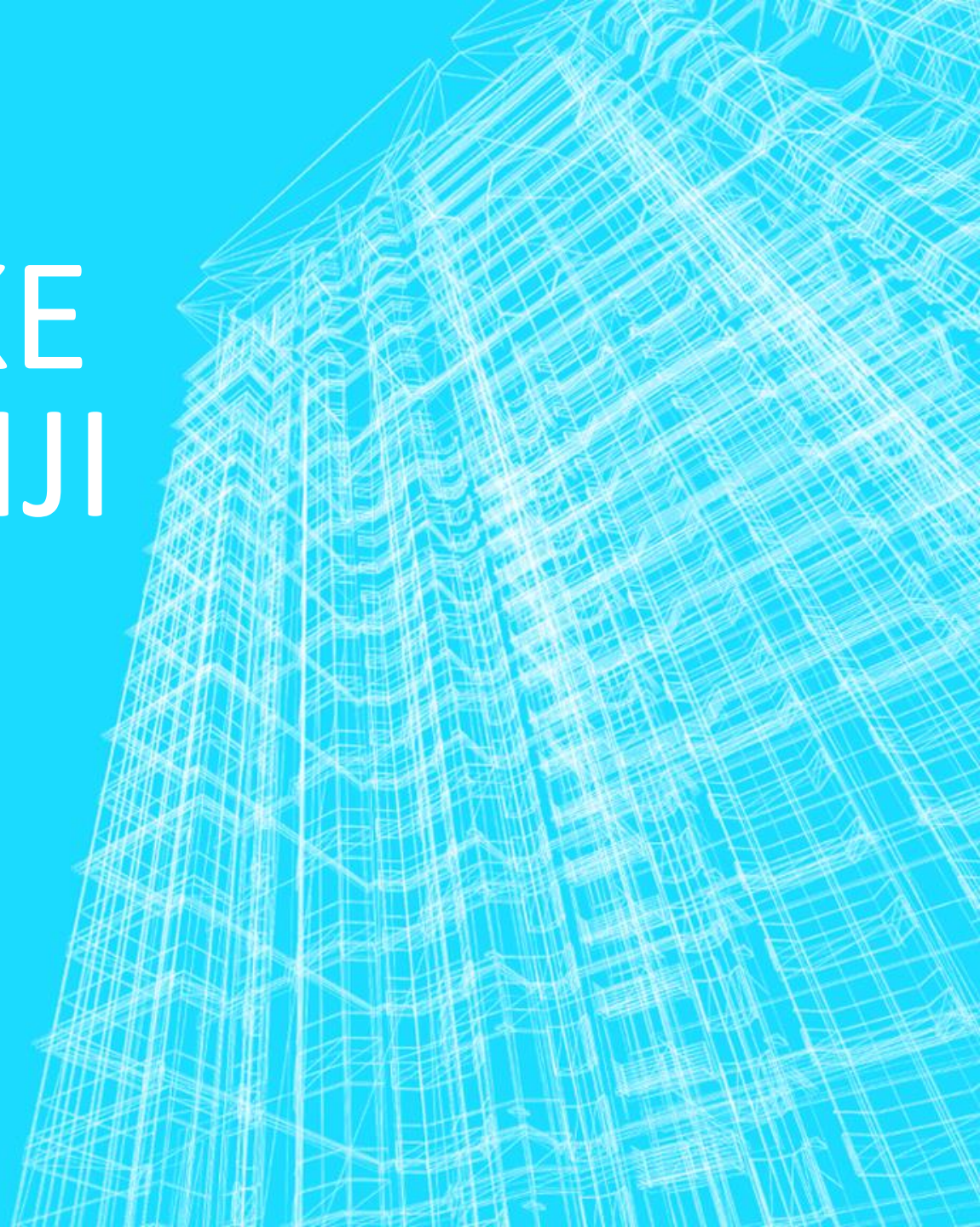
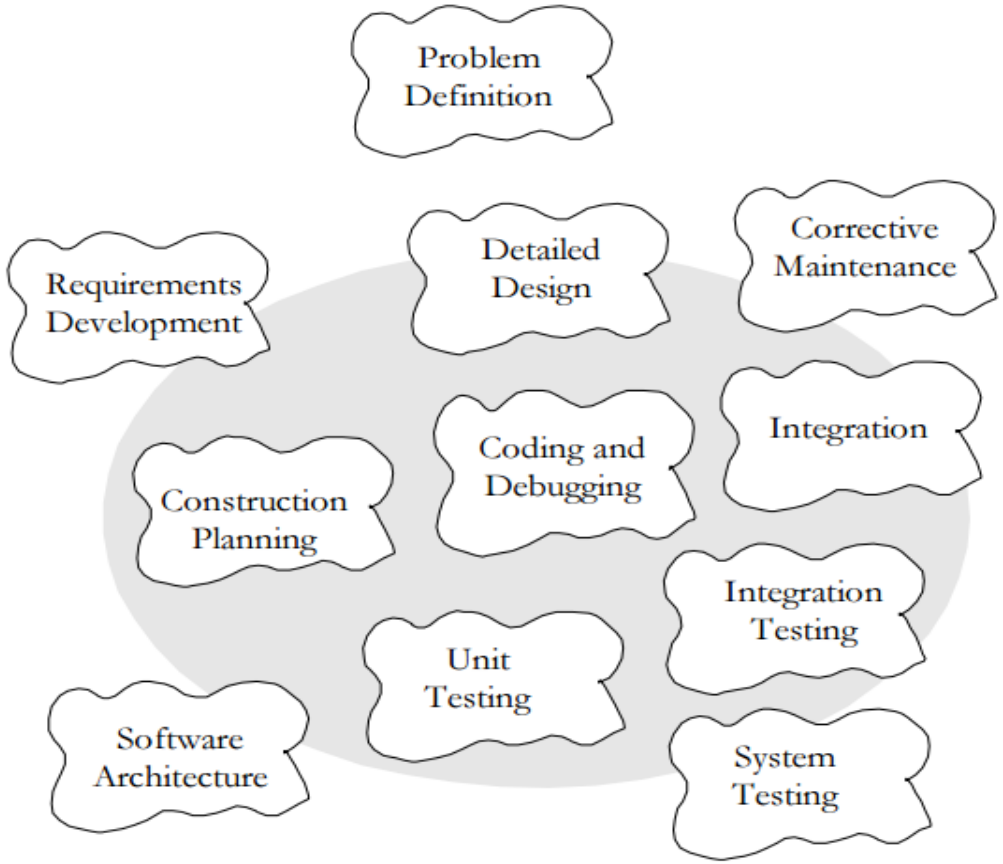
KLJUČNE ODLUKE PRI KONSTRUKCIJI SOFTVERA

