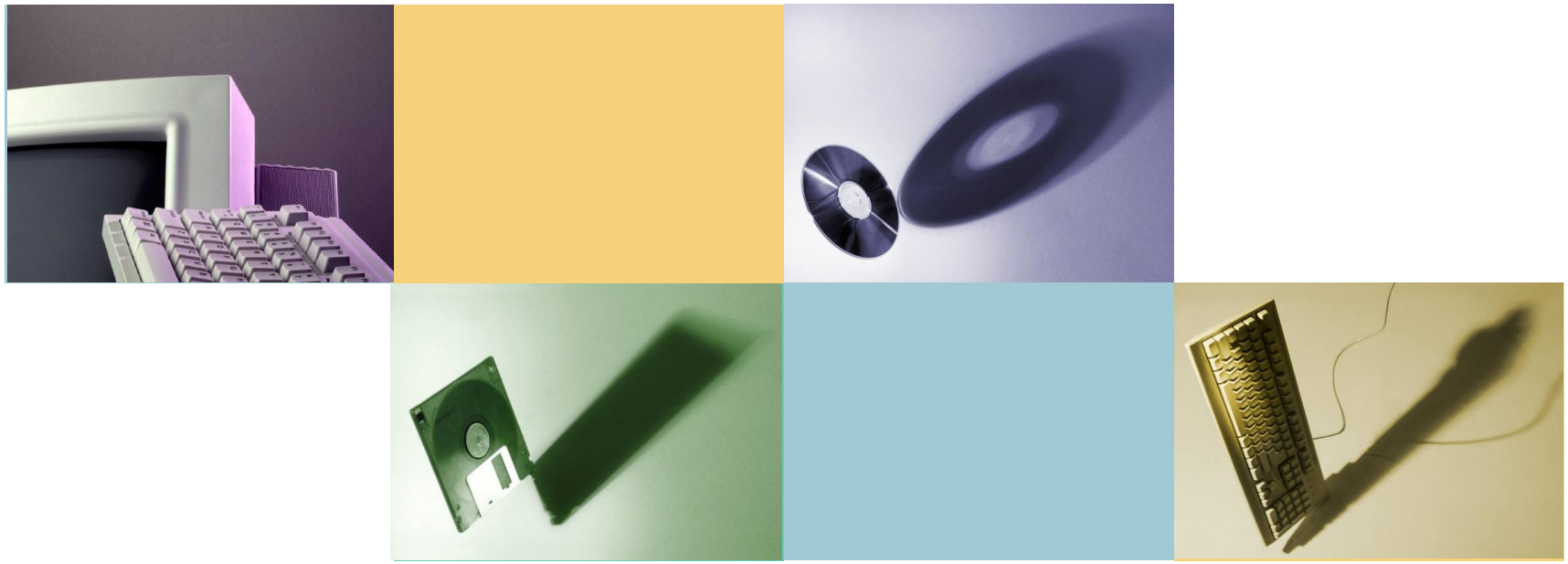
Upravljanje softverskim projektima i tehnološko preduzetništvo



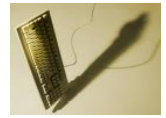
Upravljanje softverskim timovima

Prof. dr Dražen Drašković, predmetni nastavnik

Agenda

1. Ljudi i planiranje projekata
2. Softverski timovi
3. Uloge u softverskim timovima
4. Izbor ljudi u softverskim timovima





Ljudi i planiranje projekta

- Planiranje vremena
  - Lista aktivnosti (zadataka)
  - Podaci o resursima (**lista zaposlenih**, materijal, oprema)
  - Izlaz: dijagrami CPM, PERT i GANTT
- Planiranje potrebnih resursa
  - **Proceniti koliko je neophodno timova, ljudi**, drugih resursa
  - Aktivnostima dodeljivati **resurse**
  - Formira se kompletan budžet (planiranje troškova)
- **Planiranje komunikacije - pravila u timu**
- Planiranje rizika i nabavki
  - Uočiti moguće rizike i planirati ih pre početka projekta
  - Mogući rizici: **promene ljudi** u menadžmentu, **odlaska ljudi** iz organizacije/tima,...
  - Proceniti u kom mesecu projekta se rade koje nabavke; podugovaranja su nabavke!
- Planiranje kvaliteta
  - Definisati standard kvaliteta (pre početka), nakon definisanja kvaliteta prilagođavati se

SOFTVERSKI TIMOVI



Timski rad

- Mnogi softverski sistemi su suviše veliki ili kompleksni da bi bili razvijeni od strane pojedinca => stvaraju se softverski timovi!
- Tri ključna faktora koji forsiraju timski rad:
  - Širina domenskog znanja: stručnjaci iz različitih oblasti (frontend, backend, baze,...)
  - Vremenski okvir (Time-to-market): tim radi paralelno na različitim modulima
  - Održivost: tim obezbeđuje kontinuitet znanja
- Zadaci po svojoj prirodi mogu biti:
  - Deljeni (engl. *shared task*)
  - Individualni (engl. *individual*)
  - Kombinovani (engl. *combined*) - počinju individualno ali zahtevaju timsku integraciju



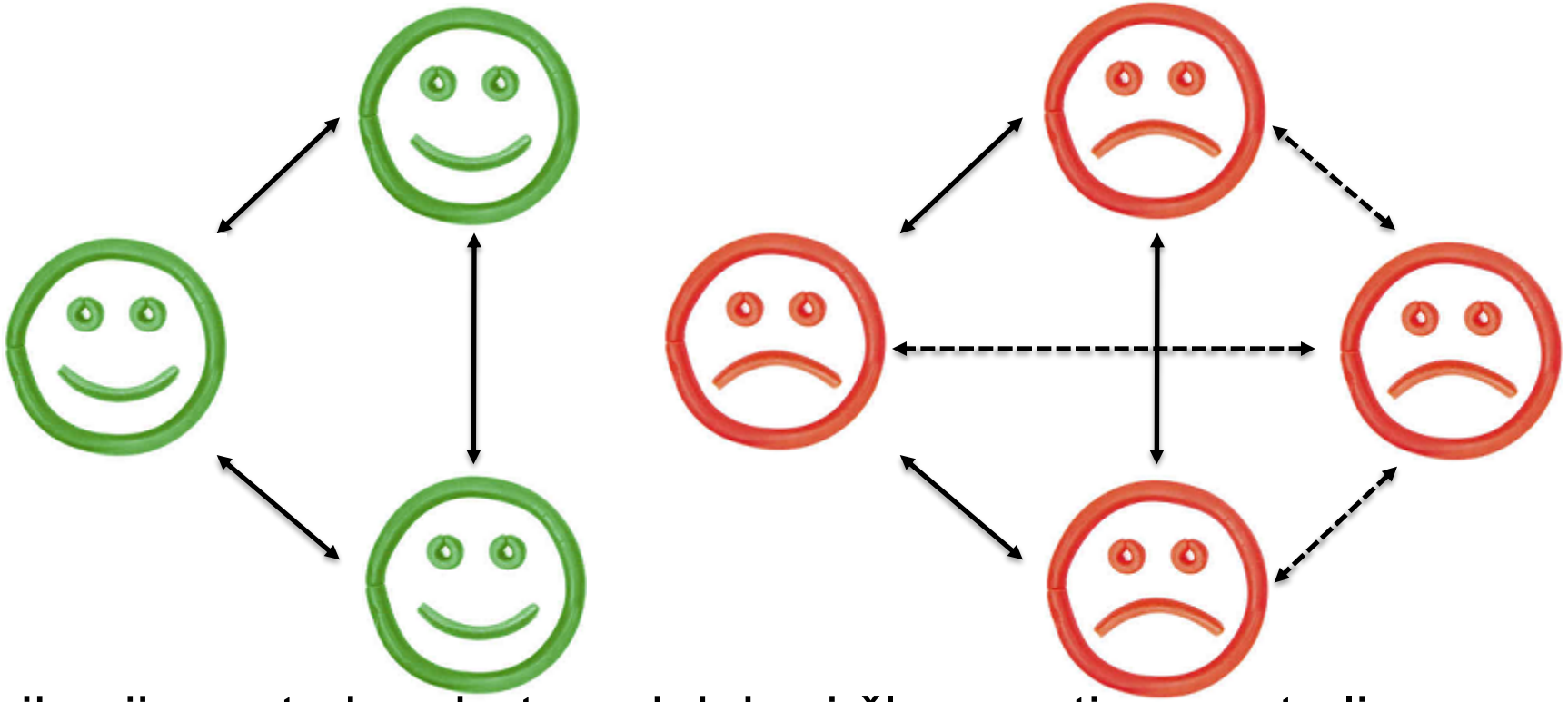
Timski rad i efikasnost

- Da bi bio efikasan, projektni tim mora biti organizovan tako da maksimizuje veštine i sposobnosti svakog člana tima.
  To je posao tim lidera.
- Mana: Šta ako tim počne da ignoriše standarde kompanije?
  Grupisanje ljudi, koji rade na nekom zadatku,
  vrlo često utiče negativno na autoritet postojećeg sistema u organizaciji.
- Prednost:
  Tim obično sadrži specijaliste za različite oblasti, koji su usmereni da završe neki važan zadatak.



Komunikaciona eksplozija

- Ako svako u timu od n ljudi, jedna osoba komunicira sa svakom drugom osobom iz tima, broj komunikacionih kanala je = $\frac{n(n-1)}{2}$



- Komunikacija raste kvadratno, dok broj članova tima raste linearno.
- Bruksov zakon:
  „Dodavanje ljudi na softverski projekat koji kasni, učiniće da on kasni još više“.

Organizacioni modeli softverskih timova

- Demokratski tim (*Egoless Programming*)
- Glavni-programer tim tzv. „šefovski tim“ (*Chief Programmer Team*)
- Modifikovani šefovski tim
- Sinhronizovan i stabilizovan tim (*Sync-and-Stabilize Team*)
- Ekstremno programiranje (XP)



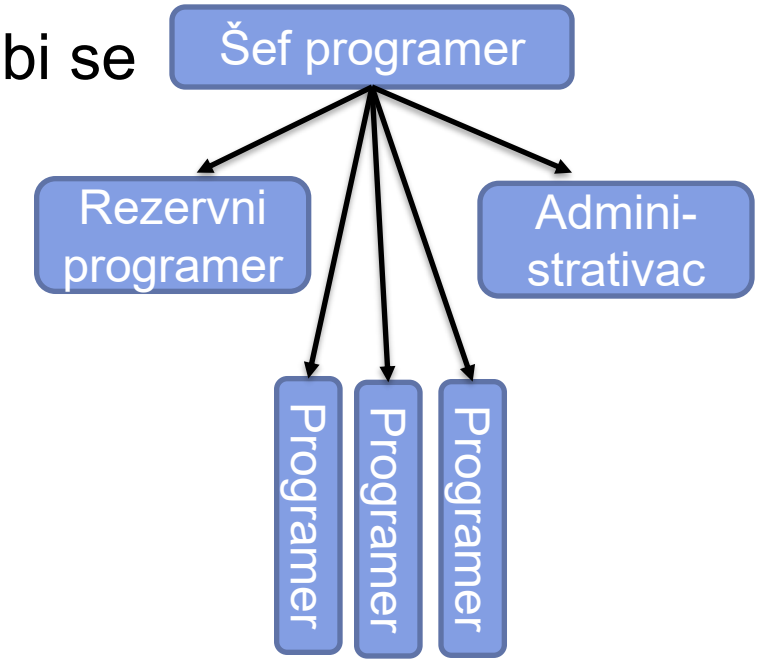
Demokratski timovi

- Ne postoji imenovani lider => u suprotnosti sa konvencionalnim principom upravljanja
- Zahtevaju „Programiranje bez ega“
  (američki informatičar *Gerald M. Weinberg*, 1933-2018, menadžer u IBM):
  - Programeri nisu previše vezani za svoj kod
  - Neophodna dobra atmosfera u timu i međusobna saradnja => **odluke su zajedničke**
  - Pronalaženje bagova daje pozitivan efekat (greške u kodu ne smatrati lično i prihvatiti pomoć kolega koji pronalaze greške)
- 10 programera „bez ega“ = demokratski tim
- Svako je jednak i tim je samoorganizovan
- Teorija tvrdi da ovakva grupa radi dobra za kompleksne probleme
- *Zaključak*: Odličan za istraživačke projekte,
  ali teško skalabilan zbog „komunikacione eksplozije“.



