Configuration Management (Upravljanje projektom)

Nemanja Veljković
1007/2012

Sastoji se od:

- Upravljanje promenama(change managment)
- Upravljanje verzijama(version managment)
- Izgradnja sistema(system building)
- Upravljanje izdanjima(release management)

Upravljanje projektom:

- Pošto se softver često menja, sistemi mogu da se posmatraju kao skup verzija, od kojih svaka mora da se održava i upravlja.
- Verzije implementiraju predloge promena, ispravke grešaka i adaptacije za različiti hardver i operativne sisteme.
- Upravljanje projektom (CM) se bavi politikama, procesima i alatima za upravljanje promenama softverskih sistema. Nama je potreban CM jer je lako izgubiti trag o tome šta se menja.

Aktivnosti CM

- Upravljanje promenama

  Vođenje evidencije o zahtevima za izmene softvera sa klijentima i programerima, izračunavanje troškova i uticaja promena i utvrđivanje koje promene treba da se sprovedu.

- Upravljanje verzijama

  Vođenje računa o višestrukim verzijama sistemskih komponenti i obezbeđivanje da promene koje su na istim komponentama načinili različiti programeri nisu u konfrotaciji jedne s drugima.

Aktivnosti CM

- Izgradnja sistema

  Proces sastavljanja komponenti programa, podataka i biblioteka, zatim kompajliranje i povezivanje kako bi se stvorio izvršni sistem.

- Upravljanje izdanjima

  Priprema softvera za spoljašnja izdanja i praćenje verzija sistema koje su objavljene na korišćenje korisnicima.

CM terminologija

Termin	Objašnjenje
Configuration item or software configuration item (SCI) – Stavka konfiguracije	Sve u vezi sa projektovanjem softvera(dizajn, kod, test podaci, dokumentacija) koji je stavljen pod kontrolu konfiguracije. Često postoje različite verzije stavke konfiguracije. Stavke konfiguracije imaju jedinstveno ime.
Configuration control – Kontrola konfiguracije	Proces obezbeđivanja da verzije sistema i komponenti se evidentiraju i održavaju tako da su promene upravljane i sve verzije komponenti su prepoznate i sačuvane za vreme trajanja sistema.
Version - Verzija	Primerak stavke konfiguracije koja se razlikuje, na neki način, od drugih slučajeva te stavke. Verzije uvek imaju jedinstveni identifikator, koji se često sastoji od naziva stavke konfiguracije na koji je dodat broj verzije .
Baseline – Osnovna linija	Osnovna linija je kolekcija komponenata verzije koje čine sistem. Uzorci su kontrolisani, što znači da verzije komponenti koje čine sistem ne mogu da se menjaju. To znači da uvek treba da bude moguće da se ponovo kreiraju osnove od njenih konstitutivnih komponenti.
Codeline – Linija koda	Linija koda je skup verzija komponenti softvera i drugih stavki konfiguracije od kojih ta komponenta zavisi .

CM terminologija

Termin	Objašnjenje
Mainline – Glavna linija	Niz osnovnih linija koje predstavljaju različite verzije sistema.
Release - Izdanje	Verzija sistema koji je pušten na upotrebu klijentima (ili drugim korisnicima u organizaciji).
Workspace – Radni prostor	Privatna radna površina, gde se softver može menjati bez uticaja na druge programere koji mogu koristiti ili menjati taj softver.
Branching-Grananje	Stvaranje nove linije koda iz verzije u postojećoj liniji koda. Nova linija koda i postojeća linija onda se mogu razvijati samostalno.
Merging - Spajanje	Stvaranje nove verzije softverske komponente spajanjem odvojenih verzija u različitim kodnim linijama. Ove kodne linije su možda stvorene od prethodne grane jedne od uključenih kodnih linija.
System building – Izgradnja sistem	Stvaranje izvršne verzije sistema kompajliranjem i povezivanjem odgovarajućih verzija komponenti i biblioteka koje čine sistem.

Upravljanje promenama

- Organizacione potrebe i zahtevi se menjaju tokom životnog veka sistema, bagovi se moraju popravljati i sistemi moraju da se prilagode promenama u svom okruženju.
- Upravljanje promenama ima za cilj da obezbedi da napredak sistema je upravljani proces i da je prioritet dat najurgentnijim i isplativim promenama.
- Proces upravljanja promenama se bavi analizom troškova i koristi od predloženih izmena, odobravanjem tih promena koje su vredne i praćenje koje komponente u sistemu su promenjene.

Parcijalno završena forma za promenu zahteva

Change Request Form

Project: SICSA/AppProcessing **Number:** 23/02
Change requester: I. Sommerville **Date:** 20/01/09
Requested change: The status of applicants (rejected, accepted, etc.) should be shown visually in the displayed list of applicants.

Change analyzer: R. Looek **Analysis date:** 25/01/09
Components affected: ApplicantListDisplay, StatusUpdater

Associated components: StudentDatabase

Change assessment: Relatively simple to implement by changing the display color according to status. A table must be added to relate status to colors. No changes to associated components are required.

Change priority: Medium
Change implementation:
Estimated effort: 2 hours
Date to SGA app. team: 28/01/09 **CCB decision date:** 30/01/09
Decision: Accept change. Change to be implemented in Release 1.2
Change implementor: **Date of change:**
Date submitted to QA: **QA decision:**
Date submitted to CM:
Comments:

Faktori u analizi promena

- Posledice od ne činjenja promena
- Koristi od promena
- Broj korisnika koji su pogođeni promenom
- Troškovi izrade promene
- Ciklus izdanja proizvoda

Upravljanje promenama i agilne metode

- U nekim agilnim metodama, kupci su direktno uključeni u upravljanje promenama.
- Predstavnik kupca predlaže promenu uslova i radi sa timom kako bi procenio njegov uticaj i odlučuje da li da promene imaju prioritet nad funkcijama planiranim za narednu iteraciju sistema.
- O promenama koje će poboljšati softver odlučuju programeri koji rade na tom sistemu.
- Refaktorisanje , gde se softver konstantno poboljšava, se ne vidi kao opterećenje već kao neophodan deo razvojnog procesa.