Razvoj Softvera 2
Igor Simović 1128/2016
Profesor: Vladimir Filipović



STA PREDSTAVLJA TERMIN KONSTRUKCIJA SOFTVERA?



Aktivnosti konstrukcije softvera su obeležene sivim krugom. Konstrukcija ima fokus na kodiranje i debagovanje, medjutim uključuje i elemente detaljnog dizajniranja, unit testiranja, i drugih elemenata.

Konstrukcija se često poistovećuje i sa “kodiranjem” ili “programiranjem”. ”Kodiranje” i nije baš pogodna reč jer predstavlja mehanički proces prebacivanja nečeg postojećeg na jezik računara dok konstrukcija u velikoj meri podrazumeva kreativnost i rasudjivanje.

KLUČNE ODLUKE

- Izbor programskog jezika
- Programerske konvencije
- Pozicija u razvoju tehnologije
- Izbor dobrih praksi na koje se stavlja akcenat

IZBOR PROGRAMSKOG JEZIKA

Značaj izbora programskog jezika u kome će sistem biti implementiran je ogroman.

Pokazalo se da izbor programskog jezika utiče na produktivnost i kvalitet koda na nekoliko različitih načina:

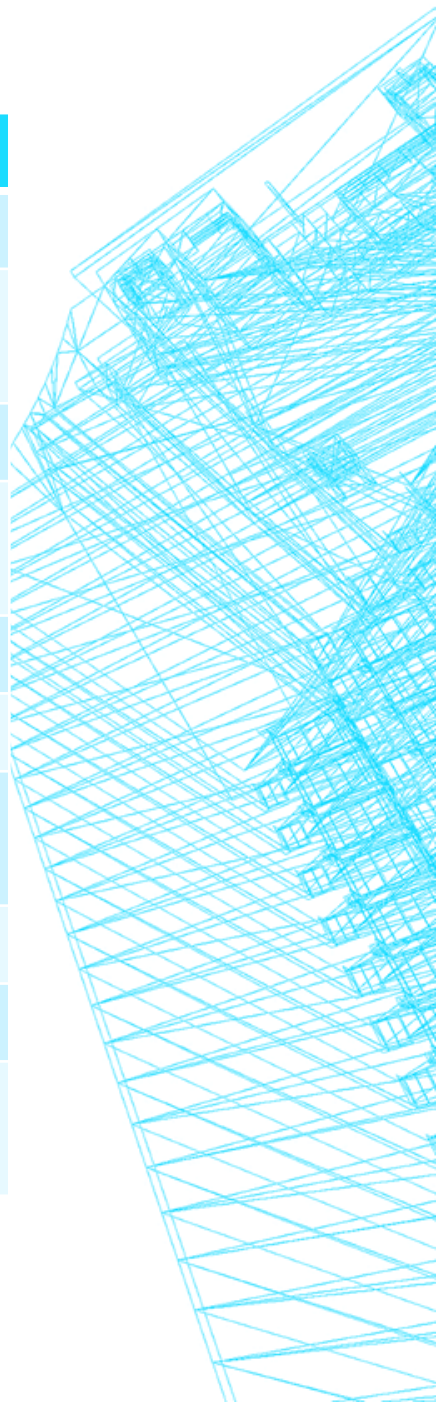
- Prethodno iskustvo i poznavanje izabranog progamskog jezika (ekspertiza)
- Nivo apstrakcije programskog jezika - jezici nižeg i višeg nivoa
- Način prevodjenja programa – interpretacija i kompilacija
- Izražajnost programerskih koncepta