Šefovski (glavni-programer) tim

- Po uzoru na hiruške timove: hijerarhija se koristi da bi se prevazišla komunikaciona eksplozija, a specijalizacija da unapredi produktivnost.
- Glavni programerski timovi pružaju uslove veoma sposobnim programerima da budu odgovorni za veći deo izrade sistema
- Glavni-programer tim od 6 osoba redukuje broj komunikacionih kanala sa 15 na 5.
- *Zaključak:* Visoka efikasnost, ali ogroman rizik ako glavni programer napusti tim (tzv. *Bus/Truck Factor*).



- Uloge:
  - Šef (glavni programer): dizajnira arhitekturu sistema, programira kompleksne delove koda i integriše kod u celinu, sveobuhvatni menadžer.
  - Rezervni (zamenik) programer: preuzima po potrebi zadatke od šefa, dizajnira testove.
  - Administrativac: održava bazu podataka i dokumentaciju, pokreće testove.

Problemi kod šefovskih timova (1)

- Ovakav pristup, u različitim oblicima,
  nesumnjivo se pokazao uspešnim, ali postoje neki problemi:
  - Talentovane inženjere, dizajnere i programere je teško naći.
    Bez izuzetnih ljudi u ovim ulogama, ovakav pristup propada.
  - Ostali članovi grupe mogu da zamere glavnom programeru (šefu) na preuzimanju odgovornosti za uspeh,
    pa mogu namerno da sabotiraju njegovu ulogu.
  - Postoji visok rizik propadanja projekta,
    ako su i glavni programer (šef) i pomoćni programer nedostupni.
  - Organizacione strukture i stepeni možda neće biti u mogućnosti da podrže ovakvu grupu



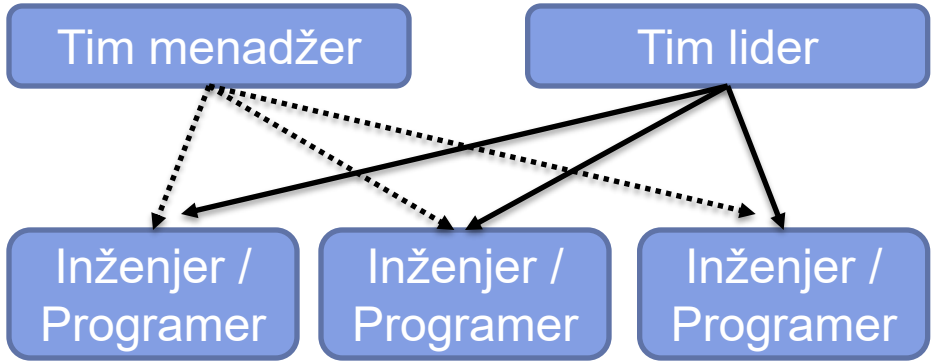
Problemi kod šefovskih timova (2)

- **Dobrog šefa je vrlo retko naći**: treba da bude presek visoko obrazovnog programera i veoma uspešnog projektnog menadžera.
- Zamenika glavnog programera je **još teže naći**: kapacitet kao šef, ali spreman da radi za manju platu i da radi u podređenom položaju (na nižoj poziciji).
- I administrativca je takođe teško naći:
  Ko želi da ne radi ništa oko razvoja softverskog sistema,
  nego samo da radi na dokumentaciji svakog dana.

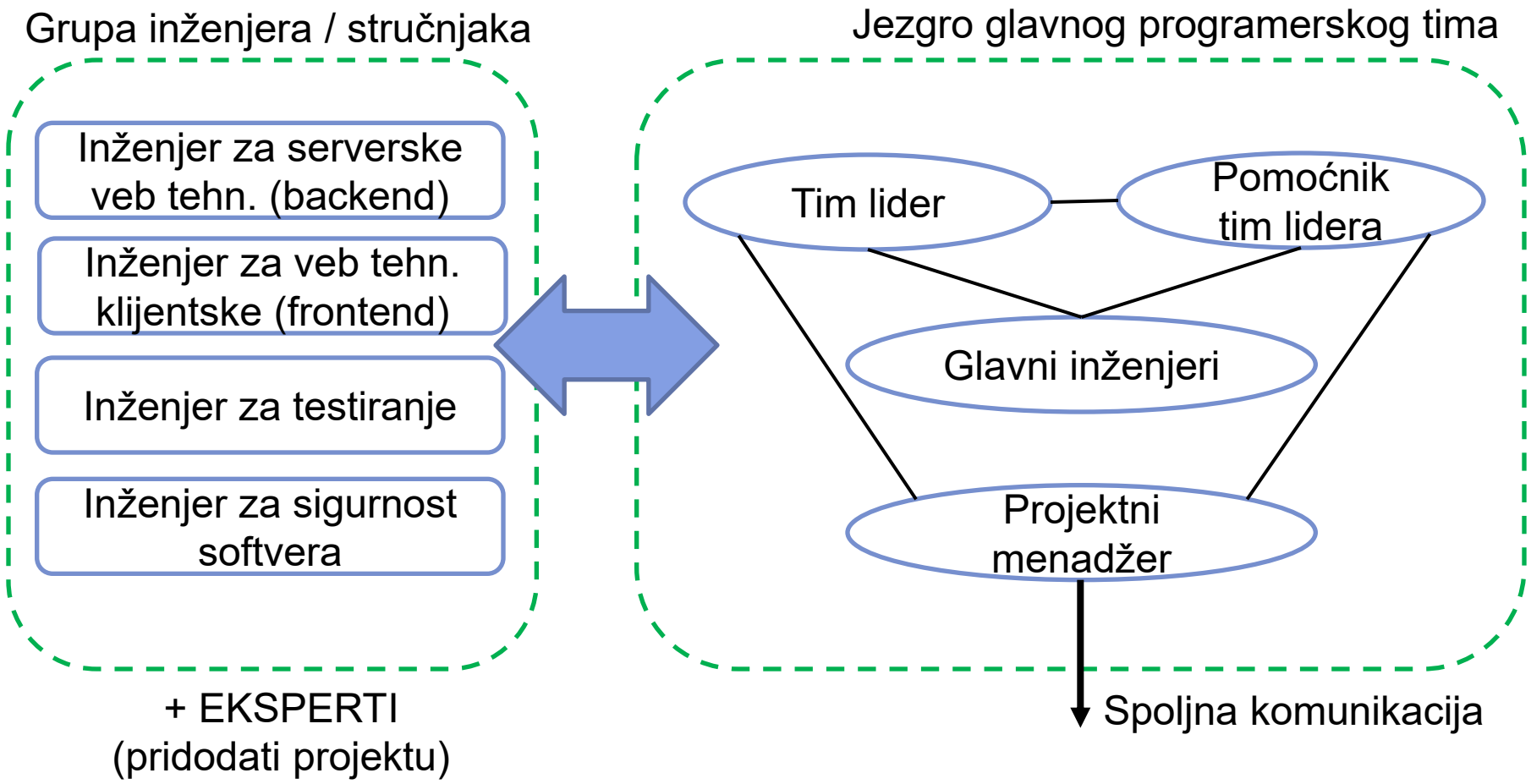
- Istorijat / *Fun Fact*:
  - „NewYork Times“ alat za automatsko sečenje fajlova = 83 000 LOC za 22 meseca (11 čovek/godina), uveo šefa u tim, kako bi bar 2-4x podigao produktivnost.
  - Polovina modula (svaki od po 200-400 LOC) bio je korektan prilikom prvog kompajliranja.
  - Alat za održavanje fajlova bio operativan nakon 20 meseci, bez greške.
  - Šef Terry Baker je jedan od najvećih programera za superračunare.

Modifikovani šefovski tim

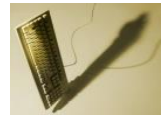
- Uvodi slojevi srednjeg menadžmenta kako bi se rasteretio glavni stručnjak
- Uloga šef programera je podeljena u 2 uloge:
  - Tim lider (za tehnička pitanja)
  - Tim menadžer (za netehnička menadžerska pitanja)
- Pažljivo odvojene odgovornosti uloga u timu, kako ne bi bilo konflikta interesa
- Ali mogu postojati neki delovi preklapanja
  (recimo kada jedan ode na godišnji odmor)
- Struktura tima može da se uveća, ako se uvede još liderskih nivoa
  (viši šefovi u hijerarhiji)



Modifikovani šefovski (glavni programer) timovi







Sinhronizovan i stabilizovan tim

- Microsoft model sa kraja 90-ih; uspešan za veoma velike projekte
  - Više od 3000 programera i testera je radilo na Win 2000
- Mali timovi od 3-8 programera i 3-8 testera, rade u paraleli.
- Pojedincima je dozvoljena sloboda da dizajnira i implementiraju specifikaciju, ali:
  - Programski kod mora biti integrisan na dnevnom nivou
  - Ukoliko tvoj programski kod sprečava kompilaciju, on mora da bude fiksiran odmah

- Timovi rade paralelno na modulima, a na kraju dana (ili nedelje) **vrši se integracija** (sinhronizacija) i **ispravljanje grešaka** (stabilizacija).
- Microsoft ima svoju poslovnu etiku već godinama, tako da njihov tim možemo nazvati „tim jake korporativne kulture“



XP tim (tim za ekstremno programiranje)

- Tim za agilne metodologije
- Baziran na programiranju u paru, stalnoj komunikaciji i brzim iteracijama:
  - šire znanje koje imaju dve osobe nego pojedinac
  - pomaže manje iskusnim programerima da se ubrzaju
- Dokazi:
  - Istraživanje na Univerzitetu Utah na naprednom kursu programiranja:
    Svoje testove parovi su radili za 60% vremena od vremena koje je bilo potrebno pojedincu. Test je prošlo 94% studenata, za razliku od 78% kada su radili pojedinačno.
- Programiranje u paru poboljšava zadovoljstvo poslom i sveukupno poverenje.



