**Softversko inzinjerstvo**

**II kolokvijum**

**Darko Stevanović**

1. *Navesti osnovne metode inzinjeringa zatheva*

Procesi inzinjeringa zahteva su obicno vodjeni odgovarajucom metodom. Te metode predstavljaju nacine za dobijanje modela sistema.(To sve vreme I pravimo)

Najpoznatije metode:

1.1. Data Flow Diagrams - DFD

1.2. Metode koje koriste relacioni model podataka

1.3. Objektno - orijentisane metode

1.4. Formalne metode

* + 1. Sintaksa
    2. Semantika
    3. Relacije

1.5. Metode zasnovane na ponasanju sistema

* + 1. Use-case specifikacija
    2. Viewpoint metode

1. *Navesti osnovne metode za projektovanje sw I objasniti svaku od njih*
   1. Strukturne metode - funkcionalno orijentisane (funkcije nad podacima)
   2. OO metode - kolekcije objekata, UML
   3. Kombinovane metode
2. *Navesti osnovne arhitekturne modele za projektovanje softvera. Objasniti Control arhitekturni model.*

Arihitekturni modeli( (stilovi) predstavljaju projektne obrasce za arhitekturu SW-a. Definisu komponente I konektere koji cine arhitekturu sistema! Zato se arhitektura sistema moze predstaviti kao graf.

Osnovni modeli //7

Control arhitekturni obrazac - koristi se za centralizovanu kontrolu

Osnovne grupe ovog modela: //4 COMF

* 1. Call-return modeli - sekvencijalni modeli
     1. Top-down pristup
  2. Menager modeli - konkurentni sistemi
     1. Jedna sistemska komponenta koja kontrolise
  3. Feed-back modeli
     1. Kontrolisu se pojedine komponente i njihove vrednosti se koriste za podesavanje sistema
  4. Open-loo modeli
     1. Kontra od feedback-a, kontrolisu se, ali se nekoriste za podesavanja

1. *Navesti metode testiranja sw I objasniti svaku od njih*

Kriterijumi:

* 1. Pristup izvornom kodu
  2. Korisnicki interfejs / Proramski interfejs

|  |  |  |
| --- | --- | --- |
| **Metod** | **Testiranje preko** | **Koristi se kod** |
| Crne kutije | interfejsa namenjenom krajnjim korisnicima |  |
| Bele kutije | preko izvornog koda | Komponentnog testiranja |
| Sive kutije | Okoline za testiranje | 1. Klijent-server test inzinjeri 2. Inzinjeri sa bazom podataka 3. Za xml fajove 4. Za Konfiguracione fajlove 5. Za internu strukturu ili alg. |

1. *Navesti I kratko opisati osobine dobro projektovanog sw: //4*
   1. Hijerarhija - hijerarhija komponenta
   2. Modularnost - moduli sa jasno definisanim interfejsom
   3. Nezavisnost - grupacija slicnih stvari u nezavisne module, lokalizovane posledice (prednost!)
   4. Jednostavan interfejs
2. *Navesti osnovne karakteristike I graficki ilustrovati broadcast arhitekturne modele*

Event -> sve komponente -> komponenta koja upravlja dogadjajem je obradjuje

Upotreba: integracije podsistema

Primer: debager + editor

1. *Sta predstavlja I koji su ciljevi inspekcije sw?*

Analiza statitckih reprezentacija sistema radi otkrivanja problema.

Koriste se alati za:

* 1. Analizu koda
  2. Pregledavanje dokumentacije

Cilj.

1. *Opisati I kratko objasniti strukturu test slucajeva*

Test slucaj - skup test podataka I pridruzenih procedura //2

Razvijen sa odredjenim ciljem

Kao sto je da se izvrsi odredjeni put u programu

Da se proveri funkcionalni ili nefunkcionalni zahtev

Test slucaj sadrzi: // 2

Opis funkcionalnosti koja se testira

Kako pripremiti okruzenje

Struktura test slucajeva: //3

* + 1. Uvod: id, naziv, namena…
    2. Aktivnost test slucaja: //4
       - 1. Konfiguracija t.s.
         2. Inicijalizacija
         3. Finalizacija
         4. Akcije I ulazni podaci
    3. Ocekivani rezultati