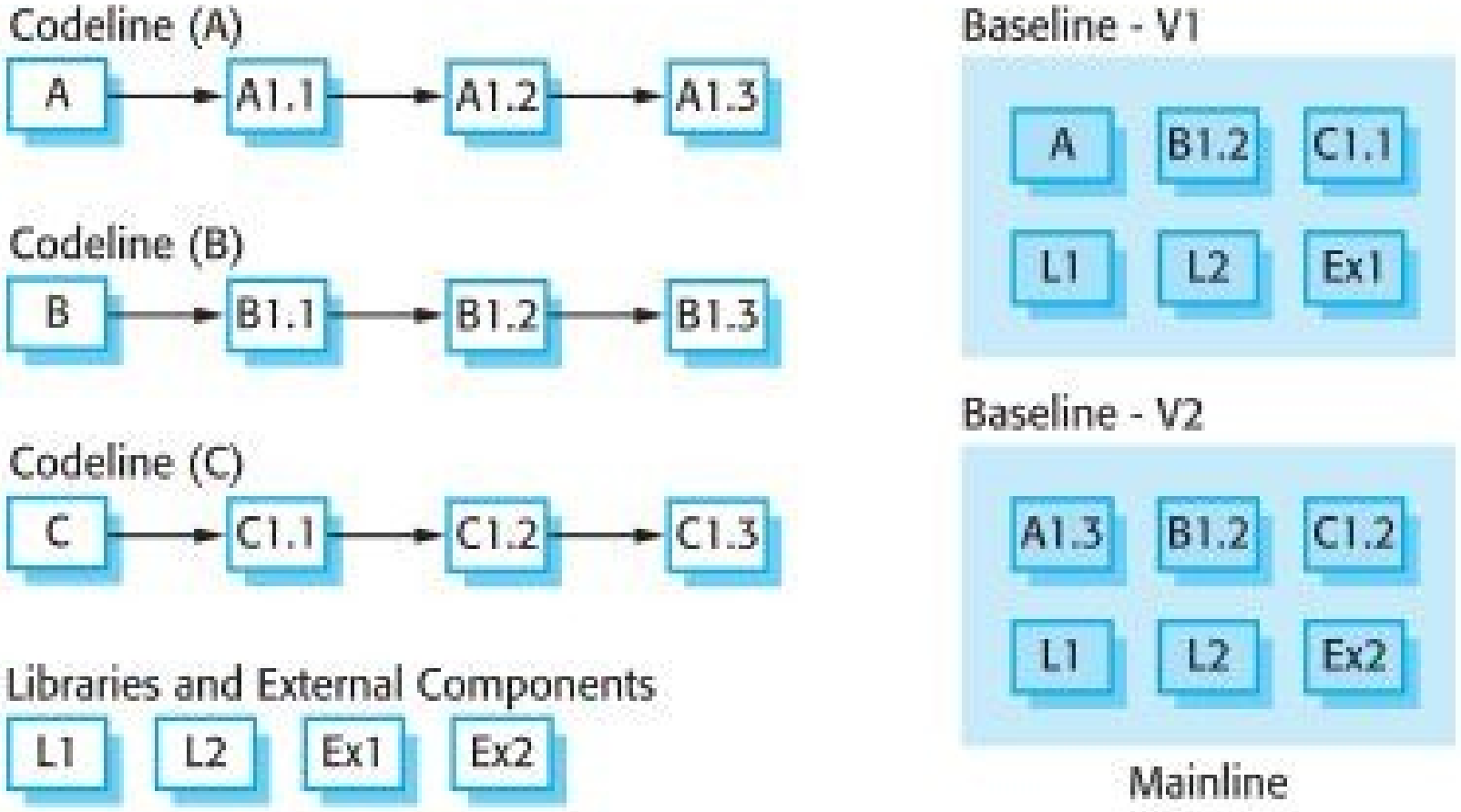
Upravljanje verzijama

- Upravljanje verzijama (VM) je proces praćenja različitih verzija softverskih komponenti ili konfiguracionih stavki i sistema u kojima se koriste ove komponente.
- To takođe podrazumeva obezbeđivanje da se promene od strane različitih programera na ovim verzijama ne mešaju jedne sa drugima.
- Zato se upravljanje verzijama može posmatrati kao proces upravljanja kodnih i osnovnih linija.

Kodne i osnovne linije

- Kodna linija je niz verzija izvornog koda sa novijim verzijama u nizu izvedenih iz ranijih verzija.
- Kodne linije se normalno primenjuju na komponente sistema, tako da postoje različite verzije svake komponente.
- Osnovna linija je definicija određenog sistema.
- Stoga osnovna liniija precizira verzije komponenata koje su uključene u sistem, plus specifikacija biblioteka koje se koriste, konfiguracioni fajlovi...

Kodne i osnovne linije



Osnovne linije

- Osnovne linije mogu biti specificirane korišćenjem konfiguracionog jezika, koji vam omogućava da definišete koje komponente su uključene u verziju određenog sistema.
- Osnovne linije su važne zato što se često mora ponovo kreirati određena verzija kompletnog sistema.
- Na primer, linija proizvoda može biti tako instancirana da postoje pojedinačni sistemi verzija za različite klijente.
- Možda ćete morati da ponovo isporučite verziju određenom kupcu, ako na primer, kupac izveštava o greškama u njihovom sistemu koje treba da se poprave.