Vežba: Odaberite model softverskog tima

- Kao projektni menadžer treba da odaberete strukturu tima za sledeće projekte:

1) Projekat za kvantne računare ima 5 istraživača. Ne postoji strogi rok za završetak. Ako uspe na ovom projektu, ovaj tim će nastaviti da radi i na kasnijim projektima.

Demokratski ili XP tim.

2) Projekat razvoja sistema za naplatu za rudarsku korporaciju. Ovaj projekat angažuje 30 čovek/godina i mora biti završen u roku od 10 meseci (striktno).

Modifikovan šefovski tim.



Strukture tima prema matrici autoriteta

- Struktura tima zavisi od stila upravljanja vašom organizacijom, broja ljudi koji će činiti tim, njihovih nivoa znanja, kao i ukupne težine problema na kome se radi.

- Postoje 3 generičke kategorije strukture tima prema moći odlučivanja:
  - Demoktratska decentralizovana (engl. *Democratic decentralized,* DD)
  - Kontrolisana decentralizovana (engl. *Controlled decentralized,* CD)
  - Kontrolisana centralizovana (engl. *Controlled centralized,* CC)



DD struktura tima

- Demokratski decentralizovani timovi
- Ovaj softverski tim nema stalnog lidera.
- Tim vode koordinatori tokom trajanja nekog zadatka,
  a zatim budu zamenjeni drugim, koji mogu voditi različite zadatke.
- Odluke se donose grupnim konsenzusom.
- Komunikacija i kontrola su horizontalni.



CD struktura tima

- Kontrolisani decentralizovani timovi
- Ovaj softverski tim ima tim lidera koji koordinira određenim zadacima i druge vođe koji imaju odgovornost za podzadatke.
- Rešavanje problema ostaje timska aktivnost, ali implementacija rešenja se deli među podgrupama od strane tim lidera.
- Komunikacija među podgrupama je horizontalna,
  ali postoji i vertikalna komunikacija u kontrolnoj hijerarhiji, ka vođi.

- *Korelacija:* Većina današnjih agilnih timova (npr. Scrum) zapravo teži ovom modelu.



CC struktura tima

- Kontrolisani centralizovani timovi
- Stroga hijerarhija. Tim lider se bavi unutrašnjom koordinacijom u timu.
- Rešavanje problema takođe na najvišem nivou
  (vođa i rešava probleme i odlučuje).
- Komunikacija između lidera i članova tima je vertikalna.

- *Korelacija:* Odgovara modelu Glavnog programer tima.



Mapiranje modela softverskih timova u kategorije

Model tima	Kategorija / Struktura	Veličina tima (tipično)	Tehničko vođstvo	Menadžerske uloge / koordinacija	Pogodan za:
Demokratski	DD (decentralizovan)	Mali (3-6 članova)	Nema (rotira se)	PM samo fasilitator	Istraživačke projekte i visoku inovativnost
Šefovski (glavni-programer)	CC (centralizovan)	Srednji (5-10 članova)	Šef programer	Projektni administrator	Kritične sisteme i strogo def. rokove
Modifikovani šefovski	CC / CD (zavisi od nivoa)	Veći (10-20 članova)	Više lidera po modulima	Projektni menadžer (PM)	Velike sisteme sa jasnom hijerarhijom
Sinhronizovan i stabilizovan	CD (kontrolisan decentralizovan)	Veliki (podeljen na manje)	Program menadžeri (po oblastima)	Menadžer izdanja (release)	Komercijalne proizvode sa fiksnim datumima isporuke
XP (agilni tim)	CD (kontrolisan decentralizovan)	Mali do srednji (5-9 članova)	Tehnički lider	*Scrum Master* i *Product Owner*	Dinamične projekte sa čestim promenama zahteva



Timovi moderne IT industrije

- Razvijeni su novi modeli koji pokušavaju da reše problem **skalabilnosti** - kako da organizacija od 1.000 ili više programera ostane brza kao dok su bili mali startap.

- 2000: Amazon-ov Two-Pizza tim
- 2007: Holakratija (*Holacracy*) u softverskim timovima
- 2012: Spotify model



Amazonov Two-Pizza tim

- Uveo ga je *Jeff Bezos* (osnivač Amazona) još početkom 2000-ih, ali je postao globalni standard za računarstvo u oblaku (*cloud*) i mikroservisni razvoj.
- **Struktura:** Broj ljudi u timu nikada ne sme biti veći nego što dve pice mogu da nahrane (obično **od 5 do 8 ljudi**).
- **Posebnost: Potpuna odgovornost („You build it, you run it“).** Tim ne samo da razvija softver, već ga sam testira, pušta u rad i održava (nema odvojenog QA ili Ops tima).
- **Karakteristike:**
  - Odgovornost - Svaki tim funkcioniše kao mala firma (startap) unutar velike, i svaki tim preuzima odgovornost za određeni proizvod/funkciju, od ideje do produkcije.
  - Inovativnost - Ekstremna decentralizacija; osnažuje timove da eksperimentišu i da ne moraju očekivati stalno odobrenje od viših instanci upravljanja.
  - Agilnost - Manje ljudi, manje sastanaka, brži konsenzus. Smanjuje „komunikacionu eksploziju“, jer zabranjuje prevelike timove.

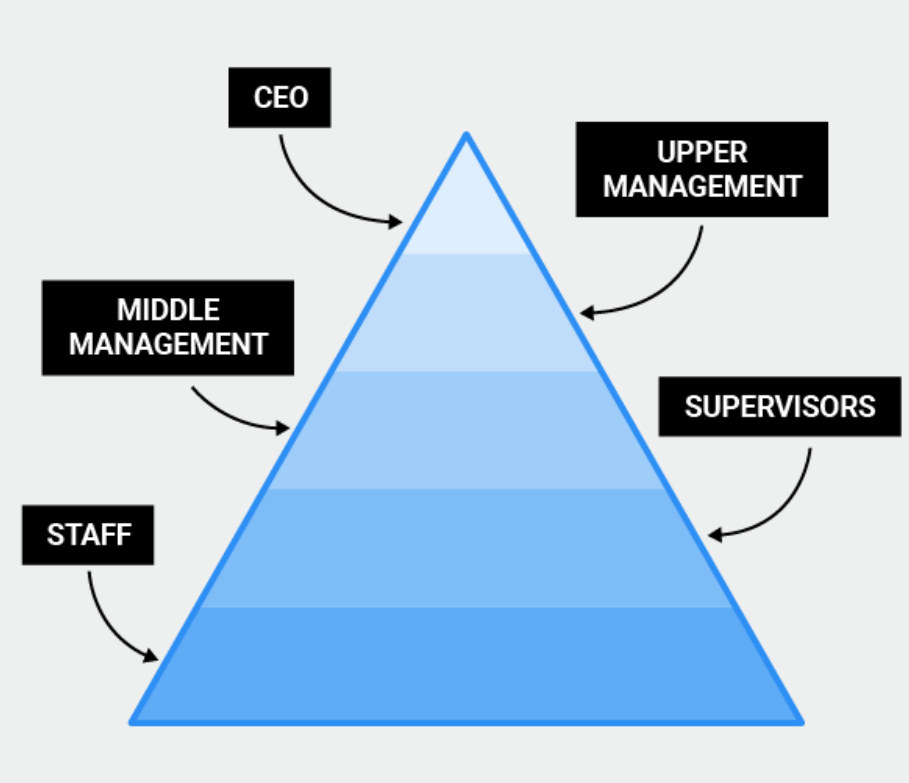


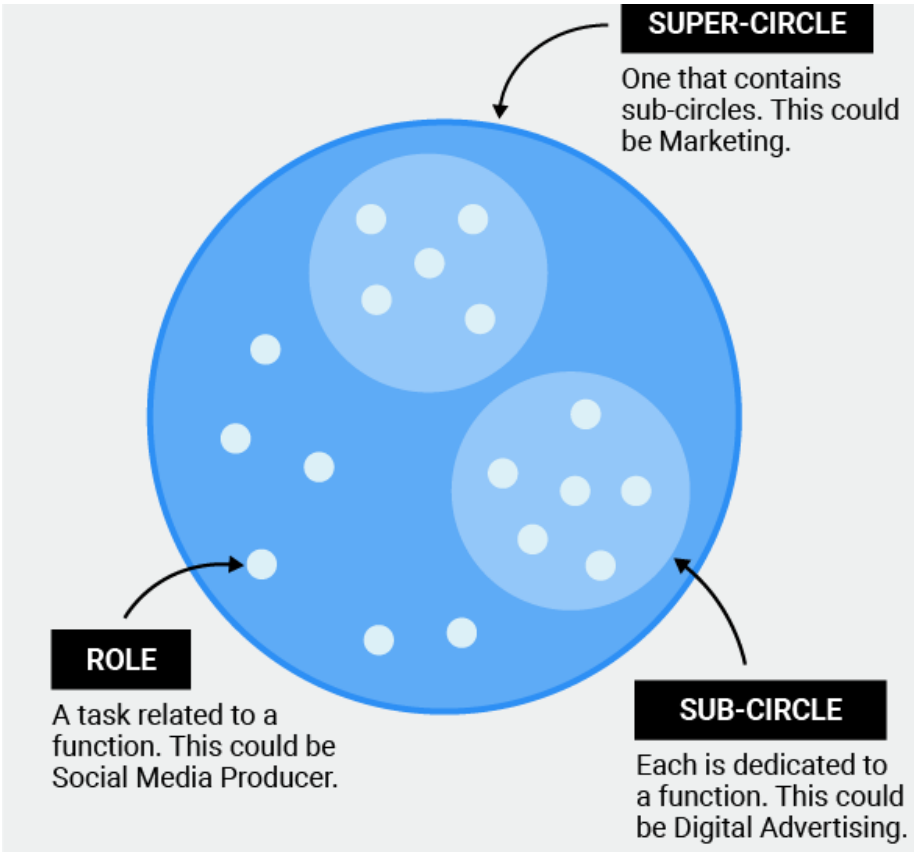
Holakratija

- Koncept je formalizovao *Brian Robertson* (2007),
  a najpoznatiju primenu je imao u kompaniji Ternary Software (USA).
- **Struktura:** Nema menadžera ni titula. Organizacija se sastoji od **krugova (*circles*)** koji se preklapaju. Ljudi nemaju pozicije već uloge koje se menjaju shodno potrebi projekta.
- **Posebnost:** Autoriteti nisu kod ljudi, već u procesu.
  Odluke se donose kroz striktan protokol na sastancima krugova.
- **Karakteristike:**
  - Ekstremni oblik **DD** strukture. Udaljavanje od fiksnog odnosa „Menadžer - podređeni“ ka modelu gde se autoritet raspoređuje na osnovu posla koji se obavlja.
  - Veoma težak za implementaciju jer zahteva visoku disciplinu i specifičnu korporativnu kulturu.

