

1

Upravljanje verzijama

2017/18.03

Upravljanje konfiguracijom

2

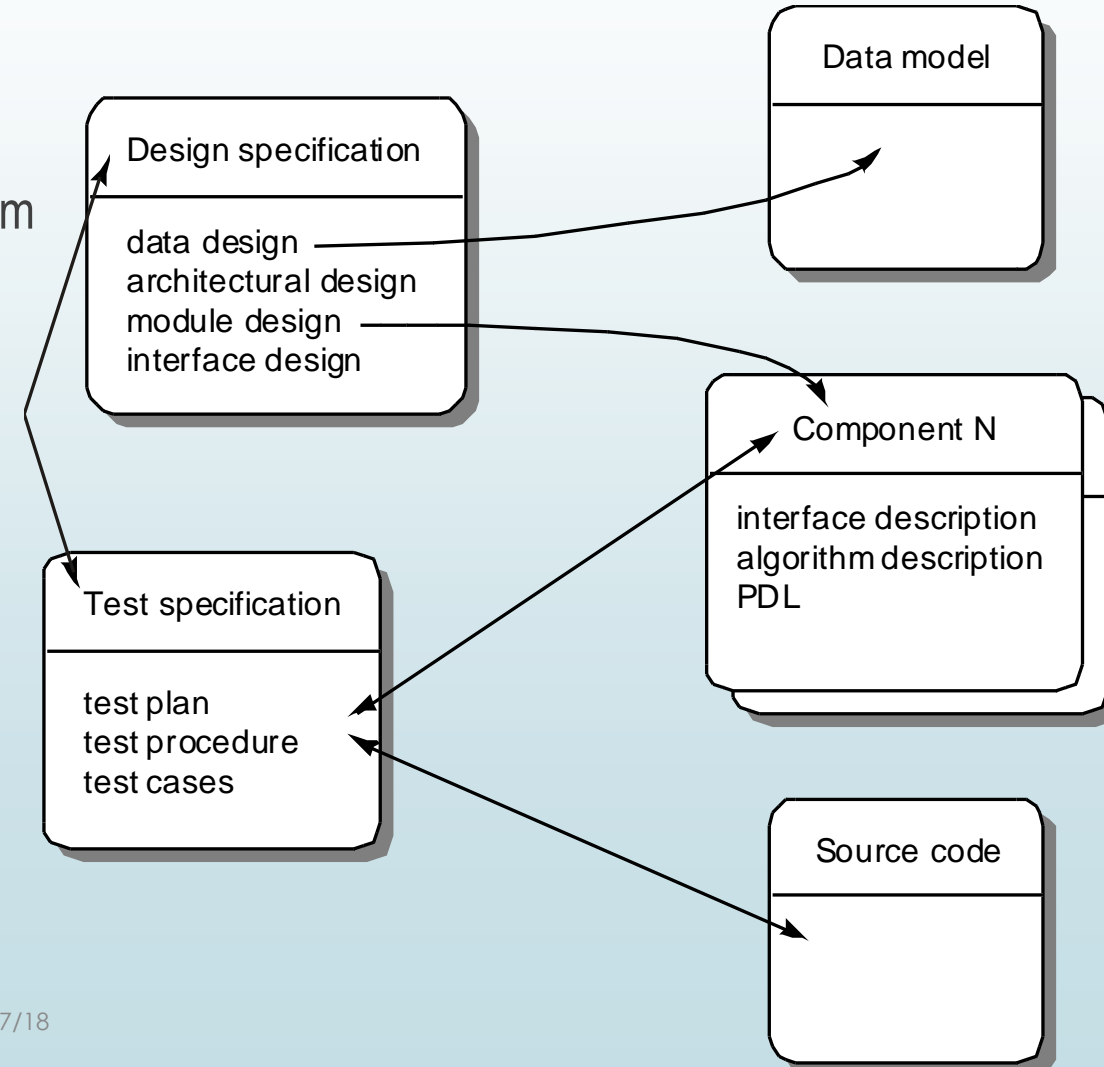
➤ Konfiguracija

- imenovani skup konfiguracijskih elemenata u određenoj točki životnog ciklusa

➤ Element konfiguracije (IEEE)

- agregacija hardvera i/ili softvera koja se tretira kao jedinka u procesu upravljanja konfiguracijom