1. *Navesti I kratko opisati koje sve (kategorije) strategija za upravljanje rizikom postoje:*

Rizik - verovatnoca da se pojavi nepovoljna prilika

|  |  |  |
| --- | --- | --- |
| **Strategija** | **Cilj** | **Primena** |
| izbegavanja | Smanjiti verovatnocu pojave rizika | Defekti u komponentama |
| minimizacije | Smanjiti uticaj rizika na projekat, ili proizvod | Bolesti osoblja |
| Planovi za nepredvidjene dogadjaje | Kako se radi sa rizikom | Finansijski problemi organizacije |

1. *Graficki ilustrovati I kratko opisati odnos izmedju evolucije I servisiranja*

*(Slika)*

* 1. Evolucija zavisi od: //3
     1. Tipa sw
     2. Koriscenog procesa razvoja
     3. Vestina I iskustva inzinjera
  2. Servisiranje - cilj odrzavanje operativnosti
     1. Otklanjaju greske
     2. Izmene
  3. Postepeno ukidanje

1. *Kratko opisati I graficki ilustrovati proces upravljanja rizikom*

Upravljanje rizikom podrazumeva:

* 1. Identifikovanje rizika
  2. Skiciranje plana minimizacije

|  |  |
| --- | --- |
| **Tip rizika - 3P** | **Uticaj rizika na** |
| projekata | resursi |
| proizvoda | kvalitet ili performanse sw proizvoda |
| poslovanja | organizijacija |

|  |  |
| --- | --- |
| **Proces upravljanja rizikom - *MAPI*** | **Zadatak** |
| identifikacija | Identifikovati sva 3 tipa rizika |
| analiza | Verovatnoca I posledice primecenih rizika |
| planiranje | Skiciranje planova izbegavanja I minimiziranja |
| monitoring | nadgledanje |

1. *Ukratko opisati I navesti prednosti I nedostatke slojevitog (eng. layered) arhitekturnog modela*

Komponente - slojevi

Konektori - interfejsi

Prednosti:

- mali uticaj promene sloja (samo na 2 nivoa)

- laka zamena identicnih interfejsa

Nedostaci:

- ne mogu svi sistemi ovako da se organizuju

1. *Sta predstavlja verifikacija, a sta validacija?*

Verifikacija - sopstvena specifikacija

Validacija - zahtevi narucioca

1. *Opisati I uporediti top-down I bottom-up pristupe projektovanje sw*

Top-down:

vrh -> podsistemi na dnu

Lose strane:

- forsiranje razvoja pojedinih grana, dok neke nisu jos zapocete

- Ne sagledavaju se vec postojece komponente

Bottom-up:

dno -> podsistemi na vrhu

- Pocinje se od gotovih komopnenata.

Losa strana:

- tehnike na najvisim nivoima se dobijaju u kasnim fazama

1. *Navesti I kratko opisati tipove odrzavanja sw*

Odrzavanje - izmena programa nakon pustanja u upotrebu

|  |  |
| --- | --- |
| **Cilj** | **Opis** |
| ispravka sw gresaka | Otklanjanje nedostataka |
| prilagodjenje sw za drugacije radno okruzenje | Drugaciji sistem od onog za koji je inicijalno impl. |
| Dodavanje nove ili izmene postojece funkcionalnosti | Novi zahtevi |

1. *Objasniti osnovne elemente Use-case dijagrama. +PRIMER*

|  |  |
| --- | --- |
| **Element** | **Opis** |
| Slucaj koriscenja | Kada slucaj koriscenja pocinje/zavrsava se  Kada interaguje sa objektima  Kada se razmenjuju objekti |
| Akteri | Skup uloga - korisnik/sistem |
| Relacije | Asocijacije  Zavisnosti(2)  generalizacije |
| Paketi | Moguce grupisanje:  Aktera  Slucajeva koriscenja  veza |

1. *Koje su prednosti, a koji nedostaci kod Client-server arhitekturnog modela?*

DS

Server + klijent + mreza

Komponente: server, klijent

Konektori: mreza, servisi servera

|  |  |
| --- | --- |
| **Prednosti** | **Mane** |
| Efikasno koriscenje mreznih sistema | Neefikasna razmena podataka |
| Omogucava slabiji hw kod klijenta (server radi) | Redundantnost podataka |
| Lako dodavanje novih servera I upgrade postojecih | Ne postoji centralni registar imena servera |

1. *Kada se po RUP metodologiji vrsi detaljno projektovanje I sta se pod tim podrazumeva?*

U fazi izrade.

