Uvod u softversko inženjerstvo - Teorija

* Projekat je **privremena** struktura koja služi za **organizaciju** i **upravljanje** aktivnostima koje za cilj imaju **izradu** jasno definisanog **proizvoda** ili **skupa proizvoda**.
* “**Proces** koji pomaže **projektnim timovima** da **koordiniraju** vlastite **napore** u cilju isporučivanja traženog **proizvoda** ili **usluge**, u **pravo vreme**, **pravom kupcu** i **u okviru raspoloživih resursa**.” - Upravljanje projektom

**Rukovodilac projektom**

brine o problemima:

* DA LI POSEDUJEMO POTREBNE RESURSE ZA REŠAVANJEODREĐENOG ZADATKA? (novac,ljudi,vreme isl)
* KOLIKO SMO SIGURNI DA SU ELEMENTI PRIMOPREDAJE KOREKTNO SPECIFICIRANI
* AKO SU DATA ODREĐENA OGRANIČENJA VEZANA ZA RESURSE, KAKO SE ONI MOGU NAJEFEKTNIJE ISKORISTITI (kako najbolje iskoristiti resurse uokviru ogranicenja)
* KOJE POUKE SE MOGU IZVUĆI IZ REALIZACIJE RAZMATRANOG ZADATKA/PROJEKTA A KOJE MOGU UNAPREDITI PERFORMANSU BUDUĆIH ZADATAKA/PROJEKATA
* NA KOJI NAČIN TEKUĆI STATUS REALIZACIJE PROJEKTA MOŽE UTICATI NA MODIFIKOVANJE PLANA

**Dimenzije procesa**

Aktivnosti - Razvoj (zahtevi, analiza, specifikacije dizajna, implementacija, testiranje), Kontrola (potvrda kvaliteta, konfigurisanje, validacija i verifikacija), Upravljanje (planiranje projekta, alokacija resursa, procena troškova, ugovaranje), Rad (obuka, prelazak, rad i gašenje)

Aspekti - Apstrakcije (fundamentalni principi i formalni modeli), Predstavljanje(notacione aspekte i jezike), Metode(formalne metode, metodologije i tekuću praksu), **Alati (obuhvataju pojedinačne softverske alate kao i integrisane skupove alata – Alati opšte namene (email, browser, tekst procesori), Alati namenjeni dizajnu i implementaciji (za modelovanje, kompajliranje, interpreteri, sintaksno osetljivi editori), Alati za podršku upravljanju projektima, Integrisana razvojna okruženja)**, Dejstva (obuhvataju merenja, analizu i evaluaciju softverskih procesa, uticaj softvera na organizacione sisteme, mere i standarde), **Komunikacije (pisana komunikacija, govorna komunikacija, dokumentacija – Online (Help, Kontekstna pomoc), Offline (Uputstva, priručnici), Inline(Komentari u programu), Tutorial(Asistencija pri ovladavanju softverskim proizvodom).**

**PRIMENA SISTEMATIČNOG, DISCIPLINOVANOG I MERLJIVOG PRISTUPA RAZVOJU, EKSPLOATACIJI I ODRŽAVANJU SOFTVERA. – Softversko inženjerstvo**

**Prvih 8 razloga obuhvataju (razlozi neuspešnosti projekata):**

**1.   nekompletna specifikacija zahteva**

**2.   nedovoljno uključivanje korisnika**

**3.   nedostatak resursa**

**4.   nerealna očekivanja**

**5.   nedostatak podrške rukovodstva**

**6.   izmena zahteva i specifikacija**

**7.   odsustvo planiranja**

**8.   kada je završen sistem nije više bio potreban**

**Trajanje** – vreme (iskazano u radnim danima, bez vikenda, praznika i drugih neradnih dana) proteklo između početka i završetka projekta, odnosno aktivnosti – realno proticanje vremena

**Napor** – rad potreban za izvršenje neke aktivnosti ili projekta – vreme rada

**Postimplementaciona analiza** - Predstavlja evaluaciju ciljeva projekta, postignutih rezultata u skladu sa planom projekta, budžetom, vremenskim odrednicama, kvalitetom predmeta primopredaje i specifikacijama naručioca ( Da li je postignut cilj projekta, da li je projekat predat na vreme u skladu sa budzetom, specifikacijama, da li je narucilac zadovoljan, da li je realizovan poslovni cilj projekta, koje su lekcije naučene, Šta je radilo? Šta nije radilo?)

**Životni ciklus softverskog** **proizvoda** obuhvata put od njegovog konceptualnog specificiranja, preko implementacije, isporuke, korišćenja, održavanja i migracija do izvođenja iz eksploatacije.

**Testiranje** – jedinično, integraciono, funkcionalno, perfomansi, prihvatanja, instalaciono

Integraciono testiranje je faza u testiranju softvera u kojoj se pojedinačni moduli softverskog sistema kombinuju i testiraju kao grupa i tako se otkrivaju greške.

