Kloniranje repozitorija je postupak kojim kopiramo cijeli repozitorij s neke udaljene lokacije na naše lokalno računalo. S tako kloniranim repozitorijem možemo nastaviti rad kao s repozitorijem kojeg smo inicirali lokalno. Kopirati repozitorij je jednostavno, dovoljno je u neki direktorij kopirati .git direktorij drugog repozitorija. I onda na novoj (kopiranoj) lokaciji izvršiti git checkout HEAD.  
Pravo kloniranje je malo drukčije. Kloniranje je kopiranje udaljenog repozitorija, ali tako da novi repozitorij ostaje ”svjestan” da je on kopija nekog udaljenog repozitorija. Klonirani repozitorij čuva informaciju o repozitoriju iz kojeg je kloniran. Ta informacija će mu kasnije olakšati da na udaljeni repozitorij šalje svoje izmjene i da od njega preuzima izmjene. Postupak je jednostavan, moramo znati adresu udaljenog repozitorija, i tada će nam git s naredbom: git clone <lokacija\_repozitorija> kopirati projekt, zajedno sa cijelom poviješću na naše račcunalo. Sad u njemu možemo gledati povijest, granati, commitati... Ne može bilo tko kasnije svoje izmjene poslati nazad na originalnu lokaciju. Za to moramo imati ovlasti. U Mercurialu ova naredba je: hg clone <lokacija\_repozitorija>.  
Jedna od prednosti korištenja hg clone je da nam omogućuje kloniranje repozitorija preko mreže. Druga je činjenica da se sjeća mjesta na kojem smo klonirali, što nam može biti korisno kada želimo dohvatiti nove izmjene iz drugog repozitorija.

Naredbom git remote <server\_URL> povezujemo naš lokalni repozitorij s udaljenim poslužiteljem. U Mercurial sustavu naredba izgleda ovako: remote <server\_URL>. Ovo nam omogućuje stvaranje, pregled i brisanje veza s drugim repozitorijima. Veze više nalikuju oznakama, a ne izravnim vezama u druge repozitorije.

U većini operacija od nas se očekuje da iniciramo interakciju s drugim repozitorijima. Bez da mi pokrenemo neku radnju, git neće nikad kontaktirati udaljene repozitorije. Slično, drugi repozitorij ne može našeg natjerati da osvježi svoju sliku. Kao što smo mi inicirali kloniranje, tako i mi moramo inicirati ažuriranje grane. To radimo pomoću naredbom git fetch. To je naredba koja povlači podatke s udaljenog repozitorija na lokalni repozitorij.

Naredba git pull u Gitu i hg pull –u u Mercurialu je naredba koja naredba povlači podatke sa udaljenog repozitorija u lokalni i zatim odmah automatski spaja udaljenu granu koju pratimo u trenutnu granu. Na taj se način pronalazi sve promjene iz repozitorija na određenom URL-u i dodaje ih u lokalni repozitorij (trenutni, osim ako nije naveden -R). Prema zadanim postavkama, ovo ne ažurira kopiju projekta u radnom direktoriju.

Naredbama git push origin <branch\_name> i hg push -B <bookmark\_name> guramo promjene na lokalnoj grani na udaljeni repozitorij odnosno na udaljeni server. Ovo je radnja s kojom aktivno mijenjamo neki udaljeni repozitorij. Prebacivanje naših lokalnih izmjena na udaljeni repozitorij ovisi o tome imamo li ovlasti za to ili ne. Udaljeni repozitorij mora biti tako konfiguriran da bismo mogli raditi push opciju. Ukoliko nemamo ovlasti, sve što možemo napraviti je zamoliti njegovog vlasnika da preuzme izmjene. Taj proces se zove pull request ili zahtjev za pull s njegove strane. Dakle, izmjene preuzimamo s git pull, odnosno hg pull –u za Mercurial. Hg povlači sve izmjene, ali ne ažurira automatski radni direktorij s promjenama

Želimo li se prebaciti u drugu granu, a imamo tekućih izmjena, git nam to ponekad neće dopustiti. S git stash možetmo privremeno spremiti izmjene koje smo radili u nekoj grani. Kad se kasnije vratimo na prvotnu granu, prethodno spremljene izmjene možemo vratiti nazad. Ekvivalent u Mercurialu je naredba hg shelve, a naredbom hg unshelve vračamo promjene u radni direktorij.