Razvoj softvera I

Uvod

Sadržaj predmeta

	Nastavna jedinica
1	Softver i Softverski inženjering
2	Modeliranje procesa i životnog ciklusa softvera
3	Planiranje i upravljanje projektom
4	Evidentiranje zahtjeva
5	Dizajniranje softvera
6	Softverski predlošci (Design Patterns)
7	Pisanje softvera
8	Testiranje softvera
9	Isporuka i održavanje softvera
10	Ocjena kvaliteta softverskog proizvoda
11	Agilne metode razvoja softvera (SCRUM, Ekstremno programiranje)

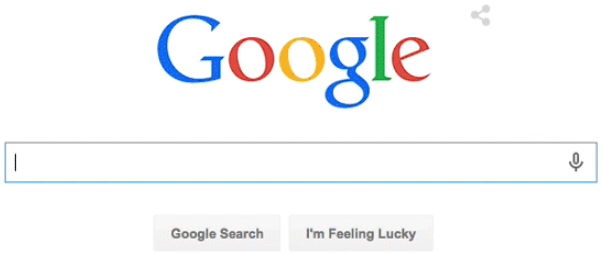
Ispitne obaveze

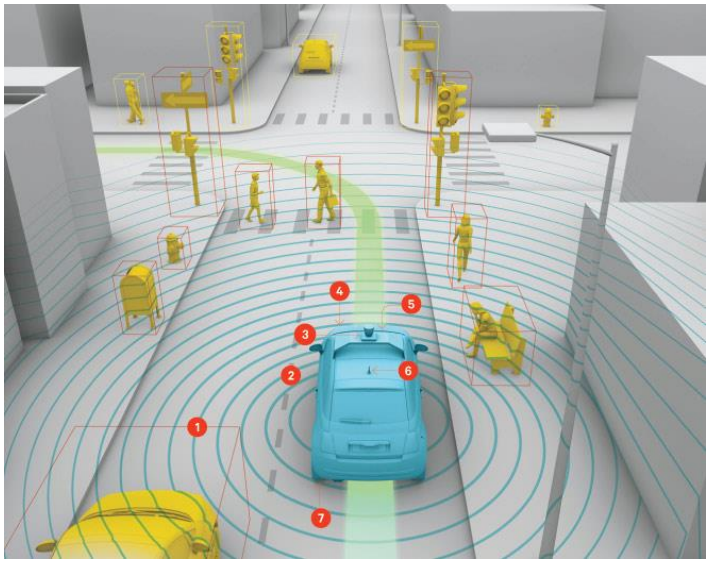
- Ispitne obaveze podrazumijevaju:
  - polaganje pismenog dijela ispita (50 bodova)
    - **teorijski i praktični dio**
  - izradu i odbranu projektnog zadatka (50 bodova)
    - **C#, ASP.NET API, MVC, Entity Framework, Angular**
- Detalji vezani za ispitne obaveze su definisani silabusom predmeta
- Posebnu pažnju obratiti na rokove za završetak pojedinih zadataka i samog projekta.

Razvoj softvera

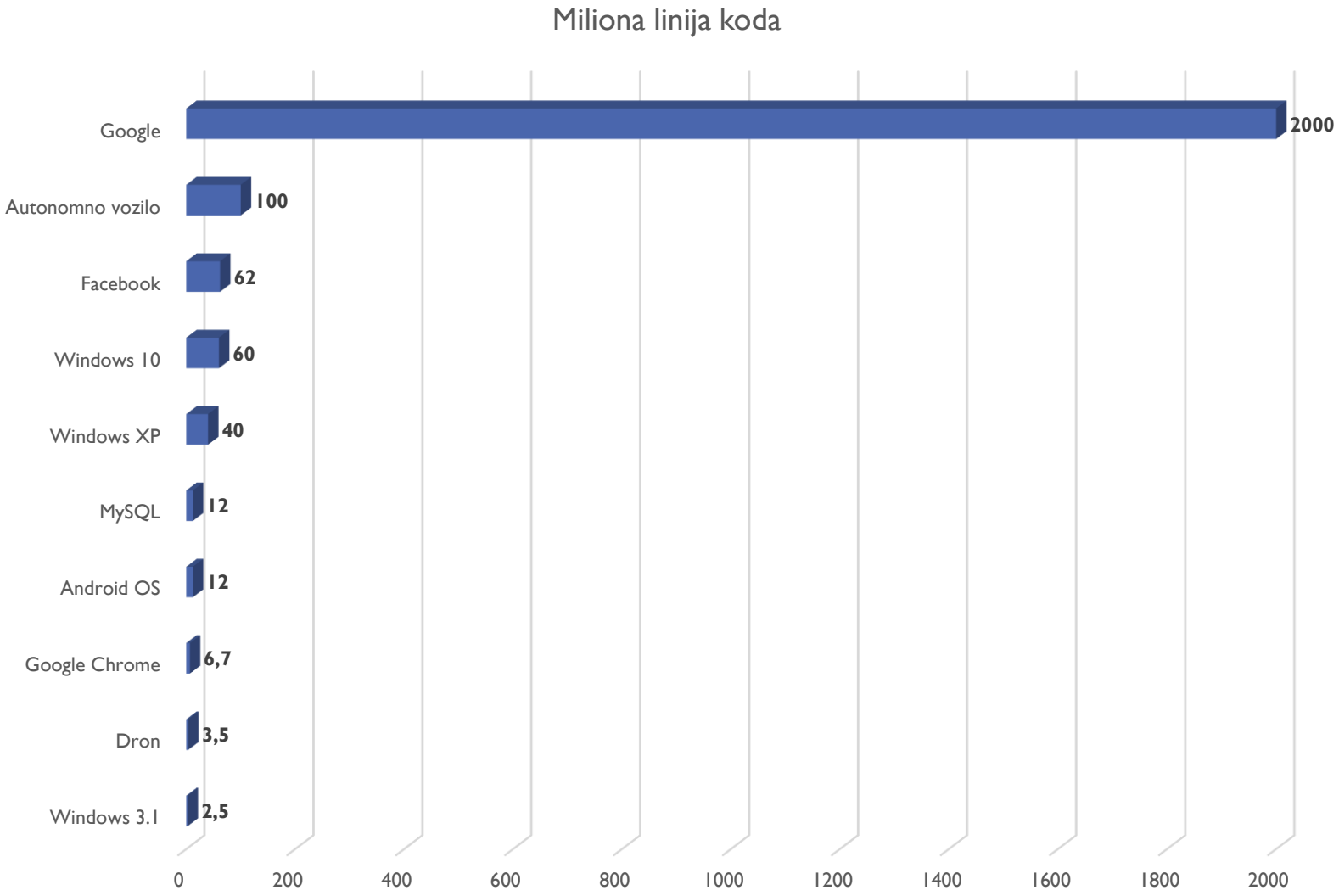
- Šta podrazumijeva pojam softver, a šta pojam inženjering?
- Šta je Softverski inženjering?
- Koliko je uspješan u praksi?
- Šta čini dobar softver?
- Ko se bavi Softverskim inženjeringom?

