



SVEUČILIŠTE U ZAGREBU



Fakultet  
elektrotehnike i  
računarstva

**Diplomski studij**

**Informacijska i  
komunikacijska tehnologija:**

Telekomunikacije i informatika

**Računarstvo:**

Programsko inženjerstvo i  
informacijski sustavi

Računarska znanost

# Raspodijeljeni sustavi

## 10. Vrednovanje nefunkcijskih obilježja raspodijeljenih sustava

Ak. god. 2020./2021.

# Creative Commons



- slobodno smijete:

- **dijeliti** — umnožavati, distribuirati i javnosti priopćavati djelo
- **prerađivati** djelo



- pod sljedećim uvjetima:

- **imenovanje:** morate priznati i označiti autorstvo djela na način kako je specificirao autor ili davatelj licence (ali ne način koji bi sugerirao da Vi ili Vaše korištenje njegova djela imate njegovu izravnu podršku).
- **nekomercijalno:** ovo djelo ne smijete koristiti u komercijalne svrhe.
- **dijeli pod istim uvjetima:** ako ovo djelo izmijenite, preoblikujete ili stvarate koristeći ga, preradu možete distribuirati samo pod licencom koja je ista ili slična ovoj.



*U slučaju daljnjeg korištenja ili distribuiranja morate drugima jasno dati do znanja licencne uvjete ovog djela.*

*Od svakog od gornjih uvjeta moguće je odstupiti, ako dobijete dopuštenje nositelja autorskog prava.*

*Ništa u ovoj licenci ne narušava ili ograničava autorova moralna prava.*

*Tekst licence preuzet je s <http://creativecommons.org/>*

# Sadržaj predavanja

- Životni ciklus sustava, nefunkcijske karakteristike sustava i važnost njihovog vrednovanja u praksi
- Postupci analize nefunkcijskih karakteristika
- Pouzdanost, raspoloživost i ukupna cijena vlasništva raspodijeljenog sustava
- Performance raspodijeljenog sustava
- Prirodne granice rasta ubrzanja i kapaciteta
- Modeliranje raspodijeljenih sustava mrežom repova

# Životni ciklus sustava

- Definicija zahtjeva na sustav
- Analiza funkcijskih obilježja (**ŠTO** sustav radi) i nefunkcijskih obilježja (**KAKO** sustav radi)
- Razvoj odabranog rješenja
- Ispitivanje
- Rad
- Mjerenja
- Modifikacija (ako zahtjevi nisu ispunjeni)



# Najvažniji nefunkcijski zahtjevi

- Performance
  - propusnost (broj zahtjeva/sekunda) i kašnjenje
- Skalabilnost
  - mogućnost povećanja ulaznih zahtjeva uz iste performance ili uz proširenje arhitekture
  - vrste:
    - Vertikalna – povećanje HW-a
    - Horizontalna – dodavanje elemenata
- Raspoloživost (*availability*)
  - postotak vremena kada sustav radi
- Pouzdanost (*reliability*)
  - Vjerojatnost da će sustav ispravno raditi u nekom vremenskom periodu
- Kapacitet
  - Isporuka funkcionalnosti za zadani ulazni teret (broj zahtjeva/sekunda)
- Sigurnost
  - zaštita sustava od malicioznih napada
- Održivost (*maintainability*)
  - jednostavnost nadogradnje sustava
- Upravljivost (*manageability*)
  - Jednostavno praćenje sustava dok radi s ciljem traženja pogrešaka i analize rada sustava
- Proširivost (*extensibility*)
  - Jednostavnost dodavanja novih funkcionalnosti
- Obnovljivost (*recovery*)
  - Mogućnost obnavljanja funkcionalnosti sustava nakon ispada
- Interoperabilnost
  - Mogućnost razmjene informacija s drugim sustavima
- Uporabljivost (*usability*)
  - Jednostavnost korištenja

# Nefunkcijske karakteristike sustava

- Nefunkcijske karakteristike sustava skupno se nazivaju **kvaliteta usluge** (QoS – *Quality of Service*)
- Definicija jamčene kvalitete usluge naziva se **ugovor o razini usluge** (SLA – *Service Level Agreement*)
  - SLA je dio ugovora između davatelja i korisnika usluga
  - SLA je sve češće je i dio opisa posla internog ICT-odjela
- Tri važne kategorije kvalitete usluge:
  - Performance (*performance*): vrijeme odziva, propusnost, kapacitet
  - Pouzdanost/raspoloživost (*reliability/availability*)
  - Ukupni trošak vlasništva (TCO – *Total Cost of Ownership*)

# Primjer: web-aplikacija za trgovinu

## Organizacija prodaje putem Interneta

Aplikacija za prodaju ima sljedeće značajke:

- Neuspješni posjeti zbog loše kvalitete usluge
  - **60 %** kupaca napušta web-stranicu aplikacije ako je odziv aplikacije **između 4 i 6 sekundi**
  - **95 %** kupaca napušta web-stranicu aplikacije ako je odziv aplikacije **veći od 6 sekundi**
- Uspješni posjeti s ostvarenom prodajom
  - **5 %** kupaca od svih koji su posjetili web-stranicu aplikacije kupi proizvode za **prosječnu cijenu 1200 kn**

# Primjer: analiza značajki web-aplikacije (1)

- Projektiranje i održavanje web-aplikacije ostvaruje se u skladu s očekivanim brojem i porastom broja korisnika
- Ako se broj posjeta web-aplikaciji – promet poveća za 30%, 60% ili 90%:
  - Hoće li odziv aplikacije biti zadovoljavajući?
  - U kojim će uvjetima odziv aplikacije preći u nezadovoljavajuće područje?
  - Koliki gubitak prihoda uzrokuje gubitak kupaca zbog slabog odziva aplikacije?
  - Koja ulaganja su potrebna da se uz povećanje prometa zadrži sav posao?
  - Kada će se, uz trenutačni trend, potreba za kapacitetom udvostručiti?



