Metodologije razvoja softvera

Uvod u metodologije razvoja softvera

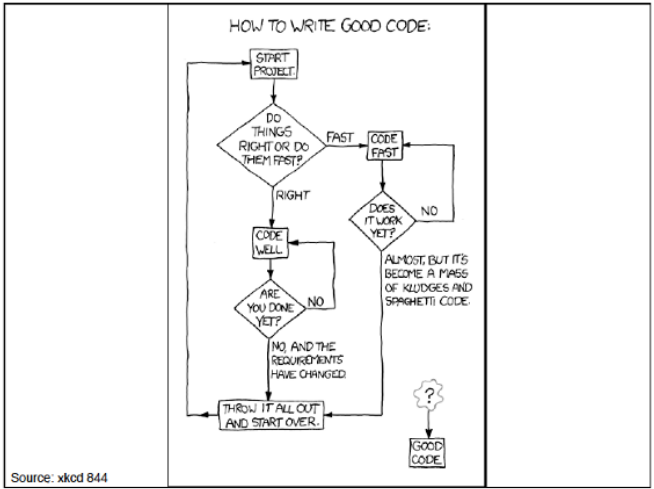
prof. dr Goran Sladić

Katedra za informatiku

2022.

Fakultet tehničkih nauka
Univerzitet u Novom Sadu

Kako napisati (dobar) softver?



Šta podrazumeva da je softverski projekat uspešan?

- *Od svih IT projekata koji su započeti od 5% do 15% će se odustati čak i pre prvog release-a. Mnogi drugi će imati prvi release koji će kasniti i/ili će probiti budžet i/ili će zahtevati ozbiljne izmene jer funkcionalnosti/performanse nisu zadovoljavajuće. Samo nekoliko će zaista i biti uspešni.* — Robert Charrette, Why Software Fails

Šta podrazumeva da je softverski projekat uspešan?

- Softver je isporučen na vreme
- Troškovi razvoja su u okviru budžeta
- Softver je zadovoljio zahteve korisnika po pitanju
  - Funkcionalnosti
  - Kvaliteta

Duke Nukem Forever

- Prva verzija 1996.
- Sledeća verzija planirana za 1997, a pojavila se tek 2011.

Tehnike za izbegavanje neuspeha

- Od 1960. pojavljuju se tehnike kako prevazići neuspehe u softverskim projektima
- Ne postoji univerzalno rešenje za sve projekte
- Mnoge od tehnika se nisu pokazale kao uspešne
- Problemi
  - Kreativnost – programiranje je u velikoj meri kreativan proces
  - Sloboda – postoji veliki stepen fleksibilnosti za skoro svaki programerski zadatak
    - Funkcionalni apsekti
    - Nefunkcionalni aspekti (skalabilnost, brzina, bezbednost, …)

Tehnike za izbegavanje neuspeha

- Jedan od elemenata koji može pomoći u izbegavanju/smanjenju neuspeha u softverskim projektima je primena odgovarajuće metodologije tokom razvoja softvera
- Ipak sama primena metodologije ne znači, *a priori*, da će neuspeh sigurno biti izbegnut

Definicija

- Metodologija razvoja softvera (Software Development Methodology - SDM)
- Okruženje za primenu prakse u softverskog inženjerstvu sa posebnim ciljem da se obezbede neophodna sredstva za razvoj softverskih sistema
- Sastoji se od dve osnovne celine
  - Skupa pravila za modelovanje, tj. jezik za modelovanje (sintaksa i semantika)
  - Procesa, koji
    - Daje smernice za redosled aktivnosti
    - Definiše koji elementi treba da se razviju koristeći jezike za modelovanje
    - Usmerava zadatke pojedinih inženjera i tima kao celine
    - Obezbeđuje kriterijume za praćenje i merenje proizvoda (softvera) i aktivnosti na projektu

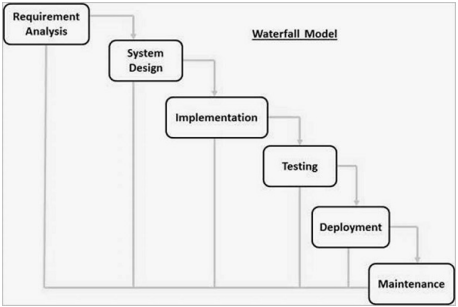
Razvoj metodologija

- Metodologije se mogu svrstati u skladu sa okolnostima koje su dovele do njihovog razvoja uključujući primenjeni pristup i metode
  - *Revolucionarne* – nove ideje i pristupi
  - *Evolutivne* – bazirane na postojećim metodologijama
  - *Integracione* – preuzima ideje/pristupe iz dve ili više metodologija
    - *Spajanje* - Obično se odvija po “design-by-committee” principu (određeno telo kreira novu metodologiju na osnovu postojećih)
    - *Ad hoc* – karakteristike preuzete iz istaknutih metodologija kako bi se zadovoljili zahtevi korisnika koji je kreirao
    - *Isplanirane (Engineered)* – nastale na osnovu analize problema u odgovarajućem domenu i na osnovu identifikovanih zahteva

