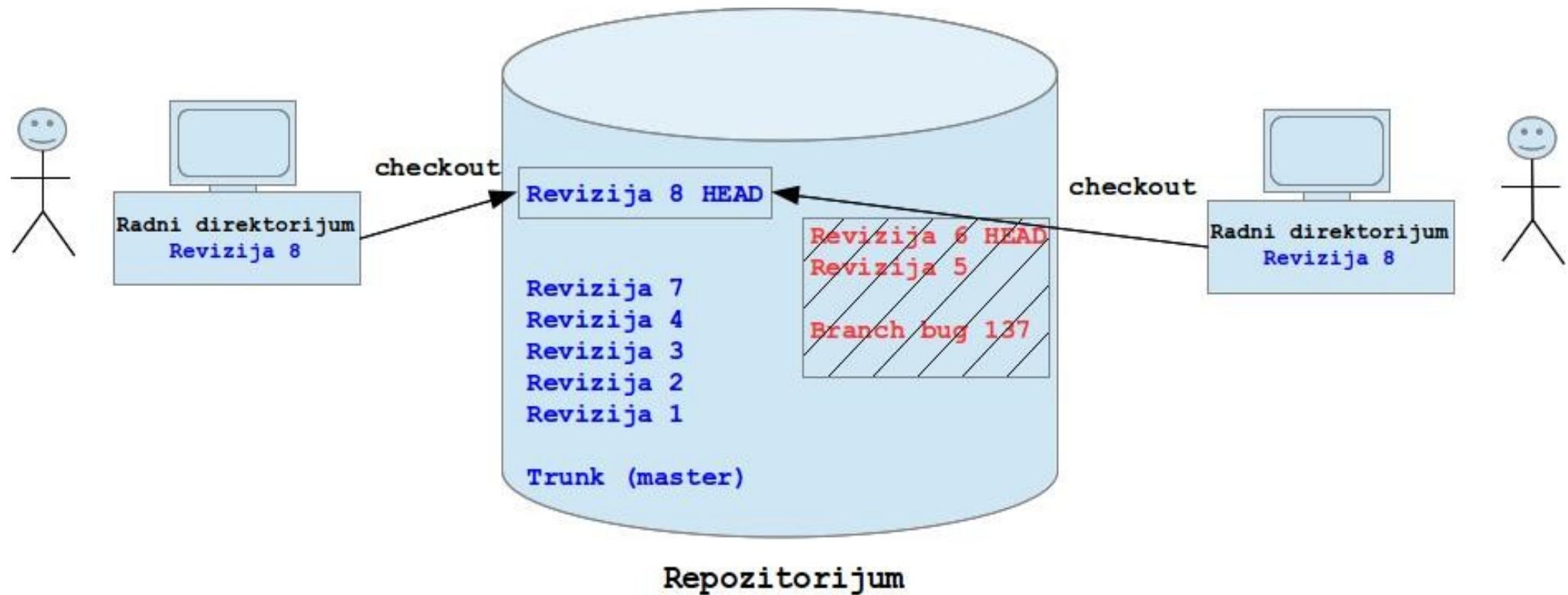
VCS osnove



VCS osnove

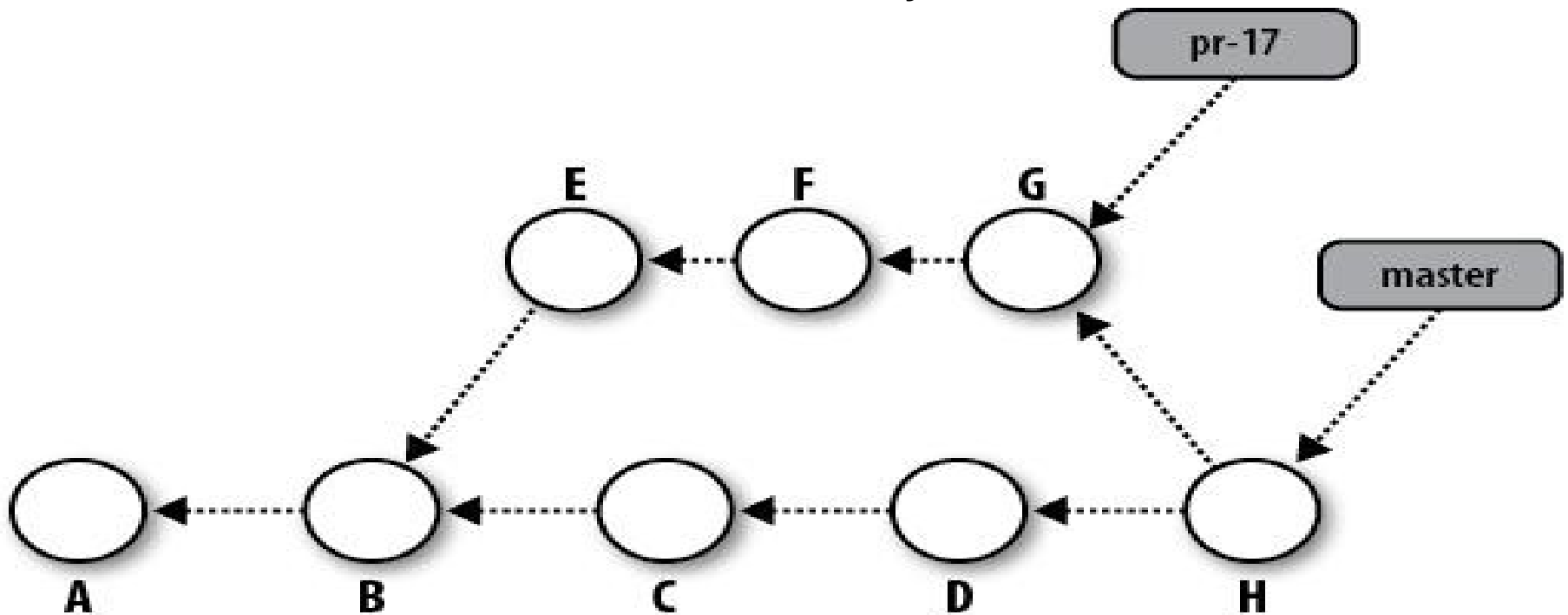


VCS osnove



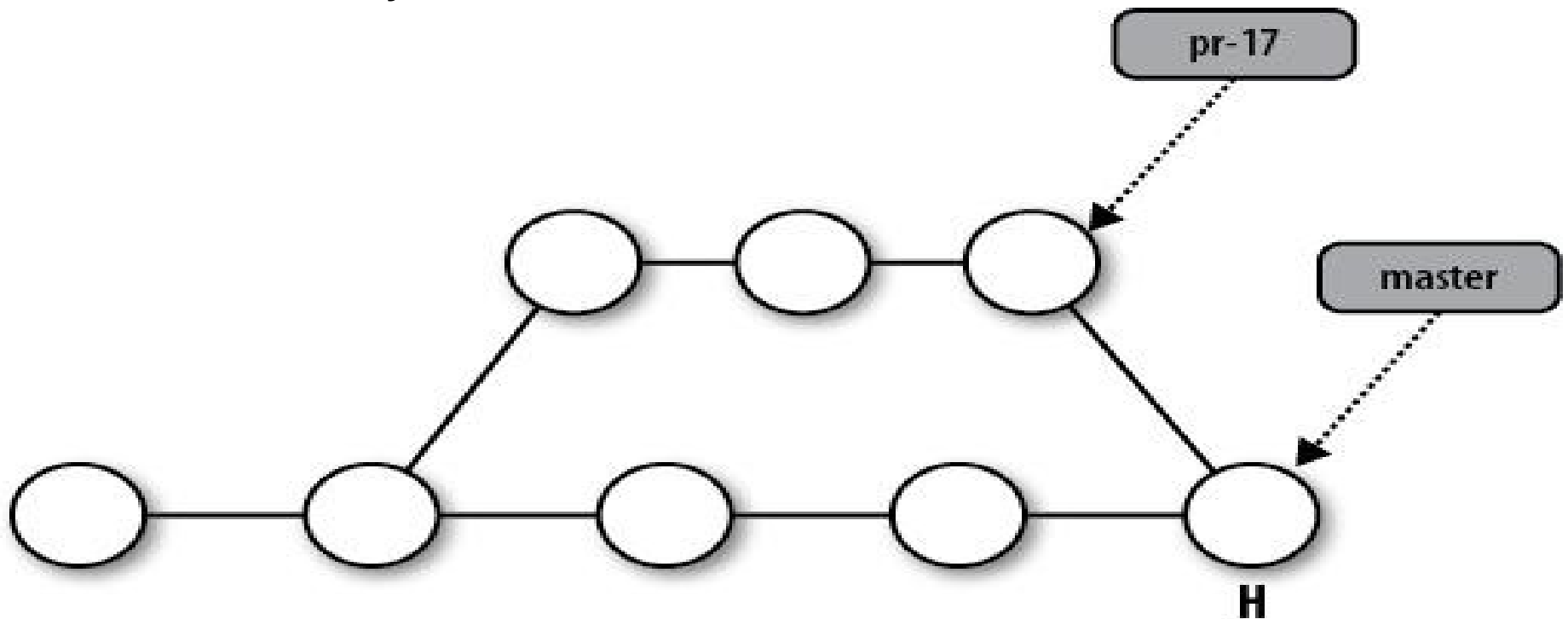
VCS osnove

- Commit graf (vide se grane „pr-17“ i „master“)
- Neciklični usmereni graf (directed acyclic graph – DAG)
- Često se i izostave strelice, revizije idu sa leva na desno

