Softversko inženjerstvo

Uvod u softversko inženjerstvo

dr Miloš Stojanović*

Visoka tehnička škola strukovnih studija Niš

2017.

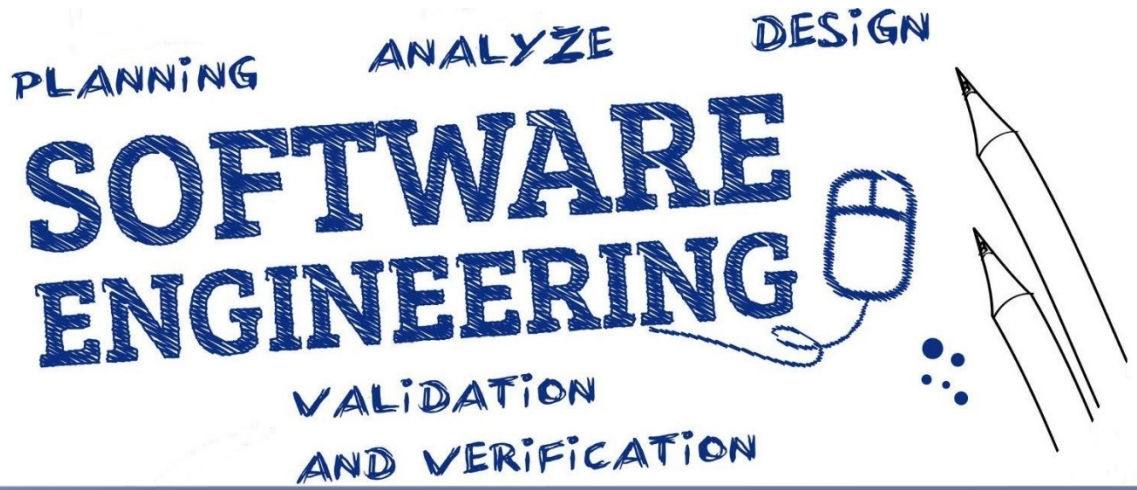
*milos.stojanovic@vtsnis.edu.rs

Informacije o predmetu

- Naziv: Softversko inženjerstvo
- ESPB: 5
- Uslov: Programski jezici II, .Net tehnologije
- Semestar: 6
- Predavanja: 2
- Auditorne vežbe: 2
- Laboratorijske vežbe: 1
- Broj časova aktivne nastave: 75

Cilj predmeta

- Upoznavanje sa savremenim softverskim inženjerstvom, teorijom i praktičnim postupcima u procesu razvoja softvera u svim fazama njegovog životnog ciklusa.
- Ishod predmeta: Studenti će biti osposobljeni da projektuju jednostavne softverske sisteme na bazi poznavanja rada alata koji će biti prikazani tokom kursa.

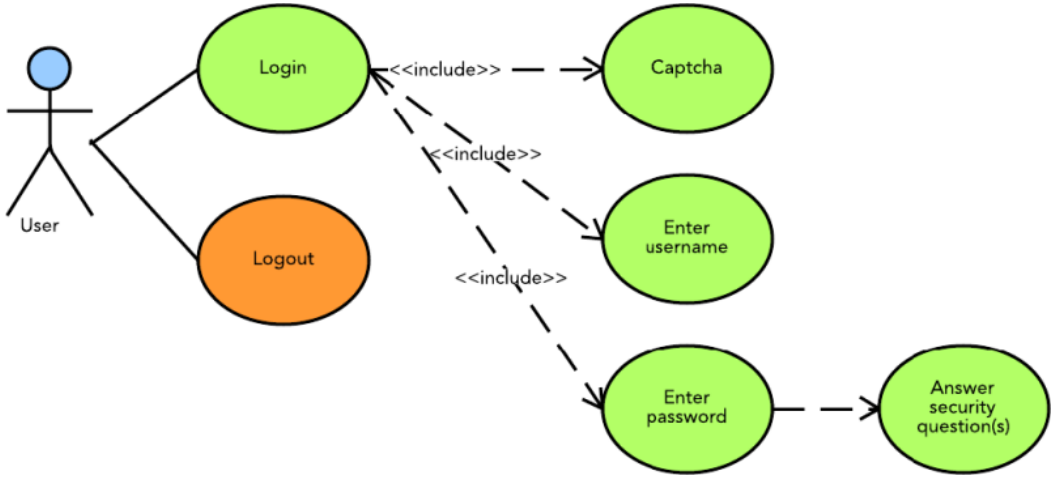


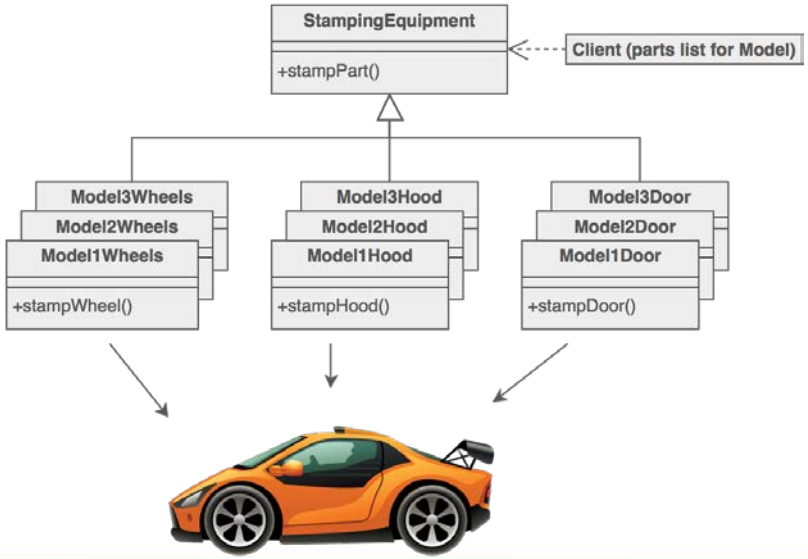
Sadržaj predmeta

1. Uvod u softversko inženjerstvo
2. Softverski procesi i metodologije razvoja softvera,
3. Agilne metode za razvoj softvera
4. RUP (Rational Unified Process) metodologija
5. Specifikacija i inženjering zahteva