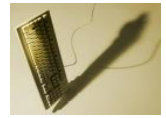


Holakratija u odnosu na hijerarhijsku strukturu



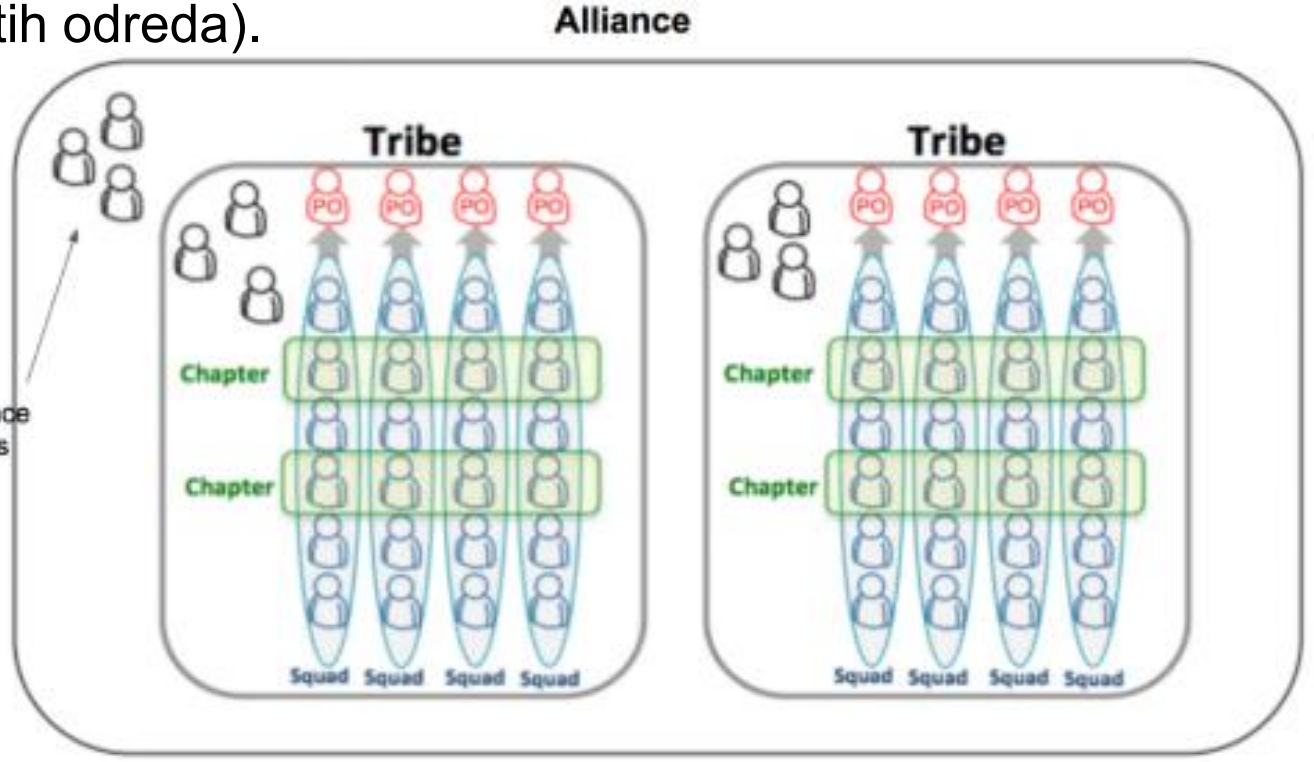






Spotify model softverskog tima

- Trenutno najpopularniji model za velike tech kompanije
- Uveli su ga u kompaniji Spotify (2007), Kniberg i Ivarsson
- Struktura:
  - **SQUAD** (timovi): potpuno autonoman tim od 6 do 12 ljudi, koji ima svog Product Owner-a.
  - **TRIBE** (pleme): skup više timova koji rade na srodnim oblastima (npr. veb ili mobilna app).
  - **CHAPTER** (poglavlje): horizontalna porodica stručnjaka (npr. svi backend programeri iz različitih odreda).
  - *Chapter Lead* - brine o karijernom razvoju ljudi, nije klasičan šef.
  - **GUILD** (udruženje): okuplja ljude iz cele kompanije koji žele da dele znanje o određenoj temi (npr. udruženje za testiranje, udruženje za Java programere,...). Dobrovoljno članstvo!
  - **ALLIANCE** (savez): koordinacija više plemena kod velikih projekata.



Timovi moderne IT industrije - Pregled

Model tima	Godina uvođenja	Fokus	Menadžerska uloga	Ključna karakteristika
AMAZON Two-Pizza model	2000	Brzina i vlasništvo	Product / tech Lead	Tim poseduje ceo životni ciklus (DevOps)
Holakratija	2007	Samoorganizacija	Nema formalnih menadžera	Uloge (role) umesto titula, krugovi umesto hijerarhije.
SPOTIFY model	2012	Skalabilnost i znanje	Chapter Lead (mentor, a ne šef)	Matrična organizacija (Timovi i Poglavlja)



ULOGE U SOFTVERSKIM TIMOVIMA



Uloge PM

- U savremenom razvoju softvera uloge projektno orijentisanih menadžera često se prilagođavaju metodologiji rada (agilna, Scrum, Kanban) ili specifičnim potrebama organizacije:
  - *Scrum Master*
  - *Product Manager* / *Product Owner*
  - Tehnički projektni menadžer (engl. *Technical Project Manager*, *TPM*)
  - Menadžer isporuke (engl. *Delivery Manager*)
  - Menadžer izdanja/puštanja u rad (engl. *Release Manager*)
  - Menadžer promena (engl. *Change Manager*)
  - PMO direktor (*Head of Project Management Office*)
- Ove uloge se često kombinuju.
- Manje firme: PM istovremeno i *Product Owner* i *Release Manager*
- Korporacije: strogo razdvojene funkcije, radi bolje kontrole kvaliteta.

Scrum Master

- U agilnom okruženju smatra se „servisnim liderom“
- U praksi softverskih kompanija:
  Scrum Master preuzima operativne aspekte PM
- Fokus na uklanjanju prepreka timu, facilitaciji sastanaka, osiguravanju da proces teče bez zastoja



Product Manager / Product Owner

- Ove uloge primarno fokusirane na proizvod - Šta se pravi?
- U velikoj meri se ipak preklapaju i sa projektnim menadžmentom - kako i kada se pravi?
- *Product Manager* upravlja životnim ciklusom proizvoda i često donosi ključne odluke o prioritetima, koje direktno utiču na plan projekta.



Tehnički PM (TPM)

- Uloga koja je specifična za IT velike kompanije i korporacije.
- TPM poseduje duboko inženjersko znanje i ne bavi se samo rokovima i resursima, već i:
  - arhitekturalnim rizicima
  - tehničkim dugom i koordinacijom između različitih razvojnih timova
- Predstavlja „most“ između poslovnih (biznis) zahteva i tehničke izvodljivosti da se oni realizuju



Menadžer isporuke (DM)

- Uslužno orijentisana uloga, prisutna često u kompanijama koje rade *outsourcing* usluge. Most između klijenta i inženjera.
- Fokus DM je na samoj isporuci klijentu, održavanju odnosa sa klijentom i osiguravanju da je kvalitet softvera u skladu sa ugovorenim nivoom usluge (SLA, *Service Level Agreement*).
- PM se fokusira na detalje:
  - Gantogram, JIRA tiketi, dnevni sastanci, precizna alokacija sati.
- DM se fokusira na ishod:
  - Da li je klijent srećan?
    (ali nekada klijentu reći i NE, tako da ne pokvarimo odnos)
  - Da li je tim efikasan? (predvideti probleme pre nego što se dese)
  - Da li je proces isporuke stabilan? (ili nešto „koči“ isporuku)

Menadžer izdanja (RM)

- Uloga na velikim i kompleksnim projektima, gde postoji mnogo zavisnosti između različitih delova koda.
- RM koordinira proces planiranja, pripreme i puštanja softvera u produkciju.
- RM usklađuje rad razvojnih timova, QA (testiranja) i DevOps.
- Dok se PM bavi širim ciljevima projekta, RM je fokusiran na samu završnicu:
- **PM:** „Da li smo završili sve planirane funkcionalnosti za ovaj kvartal?“
- **RM:** „Da li je verzija 2.4.1 prošla regresiono testiranje i da li su serveri spremni za migraciju večeras u 2 ujutru?“
- Ova uloga je kritična u:
  - **Velikim bankarskim i finansijskim sistemima** (gde svaka greška košta milione)
  - **SaaS platformama** (sa milionima korisnika gde sistem ne sme da padne)
  - **Sistemima sa mnogo zavisnosti**
    (npr. mikroservisna arhitektura gde jedan servis zavisi od pet drugih)

Menadžer izdanja (RM) - odgovornosti

- Ključne odgovornosti:
  - **Planiranje ciklusa izdanja** (*Release Train*): Definiše kalendar puštanja novih verzija. Na primer: određuje da se „veliki“ update radi jednom mesečno, a „mali“ (*bug fixes*) svake nedelje.
  - **Koordinacija između timova:** Osigurava da Tim A (koji radi na bazi podataka) završi svoj deo pre nego što Tim B (koji radi na mobilnoj aplikaciji) pokuša da pusti svoju novu verziju.
  - **Upravljanje rizicima:** Procenjuje da li je kod dovoljno stabilan. Ako QA (testiranje) prijavi kritičnu grešku pred samo puštanje, RM je taj koji ima moć da kaže: *„STANI, ovo ne ide uživo danas.“*
  - **Rollback strategija:** RM uvek mora imati „plan B“. Ako nakon puštanja koda sve krene po zlu, on vodi proces vraćanja na prethodnu stabilnu verziju.
  - **Automatizacija procesa:** U saradnji sa DevOps inženjerima, radi na tome da proces puštanja softvera bude što više automatizovan (kroz CI/CD).

Menadžer promena (CM)

- Ne bavi se samim kodom ili serverima, već **ljudskim faktorom**.
  U IT svetu, možete napraviti tehnički savršen softver, ali ako zaposleni ili klijenti odbiju da ga koriste ili ga koriste pogrešno, projekat se smatra neuspešnim.
- Ključne odgovornosti:
  - **Analiza uticaja (*Impact Analysis*):** Pre nego što se softver pusti, *CM* procenjuje ko će sve biti pogođen promenom i na koji način.
  - **Komunikacioni plan:** Kreira strategiju obaveštavanja. On objašnjava korisnicima *zašto* se promena dešava i koje su *prednosti* za njih (tzv. WIIFM - "*What's in it for me?*").
  - **Upravljanje otporom:** Identifikuje pojedince ili grupe koji se protive promeni i radi sa njima na prevazilaženju barijera.
  - **Plan obuke (*Training Plan*):** U saradnji sa timom, organizuje edukacije i priprema materijale (uputstva, video tutorijale) kako bi korisnici znali da koriste novi alat od prvog dana.
  - **Merenje uspeha:** Prati statistiku - koliko ljudi se prijavilo na novi sistem, koliko često ga koriste i kakav je njihov komentar nakon mesec dana korišćenja.

Menadžer promena - glavne veštine

- Razlike PM i CM:
  - PM: Fokusira se na to da softver bude instaliran (tehnička implementacija).
  - CM: Fokusira se na to da softver bude korišćen (ljudska implementacija).
- Kada je CM neophodan?
  - Uvođenje novog ERP ili CRM sistema (npr. prelazak cele firme na SAP ili Salesforce).
  - Digitalna transformacija (kada se tradicionalni procesi seli na Cloud).
  - Spajanje dve kompanije (*mergers & acquisitions*) gde treba uskladiti različite kulture i alate (npr. *Hewlett-Packard* akvizirao *Compaq*, 2002, 25 milijardi $).
- **Veštine CM**
  - **Empatija i aktivno slušanje:**
    Mora da razume strahove zaposlenih (npr. strah da će ih softver/AI zameniti).
  - **Psihologija promene:** Često se oslanjaju na modele kao što je **ADKAR** (Awareness, Desire, Knowledge, Ability, Reinforcement).
  - **Vrhunska komunikacija:** Sposobnost da kompleksne tehničke promene objasni jednostavnim rečnikom.