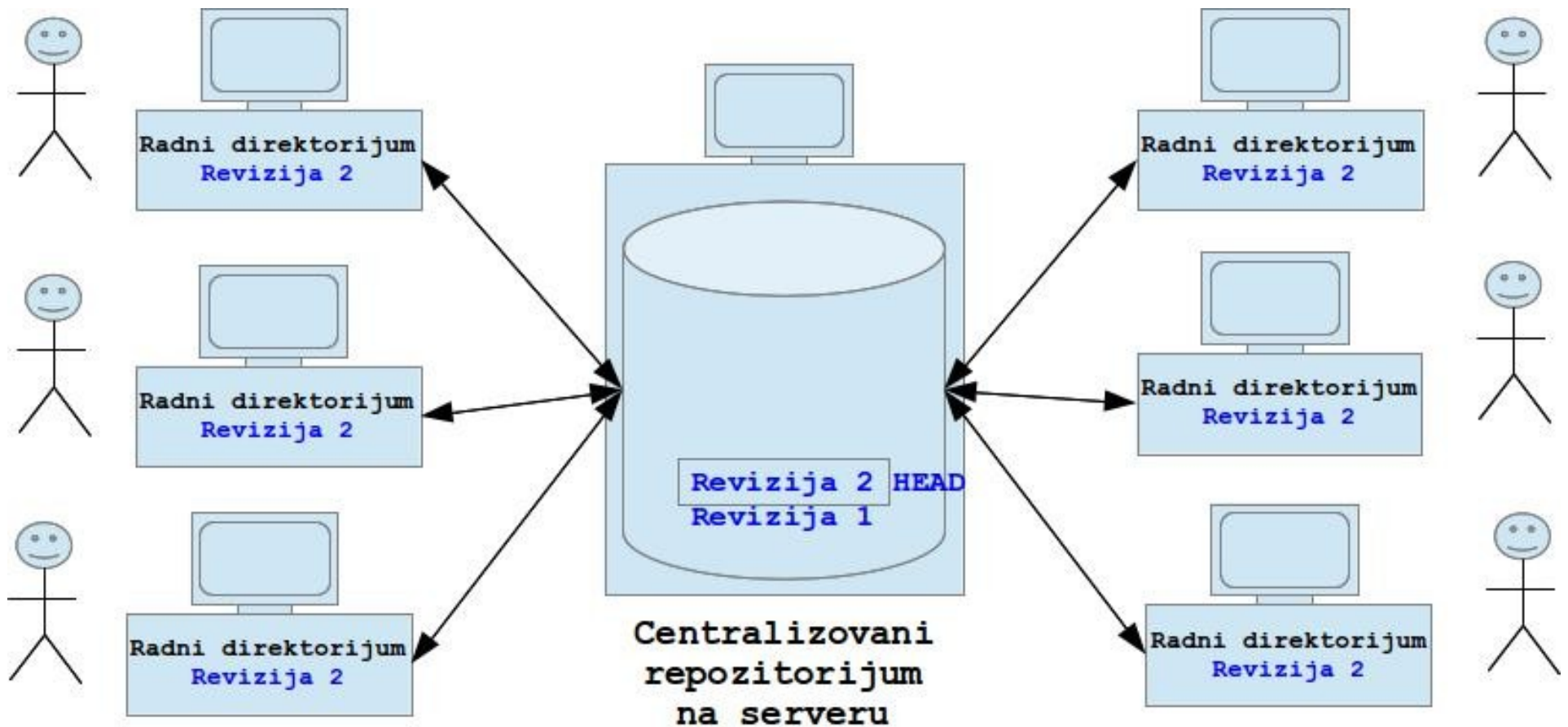


VCS osnove

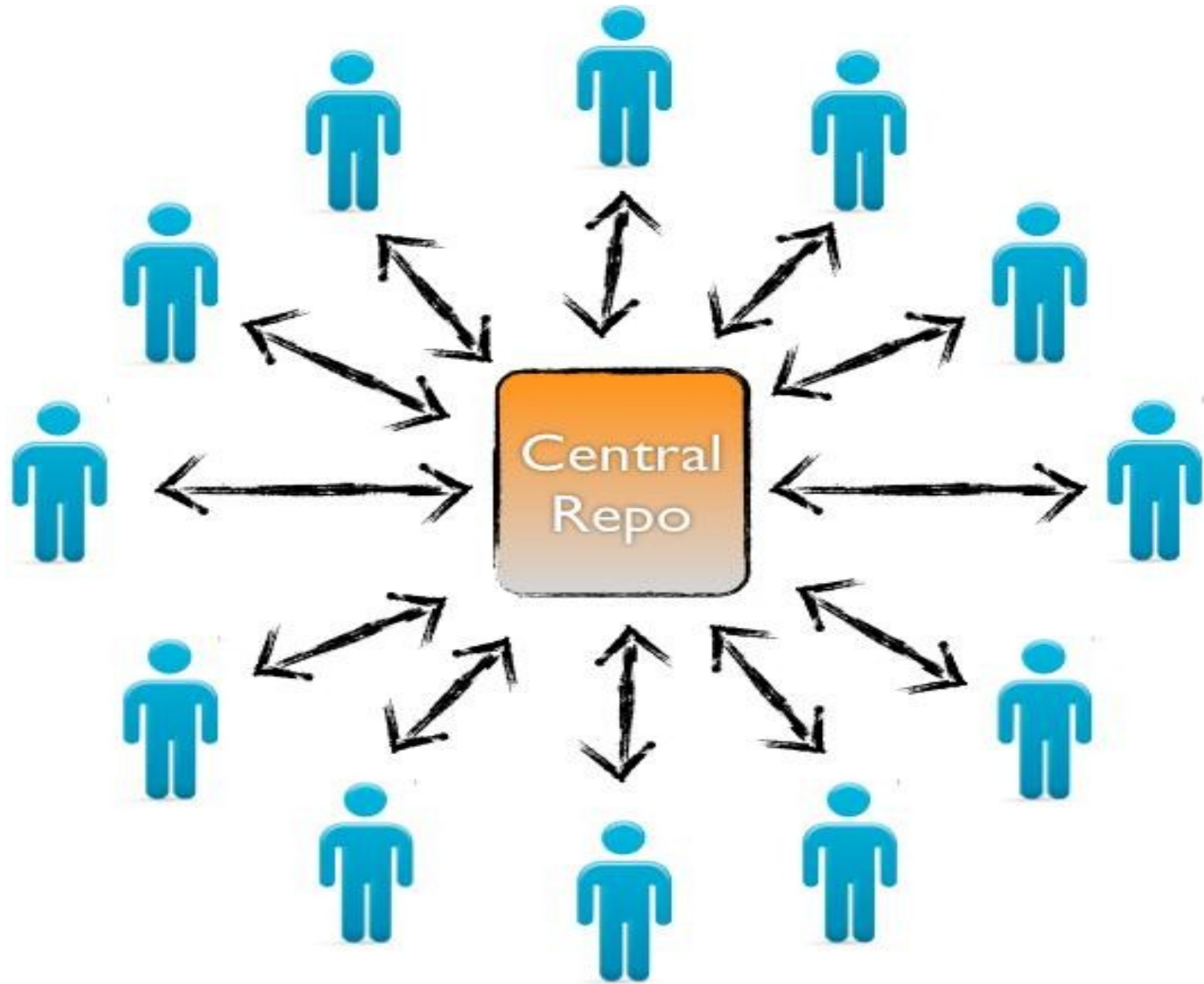
- Pojednostavljen commit graf
- Nema strelica jer se podrazumeva da su usmerene na levo
- ID samo za ključne commit-e