Sadržaj predmeta

6. Upravljanje softverskim projektima,
7. Projektovanje softvera
8. Projektovanje softvera primenom projektnih obrazaca





Sadržaj predmeta

9. Razvoj koda,
10. Refaktorisanje koda,

```
public abstract class AbstractCollection implements Collection {
  public void addAll(AbstractCollection c) {
    if (c instanceof Set) {
      Set s = (Set)c;
      for (int i=0; i < s.size(); i++) {
        if (!contains(s.getElementAt(i))) {
          add(s.getElementAt(i));
        }
      }
    } else if (c instanceof List) {
      List l = (List)c;
      for (int i=0; i < l.size(); i++) {
        if (!contains(l.get(i))) {
          add(l.get(i));
        }
      }
    } else if (c instanceof Map) {
      Map m = (Map)c;
      for (int i=0; i<m.size(); i++)
        add(m.keys[i], m.values[i]);
    }
  }
```

Duplicated Code

Duplicated Code

Alternative Classes with Different Interfaces

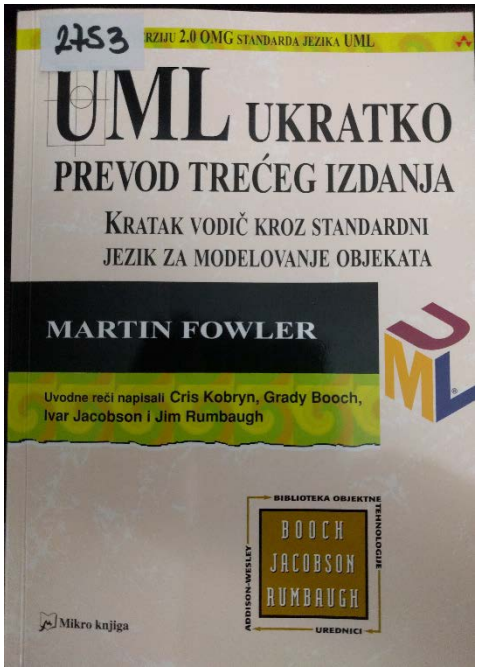
Switch Statement

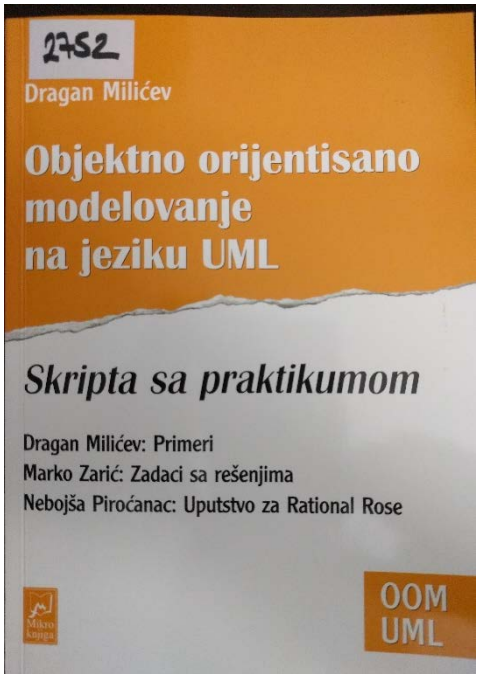
Inappropriate Intimacy

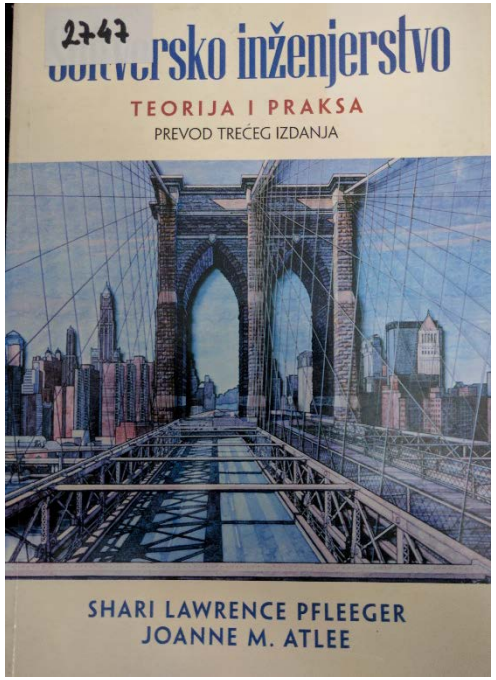
Long Method

11. Verifikacija i validacija softvera,
12. Testiranje softvera
13. Evolucija softvera
14. Održavanje softvera

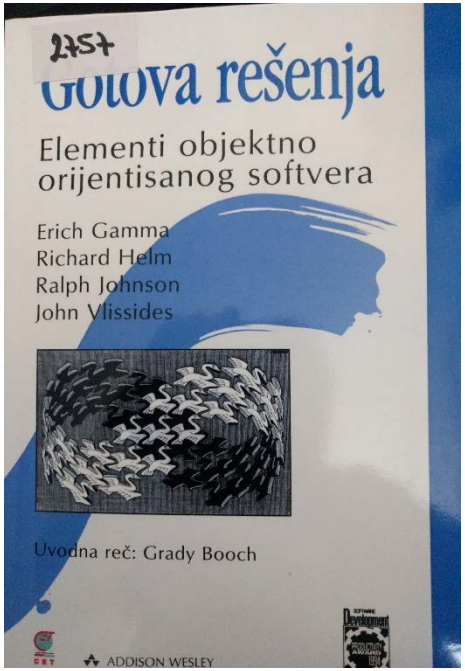
Literatura