PMO direktor

- Strateška uloga iznad nivoa pojedinačnih projekata
- PMO direktor postavlja standarde, metodologije i alate koje će svi projektni menadžeri u kompaniji koristiti
- Osigurava da su svi projekti usklađeni sa strateškim ciljevima kompanije
- Ključne odgovornosti:
  - **Upravljanje portfolijom:** Odlučuje koji projekti će se uopšte raditi. Ako kompanija ima resurse za 10 projekata, a 50 ideja, vrši selekciju na osnovu ROI i strateškog značaja.
  - **Definisanje metodologije:** Određuje „pravila igre”. Da li će firma raditi po Scrum-u, vodopadu ili hibridnom modelu? Koji alati će se koristiti (Jira, Microsoft Project, Asana)?
  - **Upravljanje resursima na nivou firme:** Prati opterećenost svih timova. Ako jedan sektor „izgara” od posla, a drugi je slobodan, on vrši stratešku preraspodelu ljudi.
  - **Izveštavanje top menadžmenta (C-level):** On je glas projekata pred direktorima (CEO, CTO, CFO). On ne priča o pojedinačnim bagovima, već o zdravlju portfolija, rizicima po budžet i napretku strateških inicijativa.
  - **Mentorstvo i razvoj PM kadrova:** Brine o karijernom putu projektnih menadžera, organizuje obuke i sertifikacije (poput PMP-a).

PMO direktor (2)

- Razlike između PMO direktor i PM:
  - Standardni PM: Da li moj projekat kasni?
  - PMO: Koliko naših projekata kasni u proseku u ovom trenutku, i kako da promenimo proces da se to više ne dešava?
- Veštine PMO direktora:
  - Strateška vizija – sposobnost da vidi 2-3 godine unapred.
  - Finansijska pismenost - duboko razumevanje budžetiranja, CAPEX/OPEX troškovi (kapitalni izdaci / operativni izdaci) i poslovni bilansi.
  - Politička veština – sposobnost balansiranja između različitih sektora u firmi (npr. sukob između prodaje koja obećava sve, i inženjerskog razvojnog tima koji ne može sve da postigne).
- U fazi skaliranja kompanije, PMO direktor je ključna figura!
- Dok je startap mali, sve je ad-hoc, ali kada kompanija poraste na 200+ inženjera, bez PMO funkcije bi nastao haos.

Uporedni pregled odgovornosti

Uloga	Odgovara na pitanje	Glavni fokus	Tipičan nivo
Project manager	Kako i kada?	Rokovi i resursi	Planiranje projekta, upravljanje budžetom, rizicima i komunikacija sa *stakeholders*.
Product manager	Šta i zašto?	Vrednost i vizija	Definisanje funkcionalnosti proizvoda, prioritizacija „backlog-a“ i usklađivanje sa potrebama tržišta.
Technical PM	Da li je izvodljivo?	Tehnička egzekucija	Koordinacija inženjerskih timova, rešavanje tehničkih blokada i arhitekturalni rizici.
Scrum master	Kako tim radi?	Protok i proces	Facilitacija agilnih ceremonija, uklanjanje prepreka (*impediments*) i zaštita tima od spoljnih pritisaka.
Delivery manager	Da li je stiglo?	Isporuka i klijent	Osiguravanje da finalni paket stigne do klijenta na vreme i u dogovorenom kvalitetu (česta uloga u outsourcing-u).
Release manager	Da li je spremno i bezbedno?	Deployment (Puštanje u rad)	Planiranje i koordinacija puštanja koda u produkciju, usklađivanje Dev-a, QA-a i Operations-a.
Change manager	Šta se menja?	Tranzicija i ljudi	Upravljanje promenama u sistemu ili organizaciji kako bi se osiguralo da korisnici prihvate novi softver/proces.
PMO Director	Da li smo usklađeni?	Portofolio i standardi	Nadgledanje svih projekata u firmi, definisanje metodologije rada i upravljanje resursima na nivou kompanije.

Zadatak 1: Transformacija kompanije

- Vi ste na poziciji **PMO direktora** u kompaniji *TechNova Solutions* koja se bavi razvojem softvera za e-trgovinu i logistiku. Kompanija je naglo porasla i tradicionalna hijerarhija više ne funkcioniše. Odlučili ste da implementirate **Spotify model** kako biste povećali autonomiju timova, ali zadržali visok nivo deljenja znanja.
- Vaš zadatak je da na osnovu sledećeg opisa mapirate organizaciju:
- 1) Operativni nivo: Imate tim pod nazivom „SmartPay“ koji se bavi isključivo integracijom kreditnih kartica. Tim je autonoman, ima svog programera, testera i Product Owner-a.
- 2) Sektorski nivo: Tim „SmartPay“, zajedno sa timovima „E-Wallet“ i „CryptoCheck“, čini širu grupu koja se bavi finansijskim uslugama i svi sede u istom delu zgrade.
- 3) Profesionalni nivo: Svi Java programeri iz svih timova u kompaniji sastaju se jednom nedeljno sa svojim mentorom kako bi usaglasili standarde pisanja koda i rešili zajedničke tehničke probleme.
- 4) Interesni nivo: Grupa entuzijasta iz različitih delova kompanije (prodaja, razvoj, marketing) pokrenula je inicijativu za istraživanje primene veštačke inteligencije i sastaju se neformalno radi razmene ideja.
- Identifikujte koji delovi opisa odgovaraju sledećim terminima Spotify modela: **Squad, Tribe, Chapter i Guild**. Nacrtajte i prikažite kako ta struktura izgleda. Broj ljudi po strukturama neka bude prilagođen Spotify modelu.

Zadatak 1: Transformacija kompanije (rešenje)

- Mapiranje termina:
- *Squad* (tim): Tim „SmartPay“ (autonoman, multidisciplinaran tim), kao i „E-Wallet“ i „CryptoCheck“.
- *Tribe* (pleme): Grupa za „Finansijske usluge“ (skup svih srodnih timova).
- *Chapter* (poglavlje): Grupa Java programera i njihov mentor (horizontalno povezivanje iste struke).
- *Guild* (udruženje): Grupa za veštačku inteligenciju (neformalna grupa ljudi, istih interesovanja, na nivou cele kompanije).

- Komentari:
- Razlikovanje vertikalne i horizontalne ose: Studenti često mešaju Poglavlje i Udruženje. Ovde jasno vide da je Poglavlje vezano za posao (Java programeri), a Udruženje za šira interesovanja (AI).
- Uloga menadžmenta: Kroz ovaj primer možete objasniti da PMO direktor ne upravlja svakim timom, već postavlja ovaj sistem koji se samoreguliše.

Zadatak 1: Transformacija kompanije (pitanje 1)

- U timu „SmartPay“, jedan programer insistira da se koristi nova verzija Jave koju je naučio u svom Udruženju ali njegov *Chapter Lead* (mentor za Javu) kaže da kompanija još uvek ne podržava tu verziju zbog stabilnosti. Ko donosi finalnu odluku?

- *Odgovor*:

U Spotify modelu, *Chapter Lead* je taj koji definiše KAKO se radi (tehnički standardi), dok Udruženje (uz Product Ownera) odlučuje ŠTA se radi. Dakle, *Chapter Lead* ima poslednju reč o tehničkom standardu kako bi se izbegao haos u kodu na nivou cele firme.

Zadatak 1: Transformacija kompanije (pitanje 2)

- Ako Pleme (*Tribe*) za „Finansijske usluge“ naraste na 250 ljudi, a Vi i dalje insistirate da svi budu deo istog Plemena kako bi bili blizu, koji rizik preuzimate?

- *Odgovor*:

Preuzimate rizik **komunikacione eksplozije**. Prema Spotify preporukama, Pleme ne bi trebalo da pređe Dunbarov broj (otprilike ~150 ljudi). Preko te granice, ljudi prestaju da se poznaju, poverenje opada, a broj komunikacionih kanala postaje nemoguć za upravljanje. Rešenje je podela na dva manja Plemena ili uvođenje Saveza (*Alliance*).



Zadatak 1: Transformacija kompanije (pitanje 3)

- Tim „*SmartPay*“ želi da pusti novu verziju koda u produkciju, ali im je potrebna promena na bazi podataka kojom upravlja drugi tim. Da li je „*SmartPay*“ u ovom slučaju pravi autonomni tim?

- *Odgovor*:

Tehnički, u idealnom Spotify modelu, nije pravi autonomni tim. Pravi tim bi trebalo da bude međufunkcionalni i da ima svog inženjera za baze podataka kako bi bio potpuno autonoman. Ako zavise od drugog tima, stvara se usko grlo (bottleneck), što je upravo ono što ovaj model pokušava da izbegne.

Zadatak 2: Skaliranje platforme

- Kompanija „RapidDeliver“ razvija aplikaciju za dostavu hrane koja je postala žrtva sopstvenog uspeha. Trenutno imaju jedan ogroman tim od 45 inženjera koji rade na monolitnom kodu. Sastanci traju satima, odluke se donose sporo, a svako puštanje nove verzije softvera je rizično jer niko ne poznaje ceo sistem. Kao konsultant za softversko upravljanje, predlažete uvođenje **Amazon-ovog Two-Pizza tim modela**.
- Vaš cilj je da razbijete ovaj veliki tim na manje jedinice.
  Na raspolaganju imate sledeće profile:
  - 6 backend programera
  - 4 frontend (Mobile/Web) programera
  - 2 DevOps inženjera
  - 2 testera (QA)
  - 2 Product Owner-a
- a) Formirajte timove prema pravilima. Koliko timova ćete napraviti i ko će činiti te timove?
- b) Definišite šta znači princip *„You build it, you run it“* u kontekstu odgovornosti ovih timova.
- c) Objasnite kako će ovi timovi međusobno komunicirati ako svaki tim razvija svoj deo koda (mikroservis).

Zadatak 2: Skaliranje platforme (rešenje)

- a) Formiranje timova
- Napravićemo dva nezavisna tima. Svaki tim će imati po 7-8 članova, što je idealno da se nahrane sa 2 pice.
- Tim A - npr. Korisnički nalozi i Plaćanja
  3 backend, 2 frontend, 1 DevOps, 1 QA, 1 Product Owner
- Tim B - npr. Katalog restorana i Logistika dostave
  3 backend, 2 frontend, 1 DevOps, 1 QA, 1 Product Owner

- b) Definisanje principa
- Ovo je ključna karakteristika Amazon modela. To znači da timovi nemaju odvojen „Operations“ tim kojem predaju kod na održavanje.
- Tim koji je napisao kod za plaćanja odgovoran je i za njegovo funkcionisanje u 3 ujutru. Ako sistem padne, članovi tog konkretnog tima primaju poziv i rešavaju problem. Ovo dramatično podiže kvalitet koda, jer niko ne želi da ga softver koji je loše napisao budi usred noći.