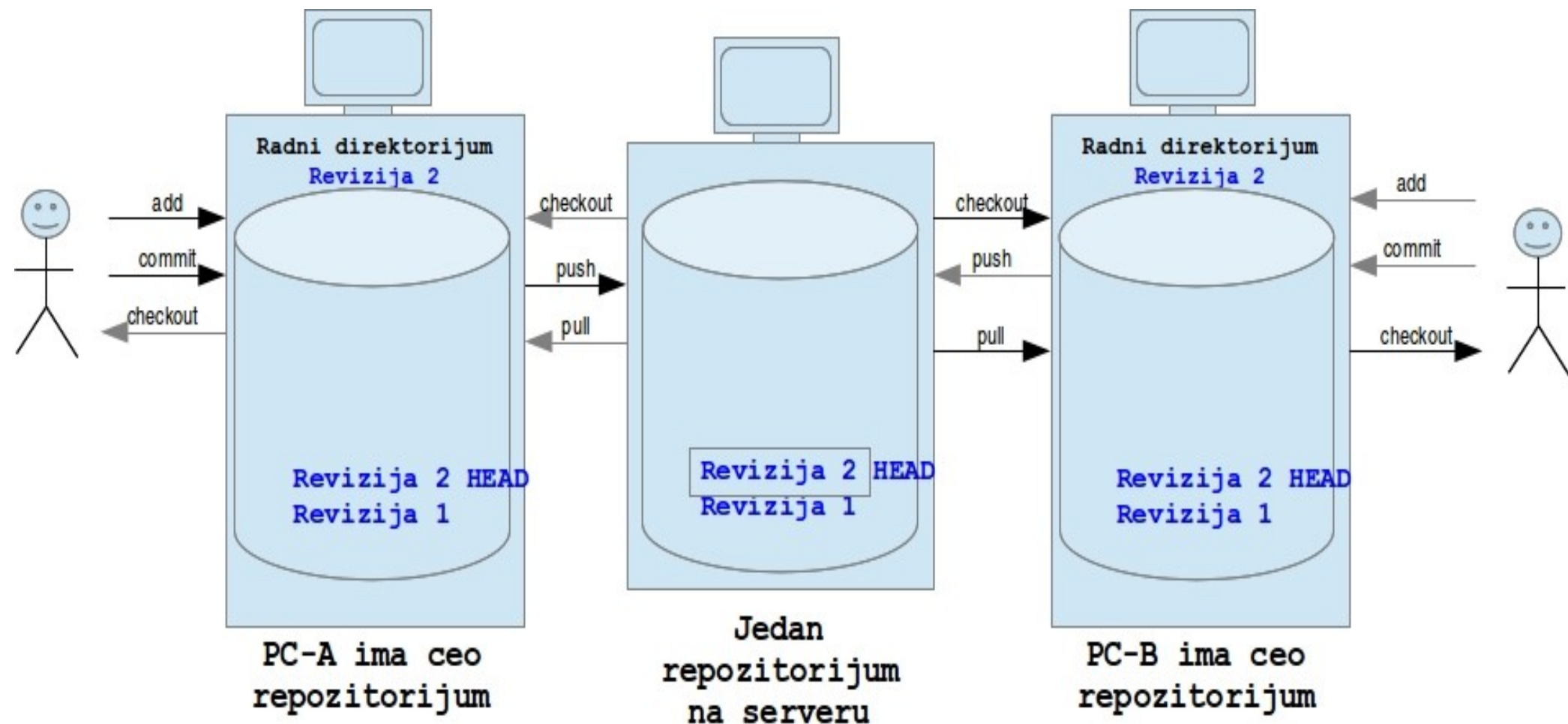
Centralizovani VCS



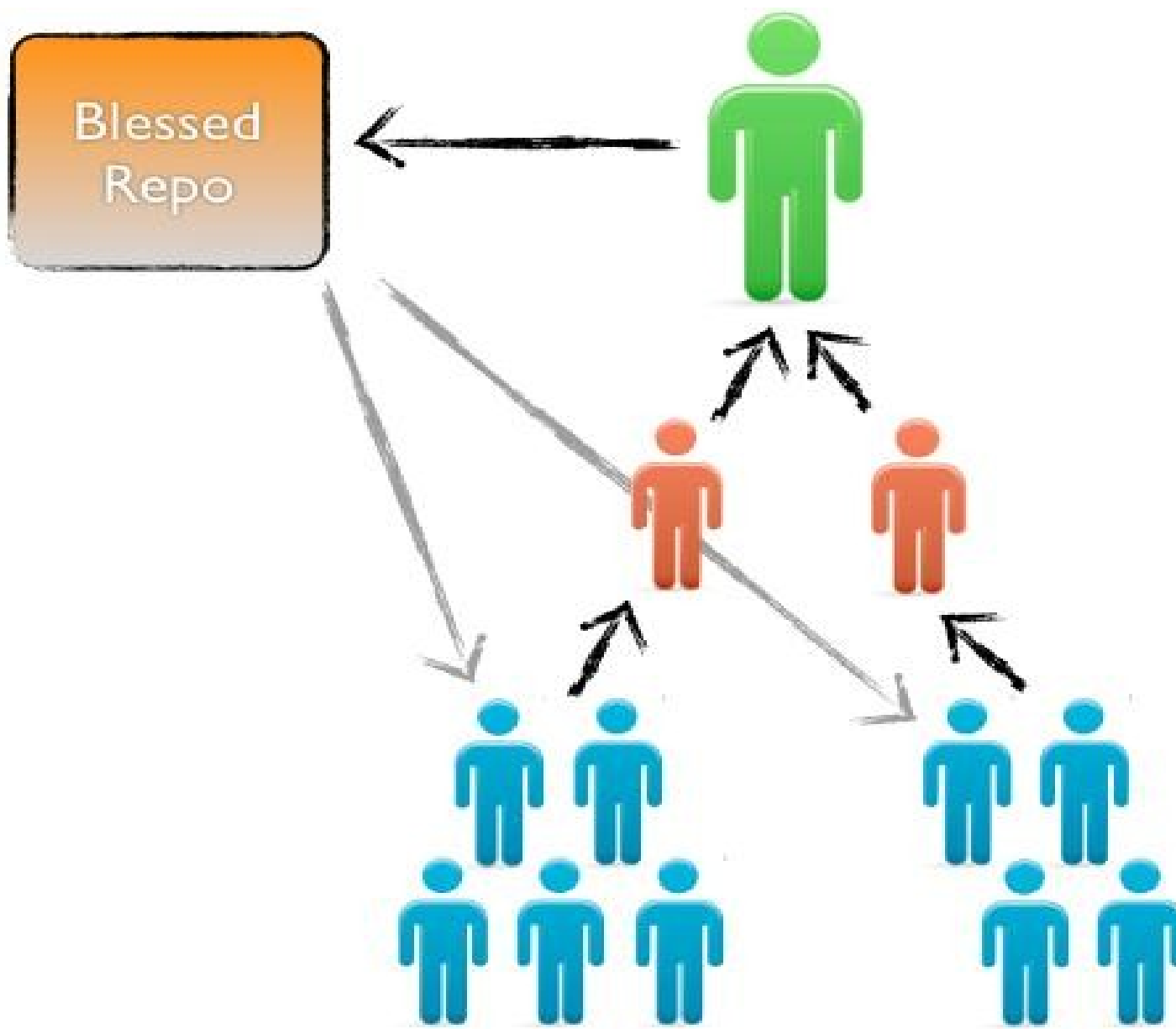
Centralizovani VCS – model timskog rada



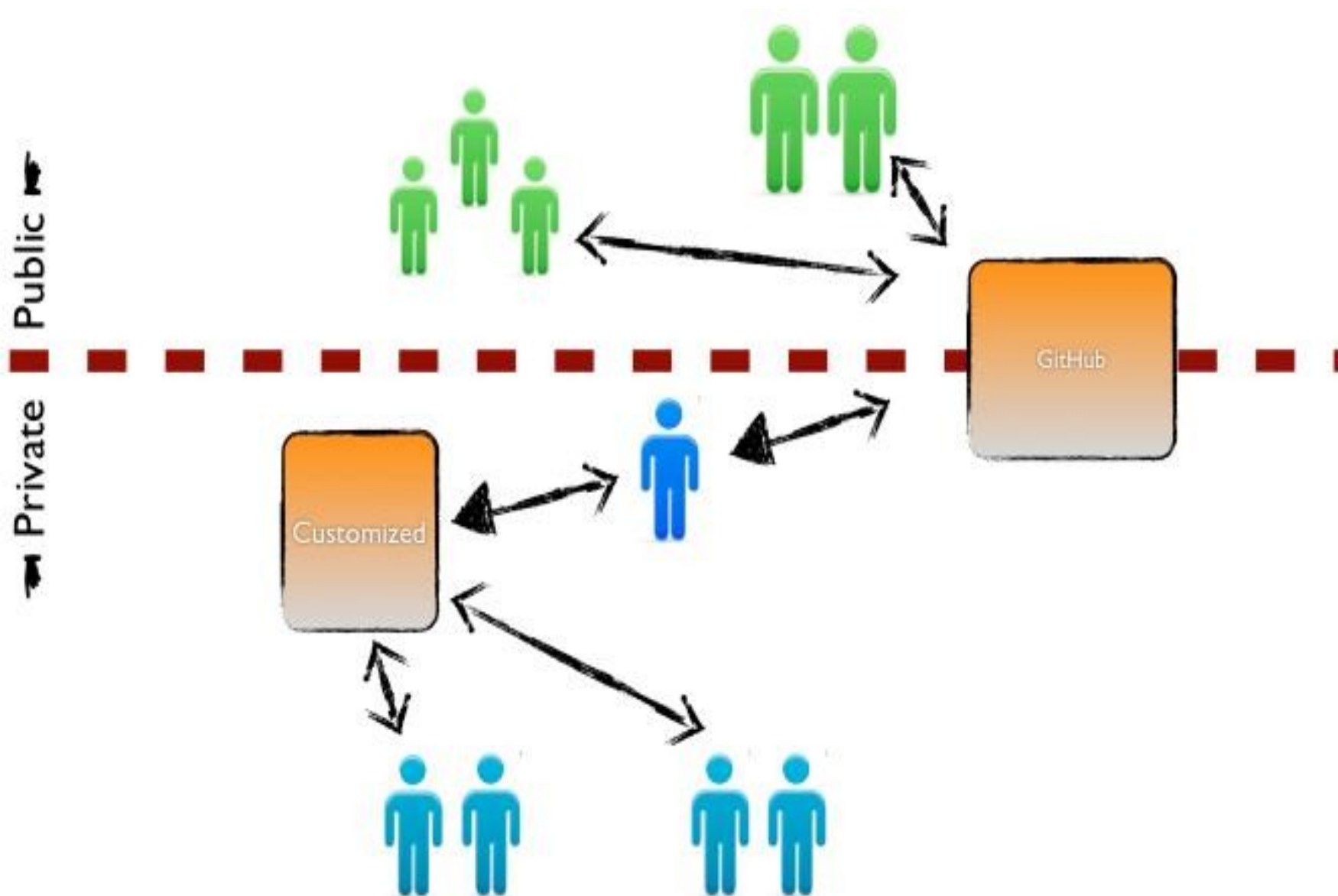
Decentralizovani VCS



Decentralizovani VCS – modeli timskog rada (poručnik i komandanti)



Decentralizovani VCS – modeli timskog rada (arbitar i delimično otvoren kod)



Git instalacija

- Windows
 - Git for Windows – besplatna aplikacija (bivši msysgit)
<https://git-for-windows.github.io/>
 - Cygwin + git paket – Cygwin je besplatna aplikacija koja omogućava korišćenje nekih GNU i open-source (Linux) alata na Windows platformi
- Linux
 - Git paket – besplatan, otvorenog koda.
- OS X (Mac)
 - Git OS X installer – besplatna aplikacija

Git instalacija

- Više načina za rad sa Git-om.
- Preko komandne linije (Windows command prompt, Git bash)
- Preko nekog okruženja za razvoj, npr.:
 - Netbeans (7.4 i noviji) Git plugin
 - Eclipse Git plugin

Osnovne Windows Command Prompt komande

- Listanje sadržaja trenutnog foldera

`dir`

`dir *.exe`

- Ulazak u folder

`cd [naziv]`

`cd My Documents`

`cd My Documents/Pictures`

- Izlazak iz foldera

`cd..`

- Promena drajva

`c:`

`d:`

- Pravljenje foldera

`mkdir [naziv]`

Osnovne Windows Command Prompt komande

- Brisanje foldera ili fajla

del [naziv]

del Documents

del index.html

del *.html

Git sintaksa i osnovne komande

- Opšti oblik git naredbe

git [opcije] komanda
[argumenti]

- Verzija

git --version