Značaj područja



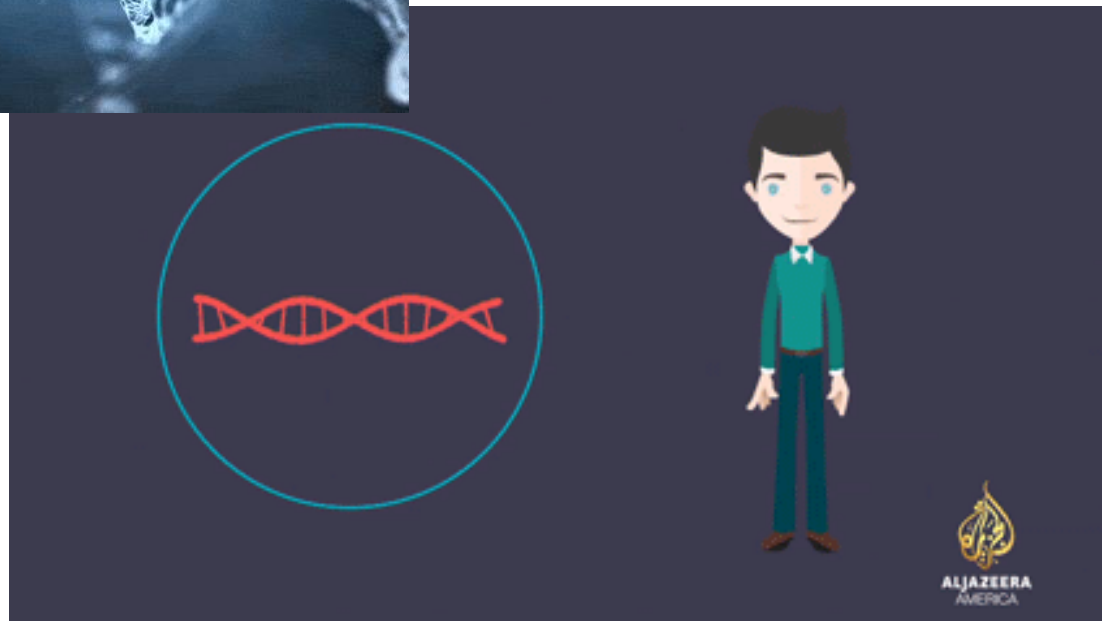


Značaj področja



Ljudski genom

3,300 biliona LoC



Pregled područja

- Koliko linija programskog koda nastaje u okviru određene kompanije?
- Da li je broj linija koda jedino mjerilo kvalitete i funkcionalnosti softvera?
- Kako upravljati cijelim procesom razvoja softvera?