Zadatak 2: Skaliranje platforme (rešenje)

- c) Komunikacija između timova
- Umesto dugih sastanaka (koji su neefikasni zbog komunikacione eksplozije), timovi komuniciraju isključivo preko interfejsa API-ja (*Application Programming Interfaces*).
- Tim A ne mora da zna kako je Tim B napisao svoj kod unutra. Oni se samo dogovore oko ugovora (interfejsa) - koje podatke Tim A šalje, a šta Tim B vraća. Ovo se zove labava sprega (*loose coupling*).

Zadatak 2: Skaliranje platforme (pitanje 1)

- U kompaniji imamo samo jednog vrhunskog stručnjaka za sigurnost (*Security Expert*). U koji od dva *Two-Pizza* tima ćete ga smestiti?

- *Odgovor*:

Ni u jedan. U ovom modelu, specifični eksperti deluju kao unutrašnji konsultanti ili se njihovo znanje pretvara u automatizovane alate (servise) koje oba tima mogu da koriste. Ako ga stavite u jedan tim, postajete usko grlo za drugi.



Zadatak 2: Skaliranje platforme (pitanje 2)

- Tim B je postao veoma uspešan i dobio je budžet da zaposli još 5 ljudi. Sada ih ima 13. Šta PM treba da uradi?

- *Odgovor*:

Tim se mora podeliti na dva nova Two-Pizza tima. Prema Bezosu, čim tim postane prevelik za dve pice, birokratija i troškovi komunikacije počinju da guše inovaciju.



Zadatak 2: Skaliranje platforme (pitanje 3)

- Može li *Two-Pizza* tim da radi na monolitnoj arhitekturi?

- *Odgovor*:

Teško. Ovaj model skoro uvek zahteva prelazak na mikroservise. Ako svi timovi menjaju isti fajl u isto vreme (monolit), stalno će se sudarati i autonomija, koja je glavni cilj ovog modela, biće izgubljena.



IZBOR LJUDI



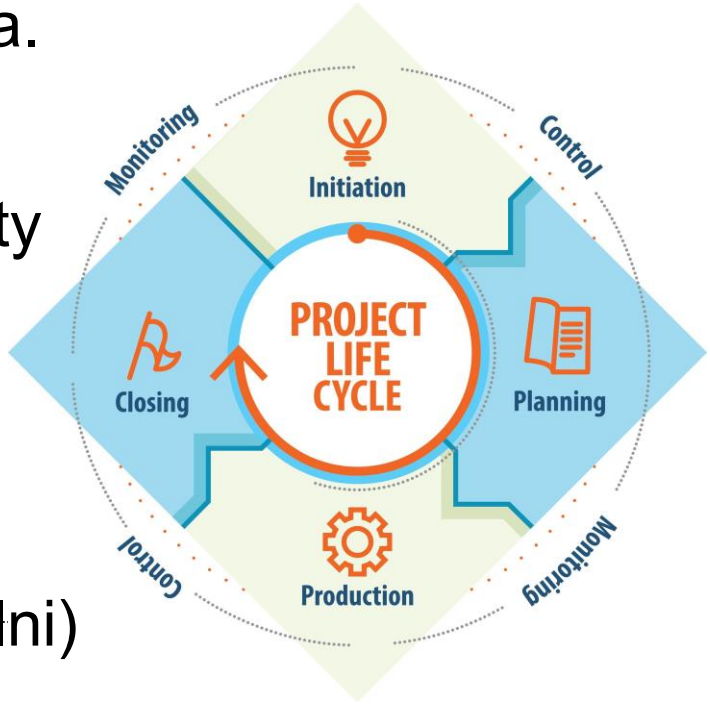
Ljudi - deo tima/organizacije

- Ljudi su najvažniji resursi svake organizacije.
  - *Produktivnost u softveru nije linearna, već intelektualna.*
- Zadaci menadžera su obično orijentisani ka ljudima.
  - Menadžer u IT-u ne popravlja kod,
    on popravlja okruženje u kojem ljudi pišu kod.
- Ukoliko ne postoji razumevanje ljudi, rukovođenje će biti neuspešno.
- Softversko inženjerstvo je primarno misaona aktivnost.
  Misaona ograničenja efektivno ograničavaju softverski proces.
  - Ako je softverski proces previše komplikovan (previše sastanaka, loša dokumentacija, haotičan kod), inženjer troši svu misaonu snagu na proces, a ne na rešenje problema.



Aktivnosti menadžmenta

- Rešavanje problema (korišćenjem raspoloživih ljudi)
  - Npr. šta ukoliko glavni senior programer ode na bolovanje usred Sprinta?
- Motivacija (ljudi koji rade na projektu)
  - Menadžer prepoznaje šta koga motiviše. Šta Vas motiviše?
- Planiranje (šta će ljudi da rade)
  - Balansiranje između želja klijenata i kapaciteta tima.
- Procena (koliko će brzo ljudi da rade)
  - Loša procena sopstvenog vremena. Scrum: Velocity
- Monitoring i praćenje (ljudskih aktivnosti)
  - JIRA alat, Burndown grafikon, GitHub metrike
- Organizacija (način na koji će ljudi da rade)
  - Izbor prave metodologije i modela (vodopad vs agilni)



Motivacija

- Važna uloga menadžera je da motiviše ljude koji rade na projektu
- Motivacija je kompleksan problem, ali se može reći da postoje različiti tipovi motivacije koji se zasnivaju na:

1) Osnovnim (psihološkim) potrebama (npr. hrana, voda, spavanje, itd.)
2) Sigurnosnim (ličnim) potrebama (npr. poštovanje, samopouzdanje)
3) Socijalnim (društvenim) potrebama (npr. biti prihvaćen kao deo grupe)

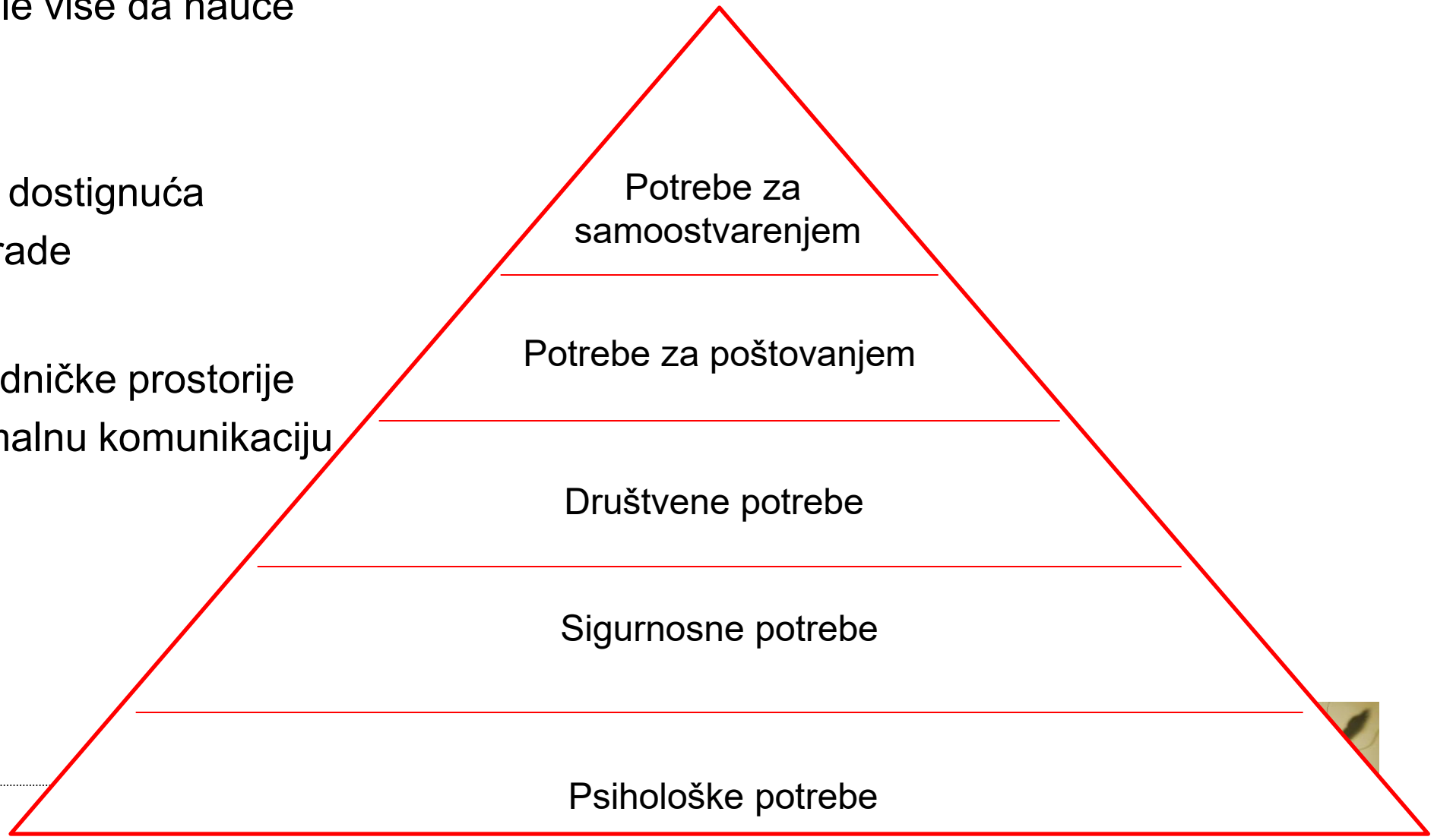
Da li u kontekstu softverskog inženjerstva, bazične potrebe kao što je 1) mogu biti zadovoljene visokom platom? => Motivacija se seli na više nivoe.



Maslovljeva hijerarhija ljudskih potreba

* Abraham Maslow, 1943, rad "A theory of Human Motivation"

- Samoostvarenje (lično ispunjenje)
  - Obuka - ljudi žele više da nauče
  - Odgovornost
- Poštovanje
  - Priznanje nekih dostignuća
  - Adekvatne nagrade
- Društvene
  - Obezbediti zajedničke prostorije
  - Dozvoliti neformalnu komunikaciju



Šta je relevantno za buduće IT lidere?

- Daniel Pink kaže da su za kreativne i misaone poslove ključna tri faktora koja nadilaze Maslovljevu piramidu:
  - Autonomija: Želja ljudi da sami upravljaju svojim radom (vreme, tehnika, tim).
  - Majstorstvo (*Mastery*): Potreba da postajemo sve bolji u onome što radimo (učenje novih tehnologija, rešavanje kompleksnih arhitektonskih problema).
  - Svrha (*Purpose*): Saznanje da to što kodiramo ima smisla i da nekom pomaže (npr. aplikacija koja spašava živote ili olakšava poslovanje).

- Razlika između spoljašnje (ekstrinične) i unutrašnje (intrinične) motivacije:
- Spoljašnja: Plata, bonusi, titule. One funkcionišu samo do određene granice (tzv. higijenski faktori).
- Unutrašnja: Uživanje u samom rešavanju problema. Softverski inženjeri su često zaljubljenici u tehnologiju i njih najviše motiviše rad na nečem izazovnom.