Waterfall metodologija

- Jedna od prvih metodologija koja se pojavila za potrebe razvoja softvera
- Često se naziva i linearnim sekvencijalnim modelom razvoja softvera
- Sekvencijalne faze:
  - Prikupljanje i analiza zahteva
  - Dizajn sistema
  - Implementacija
  - Testiranje
  - *Deployment*
  - Održavanje

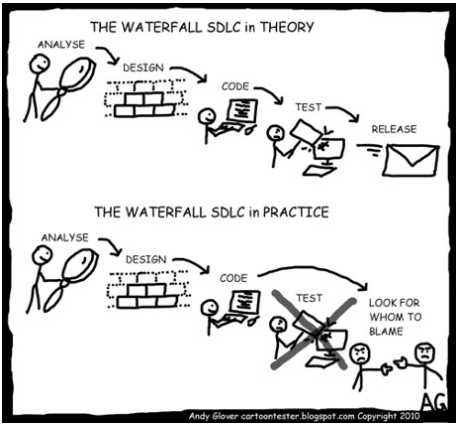
Waterfall metodologija



Waterfall metodologija

- Pomogla da se smanji stopa neuspeha u softverskim projektima
- **Ali** čak do 70% softverskih projekata koji koriste ovu metodologiju ne ispune bar deo svojih ciljeva (funkcionalnosti, vreme, resurse, finansije)

Waterfall metodologija - posledica



Waterfall metodologija - pogrešne pretpostavke

- Proces razvoja softvera može da se definiše kao predvidljiv i ponavljajući proces
- Što više planiramo na početku, imaćemo manje promena na samom projektu
- Moguće je ”prikupiti” sve zahteve unapred ili čak i ”razumeti” sve zahteve unapred
- Sve prikupljene informacije moguće je efikasno napisati u dokumentu bez neke velike kolaboracije između ljudi

Waterfall metodologija - realnost

- Uvek postoji promena
  - Neizvesnost/neodređenost se najbolje smanjuje ako se iterativno uči i o proizvodu i o procesu u okviru koga se proizvod razvija
- Zahtevi evoluiraju kako naše znanje raste – na osnovu iskustva i povratnih informacija (feedback-a)
- Razvoj novih proizvoda je evoluirajući i adaptivni proces
- Bitno znanje se gubi prilikom svakog vida “primopredaje” i ljudska interakcija je neophodna da bi se to prevazišlo

Waterfall - prednosti

- Jednostavan za razumevanje i korišćenje
- Lak za upravljanje
- Jasno definisane faze
- Faze se izvršavaju jedna po jedna
- Dobar je za male projekte gde su zahtevi dobro definisani
- Jednostavno raspoređivanje zadataka
- Proces i rezultati su dobro dokumentovani

Waterfall - mane

- Proizvod koji radi nastaje tek u kasnim fazama životnog ciklusa
- Visok rizik i dosta neizvesnosti
- Nije dobar izbor za kompleksne projekte
- Nije dobar izbor za duge projekte i projekte koji su već počeli
- Nije dobar izbor za projekte gde postoji velika verovatnoća za promenu zahteva
- Teško je meriti progres u okviru pojedinačnih faza
- Promene zahteva nisu predviđene
- Promena opsega projekta tokom razvoja može da se završi prekidom projekta

Metodologije posle waterfall-a

- Paralelni razvoj
  - Umesto da se dizajn i implementacija rade sekvencijalno, formira se neki generalni dizajn čitavog projekta i onda se projekat deli u manje delove
- V – modeli
  - Bazirani na različitim vrstama testiranja
- Spiralni modeli
  - Kombinacija waterfall modela sa iterativnim razvojem
- RAD – Rapid Application Development metodologije
  - Fazni razvoj
    - Sistem se sekvencijalno razvija kroz različite verzije
  - Razvoj prototipa
    - Prvo se razvija prototip sistema
  - Razvoj baziran na prototipu koji se odbacuje
    - Razvijaju se prototipi u različitim fazama koji se potom odbacuju