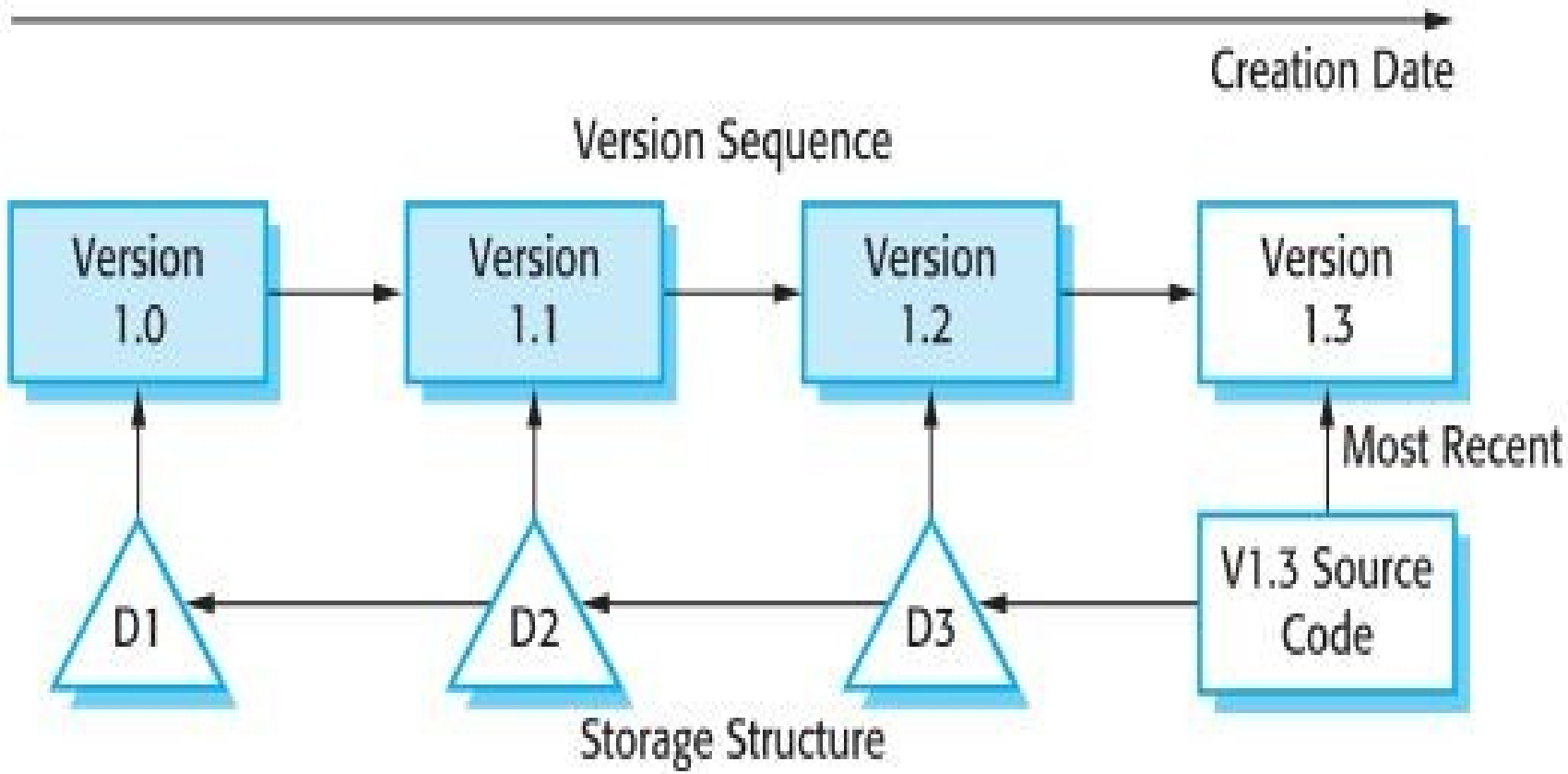
Sistemi za upravljanje verzijama

- Identifikacija verzija i izdanja
  - Upravljanim verzijama se dodeljuju identifikatori kada se podnose u sistem.
- Upravljanje skladištima
  - Da bi se smanjio prostor za skladištenje koji potražuju više verzija komponenti koje se razlikuju samo neznatno, sistemi za upravljanje verzijama obično pružaju objekte za upravljanje skladištima.
- Snimanje istorije promena
  - Sve izmene koda sistema ili komponenti se evidentiraju i navode.