Specifičnosti u IT svetu

- Zanesenost (Flow stanje)
  - To je stanje u kojem je programer toliko fokusiran na zadatak da gubi osećaj za vreme.
  - Ako je zadatak prelak: dosadno im je; ako je pretežak: anksiozni su.
  - Motivacija je pogoditi tu zlatnu sredinu.
- Izazov: Rad na zastarelim tehnologijama (*legacy code*) ubija motivaciju brže od male plate.
- Priznanje kolega: Inženjerima je često važnije šta o njihovom kodu misle kolege programeri nego sam menadžer.



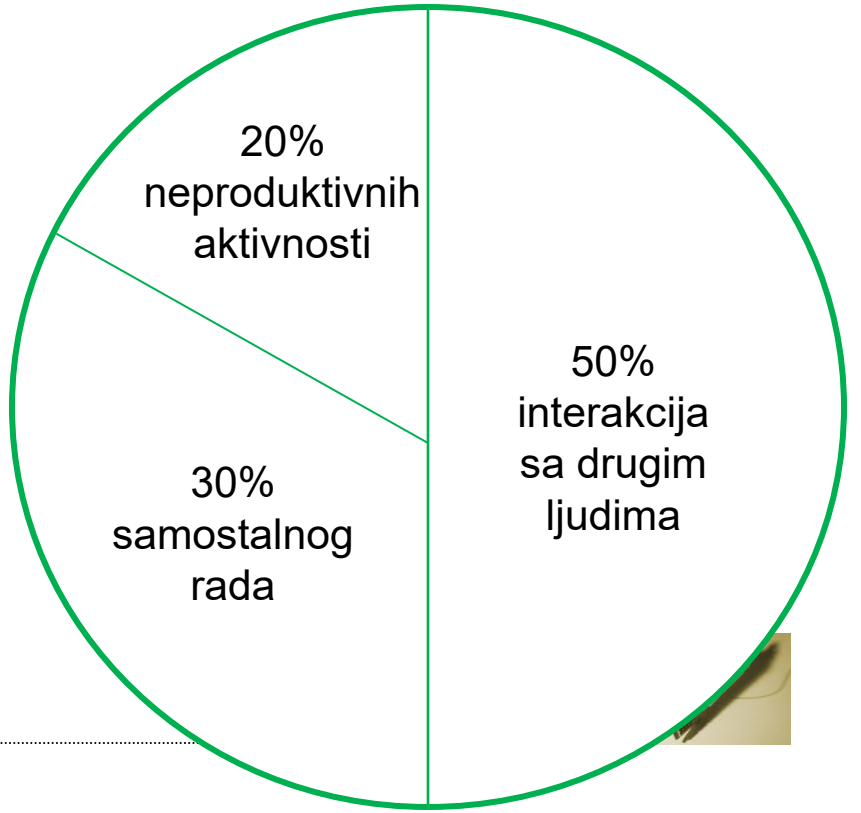
Rad u grupi

- Softversko inženjerstvo je uglavnom grupna aktivnost
  - Plan razvoja većine netrivijalnih softverskih projekata je takav da ih ne može završiti jedna osoba koja radi sama
- Grupna interakcija je ključni faktor za performanse grupe
- Fleksibilnost u formiranju grupe je ograničena
  - Menadžeri moraju da se snađu najbolje što mogu sa raspoloživim ljudima



Formiranje grupe

- Grupa formirana od članova sa istom motivacijom može biti problematična
  - Okrenuti zadatku (*task-oriented*) - svako želi da radi svoje
  - Okrenut sebi (*self-oriented*) - svako želi da bude glavni
  - Okrenuti interakciji (*interaction-oriented*) – previše ćaskanja, nedovoljno rada
- Efikasna grupa ima balans svih tipova
- Teško ostvarivo, jer je većina inženjera okrenuta zadatku
- Potreba da svi članovi učestvuju u odlukama koje utiču na celu grupu
- Statistika vremena: 50% je interakcija!



Tuckmanov model razvoja tima

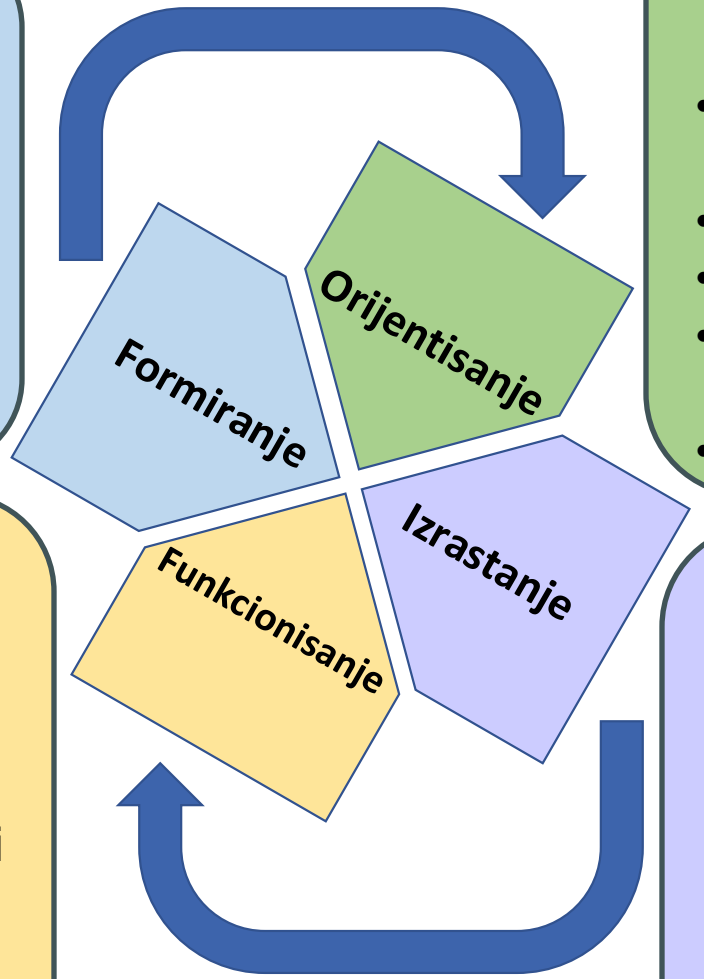
- **Tuckman-ov model razvoja tima** objašnjava faze kroz koje svaki IT tim prolazi:
  - **Forming** (Formiranje / Upoznavanje)
  - **Storming** (Orijentisanje / Konflikti oko ideja i liderstva)
  - **Norming** (Izrastanje/ Uspostavljanje pravila)
  - **Performing** (Funkcionisanje / Faza visoke produktivnost)

- **Poruka: Tim nije tim od prvog dana**,
  već da je to entitet koji se razvija kroz krize.



Tuckmanov model razvoja tima

- Članovi rezervisani i pomalo nesigurni
- Vizija i predstavljanje ciljeva.
- Upoznavanje sa pojedinačnim ulogama.
- Međusobno upoznavanje članova tima. Neformalna druženja.
- Uspostavljanje pogodne radne atmosfere.
- Menadžer mora biti direktan.



- Početak delotvornijeg rada/debata.
- Jača povezanost (kohezija) i uzajamna podrška među članovima.
- Faza entuzijazma i razočarenja (sporo dolazimo do rezultata).
- Preispitivanje uloga.
- Grupna polarizacija i konflikti.
- Učenje suočavanja i nošenja sa indiviualnim razlikama članova.
- Menadžer ohrabruje članove.

- Moral tima je visok.
- Članovi su zadovoljni i samopouzdani.
- Odnosi su uspostavljeni i stabilni.
- Svako ispunjava svoju timsku ulogu.
- Usklađena je međusobna saradnja.
- Članovi tima osećaju odgovornost jedni za druge i za obavljanje zadataka.
- Podrška minimalna i diskretna – usmerena na pohvale i učenja u timu.

- Tim se učvršćuje u skladnu celinu.
- Fokus je na timskim ciljevima.
- Usavršavaju se metode timskog rada.
- Finalizacija raspodele uloga i statusa članova.
- Učvršćivanje timskih normi ponašanja.
- Izražena saradnja i učinkovita komunikacija.

Liderstvo u grupi

- Liderstvo se oslanja na poštovanje, ne na položaj
- Mogu postojati i tehnički i administrativni lider (~modifikovani šefovski tim)
- Demokratsko liderstvo je efikasnije od autokratskog
- Treba podržavati razvoj karijere zasnovan na tehničkoj stručnosti



Kohezija grupe

- U kohezivnoj grupi, članovi smatraju grupu važnijom od bilo kog pojedinca
- Prednosti kohezivne grupe su:
  - Moguće je razviti standard kvaliteta grupe
  - Članovi grupe rade zajednički, tako da se greške izazvane neznanjem smanjuju
  - Članovi tima uče jedni od drugih i upoznaju se sa tuđim poslovima
  - Može se praktikovati nesebično programiranje gde članovi teže da međusobno poboljšaju programe



Razvijanje kohezije

- Na koheziju utiču faktori kao što su organizaciona kultura i karakter pojedinaca u grupi
- Kohezija se može podsticati kroz
  - Društvene događaje
  - Razvijanje identiteta grupe i teritorije
  - Eksplicitne aktivnosti na razvoju tima
- Otvorenost informacija je jednostavan način da se obezbedi da se svi članovi grupe osećaju kao njen deo



Lojalnost grupi

- Članovi imaju tendenciju da budu lojalni kohezivnim grupama
- „Grupno mišljenje” je očuvanje grupe bez obzira na tehničke i organizacione aspekte
- Rukovodstvo bi trebalo da radi na izbegavanju grupnog mišljenja nametanjem spoljne umešanosti u svakoj od grupa



Komunikacija u grupi

- Dobra komunikacija je neophodna za efikasan rad grupe
- Mora postojati razmena informacija o stanju projekta, razvojnim odlukama i promenama prethodnih odluka
- Dobra komunikacija jača povezanost grupe jer promoviše razumevanje
- Status članova grupe
  - Članovi višeg statusa teže da dominiraju u razgovorima (**HIPPO efekat** - *Highest Paid Person's Opinion*)
- Karakteri u grupi - Previše ljudi sličnog karaktera mogu predstavljati problem
  - Npr. svi hoće biti softverske arhitekte, niko ne želi da piše dokumentaciju ☺
- Polna struktura grupe - Polno mešovite grupe bolje komuniciraju
  - Socijalna osetljivost; Smanjuju se neproduktivni konflikti i ubrzava faza *Norming*.
- Komunikacioni kanali
  - Komunikacija usmerena ka centralnom koordinatoru je uglavnom neefikasna; dobra samo za proste zadatke



Komunikacija u grupi (2)

- Fizičko vs. Virtuelno okruženje
  - Komunikacija nije više samo razgovor u kancelariji
  - **Asinhrona komunikacija** (Slack, JIRA komentari, itd.)
  - Efikasan tim zna kada treba da piše, a kada je neophodno da se čuje/vidi
- Psihološka bezbednost (*Psychological Safety*)
  - Uverenje da član tima neće biti kažnjen ili ismejan ako napravi grešku, postavi „glupo” pitanje ili predloži radikalnu ideju.
  - Bez ovoga, komunikacija je samo prenos podataka, a ne i saradnja.
- Komunikacija u odnosu na veličinu grupe
  - Što je grupa veća, kanali moraju postati formalniji (sastanci), što smanjuje spontanost i kreativnost.