- Stara sintaksa

git-commit

- Nova sintaksa

git commit

- Pomoć

git --help

git --help [komanda]

git --help commit

git help

git help [komanda]

git help commit

git [komanda] help

git commit help

Git sintaksa i osnovne komande

- Duge opcije (počinju sa --)

```
git commit --message="Ovo je prvi commit"
```

- Kratke opcije (počinju sa -)

```
git commit -m „Ovo je prvi commit“
```

- Neke opcije se mogu naći u oba oblika (kao „message“ tj. „m“ za commit komandu), a neke u samo jednom.
- Za svaku komandu se iz help-a vidi koje su opcije i u kojem obliku

Git sintaksa i osnovne komande

- Inicijalizacija git repozitorijuma (ili prazan ili kloniranje postojećeg)

git init (prazan, nov repozitorijum)

git clone (kloniranje postojećeg repozitorijuma)

- Ako se inicijalizuje kao nov, bitno je da se trenutno nalazite u njemu pre nego što se inicijalizuje
- Git tad u njemu napravi JEDAN sakriveni folder **.git** u kojem se pamte sve revizije i koji se automatski ažurira

Git sintaksa i osnovne komande

- Dobijanje trenutnog statusa repozitorijuma
 - `git status`
- Povratna informacija o tome šta se prati, šta je stage-ovano, šta se novo promenilo....

Git sintaksa i osnovne komande

- Svaki commit mora da ima navedeno ime autora, njegov mail i commit poruku. Može da se sve navodi pri pozivu svake commit operacije, ali to je redundantno.
- Podešavanje autora i mail-a (za taj repozitorijum ili za taj korisnički nalog na računaru)

```