## Primjer: analiza značajki web-aplikacije (2)

	Povećanje broja korisnika			
	Danas	+30 %	+60 %	+90 %
Maks. posjeta/sat	900.00	1,170.00	1,440.00	1,710.00
Vrijeme odziva (s)	2.96	3.80	5.31	8.83
Izgubljeni kupci (%)	0.00	0.00	60.00	95.00
Mogući broj prodaja / sat (kn)	45.00	58.50	72.00	85.50
Mogući prihod / sat (kn)	54,000.00	70,200.00	86,400.00	102,600.00
Stvarni prihod / sat (kn)	54,000.00	70,200.00	34,560.00	5,130.00
Izgubljeni prihod / sat (kn)	0.00	0.00	51,840.00	97,470.00

Poduzeće će izgubiti više od 95% mogućeg prometa na Internetu, ako se broj potencijalnih kupaca udvostruči!

# Primjer: analiza značajki web-aplikacije (3)

- Na temelju prikazanih rezultata može se zaključiti da:
  - Pogrešno projektirana aplikacija može imati katastrofalne posljedice na poslovanje
- Upozorenje:
  - linearna ekstrapolacija najčešće ne daje dovoljno točne rezultate
  - što ako veći broj zahtjeva odstupa od statističke srednje vrijednosti: veći udjel zahtjeva na repu raspodjele vjerojatnosti nego kod normalne raspodjele → dugi rep (*long tail*)
- Na ovom predmetu naučit ćete kako pristupiti vrednovanju performanci raspodijeljenih sustava i planiranju rasta kapaciteta sustava

# Postupci analize nefunkcijskih karakteristika

- **Intuicija i iskustvo**

- Raspodijeljeni sustavi pokazuju izrazito nelinearno ponašanje pa su procjene vrlo teške

- **Modeliranje**

- Predočavanje sustava matematičkim modelom
- Podrazumijeva razvoj modela koji opisuje ovisnost performanci o pojedinim parametrima sustava

- **Simulacija**

- Postupak kojim se oponaša rad sustava
- Najtočnija metoda, često preskupa za upotrebu

# Razvoj modela raspodijeljenog sustava

- Razumijevanje funkcioniranja sustava
- Modeliranje tereta – opterećenja sustava
- **Mjerenja u radu sustava radi utvrđivanja parametara tereta – ovo ćemo vidjeti**
- Razvoj modela performanci
- Verifikacija i validacija modela performanci
  - Verifikacija: provjera ispravnosti (radi li ispravno?)
  - Validacija: provjera valjanosti (radi li ono što se očekuje?)
- Analiza mogućih scenarija promjena u budućnosti
- Procjena promjena tereta u budućnosti
- Predviđanje performanci sustava nakon puštanja u rad te u budućnosti

# Modeli tereta sustava

## Prirodni modeli tereta

- stvarne aplikacije

## Umjetni modeli tereta

- umjetne aplikacije – ogledna vrijednost (*benchmark*)
  - *Standard Performance Evaluation Corporation* (SPEC) [www.spec.org](http://www.spec.org)
  - *Transaction Processing Performance Council* (TPC-C) [www.tpc.org/tpcc](http://www.tpc.org/tpcc)

## Neizvodivi modeli tereta

- uslužni zahtjevi, intenzitet nailaska/dolaska zahtjeva, razina konkurentnog izvođenja, ...

# Primjer: utvrđivanje značajki realnog tereta (1)

- Srednja vrijednost parametara ne mora biti reprezentativna ako se pojedinačne vrijednosti nalaze u grupama koje se značajno razlikuju po vrijednostima
- U takvom slučaju potrebno je provesti grupiranje i utvrditi značajke za svaku grupu
- Postoje različite metode grupiranja i programi koji obavljaju grupiranje, a najčešće se koristi grupiranje na temelju Euklidske udaljenosti:

$$d = \sqrt{\sum_{n=1}^K (x_{in} - x_{jn})^2}$$

$K$  = broj parametara

# Primjer: utvrđivanje značajki realnog tereta (2)

- Teret web-poslužitelja sastoji se od sljedećih grupa zahtjeva:

Dokument	Veličina (KB)	Broj pristupa
1	12	281
2	150	28
3	5	293
4	25	123
5	7	259
6	4	241
7	35	75

- Grupiraj teret u tri odvojene grupe

# Primjer: utvrđivanje značajki realnog tereta (3)

- Budući da su vrijednosti jako različite treba prvo obaviti promjenu mjerila. U ovom slučaju koristimo  $\log_{10}$ .