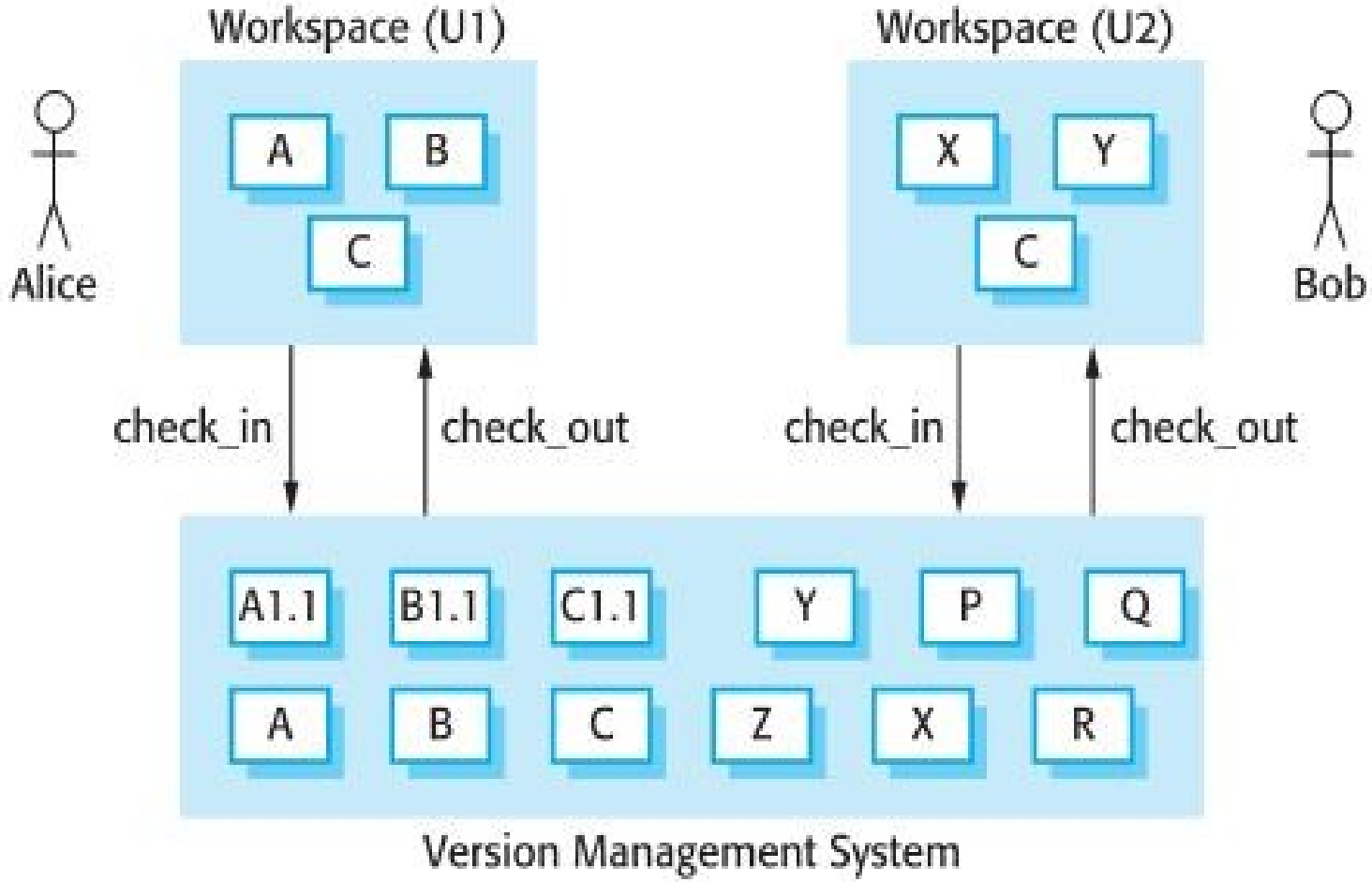
Sistemi za upravljanje verzijama

- Nezavisno programiranje
  - Sistem za upravljanje verzijama prati komponente koje su odabrane za editovanje i obezbeđuje da se promene napravljene na komponenti od strane različitih programera ne mešaju.
- Podrška projektu
  - Sistem za upravljanje verzijama može da podrži razvoj nekoliko projekata, koji dele komponente.

Upravljanje skladištima korišćenjem delti



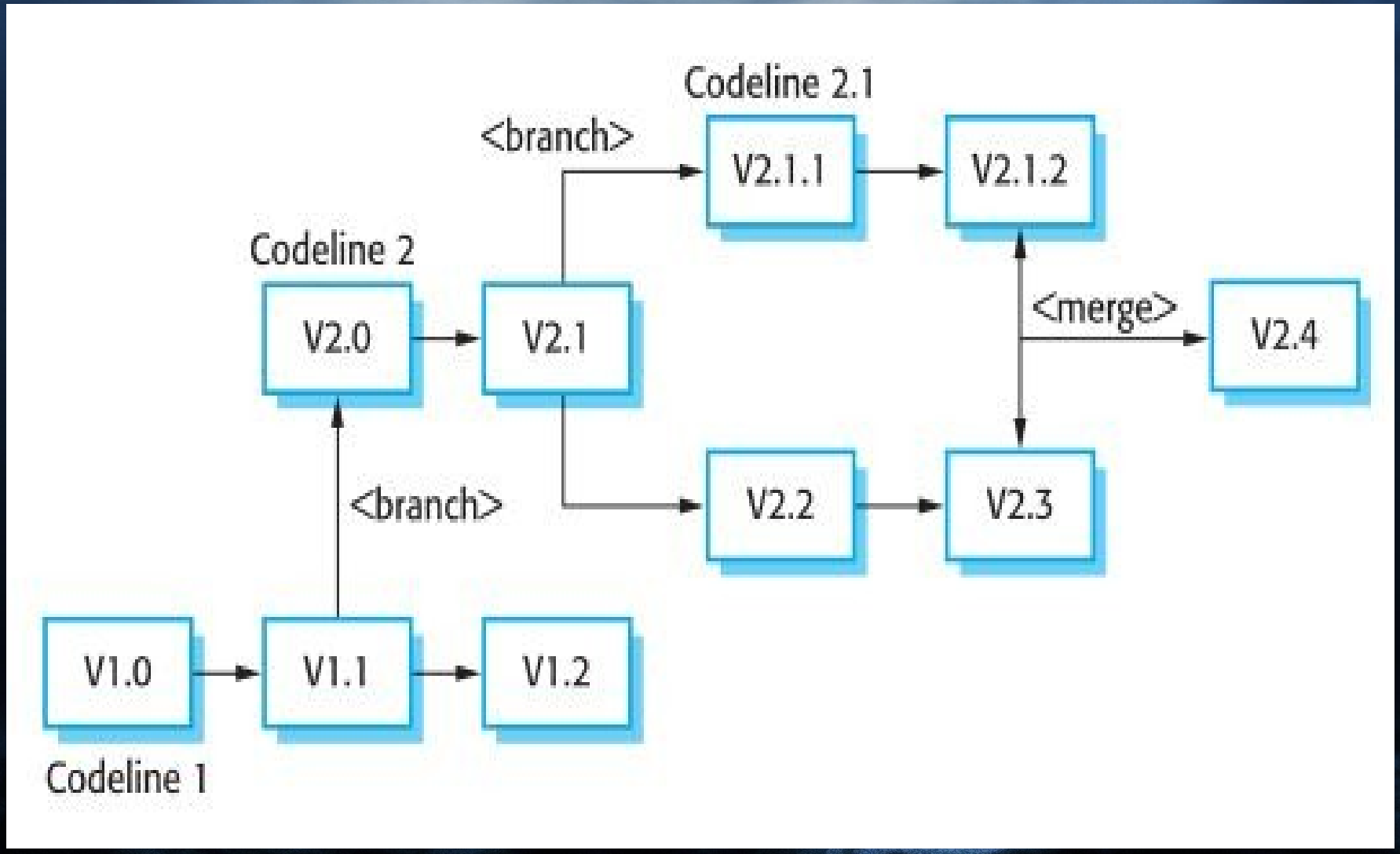
Check-in i check-out iz spremišta verzija