OPIS PROGRAMSKIH JEZIKA

- Asembler (niskog nivoa, svaka instrukcija odgovara mašinskoj)
- C (opšte namene, prilično niskog nivoa sa nekim funkcionalnostima jezika visokog nivoa, UNIX)
- C++ (objektno orjentisan, zasnovan na C-u, mogućnost upravljanja memorijom)
- C# (opšte namene.objektno-orijentisan, Microsoft, zasnovan na C#)
- Fortran (prvi jezik višeg nivoa, FORmula TRANslation, naučne i matematičke svrhe)
- Java (objektno orijentisan, zasnovan na C-u, nezavisan od platforme)
- JavaScript (skrpt jezik, interpretatorskog tipa, web aplikacije, browser)
- PHP (open-source, skript jezik, web)
- Python (interpretatorskog tipa, objektno orijentisan, ranije za pisanje skripti, sada mnogo šira upotreba)
- SQL (upitni jezik, standard za operacije sa bazama podataka, deklarativan (nema flow))

Najbolji i najgori programski jezici za odredjene svrhe (deo tabele iz knjige CC)

Kind of Program	Best Languages	Worst Languages
Command-line processing	Cobol, Fortran, SQL	-
Cross-platform development	Java, Perl, Python	Assembler, C#, Visual Basic
Database manipulation	SQL, Visual Basic	Assembler, C
Direct memory manipulation	Assembler, C, C++	C#, Java, VB
Easy-to-maintain program	C++, Java, VB	Assembler, Perl
Fast execution	Assembler, C, C++, VB	JavaScript, Perl, Python
For environment with limited memory	Assembler, C	C#, Java, VB
Mathematical calculation	Fortran	Assembler
Real-time program	C, C++, Assembler	C#, Java, Python...
Web development	C#, Java, JavaScript, PHP, VB	Assembler, C



PROGRAMERSKE KONVENCIJE

Iako na prvi pogled ne izgleda tako, konvencije imaju veoma veliki značaj u procesu razvoja softvera. Ovde se misli na konvencije o imenovanju promenjljivih, notacijama, imenovanju klasa, formatiranju, komentarisanju itd..

POZICIJA U RAZVOJU TEHNOLOGIJE

Stepen razvijenosti odredjene tehnologije i programskog jezika u velikoj meri utiče na produktivnost.

Ako se odlučimo za neku tehnologiju koja je već u velikoj meri stabilna i razijena (stara) možemo smatrati da ćemo veći deo dana provoditi u pisanju novih funkcionalnosti. U suprotnom se može desiti da dobar deo vremena ode na razumevanje nedokumentovanih ili nedovoljno dobro dokumentovanih funkcionalnosti programskog jezika

IZBOR DOBRIH PRAKSI NA KOJE SE STAVLJA AKCENAT

Deo pripreme razvoja softvera podrazumeva i donošenje odluka o tome na koju od mnogobrojnih dobrih praksi se stavlja akcenat. Na nekim projektima se upražnjava programiranje u paru i TDD, dok se na nekima isključivo programira pojedinačno. Obe tehnike mogu dovesti do zeljenih rezultata u zavisnosti od okolnosti.

SPISAK DOBRIH PRAKSI

- Kodiranje:
  - Da li su definisane konvencije imenovanja promenljivih, komentara, formatiranja
  - Da li su dogovorene prakse pri odredjenim dogadjajima
  - Da li se je identifikovan položaj u razvoju tehnologije. Da li će se programirati u odredjenom jeziku ili ćemo biti ograničeni njime
- Timski Rad:
  - Da li su definisani koraci koje svaki programer mora da prodje pre nego sto njegov kod zavrsi u glavnom repozitorijumu
  - Da li će se programirati u paru, samostalno ili kombinacija prethodne dve tehnike

- Provera kvaliteta:
  - Da li programer mora da piše skup testova pre same funkcionalnosti
  - Da li moraju postojati unit testovi bez obzira bili pisani pre ili posle
  - Da li programer mora da sprovodi testove integracije pre nego što spoji svoj kod sa ostalim funkcionalnostima
  - Da li se praktikuje recenziranje koda
- Timski Rad:
  - Da li su definisani koraci koje svaki programer mora da prodje pre nego sto njegov kod zavrsi u glavnom repozitorijumu
  - Da li će se programirati u paru, samostalno ili kombinacija prethodne dve tehnike
- Alati:
  - Da li je izabran alat za kontrolu verzija
  - Da li je izabran programski jezik I verzija programskog jezika
  - Da li doneta o odluka o korišćenju nestandardnih funkcionalnosti jezika
  - Da li je odlučeno koji alati će se koristiti: (editor, refaktoring, test framework)