Podrazumeva:

* 1. Arhitekturni projekat
  2. Detaljne dijagrame klasa
  3. 4 + 1 model sistema
  4. Projektovanje:
     1. Struktura podataka I baza podataka
     2. Korisnickog interfesja
     3. Algoritama za funkcije I modele

Izlaz: Detaljni projekat sistema

1. *Sta su kontrolne tacke, a sta izlazni produkti? Ko odredjuje kontrolne tacke I izlazne produkte projekta I na osnovu kojih parametara?*

Kontrolne tacke - krajnje tacke

Izlazni produkti - rezultati projekata

1. *Koje su prednosti, a koji nedostaci kod Event - driven arhitekturnog modela (I osnovne karakteristike)?*

Skripta.

1. *Objasniti alfa, beta I gama testiranje*
   1. Alfa (prijemno testiranje) - bela kutija, mozda crna I siva.
   2. Beta - uz posebne korisnike, crna kutija
   3. Gama - nakon eksploatisanja sw
2. *Sta predstavlja upravljanje projektima I po cemu su sw projekti osobeni?*

Upravljanje projektima: // OPR

* 1. Organizovanje
  2. Planiranje
  3. Rasporedjivanje sw projekata

Specificnosti:

* 1. Proizvod je nevidljiv, fleksibilan
  2. Swe nema status zdrave inzinjerske discipline
  3. Proces razvoja nije standardizovan
  4. Proizvodi su unikatni

1. *Skripta*
2. *Sta predstavlja I sta se dobija reinzinjeringom sistema*

Reinzinjering - restruktuiranje I redokumentovanje sw

Prednosti: smanjenje cene I rizika

Aktivnosti: //5

* 1. Prevodjenje izvornog koda
  2. Analiza programa u cilju boljeg razumevanja
  3. Unapredjenje strukture programa
  4. Modularizacija progarama
  5. Reinzinjering podataka

1. *Opisati I graficki ilustrovati proces izrade rasporeda rada na projektu I tipicne probleme vezane za ovu aktivnost*
   1. Razbiti projekat na zadatke
   2. Konkurentno
   3. Minimizirati zavisnost

Problemi:

* 1. Utvrdjivanje troskova
  2. Produktivnost nije proporcionalna broju ljudi
  3. Dodavanje ljudi
  4. Neocekivani problemi

1. *Graficki ilustrovati I kratko opisati proces testiranja sw*
   1. Komponentno testiranje
      * 1. Komponente
        2. Bele kutije
   2. Sistemsko testiranje
      * 1. Grupa komponenti
        2. Funkcionalni I nefunkcionalni zahtevi
        3. Sve kutije
   3. Prijemno testiranje
      * 1. Radno okruzenje sa podacima korisnika
        2. Crne kutije
2. *Sta je refaktoring, a sta reinzinjering? Navesti slicnosti I razlike?*

Refaktorisanje -

Unapredjenje programa

Radi smanjenja degradacije.

Predstavlja preventivno odrzavanje:

Poboljsava se struktura

Smanjuje slozenost

Povecava razumljivost

Napomena: treba izbegavati nove funkcionalnosti!

Reinzinjering - objasnjen.

1. *Objasniti razlike izmedju staticke I dinamicke verifikacije sw*

Inspekcija sw - staticka verifikacija:

* 1. analiza statickih reprezentacija sistema
  2. Alati za analizu koda

Alati za pregledavanje dokumentacije

Testiranje sw - dinamicka verifikacija:

* 1. Izvrsenje I posmatranje proizvoda
  2. Izvrsavanje sa test podacima

1. *Tipovi testiranja. Razlike izmedju testiranja I debaginga.*

Defektno testiranje

Validaciono testiranje

Testiranje - otkrivanje gresaka u sistemu

Debaging - lociranje I reparacija gresaka + glupa definicija