V model

- Izvršavanje procesa dešava se sekvencijalno u obliku slova V
- Poznat je i pod nazivom Model Verifikacije i Validacije
- Predstavlja proširenje waterfall modela i bazira se na asocijaciji faza testiranja za svaku fazu razvoja

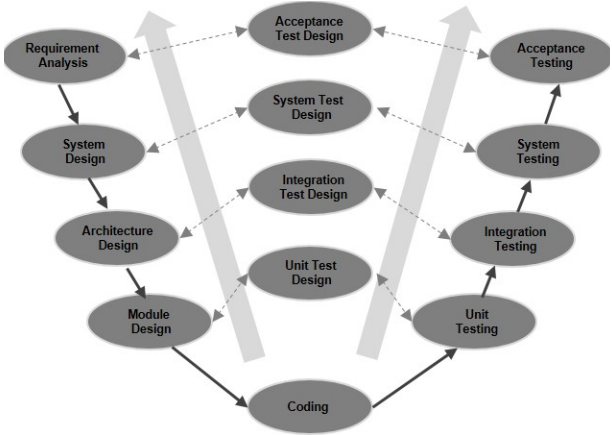
V model - Faze Verifikacije

- Analiza zahteva poslovanja - u ovoj fazi se radi planiranje dizajna testova prihvatljivosti (acceptance test design planning)
- Dizajn sistema
- Arhitektonski dizajn - High Level Design
- Dizajn modula - Low Level Design

V model - Faze Validacije

- Unit (jedinično) testiranje
- Integraciono testiranje
- Sistemsko testiranje
- Testovi prihvatljivosti (Acceptance testing)

V model


Requirement Analysis
Acceptance Test Design
Acceptance Testing
System Design
System Test Design
System Testing
Architecture Design
Integration Test Design
Integration Testing
Module Design
Unit Test Design
Unit Testing
Coding

V model - prednosti

- Zahtevi su dobro definisani i jasno dokumentovani
- Definicija proizvoda je stabilna
- Tehnologija nije promenjiva i jasna je projektnom timu
- Nema dvosmislenih ili nedefinisanih zahteva
- Projekat je kratak

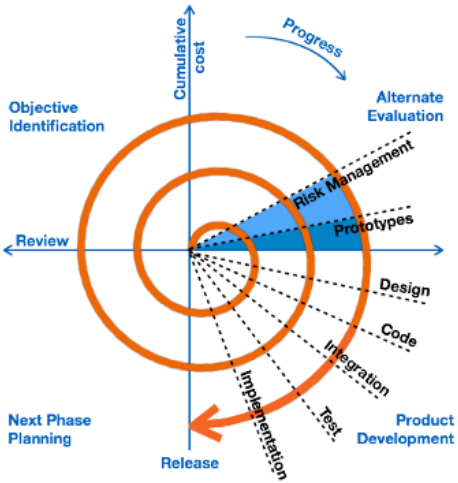
V model - mane

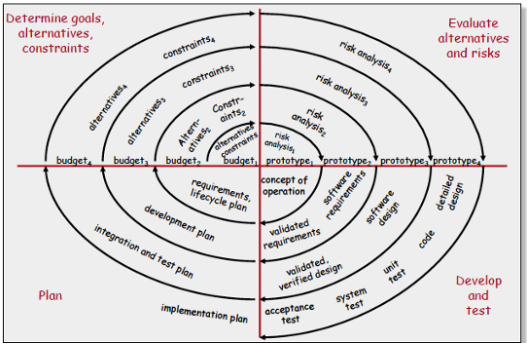
- Visok rizik i neizvesnost
- Nije dobar izbor za kompleksne projekte
- Nije dobar izbor za duge projekte i projekte koji su već počeli
- Nije dobar izbor za projekte gde postoji velika verovatnoća za promenu zahteva
- Kada projekat dođe u fazu testiranja, teško je vratiti se i promeniti funkcionalnost
- Proizvod koji radi nastaje tek u kasnim fazama životnog ciklusa

Spiralni model

- Kombinuje ideju iterativnog razvoja sa sistemski kontrolisanim aspektima waterfall modela
- Ima četiri faze:
  - Identifikacija - startuje prikupljanjem zahtevima poslovanja u osnovnoj spirali, kako proizvod biva zreliji, identifikuju se sistemski zahtevi, zahtevi podsistema i jedinični zahtevi
  - Dizajn - startuje konceptualnim dizajnom koji uključuje arhitektonski dizajn, logički dizajn modula, fizički dizajn proizvoda i finalni dizajn u daljim spiralama
  - Konstrukcija - u baznoj spirali proizvod se razvija kao Proof of Concept (POC) da se dobije mišljenje klijenata, dok se u sukcesivnim spiralama pravi model proizvoda koji zadovoljava specifičnije zahteve i dizajn
  - Evaluacija i analiza rizika - identifikuju se, estimiraju i nadgledaju tehničke mogućnosti proizvoda i upravlja se rizicima poput probijanja vremena isporuke ili budžeta