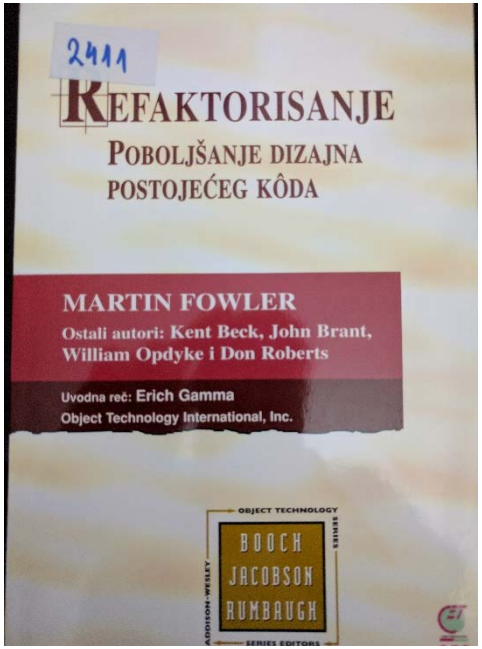

2753
RZIJU 2.0 OMG STANDARDA JEZIKA UML
UML UKRATKO
PREVOD TREĆEG IZDANJA
KRATAK VODIČ KROZ STANDARDNI JEZIK ZA MODELOVANJE OBJEKATA
MARTIN FOWLER
Uvodne reči napisali Cris Kobryn, Grady Booch, Ivar Jacobson i Jim Rumbaugh
BIBLIOTEKA OBJEKTNE TEHNOLOGIJE
BOOCH
JACOBSON
RUMBAUGH
UREDNICI
ADDISON-WESLEY
Mikro knjiga


2752
Dragan Milićev
Objektno orijentisano modelovanje na jeziku UML
Skripta sa praktikumom
Dragan Milićev: Primeri
Marko Zarić: Zadaci sa rešenjima
Nebojša Pirocanac: Uputstvo za Rational Rose
OOM UML
Mikro knjiga


2747
TEORIJA I PRAKSA
PREVOD TREĆEG IZDANJA
SHARI LAWRENCE PFLEEGER
JOANNE M. ATLEE

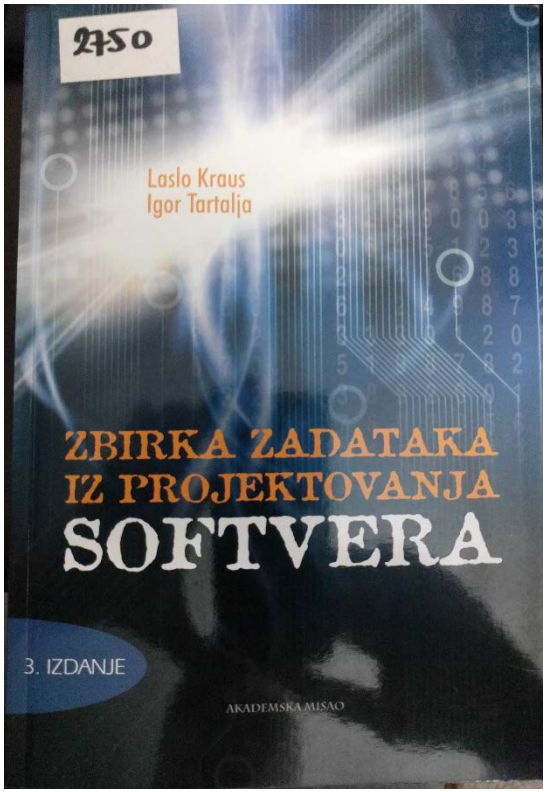
Literatura


2757
Elementi objektno orijentisanog softvera
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Uvodna reč: Grady Booch
ADDISON WESLEY


2411
REFAKTORISANJE
POBOLJŠANJE DIZAJNA POSTOJEĆEG KÔDA
MARTIN FOWLER
Ostali autori: Kent Beck, John Brant, William Opdyke i Don Roberts
Uvodna reč: Erich Gamma
Object Technology International, Inc.
BOOCH
JACOBSON
RUMBAUGH

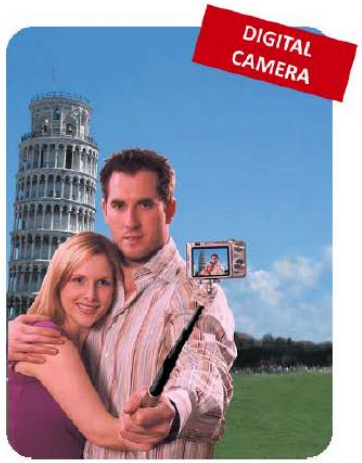

2655
Jovan Popović
TESTIRANJE SOFTVERA U PRAKSI
Pregled teorije testiranja sa primerima iz prakse
Računarski fakultet
Beograd, 2012.
CET