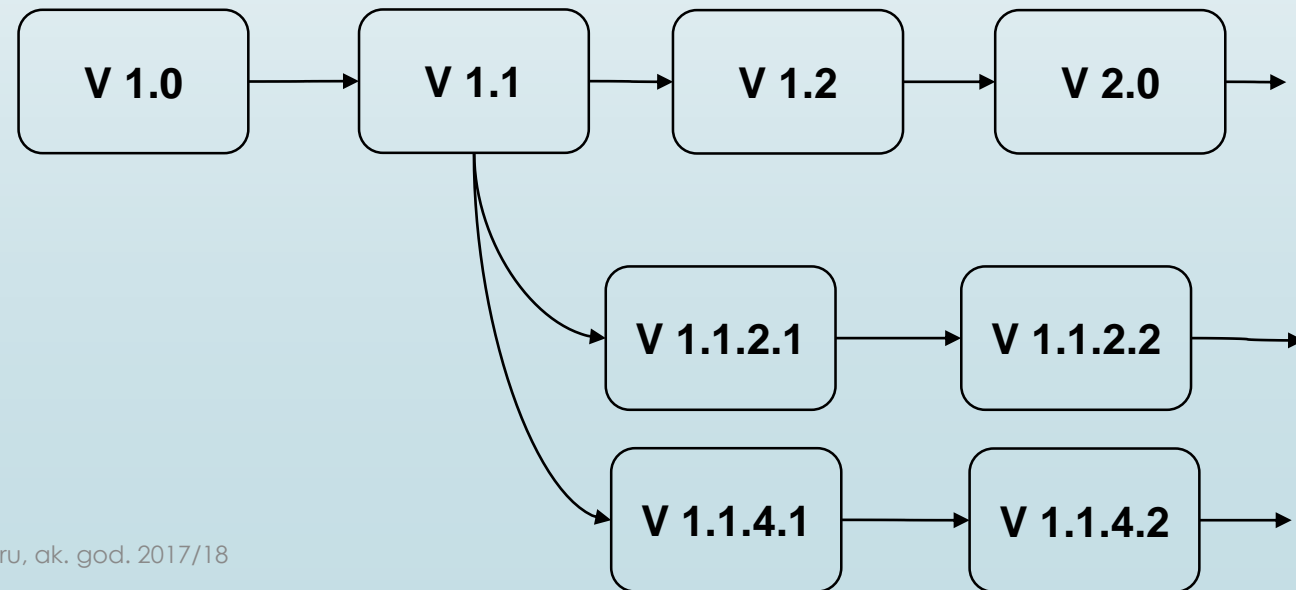
➤ Objekti konfiguracije



Verzije konfiguracije

3

- verzija, inačica (version) – određeno izdanje (issue, release) proizvoda
- objava, isporuka (release) – originalna verzija u primjeni, npr. zadnja v2.0
- revizija (revision) – ona koja se koristi umjesto originalne, podrazumijeva izmjene nastale kroz vrijeme (npr. zbog ispravljanja pogrešaka), npr. V1.2
- varijanta (variant) – alternativa originalu (hardverska platforma, različiti jezik), živi paralelno s njim, npr. v1.1.2.1
- osnovica (Baseline) – specifikacija proizvoda formalno provjerena i usvojena, koja služi kao temelj razvoja i koja se mijenja samo kroz formalnu proceduru kontrole promjena, IEEE (IEEE Std. No. 610.12-1990)



Označavanje verzija

4

- Verzija objektna datoteke u .NET Frameworku (assembly) određena je s četiri broja:

`<major version>.<minor version>.<build number>.<revision>`

- major version - mijenja se prilikom znatne promjene u (npr. kod redizajna koji prekida vertikalnu kompatibilnost sa starijim verzijama)
- minor version - mijenja se prilikom znatne promjene, ali uz zadržavanje kompatibilnosti s prethodnim verzijama
- build number - predstavlja ponovno prevođenje istog koda (npr. prilikom promjene platforme, procesora i slično)
- revision - primjenjuje se npr. prilikom izdavanja sigurnosnih zakrpa i sličnih manjih promjena

- Primjer: Properties \ AssemblyInfo

- major.minor.* (ili major.minor.build.*) automatski određuje build number i revision
 - build number: broj dana od 1.1.2000.
 - revision: broj sekundi proteklih od ponoći aktualnog dana podijeljen s 2

- .NET Core koristi Semantic Versioning: `major.minor.patch-suffix`

- Kontrola verzija (Version control) = verzioniranje
 - kombinira procedure i alate radi upravljanja različitim verzijama objekata konfiguracije, koji nastaju softverskim procesima
- Mogućnosti sustava kontrole verzija
 - baza projekata (project database) ili riznica (repository)
 - pohranjuje sve relevantne objekte konfiguracije
 - verzioniranje
 - razlikovanje pohranjenih inačica objekata konfiguracije
 - pomagalo za izradu (make facility)
 - prikuplja relevantne objekte i proizvodi određenu verziju softvera
 - praćenje problema (issues tracking), praćenje pogreški (bug tracking)
 - bilježenje i praćenje statusa tema koje se odnose na pojedine objekte konfiguracije