Grananje linija koda

- Umesto linearnog niza verzija koje odražavaju promene komponenti tokom vremena, može postojati nekoliko nezavisnih sekvenci.
- To je normalno u razvoju sistema, gde različiti programeri rade nezavisno na različitim verzijama izvornog koda i tako ga menjaju na različite načine.
- U nekom trenutku, možda će biti potrebno da se spoje grane kodnih linija da bi se stvorila nova verzija komponente koja uključuje sve promene koje su napravljene.
- Ako napravljene promene uključuju različite delove koda, komponente verzije se mogu automatski spojiti kombinujući delte koje se odnose na kod.

Grananje i spajanje



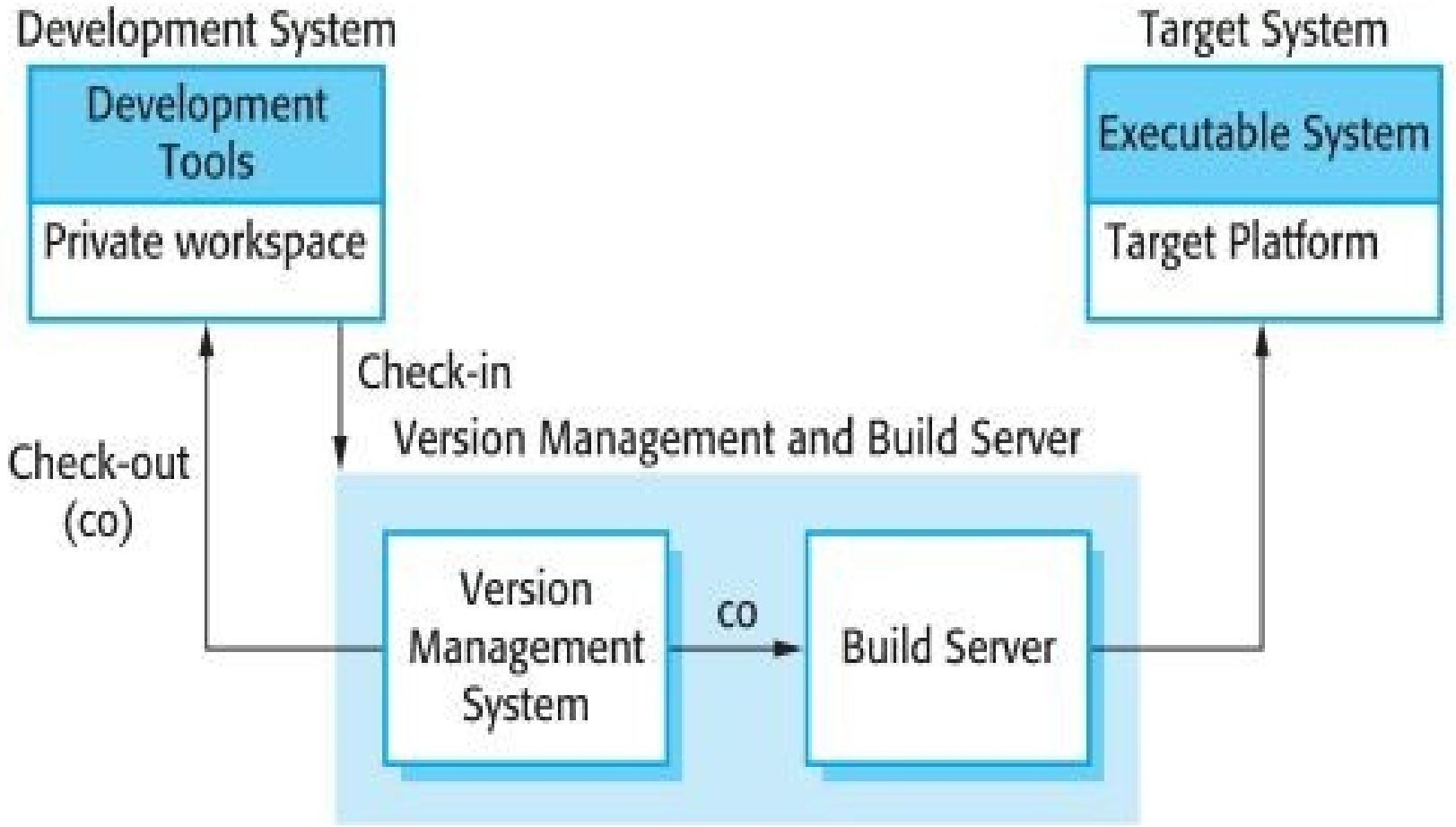
Izgradnja sistema

- Izgradnja sistema je proces stvaranja kompletnog, izvršnog sistema kompajliranjem i povezivanjem komponenti sistema, spoljašnjih biblioteka, konfiguracionih fajlova...
- Alati za izgradnju sistema i alati za upravljanje verzijama moraju komunicirati kao proces gradnje podrazumevaći proveru verzija komponenata iz spremišta koje su upravljane sistemom za upravljanje verzijama.
- Alati za izgradnju sistema koriste opis konfiguracije koji se takođe koristi za identifikaciju osnovne linije.