Literatura

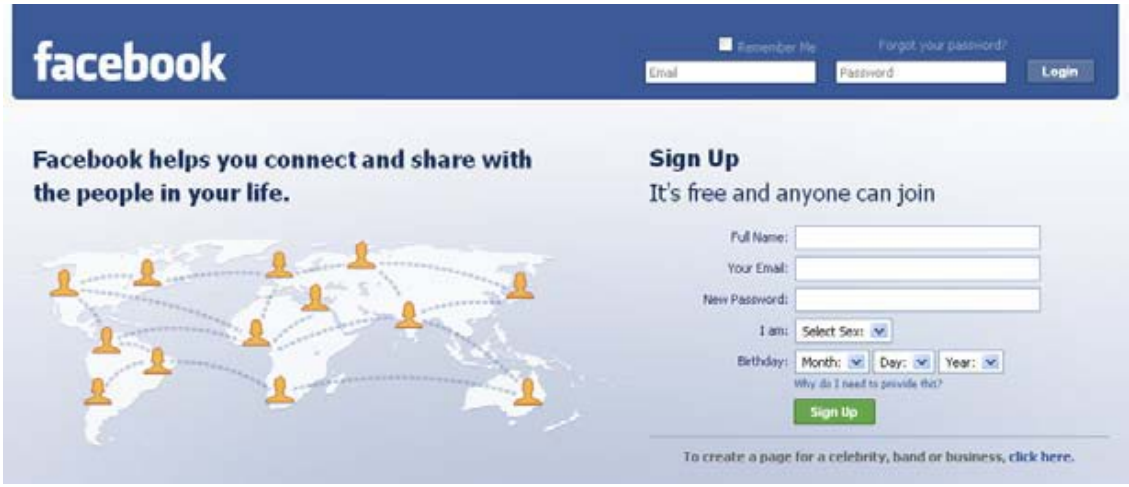

2450
Laslo Kraus
Igor Tartalja
ZBIRKA ZADATAKA
IZ PROJEKTOVANJA
SOFTVERA
3. IZDANJE
AKADEMSKA MISAO

Motivacija

- U današnje vreme softver je prisutan svuda!

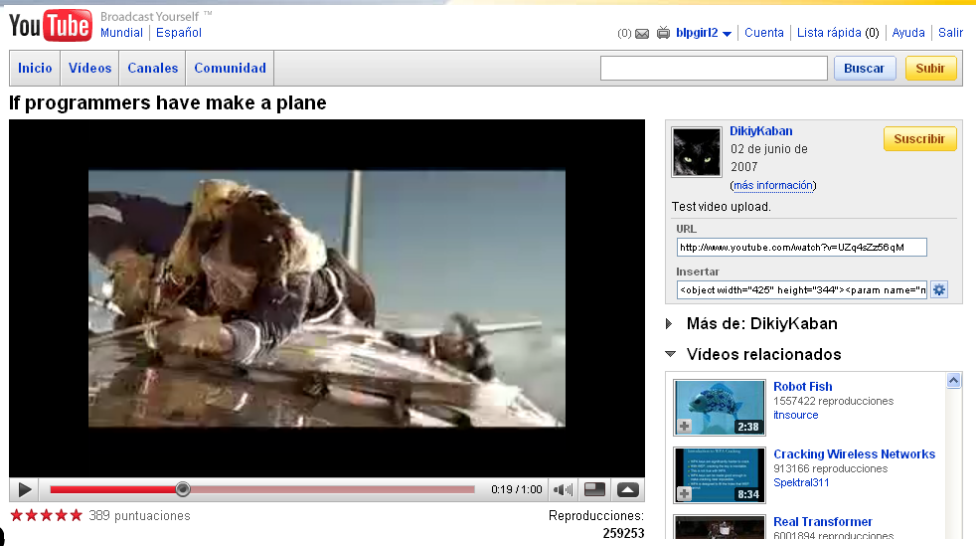


Facebook



- Socijalna mreža nastala 2004. godine
- Mark Zuckerberg (Harvard)
- Više od 500 miliona aktivnih korisnika
- 50% aktivno svaki dan
- Najbrže rastuća demografska grupa su oni od 35 i više godina
- **Više od milion software developera i preduzimača**
- Više od 350,000 aktivnih aplikacija na Facebook platformi
- Više od 250 aplikacija sa više od milion mesečno aktivnih korisnika
- Blokiran u nekoliko zemalja: Pakistan, Sirija, Kina, Vijetnam, Iran i S.Koreja.

YouTube



- Preko 78 miliona videa ukupno
- Preko 6 miliona videa mesečno
- Preko 200,000 videa dnevno
- 100 miliona posetilaca mesečno
- A sve je počelo kada su 3 momka 2005. godine napravili softver zbog problema koji su imali da međusobno razmene video fajlove sa žurke
- Novembra 2005 YouTube je zvanično započeo sa radom, a oktobra 2006. godine Google ga je otkupio za 1,65 milijardi $
- Kritike na račun: distribucije copyright materijala, zaštite privatnosti, kontraverznih sadržaja
- U nekim zemljama zabranjen pristup

Značaj softvera

- Ugnježden je u sisteme svih vrsta: transportne, medicinske, telekomunikacione, vojne, industrijske procese, održavanje, kancelarijske proizvode,... Lista je skoro beskonačna.
- Softver je praktično **neizbežan** u modernom svetu.
- Ulaskom u 21 vek, polako će postati pokretač novih napredaka u svim oblastima od osnovnog obrazovanja do genetskog inženjerstva.

Vrste softvera

- Desktop aplikacije
  - Aplikacije koje se izršavaju na lokalnim računarima. Obuhvataju sve neophodne fukncionalnosti i ne moraju biti povezani na računarsku mrežu ili Internet.
- Interaktivne aplikacije
  - Aplikacije koje se izvršavaju na udaljenom računaru a kojima korisnici pristupaju sa svoji PC računara ili terminala. Ove aplikacije obuhvataju i web aplikacije kao što su komercijalne aplikacije.
- Ugnježdeni kontrolni sistemi (embedded system)
  - Kontrolni sistemi koji kontrolišu i upravljaju hardverskim uređajima. Brojčano, najviše je ovakvih tipova aplikacija.