Spiralni model





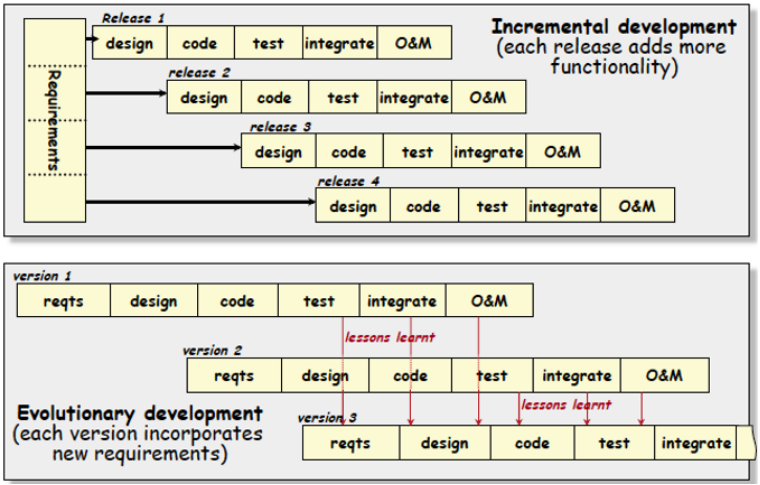
Spiralni model - prednosti

- Promena zahteva je podržana
- Dopušta upotrebu prototipa
- Zahtevi se mogu preciznije definisati
- Korisnici rano vide sistem
- Razvoj se može podeliti u manje delove i rizični delovi se mogu razvijati ranije

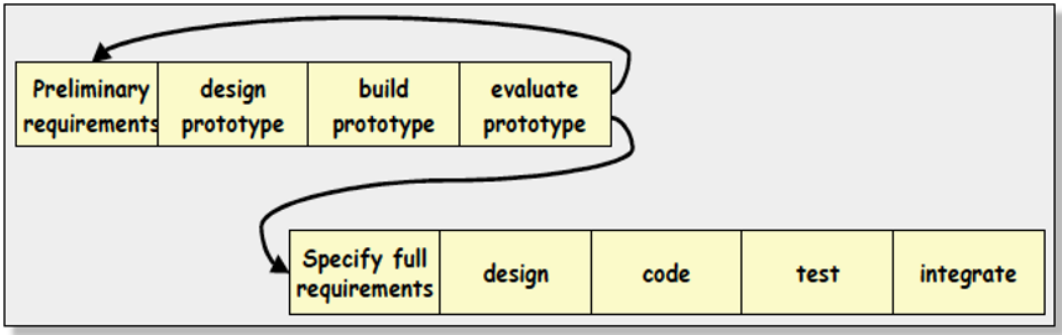
Spiralni model - mane

- Menadžment je kompleksniji
- Kraj projekta se ne mora znati u ranim fazama razvoja
- Nije pogodan za male projekte niskog rizika i može biti skup pristup
- Proces je kompleksan
- Spirala može da ide u beskonačnost
- Veliki broj međufaza zahteva dobru dokumentaciju

Fazni razvoj



Razvoj prototipa



RAD - prednosti

- Promena zahteva je podržana
- Napredak se može meriti
- Povećanje produktivnosti rada u manjim timovima za kratko vreme
- Skraćeno vreme razvoja
- Povećana ponovna iskoristivost komponenti
- Brza inicijalna revizija proizvoda
- Ohrabruje klijente da daju svoj komentar na proizvod

RAD - mane

- Samo proizvodi koji se mogu modularizovati mogu koristiti model
- Zahteva tehnički dobro potkovane inženjere i dizajnere
- Zahteva dobre veštine modelovanja
- Nije primenjiv na jeftinije projekte jer je cena modelovanja i automatskog generisanja koda visoka
- Menadžment je kompleksniji
- Zahteva učešće klijenata tokom razvoja

Agilne metodologija

- Agilne metodologije
  - Fokusirane na programiranju
  - Jako malo fiksnih pravila – ideja je u prilagođavanju
  - Otklanjaju neke nedostatke uočene kod ranijih metodologija uključujući i waterfall
  - I dalje nisu **silver bullet**, ali se u praksi pokazalo da vrlo često daju bolje rezultate nego neke druge metodologije