Program, aplikacija i softver

- Program – set instrukcija
- Aplikacija – jedan ili više programa
- Softver – sve komponente koje sačinjavaju ili opisuju softver
    - Programske instrukcije
    - Strukture podataka
    - Dokumentaciju
    - …

- Broj učesnika u razvoju?
- Softverski procesi

Inženjering

- Inženjering podrazumijeva primjenu naučnih principa i metoda u razvoju upotrebljivih struktura i komponenti, a zastupljen je u gotovo svim područjima ljudskog djelovanja: mašinstvo, građevina, i dr.
    - Predvidivost
    - Ponovljivost
    - Ponovna iskoristivost

Softverski inženjering

```
if (inzenjerstvo == primijenjena_nauka)
        softversko_inzenjerstvo = primijenjena_racunarska_nauka;
```

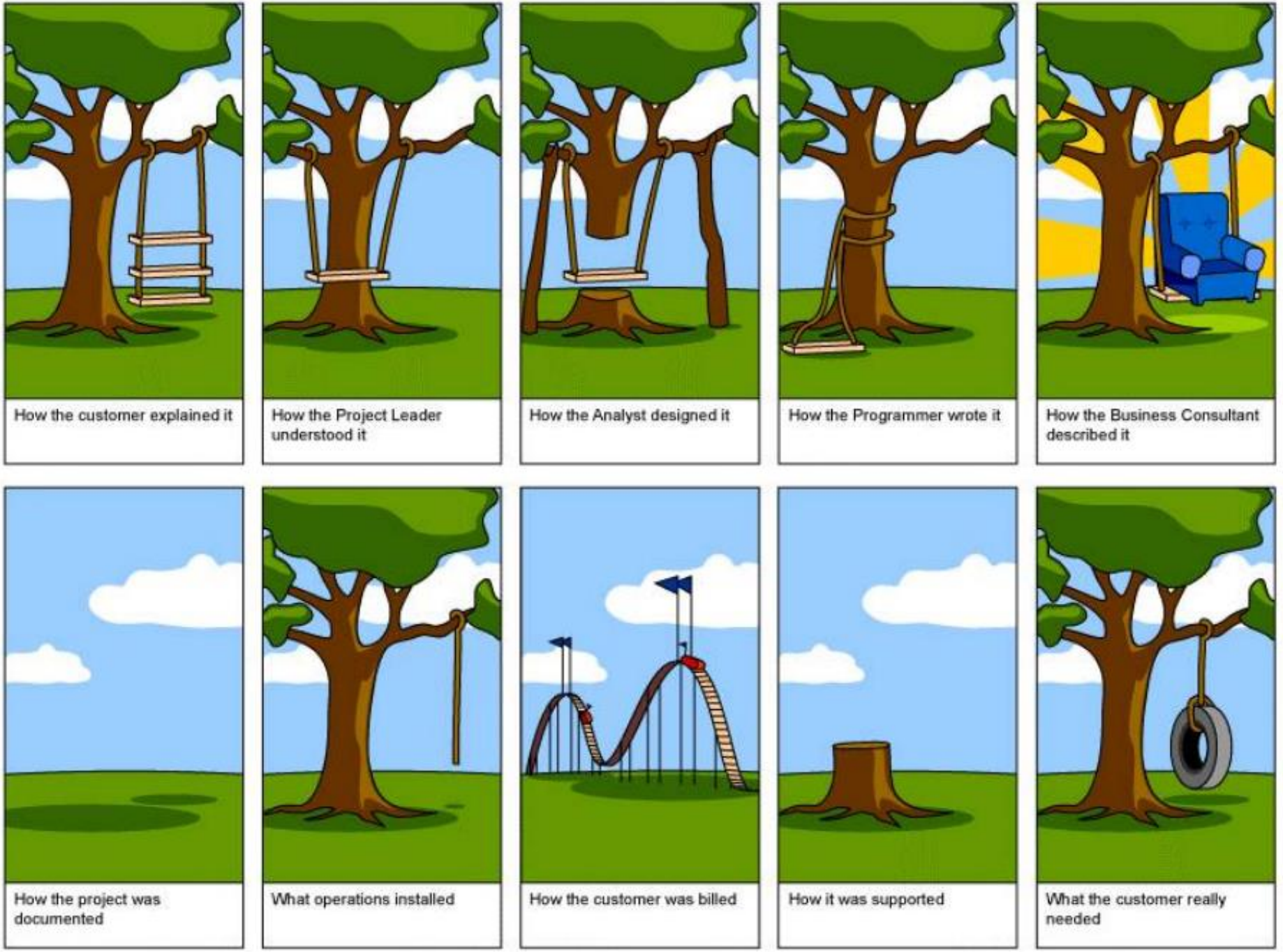
- Računarska nauka – osigurava teorijsku osnovu za dizajn i korištenje savremenih računarskih komponenti
- Zbog čega se javila potreba za softverskim inženjerstvom?
    - Softverska kriza sredinom 60-ih godina
        - Kašnjenja u isporuci softvera
        - Isporučeni softver je imao mnogo grešaka u radu
        - Isporučeni softver nije ispunjavao postavljene zahtjeve
        - Softver je bilo jako teško nadograđivati i održavati
    - Kompleksniji problemi zahtijevaju kompleksnija softverska rješenja

Softverski inženjering::Definicija

- Definicije Softverskog inženjeringa:
  - Softverski inženjering se bavi ekonomskim aspektima razvoja softvera visokog kvaliteta (Pagel, 94)
  - Softverski inženjering je inženjerska disciplina koja se bavi praktičnim problemima razvoja velikih softverskih sistema (Sommerville, 92)
  - Softverski inženjering je primjena sistematskih, disciplinovanih i mjerljivih pristupa razvoju, rukovanju i održavanju softvera (IEEE SI riječnik, 90)
  - Softverski inženjering je proces koji olakšava specificiranje, projektovanje, implementaciju i testiranje softvera za skup iznijetih zahtjeva, na najbrži i najprofitabilniji mogući način (Kehoe, 95).