Vrste softvera

- Sistemi za serijsku obradu

➢ Poslovni sistemi namenjeni za obradu podataka u velikim serijama. Obrađuju veliki količinu ulaznih podataka i kreiraju izlaz.

- Aplikacije za zabavu

➢ Softver namenjen prvenstveno ličnoj upotrebi i zabavi korisnika.

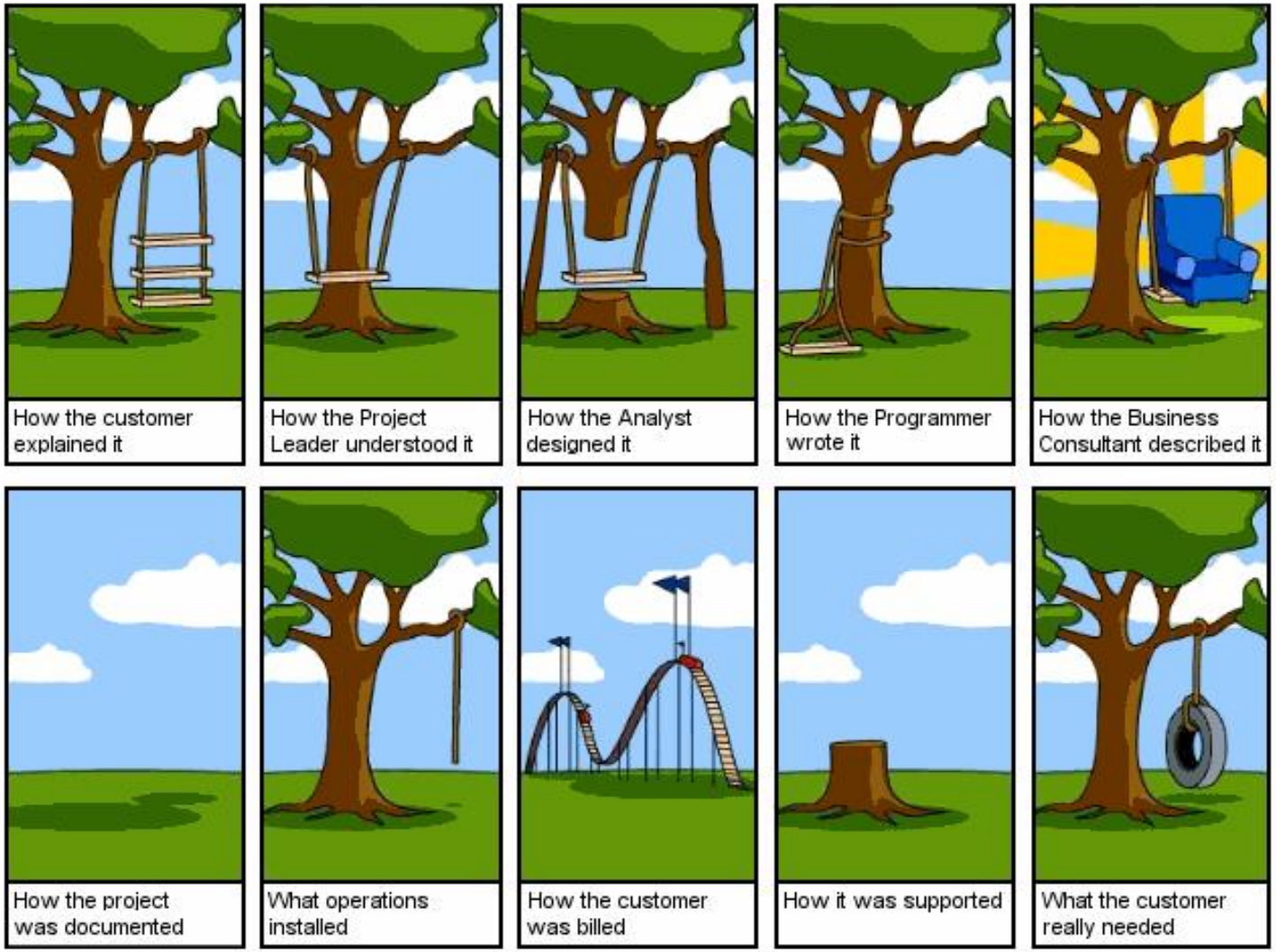
- Sistemi za modelovanje i simulacije

➢ Ovo su sistemi koje su razvili naučnici i inženjeri za modelovanje fizičkih procesa i situacija.