➤ Automatsko označavanje

➤ prednosti:

- eliminacija ručnog rada (npr. pisanja i izvedbe skripti)
- ne postoje dvije inačice s istom oznakom

➤ nedostaci:

- oznaka elementa ne podudara se s oznakom cijelog sustava
- novi brojevi ovise o danu i vremenu prevođenja
- verzija se mijenja pri svakom prevođenju, neovisno o tome jesu li se dogodile promjene ili ne

➤ Ručno verzioniranje

➤ prednosti:

- potpuna kontrola nad brojevima verzije
- moguća je sinkronizacija između verzije pojedinih komponenti i verzije cijelog sustava

➤ nedostaci:

- verzioniranje se mora raditi ručno
- moguće je napraviti više različitih objektnih datoteka s istim oznakama

Sustavi za upravljanje verzijama izvornog koda

7

- Subversion, CVS, Microsoft Team Foundation Server (TFS), Git, ...

- Centralizirani sustavi

- Subversion, CVS, TFS, ...

- Repozitorij se nalazi na zajedničkom serveru

- Korisnik sinkronizira sadržaj svojih mapa sa sadržajem na serveru

- Distribuirani

- Git kao najpoznatiji primjer distribuiranog sustava.

- Svaki korisnik ima kompletnu kopiju repozitorija zajedno s poviješću promjena

- Omogućavaju lokalno upravljanje promjenama i naknadnu sinkronizaciju

Karakteristike sustava za upravljanje verzijama

8

➤ Identifikacija verzija i izdanja

- Svaka verzija pohranjena u repozitoriju ima jedinstveni identifikator

➤ Kompaktna pohrana

- Umjesto kopije svake od verzija čuva se zadnja verzija te lista razlika između susjednih verzija