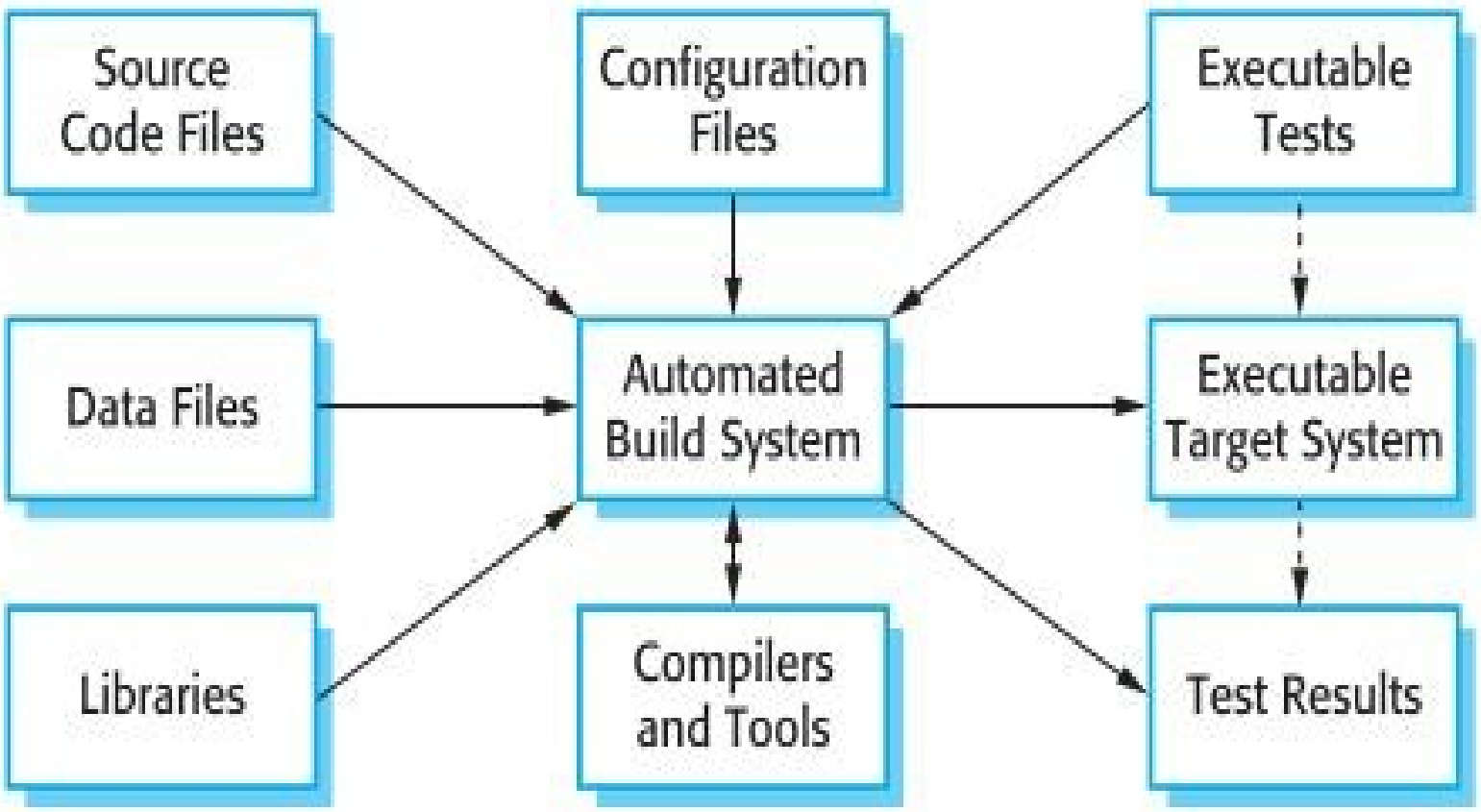
Platforme za izgradnju sistema

- Razvojni sistem, koji obuhvata razvojne alate kao što su kompajleri, editori izvornog koda...
  - Programeri proveravaju kod iz sistema za upravljanje verzijama u privatnom workspace-u pre nego što promene na sistemu.
- Server za izgradnju, koji se koristi za izgradnju konačne, izvršne verzije sistema.
  - Programeri check-inuju kod na sistem za upravljanje verzijama pre same izgradnje. Izgradnja sistema može da se osloni na spoljne biblioteke koje nisu uključene u sistem za upravljanje verzijama.
- Ciljno okruženje, koje je platforma na kojoj se sistem izvršava.

Razvoj, izgradnja i ciljne platforme



Izgradnja sistema



Funkcionalnost izgradnje sistema

- Generisanje build skripti
- Integracija sistema za upravljanje verzijama
- Minimalno rekompajliranje
- Stvaranje izvršnog sistema
- Automatizacija testova
- Izveštavanje
- Generisanje dokumentacije

Minimiziranje rekompajliranja

- Alati za podršku izgradnji sistema su obično dizajnirani da smanje količinu kompilacije koja je potrebna.
- Oni to rade tako što proveravaju da li je kompajlirana verzija komponente na raspolaganju. Ako je tako, nema potrebe da ponovo kompajliraju tu komponentu.
- Jedinstveni potpis identifikuje svaku verziju izvornog i objektnog koda i menja se kada je izvorni kod promenjen.
- Upoređivanjem potpisa na izvornom i objektnom kodu fajlova, moguće je odlučiti da li je izvorni kod korišćen za generisanje komponente objektnog koda.