Problemi u razvoju softvera

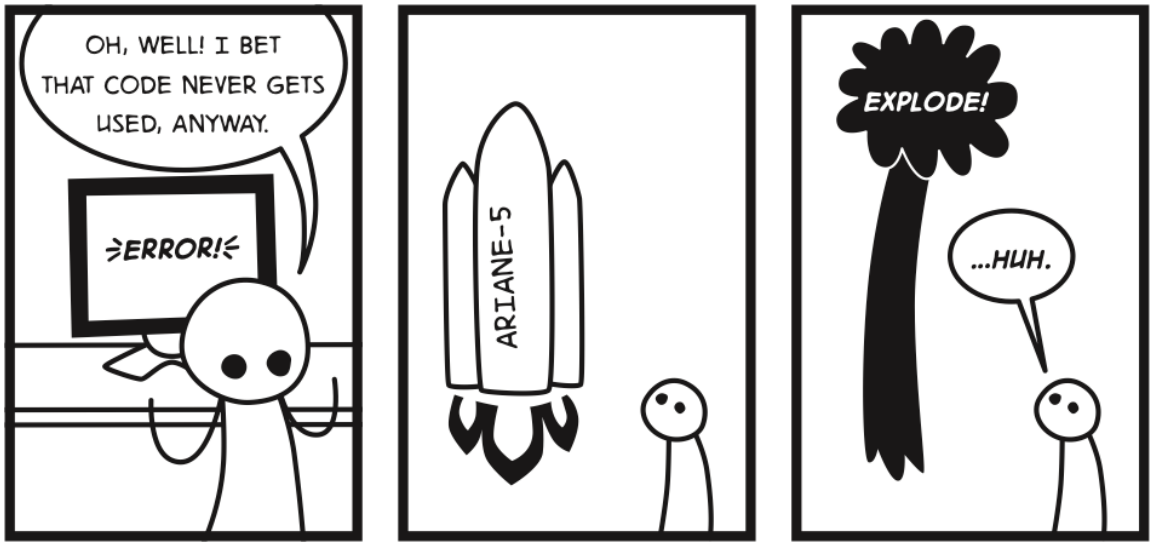
- Finalni softverski proizvod ne ispunjava očekivanja korisnika.
- Teško ga je proširiti i unaprediti: Ukoliko kasnije želite da dodate novu funkcionalnost to je skoro nemoguća misija.
- Loša dokumentacija.
- Loš kvalitet: česte greške, komplikovano korišćenje,...
- Više vremena i veći troškovi nego što je očekivano.

Problemi u razvoju softvera



Primer

- Greška na lanseru satelita Ariane 5 je uzrokovana greškom u softverkoj funkciji za konvertovanje iz 64-bitne float vrednosti u 16-bitnu celobrojnu vrednost.



Primer

- Softver za spejs šatl
- Cena: 10 milijardi $, milioni više od planiranog
- 100.000 redova koda softver u šatlu
- 3.000.000 redova koda softver za podršku šatlu
- Vreme: 3 godine zakašnjenja
- Kvalitet: Prvo lansiranje Kolumbije je otkazano zbog problema sa sinhronizacijom računara na šatlu
- Greška je nastala 2 godine ranije kada je programer promenio jedinicu za kašnjenje rutine prekida sa 50 na 80 milisekundi
- Potencijalne greške i danas postoje
- Astronautima se daju priručnici sa poznatim softverskim greškama i uputstvima kako ih rešiti.

Primer

- “Milenijumska buba” - Y2K
- Umesto 4 pozicije za godinu u većini sistema postoje samo dve
- Šta nakon 99 – 00?
- Ukupni troškovi pripreme procenjeni na 300 milijardi dolara

Šta je softver?

- Softver je računarski program, pridružena dokumentacija i konfiguracioni podaci neophodni da bi sistem radio korektno.
- Softverski sistem se obično sastoji od:
  - Određenog broja programa
  - Konfiguracionih fajlova
  - Sistemske dokumentacije koja opisuje strukturu sistema
  - Korisničke dokumentacije
  - Web sajtova za podršku korisnicima
- SW inženjerstvo se bavi razvojem softverskih **proizvoda** (softver koji se može prodati kupcu).

Tipovi SW proizvoda

- Postoje 2 osnovna tipa:
  - Generički (generic) – samostalni sistemi namenjeni za prodaju na slobodnom tržištu
  - Ugovorni (custom) – razvijeni za jednog korisnika prema njegovim zahtevima
- **Osnovna razlika je ko definiše šta će se raditi** (specifikacija)

Softverska kriza

- Ispoljava se kroz:
  - Loš kvalitet softverskih projekata,
  - Nisku produktivnost,
  - Visoku cenu razvoja softvera,
  - Dugo vreme realizacije softvera.
- Razlozi: nestrukturno programiranje (hacking), nedostatak **kompletne** specifikacije zahteva i **odgovarajuće** metodologije razvoja.
- Kao rešenje za SW krizu predlaže se sistematski prilaz razvoju i evoluciji softvera čime se bavi softversko inženjerstvo.

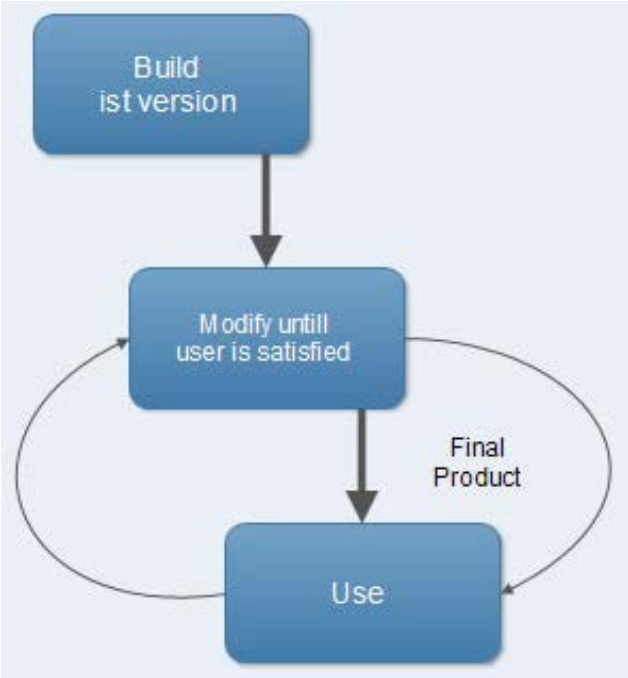
Šta je softversko inženjerstvo

- Termin se prvi put upotrebio 1968. na NATO Software Engineering konferenciji.
- SW inženjerstvo je inženjerska disciplina koja obuhvata **sve aspekte proizvodnje softvera** od ranih faza specifikacije sistema do održavanja sistema nakon stavljanja u upotrebu.
- **Svi aspekti produkcije softvera**

➢ Ne samo tehnički proces razvoja. Takođe obuhvata upravljanje projektom, razvoj alata, metoda, itd., koji podržavaju proces produkcije softvera.