git config user.name „Pera Peric“
```

```
git config user.email „pera@peric.net“
```

```
git config --global user.name „Pera Peric“
```

```
git config --global user.email „pera@peric.net“
```

Git sintaksa i osnovne komande

- Svaki commit mora da ima navedeno ime autora, njegov mail i commit poruku. Može da se sve navodi pri pozivu svake commit operacije, ali to je redundantno.
- Podešavanje autora i mail-a (za taj repozitorijum ili za taj korisnički nalog na računaru)

```
git config user.name „Pera Peric“
```

```
git config user.email „pera@peric.net“
```

```
git config --global user.name „Pera Peric“
```

```
git config --global user.email „pera@peric.net“
```

Git sintaksa i osnovne komande

- Prema prioritetima podešavanja se prvo gledaju:
 - Za taj repozitorijum (fajl `.git/config`)
 - Za taj korisnički nalog na računaru (fajl `.gitconfig` u okviru „home“ foldera naloga)
 - Globalne promenljive za taj računar (`GIT_AUTHOR_NAME` i `GIT_AUTHOR_EMAIL`)
- Poruka u commit operaciji može da se navede u pozivu operacije, ali može i da se definiše startovanje editora

```
export GIT_EDITOR emacs
```


Git sintaksa i osnovne komande

- Pre commit operacije postoji i međukorak, tzv. stage tj. proba.
- Na stage (u tzv. indeks) se postavljaju sve promene koje se žele commitovati.

```
git add [naziv fajla ili foldera]
```

```
git add index.html
```

```
git add source/
```

- Ako se stavi naziv foldera, biće obuhvaćeni i svi njegovi fajlovi ili podfolderi.