Dokument	Veličina (KB)	Broj pristupa
1	1.08	2.45
2	2.18	1.45
3	0.70	2.47
4	1.40	2.09
5	0.85	2.41
6	0.60	2.38
7	1.54	1.88



# Primjer: utvrđivanje značajki realnog tereta (4)

- Podrazumijevajući da je težište grupe od jedne točke ta točka, izračunavamo Euklidske udaljenosti između težišta:

Grupa	G1	G2	G3	G4	G5	G6	G7
G1	0	1.49	0.38	0.48	0.24	0.48	0.74
G2		0	1.79	1.01	1.01	1.64	1.83
G3			0	0.79	0.16	0.13	1.03
G4				0	0.64	0.85	0.26
G5					0	0.25	0.88
G6						0	1.07
G7							0

- Budući da je udaljenost između G3 i G6 najmanja, svrstavamo ih u novu grupu G36 i računamo joj težište:

$$(0.7+0.6)/2 = 0,65 \quad \text{i} \quad (2.47+2.38)/2 = 2,43$$

# Primjer: utvrđivanje značajki realnog tereta (5)

- Sad računamo Euklidske udaljenosti između novih težišta:

Grupa	G1	G2	G36	G4	G5	G7
G1	0	1.49	0.43	0.48	0.24	0.74
G2		0	1.81	1.01	1.64	0.76
G36			0	0.82	0.19	1.05
G4				0	0.64	0.26
G5					0	0.88
G7						0

- Budući da je udaljenost između G36 i G5 najmanja izračunamo težište nove grupe G365:

$$(0.65+0.85)/2 = 0.75 \quad \text{i} \quad (2.43+2.41)/2 = 2.42$$

# Primjer: utvrđivanje značajki realnog tereta (6)

- Postupak ponavljamo dok ne dobijemo željene tri grupe:

Grupa	G1365	G2	G47
G1365	0	1.60	0.72
G2		0	0.89
G47			0

- Nakon vraćanja parametara u izvorno mjerilo dobivamo novi model tereta:

Grupa	Tip dokumenta	Velicina (KB)	Broj zahtjeva
G1356	Mali	8.19	271.51
G47	Srednji	20.58	96.05
G2	Veliki	150.0	28

- Ovakvo grupiranje daje puno realističnije postavke za modeliranje sustava

# Najčešće pogreške kod modeliranja

- Presložena analiza problema
- Nije definiran specifični cilj
- Prejudiciranje rezultata
  - model treba dokazati da je “naše” rješenje bolje od “njihovog”!
- Nedovoljno razumijevanje sustava
- Neadekvatne mjere performanci
- Nereprezentativni teret
- Neuključivanje važnih parametara
- Promatranje u krivom intervalu vrijednosti parametara
- Pogrešno rukovanje ekstremnim vrijednostima
- Nedovoljno promatranje evolucije sustava i tereta
- Kriva interpretacija rezultata

# Pouzdanost sustava

## Pouzdanost:

- Vjerojatnost da sustav funkcionira u definiranom vremenskom intervalu  $T$  pod danim uvjetima okružja

U radu sustava događaju se kvarovi koji se moraju otkloniti, što se opisuje parametrima:

- Srednje vrijeme između kvarova (MTBF – *Mean Time Between Failure* )
- Srednje vrijeme popravka (MTTR – *Mean Time To Repair*)

# Raspoloživost sustava

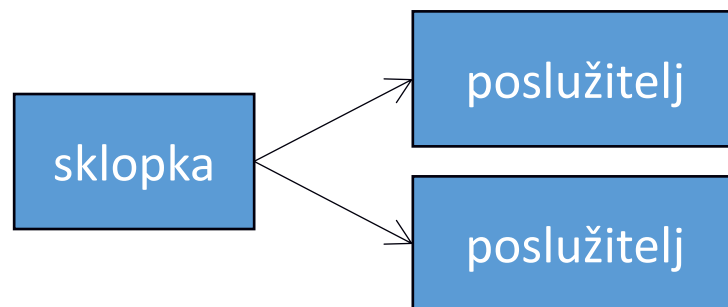
## Raspoloživost:

- Vjerojatnost da će sustav funkcionirati u trenutku  $t$   
(ili: postotak ukupnog vremena koje je sustav u radu)

Računanje raspoloživosti:  $MTBF/(MTBF + MTTR)$

- Raspoloživost od 0,9999 znači da će sustav biti izvan funkcije:  $(1 - 0,9999) \times 30 \times 24 \times 60 = 4,32$  min/mjesec
- Raspoloživost serijske kombinacije dva podsustava jednaka je umnošku raspoloživosti pojedinih podsustava:  
 $R_s = R_1 \times R_2$
- Raspoloživost paralelne kombinacije dva podsustava jednaka je:  
 $R_p = R_1(1 - R_2) + R_2(1 - R_1) + R_1R_2$   
(pretpostavka: jedan raspoloživi podsustav dovoljan za ispravno funkcioniranje sustava)

# Primjer: raspoloživost web-poslužitelja



Izračunajte raspoloživost sustava koji uključuje dva paralelna web-poslužitelja s raspoloživošću od 0,99 te jedne mrežne sklopke za raspodjelu tereta između poslužitelja s MTBF = 1 godina i MTTR = 2 sata.