Cilj softverskog inženjerstva

- Cilj softverskog inženjerstva je da razvoj softvera približi nauci i inženjerstvu i udalji ga od ad-hoc pristupa razvoju čiji su ishodi nepredvidljivi a koji su intenzivno korišćeni u prošlosti i dalje se koriste.



Osnovni principi softverskog inženjerstva

- **Sisteme treba razvijati korišćenjem upravljivog i jasnog razvojnog procesa.** Naravno koriste se različiti procesi za različite tipove softvera.
- **Pouzdanost i performanse su značajne za sve vrste sistema.**
- Veoma je bitno razumevanje i upravljanje **specifikacijom i zahtevima softvera** (šta softver treba da radi).
- Tamo gde je to moguće **bolje je koristiti postojeći softver nego razvijati novi**.

Značaj softverskog inženjerstva

- Sve više, pojedinci i društvo zavise od složenih softverskih sistema. Neophodno je da **brzo** i **ekonomično** proizvodimo pouzdane sisteme.
- Mnogo je **jeftinije** (gledano dugoročno) korišćenje metoda i tehnika softverskog inženjerstva za izradu softverskih sistema nego jednostavna implementacija programa kao što se to radi na malim programima za ličnu upotrebu.
- Za većinu tipova sistema, većina **troškova** su troškovi promene softvera nakon što je stavljen u upotrebu.

Troškovi SW inženjerstva

- **Grubo 60% troškova su razvojni troškovi, 40% troškovi testiranja.**
- **Troškovi evolucije često prelaze troškove razvoja.**
- Troškovi variraju zavisnosti od tipa sistema koji se razvija i zahtevanih atributa sistema.

➢ Npr. performanse i pouzdanost sistema.

Koji su ključni izazovi softverskog inženjerstva

- **Postojeći sistemi** (Legacy Systems)

➢ Stari, postojeći sistemi koje treba održavati i ažuirati.

- **Sve veća raznovrsnost** (Heterogenity)

➢ Sistemi su distribuirani i uključuju raznovrstan hardver i softver.

- **Isporuka** (Delivery)

➢ Zahtevi da se skrati vreme izrade softvera.

Istorija razvoja softverskog inženjerstva

Godina	Kratak opis razvoja
1940-te	▪ ručno pisanje mašinskog koda (0 i 1)
1950-te	▪ makro asembleri, interpreteri i prva generacija kompajlera
1960-te	▪ funkcionalno programiranje (Basic, Fortran, Cobol …) ▪ *mainframe* računari i softveri za velike korporacije
1970-te	▪ kolaborativni alati ▪ rast manjih poslovnih softvera
1980-te	▪ personalni računari (PC) i radne stanice ▪ rast potrošačkih softvera (MRP I, MRP II …)
1990-te	▪ objektno-orjentisano programiranje (C++, C#, Java …) ▪ agilni procesi ▪ integrisana poslovna rešenja (ERP, CRM …)
2000-te do danas	▪ veb servisi i servisno-orijentisano programiranje ▪ inteligentna poslovna rešenja (BI) ▪ servisi u *Cloud Computing* okruženju

Softversko inženjerstvo za Web

- Web je danas platforma za izvršenje aplikacija i ogranizacije češće koriste Web aplikacije nego lokalne sisteme.
- Web servisi omogućavaju da se aplikativnim funkcijama pristupa preko Weba.
- Cloud computing je pristup obezbeđivanju računarskih servisa gde se pokreću udaljene aplikacije.
- Korisnici ne kupuju softver nego plaćaju njegovu upotrebu.

Softversko inženjerstvo za Web

- **Ponovna upotreba softvera** je dominantan pristup za konstruisanje Web aplikacija.
  - Prilikom izgradnje ovakvih sistema treba razmišljati o upotrebi postojećih softverskih komponenti i sistema.
- Web sistemi se mogu razvijati i isporučivati **inkrementalno**.
  - Za ovavke sisteme nije praktično definisanje svih zahteva unapred.
- Korisnički interfejs je ograničen mogućnostima Web browsera.
  - Tehnologije kao što je AJAX omogućavaju bogat korisnički interfejskoji se kreira unutar Web browsera.

Koje su razlike između SW inženjerstva i informatike

- Nauka o računarstvu ili informatika se bavi **teorijom** i osnovama računarstva.
- SW inženjerstvo se bavi **praktičnom** stranom razvoja i isporuke korisnog softvera.
- Teorija nauke o računarstvu je trenutno dobro razvijena i obezbeđuje solidnu osnovu za softversko inženjerstvo.

Koje su razlike između softverskog i sistemskog inženjerstva

- Sistemsko inženjerstvo se bavi svim aspektima razvoja sistema zasnovanih na računarima, uključujući hardverski, softverski i procesni inženjering.
- **Softversko inženjerstvo je deo tog procesa**.
- Sistemsko inženjerstvo je starija disciplina od SW inženjerstva.

Šta je softverski proces

- Struktuiran skup aktivnosti neophodan za razvoj softverskog sistema.
- Aktivnosti zajedničke za sve softverske procese:

➢ **Specifikacija softvera** (Software Specification) - korisnici i inženjeri definišu softver koji treba izraditi i ograničenja u toku tog procesa.