**Taksonomija obrazovanja**

----------------------------------

Znanje (**Posmatranje i izvođenje informacija;Poznavanje datuma, događaja, mesta; Poznavanje osnovnih ideja; Vladanje pojedinim predmetima**)

Razumevanje (**Razumevanje informacija; Shvatanje značenja;Prevođenje znanja u nove kontekste;Interpretiranje činjenica, poređenje, suprotstavljanje;Sortiranje, grupisanje, spoznavanje uzroka; Predviđanje posledica**)

Primena (**Korišćenje informacija;Korišćenje metoda, koncepata, i teorija u novim situacijama; Rešavanje problema korišćenjem potrebnih veština i znanja**)

Analiza (**Uočavanje šablona;Organizovanje delova;Prepoznavanje skrivenih značenja;Identifikacija delova)**

Sinteza (**Korišćenje starih ideja za kreiranje novih;Na osnovu datih činjenica obaviti poopštavanje;Povezivanje znanja iz više oblasti;Predviđanje, izvođenje zaključaka)**

Ocena (**Poređenje i pravljenje razlika između ideja;Ispitivanje valjanosti teorija, prezentacija; Izbor na bazi racionalnih argumenata; Verifikacija vrednosti dokaza;Prepoznavanje potencijalne subjektivnosti)**

3 činioca u procesu rešavanja problema:

1. Znanje
2. Iskustvo
3. Raspoloživa (materijalno-tehnička) sredstva

**Identifikacija problema** – Odvajanje poznatog od nepoznatog, šta je problem i koji je skup ograničenja pri realizaciji

**Dekompozicija problema**–Uočiti njegovu unutrašnju strukturu i veze između delova njegove unutrašnje structure, dekompozicija se završava kad se po svakom od delova rešavanja stigne do elementarnog problema (problem koji rešavač može rešiti na osnovu postojećeg znanja, iskustva i raspoloživih materijanlo-tehničkih sredstava)

1. Identifikacija problema
2. Dekompozicija problema
3. Definisanje algoritama (postupak rešavanja)
4. Implementacija algoritama zadatim ili izabranim tehničkim sredstvima
5. Testiranje funkcionalnosti, verifikacija i validacija
6. Integracija u radno okruženje I obuka korisnika za korišćenje
7. Dokumentovanje
8. Operativna eksploatacija i održavanja
9. Kapitalna unapređenja, migracija i izvođenje iz eksploatacije

Strategije za poboljšanje uspešnosti procesa izrade softvera

1. Uključivanje korisnika
2. Podrška rukovodnih struktura
3. Jasni poslovni ciljevi
4. Optimiziran obim projekta
5. Agilan process
6. Ekspertska znanja rukovodioca projekta
7. Upravljanje budžetom
8. Vrhunski resursi (vešti saradnici)
9. Formalna metodologija
10. Standardizovani alati I infrastruktura

Osnovni principi objektne orijentisanosti

1. Apstrakcije (omogućavaju rukovanje složenošću koncentracijom na esecijalne karakteristike po kojima se jedan entitet razlikuje od drugih)
2. Enkapsulacija (razdvaja implementaciju od njegovog javnog interfejsa, koristi se ponašanje objekta bez poznavanja na koji način je to implementirano)
3. Modularnost (Dekompozicija nečeg što je složeno na manje delove kojima se može lakše rukovati)
4. Hijerarhija (Bilo koje rangiranje ili uređenje apstrakcija u formi drveta)

Svaki sistem poseduje šest esencijalnih elemenata

1. Misiju (Cilj zbog kog sistem postoji)
2. Organizacionu strukturu (Skup delova sistema koji formiraju njegovu arhitekturu – unutrašnju strukturu)
3. Funkcionalnu strukturu (Predstavlja aktivnosti koje sistem sprovodi – funkcije)
4. Informacionu strukturu (Skup elemenata sa kojima I nad kojima se izvršavaju funkcionalne transformacije)
5. Upravljačku strukturu (Prinuda, spoljna ili unutrašnja, koja drži sistem na okupu, očuvanje sistema kao celine)
6. Veze sa okruženjem (Kontrolisane tačke preko kojih se obezbeđuje interoperabilnost)

Aspekti dizajna softverskih sistema

1. **Konceptualni (sistemski) dizajn**- precizna specifikacija iz koje **korisnik** nedvosmisleno i jednoznačno dobija informaciju o tome **šta će da radi softver.**
2. **Tehnički dizajn -** omogućava projektantima i programerima razumevanje stvarno potrebnog hardvera i softvera za rešavanje problema.
3. **Arhitektonski dizajn**  - bavi se izborom strategije rešavanja I modularizacijom sistema
4. Strategija
5. Klijent/server arhiktetura
6. Troslojna arhitektura
7. Agentska arhitektura
8. Dekompozicija
9. Bazirana na modulima
10. Bazirana na podacima
11. Bazirana na događajima
12. Bazirana na Ulaz/Izlaz
13. Bazirana na objektima
14. **Detaljni dizajn -** bavi se formulacijom detaljnih algoritama I strukture podataka neophodne za implementaciju modula