Softverski inženjering

- Primjeri (ne)uspješnosti u razvoju ili nadogradnji softvera
  - Poreska uprava SAD-a – projekat koštao 94 miliona dolara više u odnosu na planirane troškove
  - Rengen Therac 25 – uzrokova smrt više osoba
  - Siemens softver za Fond zdravstvenog osiguranja Njemačke – zbog kašnjenja projekta nastali troškovi u visini od 1 milijarde maraka
  - Deutsche Telecom – zbog greške u obračunu cijene impulsa nastalo stotine miliona maraka gubitka
  - Amazon.com – zbog problema prilikom nadogranje sistem nije bio dostupan 90 minuta što je dovelo do smanjenja prihoda za 2.8 miliona dolara
- Tržište zahtijeva brz razvoj softvera! Kako osigurati kvalitet?
  - Efikasno testiranje
  - Greške otkrivene tokom analize su 10 puta jeftinije od onih otkrivenih nakon isporuke


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

Softverski inženjering

- Kako odgovoriti na sve kompleksnije zahtjeve?
  - SDLC (Software Development Life Cycle) – fazni razvoj

- Komunikacija
- Planiranje
- Modeliranje
  - Analiza zahtjeva
  - Dizajn
- Razvoj
  - Progrmiranje
  - Testiranje
- Isporuka

- Upravljanje projektom
- Tehnička evaluacija
- Osiguranje kvaliteta
- Evaluacija napretka
- Upravljanje rizicima

Softverski inženjering::Uvod

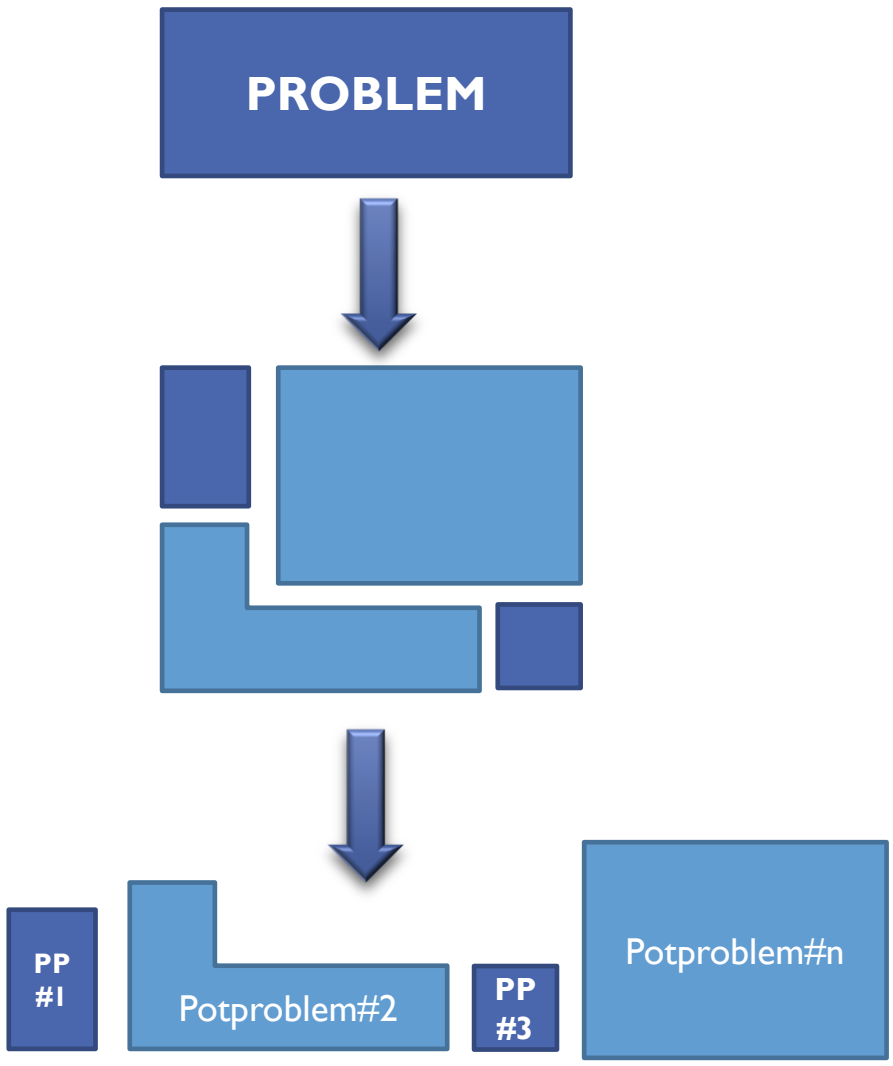
- Programiranje je relativno mala komponenta softverskog inženjeringa!