➢ **Razvoj softvera** (Software Development) - softver se dizajnira i implementira.

➢ **Validacija softvera** (Software Validation) - softver se proverava da bi se utvrdilo da li ispunjava korisničke zahteve.

➢ **Evolucija softvera** (Software Evolution) - softver se modifikuje u skladu sapromenama zahteva korisnika i tržišta.

Šta je **model** softverskog procesa

- Uprošćena reprezentacija softverskog procesa koja predstavlja **jedan pogled** na ovaj proces.
- Opšti modeli (paradigme):

➢ Model vodopada

➢ Iterativni razvoj

➢ Razvoj zasnovan na korišćenju gotovih komponenti

Šta su **metode** SW inženjerstva

- Struktuirani prilaz razvoju softvera.
- Metode obuhvataju:

➢ Opise modela (opisi grafičkih modela koji nastaju u toku razvoja)

➢ Pravila (ograničenja primenjena na model sistema)

➢ Preporuke (saveti za dobro projektovanje)

➢ Vodič kroz proces (kako teku aktivnosti)

Šta su CASE alati

- CASE alati su softverski sistemi namenjeni pružanju automatske podrške aktivnostima softverskog procesa.
- CASE višeg nivoa (**Upper-CASE**)
  - Alati koji podržavaju aktivnosti procesa (inženjering zahteva i projektovanje)
- CASE nižeg nivoa (**Lower-CASE**)
  - Alati koji podržavaju kasnije aktivnosti (programiranje, debagiranje i testiranje)

Koji su atributi kvalitetnog softvera

- **Pogodnost za održavanje** (Maintainablitity)

➢ Treba da je u stanju da može da se lako menja.

- **Stabilnost** (Dependability)

➢ Mora da uliva poverenje što podrazumeva da je:

✓ Pouzdan (Reliability),

✓ Bezbedan (Security) i

✓ Siguran (Safety).

- **Efiksanost**(Efficiency)

➢ Mora da ekonomično koristi resurse sistema

- **Upotrebljivost** (Usability)

➢ Mora da bude pogodan za korišćenje

Pogledi na kvalitet

- Transcedentalni pogled – kvalitet je nešto što možemo da prepoznamo, ali ne i definišemo
- Korisnički pogled – kvalitet je usklađen sa namenom
- Pogled sa aspekta proizvodnje – kvalitet je usklađen sa specifikacijom
- Pogled sa aspekta proizvoda – kvalitet je vezan za karakteristike samog proizvoda
- Pogled na bazi vrednosti – kvalitet zavisi od toga koliko je kupac spreman da za njega plati

Šta je to kvalitetan softver

- **Kontekst pomaže da se utvrdi odgovor.**

➢ Nedostaci koji mogu da se tolerišu u softveru za obradu teksta, ne mogu da se prihvate u sistemima gde je bezbednost faktor od izuzetnog značaja.

Šta je to kvalitetan softver

- **Kvalitet moramo posmatrati sa najmanje tri aspekta.**

➢ Kvalitet proizvoda.

➢ Kvalitet postupka izrade proizvoda.

➢ Kvalitet proizvoda u kontekstu okruženja u kome će se on koristiti.

Najčešće postavljana pitanja u vezi softverskog inženjerstva

Pitanje	Odgovor
Koji su to ključni izazovi sa kojima se suočava softversko inženjerstvo?	Borba sa sve većim razlikama, zahtevima za smanjenjem vremena isporuke i razvojem pouzdanog softvera.
Koji su troškovi softverskog inženjerstva?	Oko 60% su troškovi razvoja softvera, 40% su troškovi testiranja. Za softver prilagođen korisniku, troškovi evaluacije često prevazilaze troškove razvoja.
Koje su najbolje tehnike i metode softverskog inženjerstva?	Upravljanje svim softverskim projektima i njihov razvoj moraju biti profesionalni. Različite tehnike su pogodne za različite tipove sistema. Na primer, igre se uvek razvijaju korišćenjem velikog broja prototipova dok se sistemi za kontrolu zahtevaju razvoj kompletne i detaljne specifikacije. S toga, nema najbolje metode.
Koje novine je uneo Web u softversko inženjerstvo?	Web je doveo do pojave softverskih servisa i mogućnosti razvoja distribuiranih servisno orjentisanih sistema. Razvoj Web sistema je doveo do značajnog napredka u razvoju programskih jezika i višestruke upotrebe softvera.

Najčešće postavljana pitanja u vezi softverskog inženjerstva

Karakteristika proizvoda	Opis
Održivost	Softver mora biti napravljen na takav način da može lako da evaluaira u skladu sa promenama zahteva korisnika. To je kritična karakteristika s obzirom na promenljivost poslovnog okruženja.
Poudanost i bezbednost	Softverska pouzdanost obuhvata skup karakteristika kao što su pouzdano izvršenje funkcija, bezbednost i sigurnost. Pouzdan softver ne sme da uzrokuje dovede do fizičkih ili ekonomskih posledica u slučaju otkaza. Zlonamerni korisnici ne smeju biti u mogućnosti da pristupe sistemu ili ga ugroze.
Efikasnost	Softver ne sme nepotrebno da troši sistemske resurse kao što su memorija i procesorsko vreme. Efikasnost stoga obuhvata vreme odgovora, vreme obrade, iskorišćenje memorije, itd.
Prihvatljivost	Softver mora biti prihvatljiv za one korisnike za koje je napravljen. Ovo znači da mora biti razumljiv, upotrebljiv i kompatibilan sa drugim sistemima koji se koriste.