Identifikacija fajlova

- Modifikacija timestamps-a
  - Potpis na datoteci izvornog koda je vreme i datum kada je datoteka izmenjena. Ako je datoteka izvornog koda komponente izmenjena nakon povezane datoteke objektnog koda, tada sistem pretpostavlja da je rekompajliranje neophodno kako bi se kreirala nova datoteka objeknog koda.
- Checksume izvornog koda
  - Potpis na datoteci izvornog koda je checksuma izračunata na osnovu podataka u datoteci. Funkcija cheksume izračunava jedinstveni broj koristeći izvorni tekst kao ulaz. Ako promenite izvorni kod (čak i za 1 karakter) , to će generisati drugačiji checksum. Zato možete biti sigurni da datoteke izvornog koda sa različitim checksumama su zapravo različite.

Timestamps vs checksums

- Timestamps
  - Izvorne i objektne datoteke su povezane po imenu pre nego po eksplicitnom potpisu izvorne datoteke, tada obično nije moguće izgraditi različite verzije komponenti izvornog koda u istom direktorijumu u isto vreme, kao što se mogu generisati objektne datoteke sa istim imenom.
- Checksume
  - Kada rekompajlirate komponentu, tada se ne prepiše objektni kod, kao što bi inače bio slučaj kada se timestamp koristi. Umesto toga, stvara se nova datoteka objektnog koda i označi sa potpisom izvornog koda. Paralelna kompilacija je moguća i različite verzije komponenti mogu biti kompajlirane u isto vreme.

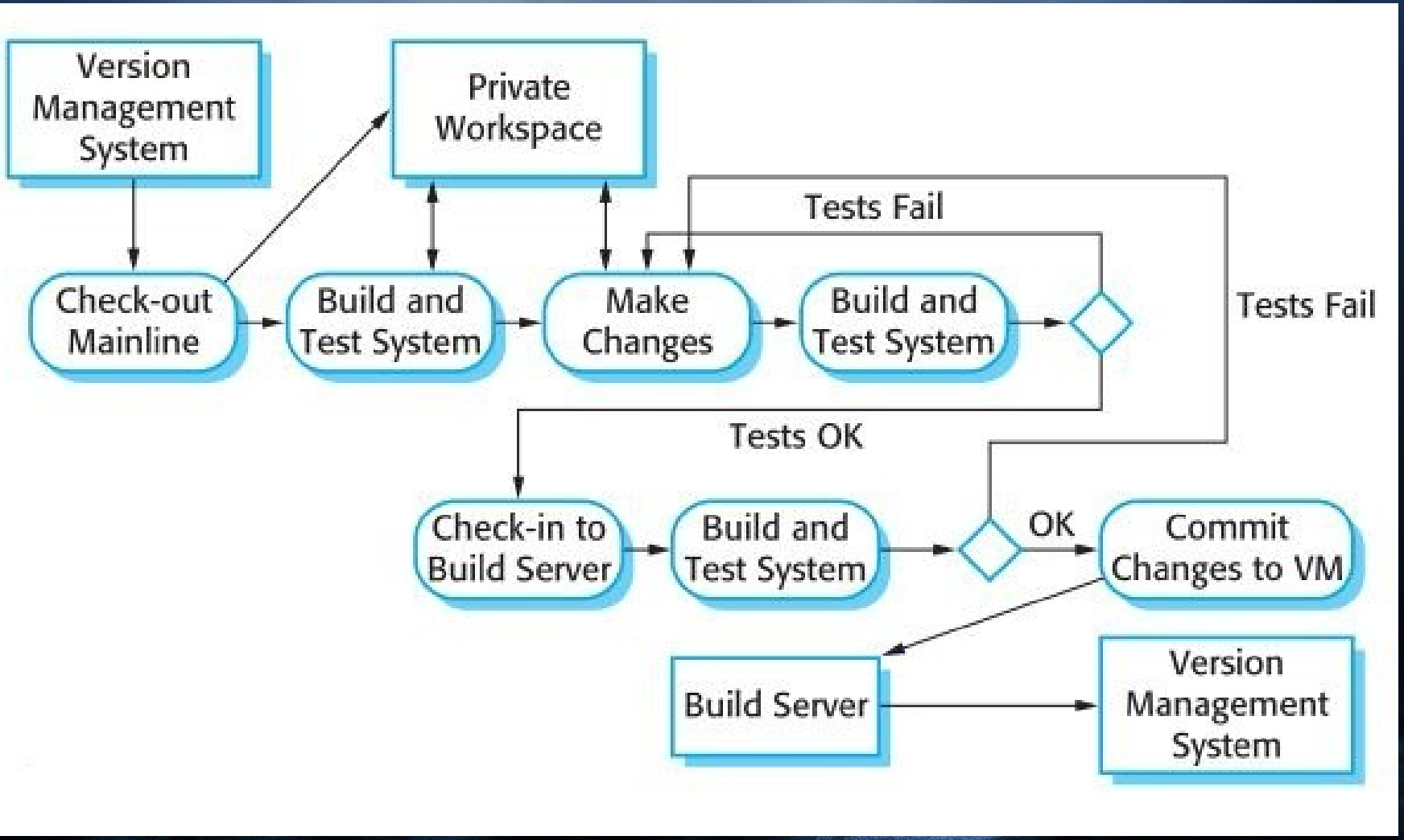
Agilna izgradnja

- Čekirajte sistem glavne linije od sistema za upravljanje verzijama u privatni radni prostor programera.
- Izgradite sistem i pokrenite automatske testove kako bi se osiguralo da izgrađeni sistem prolazi sve testove. Ako ne, konstrukcija je oštećena i vi treba da obavestite sve koji su čekirali u poslednjem osnovnom sistemu. Oni su odgovorni za popravku problema.
- Napravite promene u komponentama sistema.
- Izgradite sistem u privatnom radnom prostoru i ponovo pokrenite sistemske testove. Ako testovi ne prođu, nastaviti editovanje.