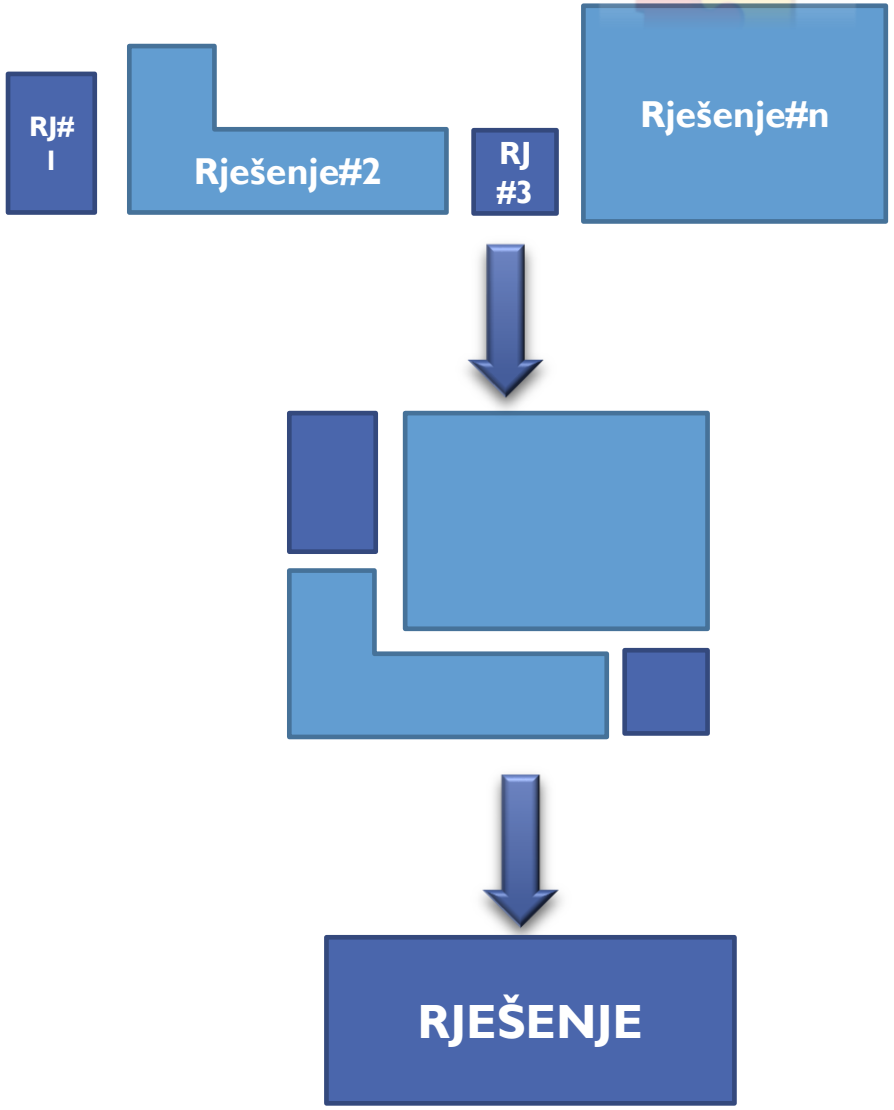
Softverski inženjering

Upravljanje kompleksnošću i neočekivanim pojavama

- Softverski inženjering, kao nauka, omogućava efikasnije rješavanje postavljenih problema
  - Razumijevanje problema (potrebe naručioca projekta - softvera)
  - Definisanje plana za rješavanje problema
  - Izvršenje plana
  - Evaluacija rezultata
- Kompleksnost problema zahtijeva ***analizu*** (razlaganje problema na manje dijelove) ili dekompoziciju
  - **Divide and conquer**
- Nakon analize koristimo proces ***sinteze*** u cilju kreiranja kompletnog rešenja.