Git sintaksa i osnovne komande

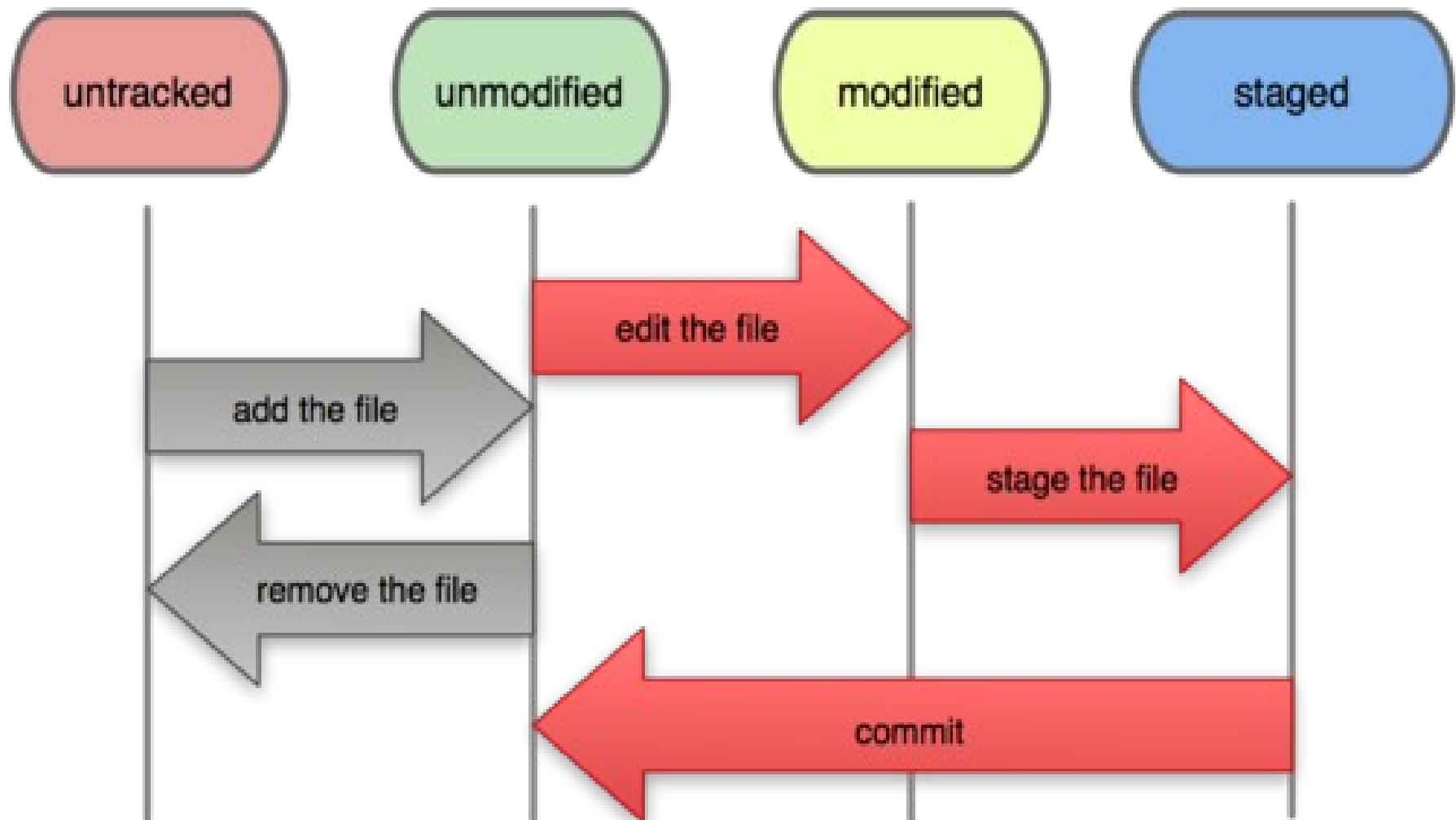
- Bitno je da se na stage stavi poslednja verzija fajla ili foldera, inače neće ažurna verzija biti commitovana.
- Za brisanje sa stage-a se koristi sledeća komanda

```
git rm --cached [naziv fajla ili foldera]
```

```
git rm --cached index.html
```

Git sintaksa i osnovne komande

File Status Lifecycle



Git sintaksa i osnovne komande

- Commit operacija izgleda ovako (kompletna):

```
git commit -m "Poruka" --author="Pera Peric<pera@peric.net"
```

- Ali, ako su autor i editor podešeni onda ovako, pri čemu se odmah po pozivu otvar editor u koji se ukucava poruka:

```
git commit
```

```
git commit -all (automatski proverava sve, stage-uje sve osim neprijavljenih foldera i commituje)
```

- Po izvršenju, ceo sadržaj indeksa tj. stage-a se memoriše u repozitorijum kao nova revizija.