- Raspoloživost sklopke:  $(365 \times 24) / (365 \times 24 + 2) = 0,9998$
- Raspoloživost 2 paralelna poslužitelja:  
 $0,0099 + 0,0099 + 0,9801 = 0.9999$
- Raspoloživost sustava:  $0,9998 \times 0,9999 = 0,9997$

# Ukupni trošak vlasništva (1)

## Kapitalni trošak (CAPEX – *Capital Expenditure*)

- trošak nabavke i izvedbe sustava (oprema)
- trošak amortizacije – zavisi o propisanom vremenu trajanja
  - za računalne sustave 3 godine (33,33% nabavne cijene godišnje)

## Operativni trošak (OPEX – *Operating Expenditure*)

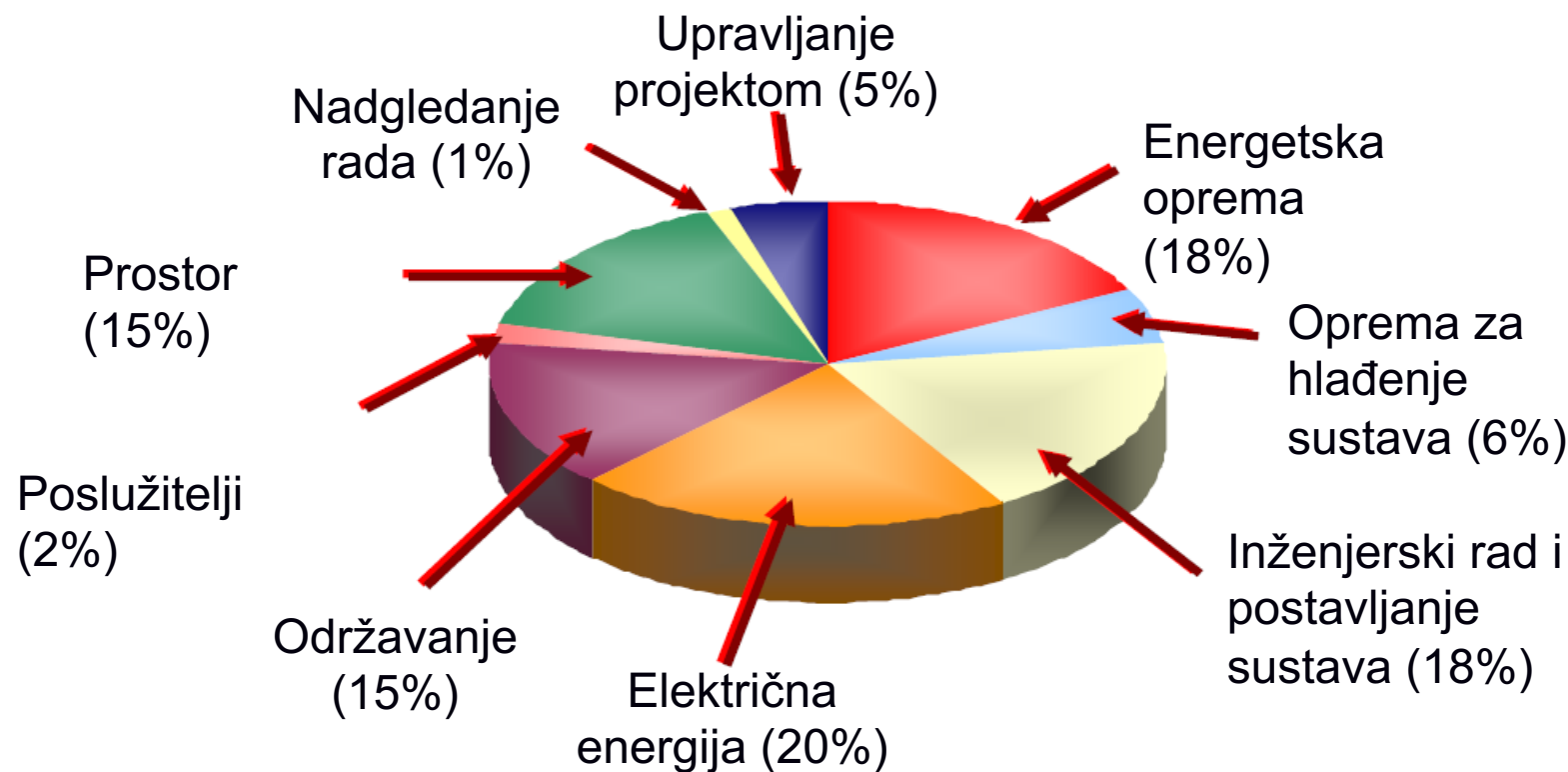
- trošak rada/pogona sustava:
  - zaposlenici (ICT odjel)
  - prostor i infrastruktura
  - režije: energija, komunikacijske usluge, fizička sigurnost, ...



# Ukupni trošak vlasništva (2)

## Prosječni trošak uporabe poslužiteljskog sustava:

- CAPEX/OPEX = 50/50, kroz tri godine korištenja, ovisno o „geografiji” i godini



Izvor: Intel (internetska anketa 2008.)