Softverski inženjering::Analiza i sinteza





Softverski inženjering

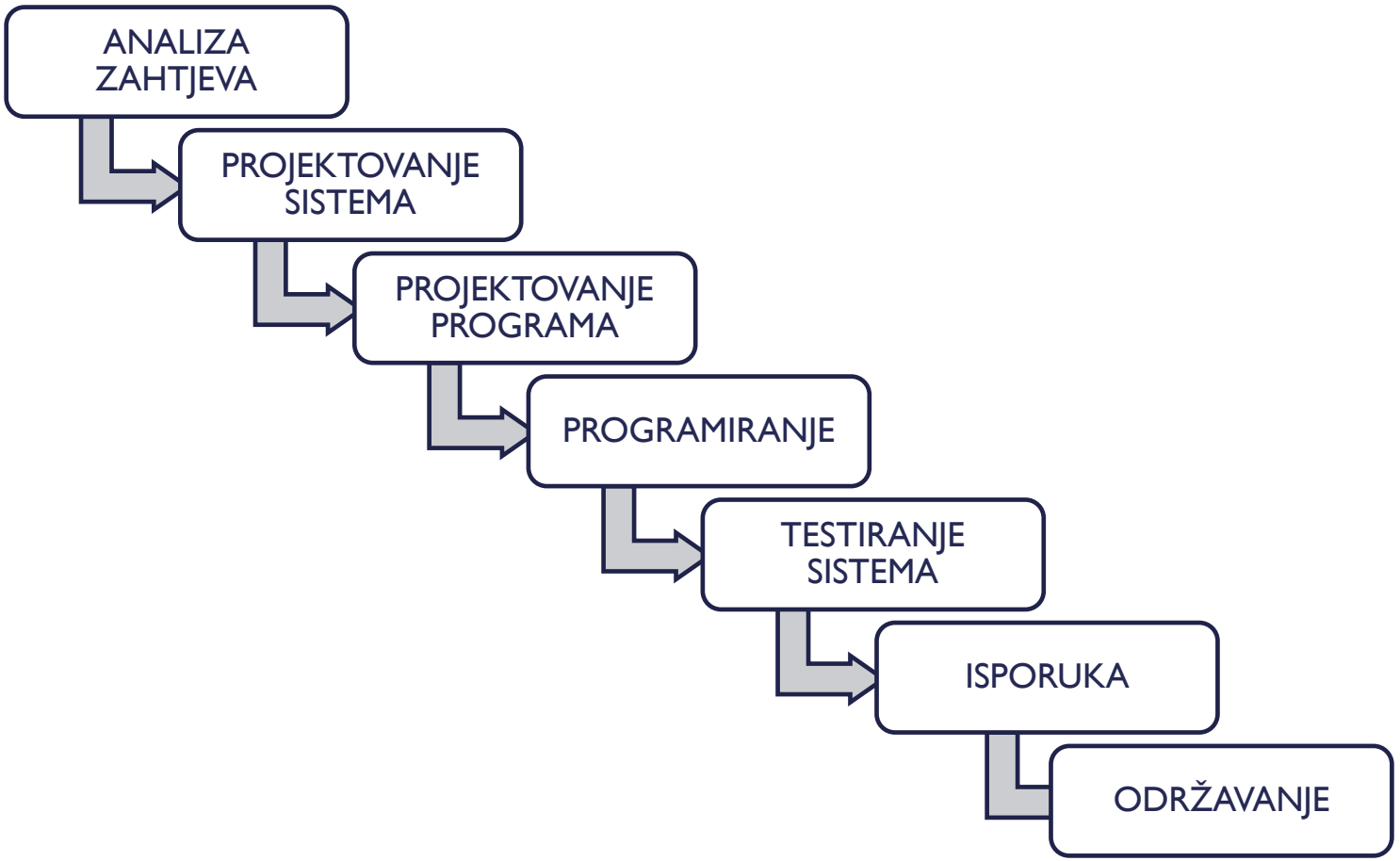
- Rješavanje problema zahtijeva korištenje različitih metoda, alata, procedura i paradigmi
  - **Metoda** (tehnika) je formalni postupak za postizanje nekog rezultata
  - **Alat** je instrument (automatizovan sistem) koji omogućava da se određena aktivnost obavi na kvalitetniji način
  - **Procedura** je kombinacija alata i metoda koji u međusobnom skladu proizvode dati proizvod
  - **Paradigma** predstavlja pristup ili filozofiju razvoja softvera

- Korištenjem alata, tehnika, procedura i paradigmi softverski inženjeri su u stanju poboljšati kvalitet softverskog proizvoda.

Procesni modeli

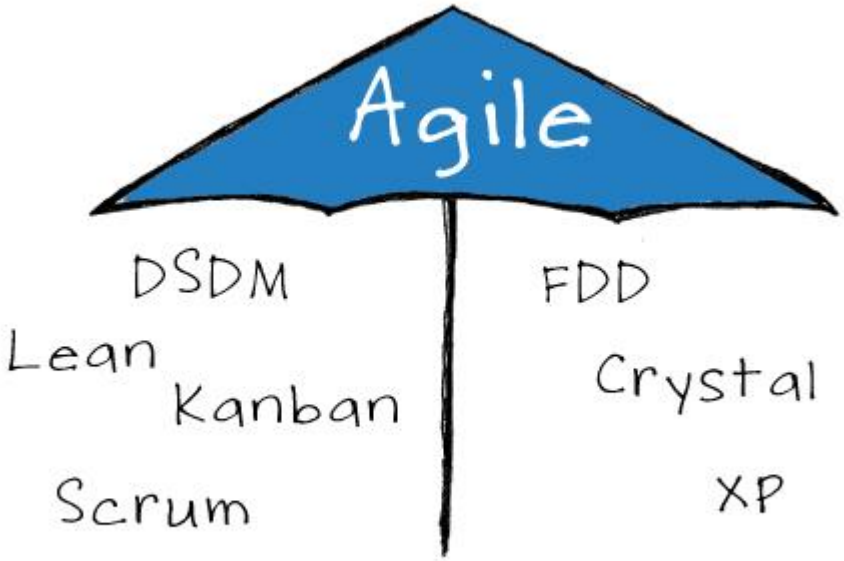
- Preskriptivnim
  - Model vodopada
  - Inkrementalni
  - Spiralni
  - …
- Agilni
  - Extreme Programming (XP)
  - Scrum
  - Kanban
  - Crystal
  - …
- Koji je model najbolji?