Agilna izgradnja

- Kada je sistem prošao svoje testove, proverite u izgrađenom sistemu, ali nemojte komitovati kao novu sistemsku osnovnu liniju.
- Izgradite sistem na serveru i pokrenite testove. Vi treba da uradite to u slučaju da su drugi menjali komponente od kada ste se odjavili sa sistema. Ako je ovo slučaj, proverite komponente koje nisu uspele i izmenite tako da ovi testovi prođu na vašem privatnom radnom prostoru.
- Ako sistem prođe svoje testove na izgradjenom sistemu, onda komitujte izmene koje ste napravili kao novu osnovnu liniju u sistemskoj glavnoj liniji.

Kontinuirana integracija



Dnevna izgradnja

- Razvojna organizacija postavlja vreme isporuke (recimo 14h) za sistemske komponente.
- Ako programeri imaju nove verzije komponenti koje su pisali, oni moraju da ih dostave do tog vremena.
- Nova verzija sistema je izgrađena od ovih komponenti kompajliranjem i povezivanjem tako da formiraju kompletan sistem.
- Ovaj sistem se onda dostavlja test timu koji sprovodi niz predefinisanih sistemskih testova.
- Greške koje su otkrivene tokom testiranja sistema su dokumentovane i vraćene sistemskim programerima. Oni popravljaju ove greške u narednoj verziji komponente.

Upravljanje izdanjima

- Izdanje sistema je verzija softverskog sistema koji se distribuira potrošačima.
- Za masovno tržište softvera, obično je moguće identifikovati dve vrste izdanja: Glavna (Major) izdanja koja pružaju značajnu novu funkcionalnost sistemu i manja (minor) izdanja, koja donose popravke bagova i rešavaju probleme korisnika koji su prijavljeni.
- Za korisnički softver ili linije softverskih proizvoda, izdanja sistema moraju da se proizvode za svakog kupca, jer individualni potrošači mogu biti koristiti nekoliko različitih izdanja sistema u isto vreme.

Praćenje izdanja

- U slučaju problema, može biti neophodno da se tačno reprodukuje softver koji je isporučen određenom kupcu.
- Kada je izdanje sistema proizvedeno, ono mora biti dokumentovano kako bi se osiguralo da se može ponovo stvoriti upravo u budućnosti.
- Ovo je posebno važno za prilagođene , dugoživotne ugrađene sisteme, kao što su oni koji kontrolišu složene mašine.
- Kupci mogu da koriste jedno izdanje ovih sistema dugi niz godina i mogu da zahtevaju određene promene u određenom softverskom sistemu dugo nakon prvobitnog datuma izdavanja.

Reprodukcija izdanja

- Da bi se dokumentovalo izdanje, morate da snimite posebne verzije komponenti izvornog koda koje su korišćene za kreiranje izvršnog koda.
- Morate držati kopije datoteka izvornog koda, odgovarajuće izvršne i sve podatke i konfiguracione fajlove.
- Takođe treba zabeležiti verzije operativnog sistema, biblioteke, kompajlere i druge alate korišćene za izgradnju softvera.

Planiranje izdanja

- Kao što je tehnički posao uključen u stvaranje izdanja distribucija tako i reklamni i promotivni materijali moraju da budu pripremljeni, kao i marketinške strategije da ubede kupce da kupuju novo izdanje sistema.
- Pravo vreme za izdanje
- Ako su izdanja suviše česta ili zahtevaju nadogradnju hardvera, korisnici se neće premestiti na novo izdanje, naročito ako trebaju da plate za to.
- Ukoliko su izdanja sistema suviše retka, tržišni udeo može biti izgubljen, jer su potrošači prešli na alternativne sisteme.

Komponente izdanja

- Osim izvršnog koda sistema, izdanje može obuhvatiti :
- konfiguracione fajlove koji definišu kako izdanje treba biti konfigurisano za određene instalacije:
  - datoteke sa podacima, kao što su datoteke sa porukama o grešci, koje su neophodne za uspešno funkcionisanje sistema;
  - instalacioni program koji se koristi da pomogne instaliranje sistema na ciljnom hardveru;
  - elektronsku i papirnu dokumentaciju koja opisuje sistem;
  - pakovanje i pridruženi publicitet koji su dizajnirani za to izdanje.

Faktori koji utiču na sistem za planiranje izdanja

Faktor	Objašnjenje
Tehnički kvalitet sistema	Ako su prijavljene ozbiljne sistemske greške koje utiču na način na koji mnogi kupci koriste sistem, može biti neophodno da se izda popravka greške. Manje greške sistema mogu se popraviti donošenjem zakrpe - patcha (obično se distribuiraju preko Interneta) koje se mogu primeniti na trenutno izdanje sistema.
Promena platforme	Možda ćete morati da stvorite novo izdanje za softversku aplikaciju, kada se izda nova verzija platforme operativnog sistema.
Lehmanov peti zakon	Ovaj 'zakon' sugeriše da ako dodate mnogo novih funkcionalnosti u sistem, takođe ćete uneti bagove koji će ograničiti količinu funkcionalnosti koje mogu biti uključene u sledećem izdanju. Dakle, izdanje sistema sa značajno novim funkcionalnostima možda će morati da bude praćeno izdanjem koje se fokusira na popravku problema i poboljšanje performansi.

Faktori koji utiču na sistem za planiranje izdanja

Faktor	Objašnjenje
Konkurencija	Za masovno tržište softvera, novo izdanje sistema može biti neophodno, jer je konkurentski proizvod uveo nove opcije i udeo na tržištu se može izgubiti ako se one ne obezbede postojećim klijentima.
Zahtevi marketinga	Odeljenje marketinga organizacije možda se obavezalo da ponudi novo izdanje da bude dostupno do određenog datuma.
Predlozi promene korisnika	Za prilagođene sisteme, kupac je možda napravio i platio za određeni set predloga promena sistema, pa očekuje izdanje sistema čim pre bude implementirano.

Pitanja?

Hvala na pažnji :)