# Mogući načini ostvarenja sustava

## Izgradnja vlastitog poslužitelja

## Iznajmljivanje poslužitelja

- Udomljivanje sustava (*hosting*)
  - Udomitelj: pruža i upravlja fizičkom infrastrukturom (zgrada, napajanje, pristup Internetu, poslužitelji)
  - Zakupnik: postavlja i upravlja sredstvima koja se poslužuju
  - Primjer: *web-hosting*

## Računalni oblak (*computing cloud*)

- Infrastruktura kao usluga (*Infrastructure as a Service, IaaS*)
- Platforma kao usluga (*Platform as a Service, PaaS*)
- Softver kao usluga (*Software as a Service, SaaS*)

# Performance sustava

## Vrijeme odziva sustava (*response time*)

- Vrijeme potrebno da sustav odgovori na zahtjev za posluživanjem (npr. vrijeme između pritiska na miša i pojave web-stranice na zaslonu)

## Propusnost sustava (*throughput*)

- Broj posluženih zahtjeva u jedinici vremena (različito od  $1/\text{vrijeme odziva}$ , jer se zahtjevi mogu posluživati paralelno)
  - Propusnost je funkcija tereta i kapaciteta sustava

## Kapacitet sustava (*capacity*) = maksimalna moguća propusnost sustava

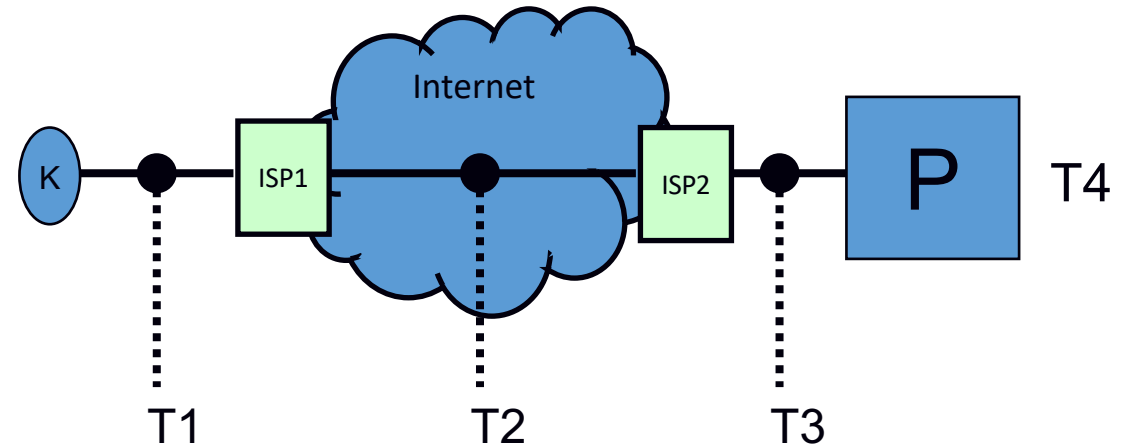
- Sustav je zauzet 100% vremena

# Vrijeme odziva u raspodijeljenom sustavu

## Vrijeme odziva

- Vrijeme u mreži ( $T1 + T2 + T3$ )

- Kašnjenje
- Vrijeme prijenosa




- Vrijeme na poslužitelju (T4)

- Vrijeme posluživanja (CPU, disk, LAN, ...)
- Vrijeme čekanja na resurse (CPU, disk, LAN, baza podataka, ...)

# Primjer: dobavljanje web-stranice <https://www.websitepulse.com>

24/7 Live chat24/7 1-888-WSPULSE

LoginSign Up

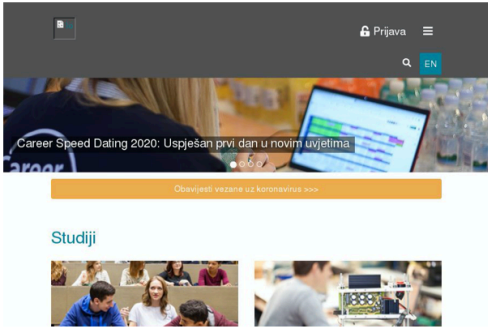
WebsitePulse™  
take IT easy

[Perform New Test](#)[Start a FREE Trial](#)

www.fer.unizg.hr




Test ResultsRecommendations

Test performed from Munich, Germany on December 12, 2020 at 11:16



initial load screenshotreturn load screenshot

performance grade <b>79</b> / 100	first visit <b>9.39</b> <b>7.86</b> <b>4 MB</b> load timeDOM readypage size
requests <b>70</b>	return visit <b>3.43</b> <b>3.10</b> <b>476 KB</b> load timeDOM readypage size

Share your resultsget a permalinkdownload report data

Performance grade 79 out of 100. Your website's performance grade is very good.  
[See our recommendations on how to improve performance](#) →

# Propusnost raspodijeljenog sustava

## Broj posluženih zahtjeva u jedinici vremena (Z/S)

- „Zahtjev” ovisi o razini sustava na kojoj se promatra
- Primjeri tipičnih zahtjeva i jedinica za propusnost:
  - Broj transakcija u sekundi
  - Broj pretinaca u sekundi
  - Broj web-stranica u sekundi
  - Broj poruka u sekundi
  - Broj paketa u sekundi
  - Broj instrukcija u sekundi
  - ...

Propusnost



# Primjer: izračun propusnosti

Kolika je maksimalna propusnost tj. kapacitet diska u sustavu za *on-line* transakcije?

- U/I operacija diska traje prosječno 10 ms
- Iskorištenje diska je 100% (tj. stalno zauzet)
- Maksimalna propusnost tj. kapacitet diska:  
 $1/0.01 = 100$  operacija u sekundi

Napomena: Kapacitet se određuje kod 100% zaposlenosti!

# Ostvarivanje razmjernog rasta (skaliranje)

## Osnovni modeli ostvarivanja razmjernog rasta kapaciteta sustava

- **Vertikalno skaliranje** podrazumijeva prijelaz na poslužitelja s većim kapacitetom
- **Horizontalno skaliranje** podrazumijeva dodavanje poslužitelja (obično istog kapaciteta).
- **Skaliranje prema gore** (većem kapacitetu) je jednako važno kao i **skaliranje prema dolje** (manjem kapacitetu) zbog potrebe prilagodbe troškova prihodima!