Model vodopada



Agile metode

- Osnovni koncepti agilnih metoda su:
  - Interakcija i kolaboracija sa korisnikom
  - Softver prije detaljne dokumentacije
  - Prilagođavanje umjesto krute administracije



Procesni modeli

The sooner you start to code, the longer the program will take...

Softverski inženjering

- Pisanje softvera je ujedno i umjetnost i nauka
- Kako osigurati kvalitet?
  - Efikasno testiranje
  - Greške otkrivene tokom analize su 10 puta jeftinije od onih otkrivenih nakon isporuke
- Većina problema nema jedinstveno rešenje
  - umjetnost, inovativnosti i vještina
- Većina razvijenog softvera „radi”, ali se postavlja pitanje:
  - ispunjenja potreba naručioca
  - stabilanosti
  - razumljivosti
  - održavanja i nadogradnje

Softverski inženjering

- ***Suština softverskog inženjeringa je projektovanje i razvoj visoko kvalitetnog softvera.***

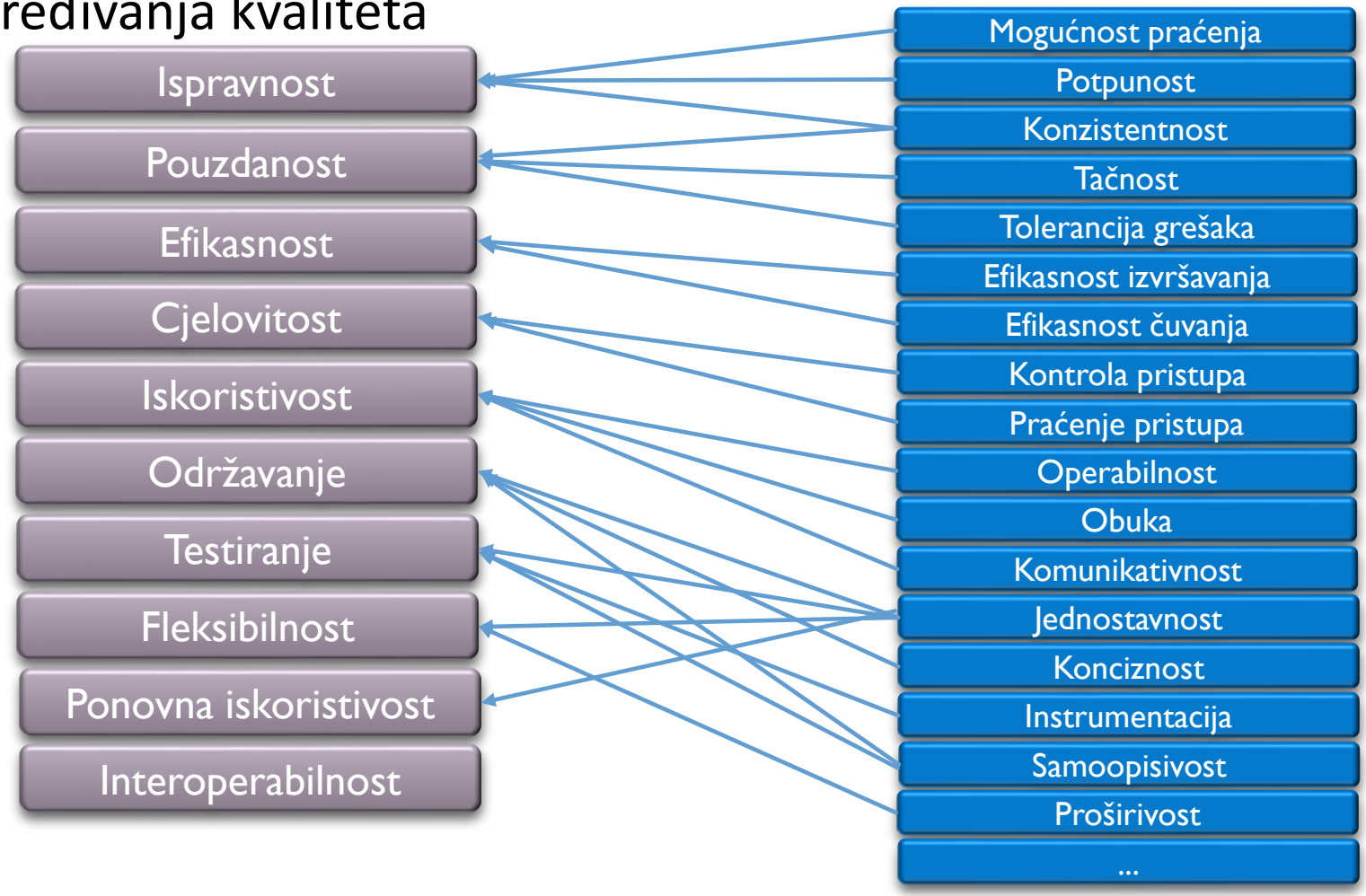
Ako ovaj momak može nešto da pokvari, pokvarit će!
Edward Murphy