Karakteristike kvalitetnog dizajna

1. **Nezavisnost komponenti**
2. Veze među modulima
   1. Nepovezani
   2. Veze preko podataka
   3. Veze preko structure podataka
   4. Veze preko kontrola
   5. Veze preko zajedničkih elemenata
   6. Veze preko sadržaja

b) Unutrašnja funkcionalna snaga modula

1. Funkcionalna
2. Sekvencijalna
3. Komunikaciona
4. Procedurna
5. Temporalna
6. Logička
7. Koincidentna
8. **Identifikacija I rukovanje izuzecima** (situacija u suprotnosti sa onim što softver treba da radi)
9. **Otpornost na greške I rukovanje greškama** (posledica otkaz, rukovati njima u clju minimizacije vremena u kome sistem nije funkcionalan, otkriva se na osnovu otkaza)
10. **Smanjenje složenosti** (pojednostavljivanje strukture softverskog sistema i strukture podataka do nivoa potpune funkcionalnosti uz minimalnu složenost)

Dva ključna elementa koja podrazumeva izrada softvera

1. **PROCES**

*Proces* – niz koraka koji uključuju aktivnosti, ograničenja I resurse I rezultuju željenim postignućem

Modeli životnog ciklusa softvera

* **Modeli oslonjeni na jednu verziju softverskog proizvoda**

Big Bang, Model vodopada (Analiza zahteva, Dizajn sistema, Dizajn programa, Kodiranje, Testiranje I integracija, Testiranje sistema, Prihvatanje, Rad I održavanje), Modifikovani model vodopada ( Mogućnost vraćanja jedan nivo) V model – integrisano testiranje (Kodiranje u dnu)

* **Inkrementalni modeli** – jedna verzija uz izradu prototipa (dopunjuje proizvod u svakoj fazi)
  + Scrum Model- mali tim, niz sprintova kao etapa u razvoju softverskog proizvoda
    - Fountain model
* **Iterativni modeli** – više verzija (prerađuje proizvod u svakoj fazi)
  + Spiralni model I varijante
    - Boehm-ov
    - ROPES model
    - CIT (RUP) (kontrolisane iteracije)

TimeBox model

Ciklički model životnog ciklusa

Koevolutivni model

Keller-ov “Uradi sam” model životnog ciklusa softvera

**Ekstremno programiranje** – Metodologija razvoja softvera čiji je osnovni cilj poboljšanje kvaliteta I brzi odziv na zahteve korisnika (testovi se pišu pre kodiranja, korisničke price se koriste kao podloga za testove prihvatanja, koji se cesto rade, sastanci “s nogu”, kod - zajednickog vlasnistvo)

1. **PROIZVOD**
2. Klase softverskih sistema
   1. Odnost prema okruženju
      1. Batch
      2. Interaktivni
      3. Reaktivni
      4. Real Time
      5. Konkurentni
      6. Mrežni
      7. Distribuirani
   2. Interne karakteristike
      1. Tablično upravljanje
      2. Procesno upravljanje
      3. Upravljanje zasnovano na znanju
   3. Generičke oblasti primene
      1. Avio saobraćaj
      2. Komunikacioni sistemi
      3. Operativni sistemi
      4. Sistemi za rukovanje bazama podataka
      5. CASE, CAD, CAM, Alati
3. Opšti zahtevi prema softverskom proizvodu

Prenosivost, Zaštita, Pouzdanost, Ponovna iskoristivost, Robustnost, Bezbednost, Sigurnost, Mogućnost testiranja, Mogućnost korišćenja, Efikasnost, Otpornost na greške, Korektnost…

**Strategije niveliranja resursa**

* Korišćenje raspoloživog vremena klizanja – odnos između najranijeg početka (ES) i najkasnijeg završetka (LF) (najprihvatljivija)
* Pomeranje roka završetka projekta (najnepopularnija)
* Uravnotežavanje radnog vremena – (proglasiti vikend za radne dane)
* Dalje razlaganje aktivnosti u cilju restrukturiranja opterećenja
* Razvlačenje aktivnosti – smanjenje dnevnog procenta angažovanja resursa
* Angažovanje zamenskih resursa – moguće sa nižim stepenom veštine ali u dužem vremenskom periodu.

**Metode za procenu trajanja aktivnosti**

* Sličnost sa drugim aktivnostima – po analogiji (Similarity methode)
* Korišćenje istorijskih podataka (History data)
* Ekspertske procene (Expert advice)
* Delfi tehnika (Delphy technique)
* Modifikacija originalne Delfi-tehnike
* Tehnika tri tačke(tree point technique)
* Širokopojasna Delfi tehnika (Wide-band Delphy)

IZUZETAK – poznati dogadjaj koji u normalnim uslovima ne bi trebao da se desi

GRESKA – nesto sto nismo mogli predvideti, a moze da otkaz