# Poboljšanje performanci

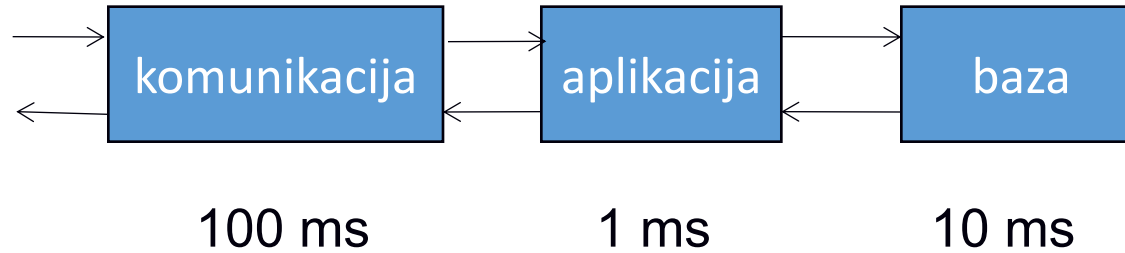
## Arhitekturne promjene za poboljšanje performanci:

- Serijsko preklapanje (*pipeline*)
- Paralelno preklapanje
- Paralelna obrada
- Privremena pohrana (*cache*)

## Primjer: web-poslužitelj s 3 modula

- Mrežni modul s odzivom od 100 ms
- Modul aplikacije s odzivom od 1 ms
- Modul baze podataka s odzivom od 10 ms
- Napomena: deterministički pristup, bez razmatranja raspodjele nailaska zahtjeva, vremena posluživanja i čekanja te interakcije paralelnih aktivnosti

# Serijsko preklapanje



- Svaki 100 ms prihvaća se novi zahtjev, dok prethodni još nije obrađen i završen
- Serijskim preklapanjem ne smanjuje se vrijeme odziva, a povećava se propusnost:

$$T_{\text{odziva}} = 100 \text{ ms} + 1 \text{ ms} + 10 \text{ ms} = 111 \text{ ms}$$

$$\text{Propusnost} = 1/0,1\text{s} = 10/\text{s}$$

# Mjerenje odziva web-aplikacije

- alat [Apache JMeter](#)
- za testiranje performanci statičkih i dinamičkih resursa
- podržani protokoli:
  - Web - HTTP, HTTPS (Java, NodeJS, PHP, ASP.NET, ...)
  - SOAP / REST Webservices
  - FTP
  - Database via JDBC
  - LDAP
  - Message-oriented middleware (MOM) via JMS
  - Mail - SMTP(S), POP3(S) and IMAP(S)
  - Native commands or shell scripts
  - TCP



# Web-aplikacija u Spring Bootu u Javi

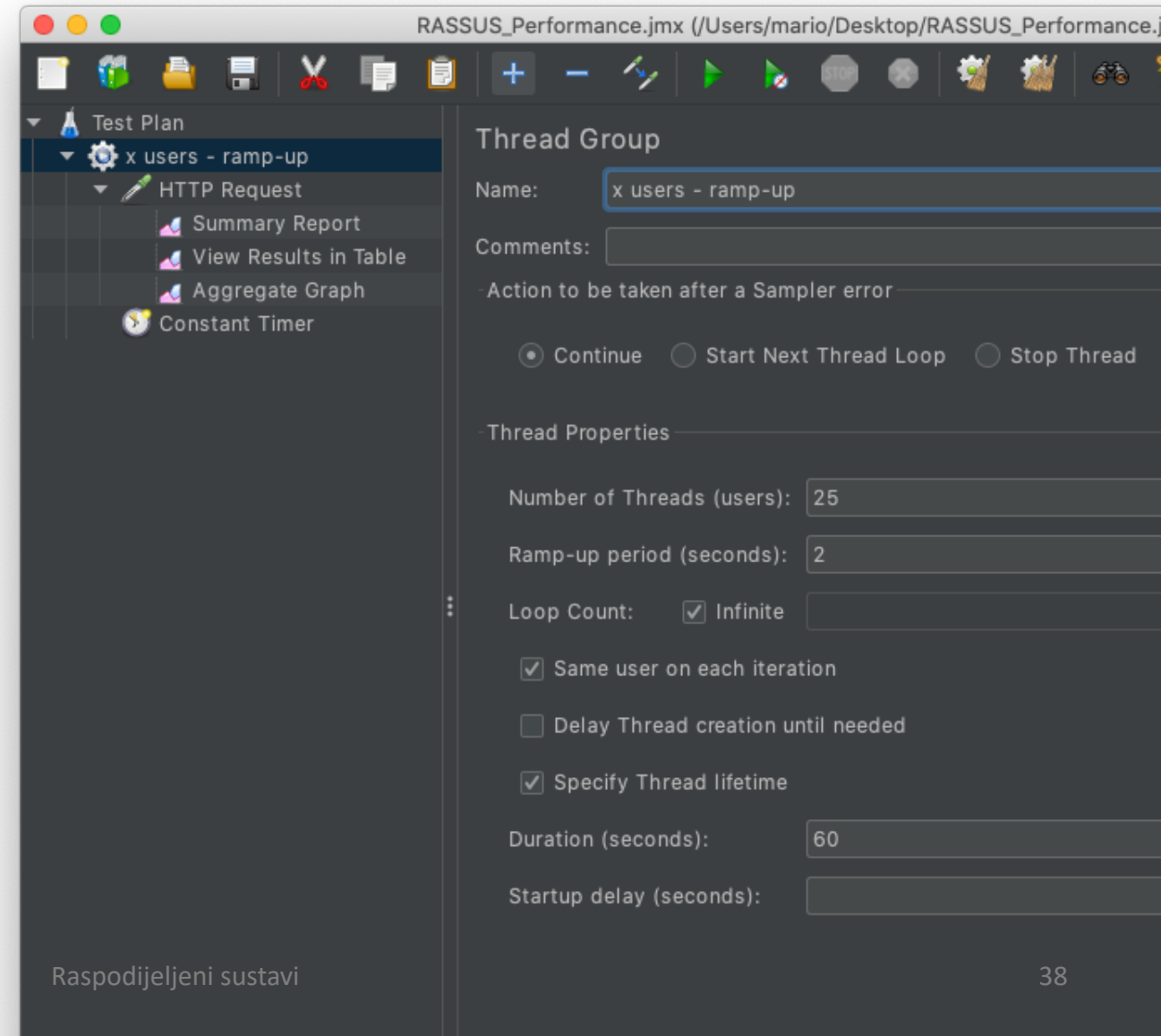
- Problemi kod mjerenja:
  - Koliko paralelnih niti/dretvi koristi?
  - Koliko konekcija ima prema bazi podataka?
  - Spremanje podataka iz baze u privremenu memoriju?
    - Koliko često se to događa (vjerojatnost)?
  - JVM radi optimizacije za vrijeme rada. Kako anulirati taj dio?
  - Kako mjeriti pojedine komponente (mrežni dio, izvršavanje, dohvaćanje iz baze)?