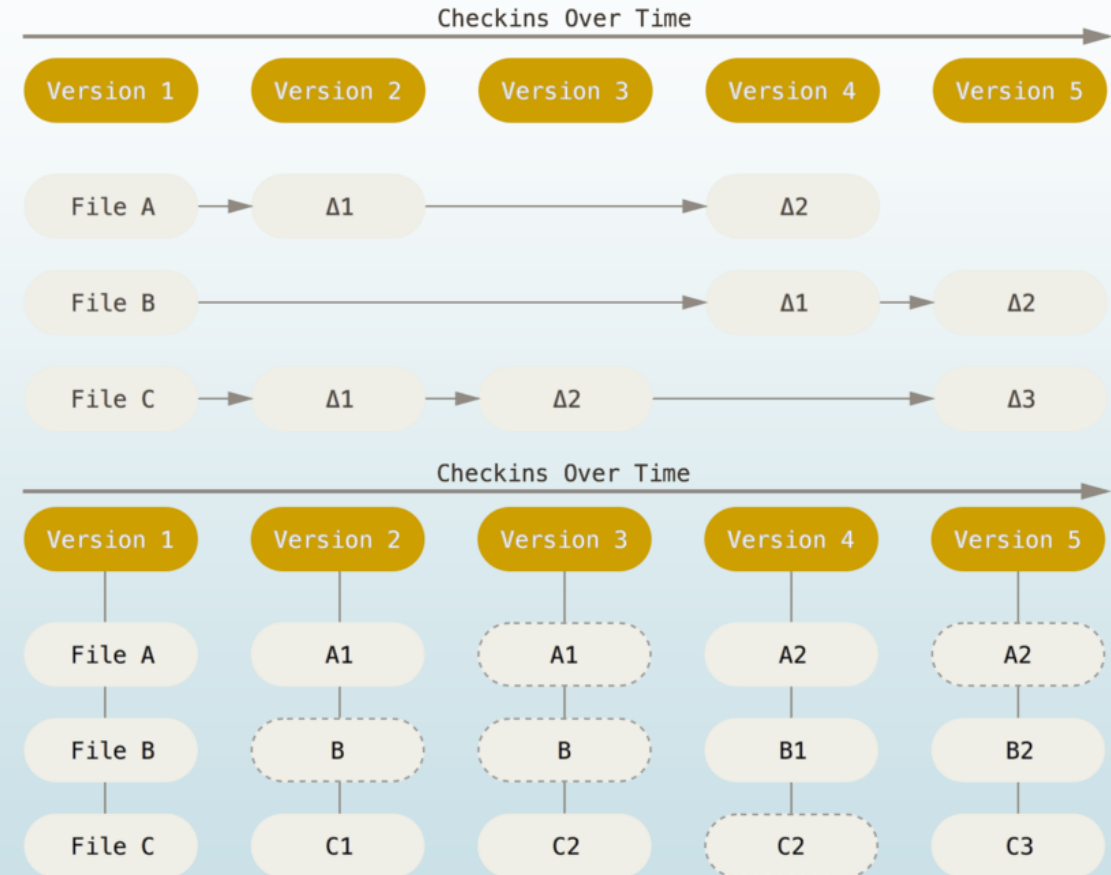
- Razlike se nazivaju deltama

➤ Evidencija povijesti promjena

- Svaka promjena se bilježi te se može rekonstruirati iz povijesti verzija

➤ Podrška za istovremeni razvoj

- Više korisnika može istovremeno koristiti raditi nad istim repozitorijem
- U slučaju istovremene izmjene istih datoteka sustava pruža mehanizam za razrješavanje konflikata

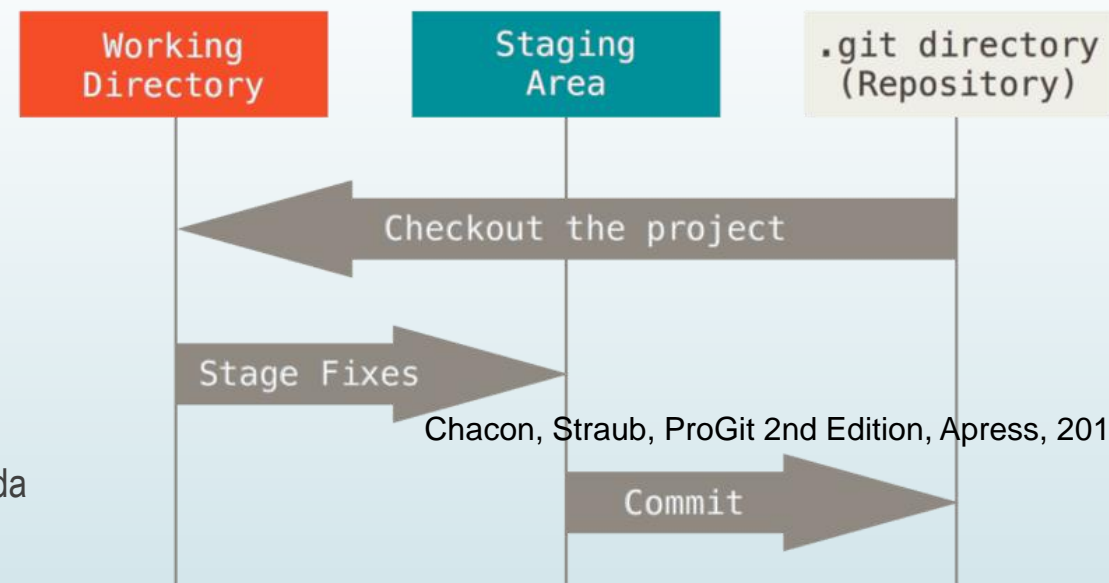


Chacon, Straub, ProGit 2nd Edition, Apress, 2014.

Neki od osnovnih pojmova

9

- Checkout
 - stvaranje lokalne radne kopije datoteka spremljenih u repozitoriju
 - Ovisno o alatu može ali ne mora uključivati dohvat određene verzije i odbacivanje trenutnih promjena
- Add/Stage
 - Odabir datoteka čije izmjene će se evidentirati
- Commit (check-in na TFS-u)
 - Pohrana izmjena u repozitorij
 - Nedjeljivi skup promjena se naziva commit u Git-u (changeset na TFS-u)
- Branch
 - stvaranje nove „kopije” (paralelne grane, verzije) izvornog koda
- Merge
 - spajanje više grana izvornog koda ili više verzija jedne datoteke u zajedničku granu
- Pull (Get Latest Version na TFS-u)
 - Dohvat zadnje verzije iz repozitorija
- Push
 - Slanje promjena iz lokalnog repozitorija na udaljeni repozitorij



Literatura za Git

10

- S. Chacon, B. Straub: Pro Git, 2nd Edition 2014. <https://git-scm.com/book/en/v2>
- T. Krajina: Uvod u Git: <https://tkrajina.github.io/uvod-u-git/git.pdf>