Git sintaksa i osnovne komande

- Dodavanje tagova u commit (može i uz druge objekte)
git tag -m „opis“ [tag ID] [commit ID]
git tag -m „Tag version 1.0“ v1.0 3ede462
- Mora da bude već gotov commit

Git sintaksa i osnovne komande

- Pregledanje commit-a i commit istorije na više načina

- Log

`git log`

`git log --pretty=short`

`git log --pretty=oneline`

`git log --pretty=full`

`git log --stat`

`git log --abbrev-commit`

`git log --pretty=full --stat`

`git log [revizija]`

`git log [start..end]`

`git log [start..]`

`git log [..end]`

- Grafički prikaz (DAG)

`gitk`

`git log --graph`

Git sintaksa i osnovne komande

- Diff – automatizovano poređenje dve revizije

```
git diff [ID revizije 1] [ID revizije 2]
```

- Brisanje fajlova/foldera (automatski stage)

```
git rm [naziv fajla ili foldera]
```

```
git rm index.html
```

- Brisanje se vrši i iz indeksa i iz radnog direktorijuma i to samo ako nije menjan od poslednjeg commita (opcija -f)
- Ne briše se istorijat tog fajla ili foldera
- Sa opcijom --cached se briše samo iz indeksa

Git sintaksa i osnovne komande

- Preimenovanje/pomeranje fajlova/foldera (autom. stage)

```
git mv [stara putanja] [nova putanja]
```

```
git mv index.html stari.html
```

```
git mv source/source.html source/index2.html
```

- Log funkcioniše i za fajlove (pun istorijat jednog fajla)

```
git log [naziv fajla]
```

```
git log --follow [naziv fajla]
```


Git sintaksa i osnovne komande

- Novi repozitorijum može da se napravi i kloniranjem
 - `git clone [repo1] [repo 2]`
 - `git clone html_public html_clone`
- Kloniraju se tj. kopiraju sve revizije
- Ne kopiraju se podešavanja (user, email...)

Git sintaksa i osnovne komande

- Svaki fajl u radnom direktorijumu može biti u jednom od tri statusa:
 - tracked – prati se i njegove verzije se skladište sa svakom novom revizijom (npr. source fajlovi)
 - ignored – nalazi se fizički u radnom direktorijumu, ali se namerno ignoriše, ne skladišti se u okviru repozitorijuma (svi automatski generisani fajlovi razvojnog okruženja)
 - untracked = tracked – ignored (trebalo bi da nema fajlova u ovom statusu, treba ih ili pratiti ili ignorisati)

Git sintaksa i osnovne komande

- Ignorisanje određenog fajla ili grupe fajlova se radi na sledeći način:
- Napravi se u repozitorijumu `.gitignore` fajl i doda se u njega sadržaj tj. naziv fajla koji se ignoriše

```
echo Osoba.class > .gitignore
```

- Ovaj fajl se stavlja u indeks i onda i on ulazi u svaku reviziju

```
git add .gitignore
```

Git sintaksa i osnovne komande

- .gitignore fajl može da se nalazi u korenu repozitorijuma ali i u svakom podfolderu (ovi u podfolderima su prioritet)
- Osim naziva fajlova, mogu da se unose i paterni

echo *.class > .gitignore (ignorišu se svi class fajlovi u svim folderima)

echo source/ > .gitignore (ignoriše se source folder i sve u njemu)

echo !st.class > .gitignore (ne ignoriše se st.class fajl iako je već navedeno da ignoriše class fajlove)

Git sintaksa i osnovne komande

- Pravljenje grane

git branch [naziv grane] [commit ID]

git branch bugs/bug17 (ako se izostavi commit ID, grana se otvara od HEAD revizije)

- Osnovna grana u projektu je master

- Prikaz grane

git branch (sve grane u projektu, aktivna grana označena sa *)

git show-branch [naziv konkretne grane ili patern]

git show-branch bugs/bug17

Git sintaksa i osnovne komande

- Da bi moglo da se radi na grani, mora na nju da se pređe
- To znači da se sadržaj grane prekopira u radni direktorijum

```
git checkout [naziv grane]
```

```
git checkout bugs/bug17
```

- Neće se preći na drugu granu ako ima nesačuvanih tj. necommitovanih promena u radnom direktorijumu da se ne bi izgubile te promene (opcija -f)

Git sintaksa i osnovne komande

- Brisanje grane

```
git branch -d [naziv grane]
```

```
git branch -d bugs/bug17
```

- Može se brisati samo grana koja nije aktivna (prvo preći na drugu granu, pa onda brisati)
- Brisanje se neće raditi ako na grani postoji commit koji ne postoji više nigde u repozitorijumu (uraditi prvo merge ili opcija -D)

Git sintaksa i osnovne komande

- Spajanje grana

`git merge [naziv grane]` (spajanje aktivne gr. sa navedenom)

`git merge bugs/bug17`

- Pri spajanju grana bitan je redosled

- Mogući su i konflikti ako se menjaju isti fajlovi

Git sintaksa i osnovne komande

- Rad sa udaljenim repozitorijumom, preuzimanje tj. kloniranje (prvi korak)

```
git clone [url]
```

```
git clone git://github.com/schacon/grit.git
```

- Bare i development repozitorijum (prvi ima samo revizije koje se samo u njega dodaju, drugi ima i radni direktorijum)

```
git clone --bare [url] [putanja do novog repozitorijuma]
```

- Udaljeni repozitorijum može da bude na istoj mašini

Git sintaksa i osnovne komande

- Da bi se omogućio timski rad, potrebno je da se ovaj klonirani repozitorijum doda kao remote repozitorijum (remote referenca)

```
git remote [url]
```

- Remote tracking branches – grane koje prate stanje udaljenog repozitorijuma

„remotes/origin“ grana

Git sintaksa i osnovne komande

- Preuzimanje fajlova i merge iz udaljenog repozitorijuma
git pull
- Postavljanje novih commitova u udaljeni repozitorijum
git push