Softverski inženjering

- Procjena dobrog ili lošeg softvera značajno zavisi od kontaksta poslovanja
- Kvalitet softvera se posmatra kroz:
  - Kvalitet proizvoda
    - da li softver ispunjava zahtjeve?
    - da li softver stabilo funkcioniše?
    - kakva je arhitektura, organizacija koda i mogućnost proširenja?
  - Kvalitet postupka izrade (procesa)
    - podjednako značajan kao i kvalitet proizvoda
  - Kvalitet proizvoda u kontektsu poslovnog okruženja za koji je namijenjen
    - povrat investicije

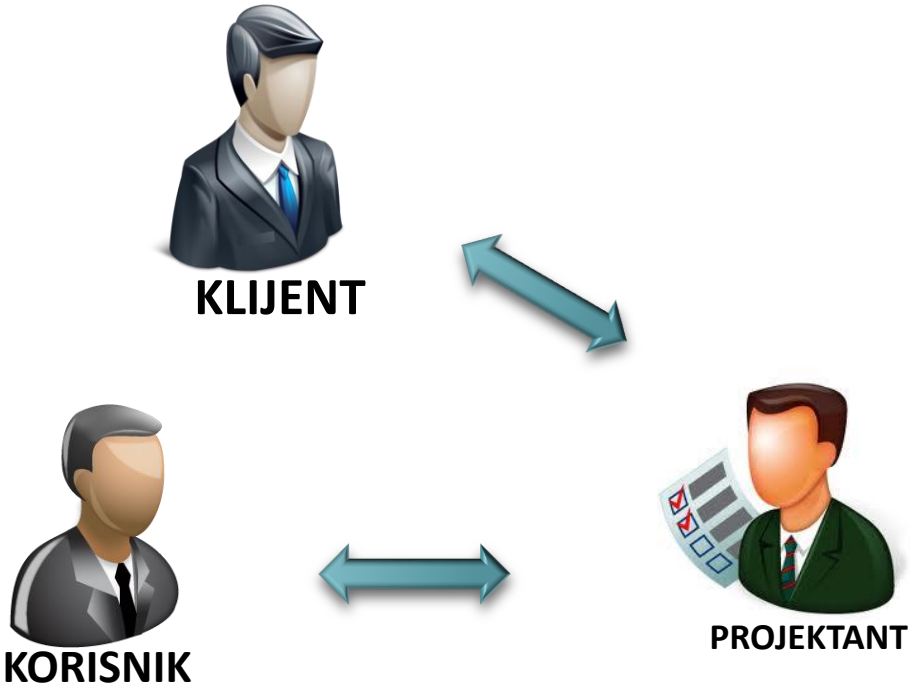
Softverski inženjering

- Potrebno je na neki način približiti metriku i kontekst određivanja kvaliteta



Softverski inženjering

- Učesnici i aktivnosti u razvoju softvera:
  - Analiza i definisanje zahtjeva
  - Projektovanje sistema
  - Pisanje programa
  - Testiranje jedinica
  - Integrativno testiranje
  - Testiranje sistema
  - Isporuka sistema
  - Održavanje



Softverski inženjering::Akteri

INSTRUKTOR

ANALITIČAR

KLIJENT

TESTER

PROJEKTANT

PROGRAMER

Softverski inženjering::Faze razvoja

ANALIZA I DEFINISANJE ZAHTJEVA

PROJEKTOVANJE SISTEMA

PROJEKTOVANJE PROGRAMA

IMPLEMENTACIJA PROGRAMA

TESTIRANJE JEDINICA

INTEGRATIVNO TESTIRANJE

TESIRANJE SISTEMA

ISPORUKA SISTEMA

ODRŽAVANJE

Softverski inženjering::Sistemski pristup

- Pojam granice sistema
- Pojam entiteta (objekata) i aktivnosti
- Sistem – skup entiteta, aktivnosti, njihovih odnosa i granica sistema

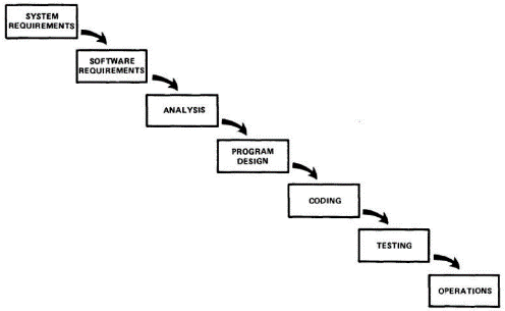
- Entiteti:
  - Student
  - Predmet
  - Profesor
  - Plan i program
  - ...

- Aktivnosti:
  - Upis godine
  - Prijava ispita
  - Zahtjev za potvrdu
  - ...

Softverski inženjering::Disciplina

- Pojmovi koji sačinjavaju disciplinu Softverskog inženjeringa:
  - Apstrakcija
  - Metoda analize i dizajna
    - standardi koji osiguravaju timski rad
  - Prototip korisničkog interfejsa
    - često identifikuje nejasnoće
  - Arhitektura softvera
    - dekompozicija
  - Softverski procesi
    - veličina projekta i kontekst primjene
  - Ponovna iskoristivost
  - Mjerenja
    - uspjeh i kvalitet
  - Alati i integrisana okruženja

Softverski inženjering::Stalne promjene



KRAJ PREZENTACIJE