Izbor i zadržavanje ljudi

- Izbor ljudi za rad na projektu je glavna menadžerska dužnost
- Odluke se obično zasnivaju na:
  - specifikacija posla/pozicije u timu
    - Da li nam treba izvršilac ili donosilac odluka?
  - kreiranje profila osobe koju tražimo
    - Profil osobe mora da odgovara „kulturnom kodu" kompanije (tzv. *Cultural Fit*)
  - informacijama dobijenim od kandidata (CV) + GitHub/StackOverflow
    - Portfolio (kod koji je pisao) je i bitniji u IT sektoru
  - intervju, informacijama dobijenim u razgovoru za posao
  - referencama, preporukama dobijenim od drugih ljudi koji poznaju kandidata (mentor, bivše kolege - saradnici, itd.)
- Neke kompanije koriste psihološke i testove sposobnosti



Intervju - tehnički

- Tehnički intervju (provera znanja)
  - **Kodiranje uživo** (Live coding): Da li zna da reši problem?
  - Fokus nije samo na tačnom rešenju, već na **razmišljanju naglas**.
  - Ispituje se kako kandidat pristupa problemu, kako tretira granične slučajeve (*edge cases*) i kolika je optimizovanost koda?
  - Primeri:
    - Implementiraj funkciju koja balansira binarno stablo?
    - Napravi API krajnju tačku za filtriranje korisnika po ...
  - **Pair Programming sa seniorom:** Ovo je modernija i opuštenija varijanta. Kandidat i senior inženjer zajedno rešavaju zadatak.
  - Pored koda, ovde se gleda **kako kandidat sarađuje**.
    Da li prihvata sugestije?
    Da li zna da objasni svoj proces? Ovo simulira realan radni dan u timu.

Intervju - sistemski dizajn

- Sistemski dizajn intervju
  - Ovaj intervju je obavezan za senior pozicije/softverske arhitekte.
    Ne piše se kod, već se crtaju komponente sistema na visokom nivou.
  - **Zadatak:** Obično je vrlo širok, npr. „Dizajniraj sistem kao što je Instagram” ili „Dizajniraj servis za skraćivanje URL adresa”.
  - **Šta se ocenjuje:**
    Kandidat mora da objasni kako bi povezao baze podataka, servere, keširanje (*Caching*), balansere opterećenja (*Load Balancers*) i kako bi osigurao da sistem ne padne, ako ga koristi milion ljudi istovremeno.
  - **Ključni koncepti:** Skalabilnost, dostupnost i pouzdanost sistema.
  - Ovde se vidi da li osoba razume širu sliku projekta.



Bihejvioralni intervju: Provera „karaktera”

- Dok prethodna dva testiraju šta kandidat **zna**, ovaj testira ko kandidat **jeste**.
- Cilj je predvideti buduće ponašanje na osnovu prošlih iskustava.
- **Metoda STAR -** Od kandidata se očekuje da odgovori koristeći formulu: **S**ituation (Situacija), **T**ask (Zadatak), **A**ction (Akcija koju je preduzeo) i **R**esult (Rezultat).
- **Tipična pitanja:**
  - „Ispričaj nam o situaciji kada si zakasnio sa rokom. Kako si to rešio?”
  - „Opiši konflikt koji si imao sa kolegom oko tehničkog rešenja i kako ste došli do kompromisa.”
- **Šta se traži od kandidata:** Emocionalna inteligencija (EQ), sposobnost rešavanja konflikata, timski duh i usklađenost sa vrednostima kompanije (*Cultural Fit*).



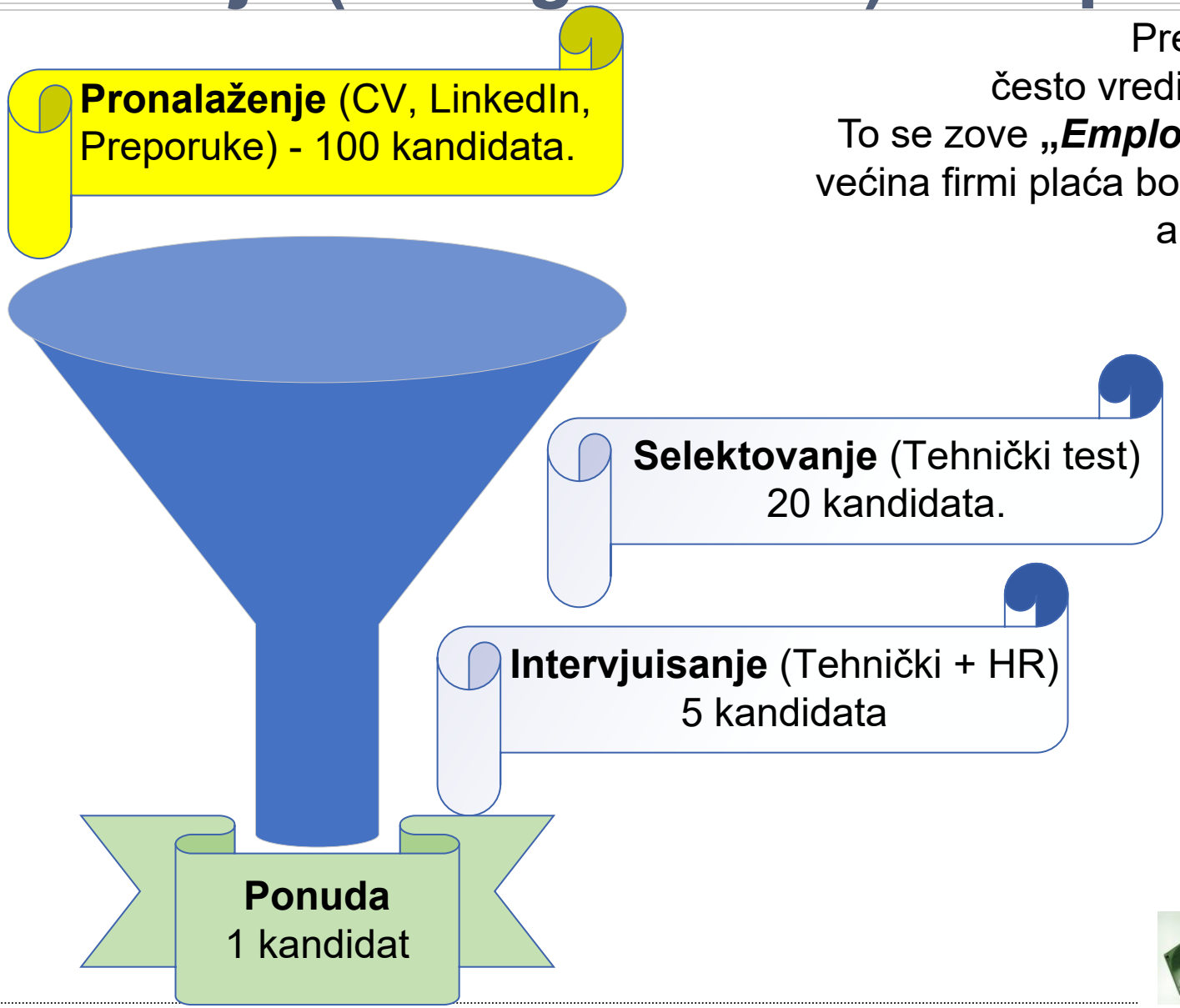
Uporedna analiza intervjuisanja

Vrsta intervjua	Fokus	Pitanje na koje intervju odgovara
Tehnički (koderski)	Sintaksa, algoritmi, logika	Zna li on/ona da programira?
Sistemski dizajn	Arhitektura, skalabilnost	Razume li on/ona kako softver radi u produkciji?
Bihejvioralni	Komunikacija, karakter	Želimo li da radimo sa ovom osobom 8 sati dnevno?

- Dobar menadžer nikada ne donosi odluku na osnovu samo jednog od ovih intervjua.
- Primer: genije za algoritme, ali toksična osoba koja uništava koheziju tima.
- **Balans između tehničke ekspertize i emocionalne/socijalne inteligencije je ono što čini idealnog kandidata za rad u grupi.**

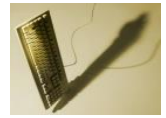


Levak selekcije (*Hiring Funnel*) u IT poslovima



Preporuka od bivšeg kolege
često vredi više nego deset testova.
To se zove ***„Employee Referral Program“*** i
većina firmi plaća bonuse svojim zaposlenima
ako dovedu dobrog kolegu.





Radno okruženje u kontekstu IT sektora (1)

- Udobnost i zdravlje
  - Programeri provode 8 do 12 sati ispred ekrana.
  - Loša ergonomija direktno vodi do hroničnih zdravstvenih problema koji izbacuju ljude iz stroja.
  - Šta je važno?
  - Udoban (a ne lep) nameštaj - Ergonomska stolica i **stolovi sa podesivom visinom** (*sit-stand desks*).
  - **Oprema:** Kvalitetna tastatura, miš i monitori sa filterima za plavu svetlost smanjuju zamor zglobova (Karpalni tunel) i očiju.



Radno okruženje u kontekstu IT sektora (2)

- Privatnost ili saradnja (akustični komfor)
  - Softversko inženjerstvo zahteva stanje dubokog rada (Deep Work).
  - Problem otvorenih (tzv. „Open-space“) kancelarija:
    Iako su popularne zbog komunikacije, one su pogubne za koncentraciju zbog buke.
  - Rešenje: Obezbeđivanje „tihih zona“ ili kabina za fokusiran rad, kao i kvalitetnih slušalica sa poništavanjem buke (*noise-cancelling*).
  - Privatnost u softverskom inženjerstvu znači „pravo na neometan misaoni proces“.



Radno okruženje u kontekstu IT sektora (3)

- Ambijentalni uslovi (osvetljenje i grejanje)
  - Osvetljenje: Najbolje je prirodno svetlo, ali bez odsjaja na monitoru. Veštačko osvetljenje mora biti indirektno i prilagodljivo.
  - Grejanje i ventilacija (HVAC): IT oprema (serveri, računari) generiše toplotu. Dobra klimatizacija i svež vazduh su neophodni da bi mozak funkcionisao efikasno; zagušljive prostorije drastično smanjuju kognitivne performanse.
- Sanitarne i prateće prostorije
  - Kuhinje i „chill“ zone: Mesta gde se ljudi sreću neformalno. Mnoga najbolja tehnička rešenja nastaju upravo ovde, u opuštenom razgovoru pored aparata za kafu.
  - Tuševi: Sve više IT kompanija ih ima kako bi podržale ljude koji dolaze biciklom na posao ili vežbaju tokom pauze, što doprinosi opštem zdravlju tima.
- Istraživanje

  (Peopleware: Productive Projects and Teams, Book by Timothy Lister and Tom DeMarc, 2013):

  Inženjeri koji rade u tišim i prostranijim kancelarijama bili su **2.6 puta produktivniji** od onih u bučnim i prenatrpanim prostorima.