# Web-aplikacija u Spring Bootu u Javi

- Koliko paralelnih niti/dretvi koristi?
  - Postaviti konfiguraciju: `server.tomcat.max-threads`
- Koliko konekcija ima prema bazi podataka?
  - Postaviti konfiguraciju (v 2.): `spring.datasource.hikari.maximum-pool-size`
- Spremanje podataka iz baze u privremenu memoriju?
  - Koliko često se to događa?
    - Unutar jedne sjednice nije moguće isključiti, ali se može napraviti analiza koda i na osnovu toga izračunati vjerojatnost
- JVM radi optimizacije za vrijeme rada. Kako anulirati taj dio?
  - Treba „zagrijati” JVM da ne radi prevođenje *byte* koda (cca 8-10 tisuća zahtjeva) tj. prije mjerenja poslati zahtjeve na poslužitelj, a nakon toga raditi mjerenja
- Kako mjeriti pojedine komponente (mrežni dio, izvršavanje, dohvaćanje iz baze)?
  - JRF (Java Flight Control) - <2% utjecaja na izvođenje aplikacije
  - Koristiti Aspektno orijentirano programiranje (AOP) za ubacivanje kontrolnih točki

# JMeter test

- Korisnik je jedna nit/dretva
- Šalje se jedan GET zahtjev
  - Dohvaća podatke iz baze i računa
- Pauza između zahtjeva 100ms
- Test traje 60s
- Mijenjamo broj korisnika 25-300



# JMeter – rezultati

Aggregate Graph

Name:

Aggregate Graph

Comments:

Write results to file / Read from file

Filename

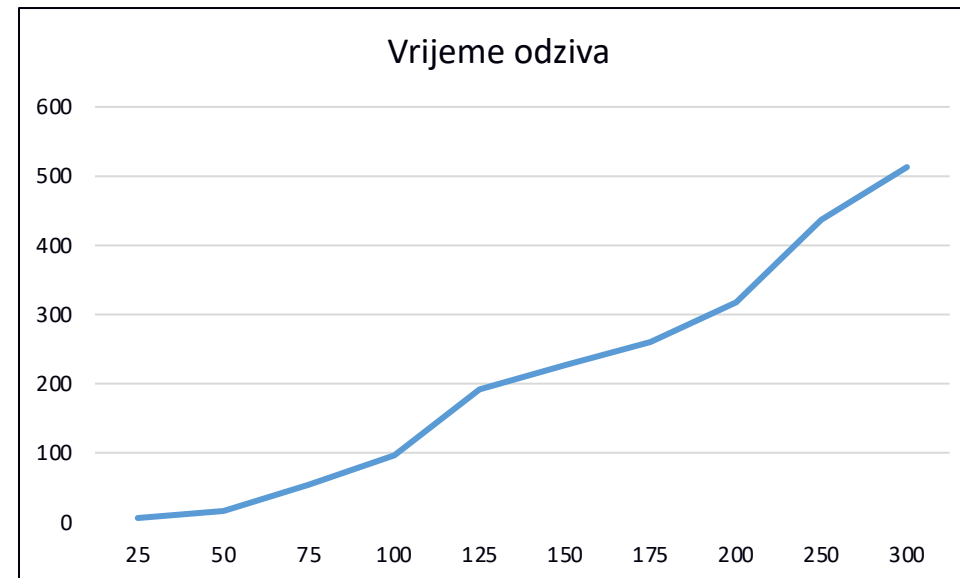
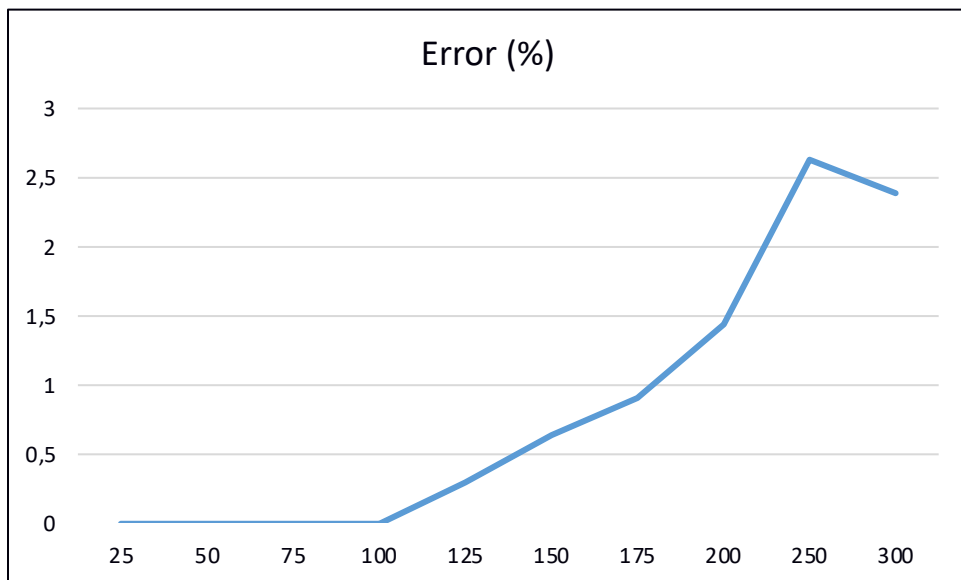
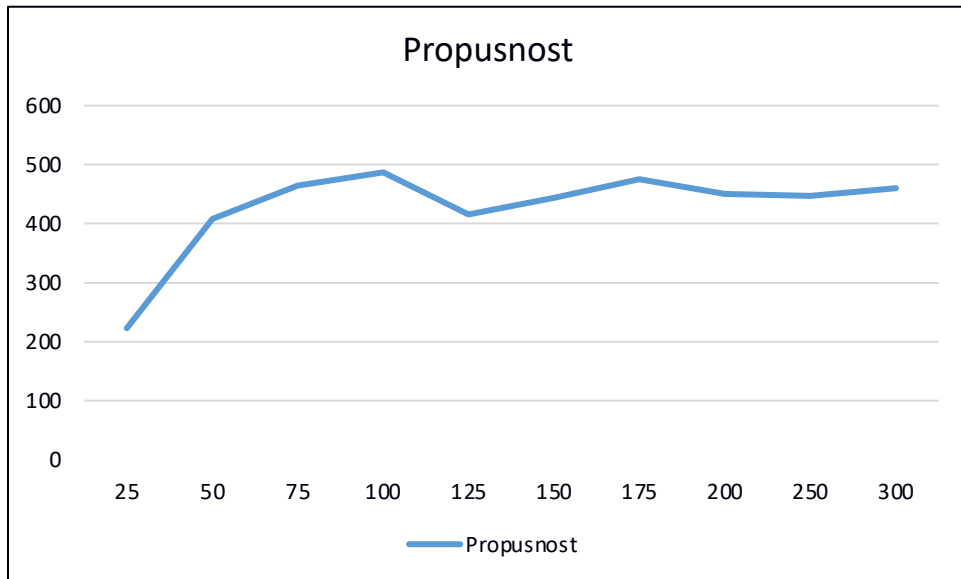
Browse...

Log/Display Only: ☐ Errors ☐ Successes

Configure

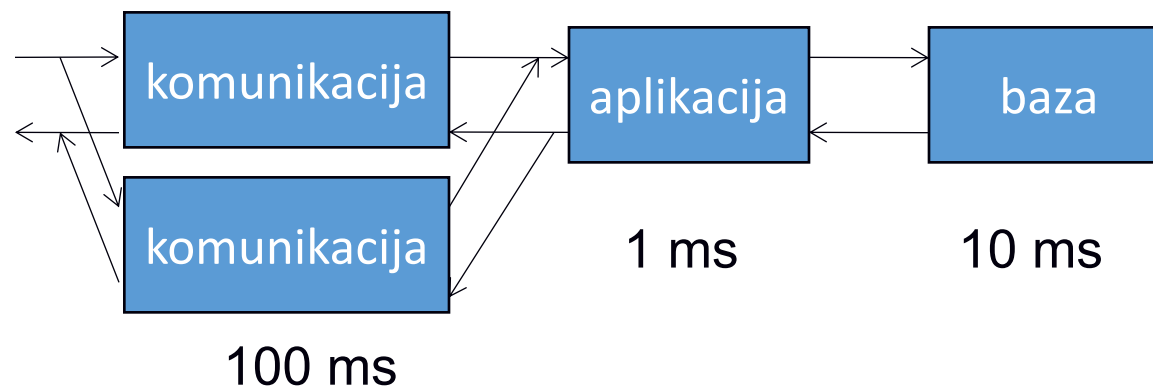
Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Maximum	Error %	Throughput	Received KB/sec	Sent KB/sec
HTTP Request	13344	6	4	6	8	26	0	791	0.00%	222.9/sec	161.07	27.21
TOTAL	13344	6	4	6	8	26	0	791	0.00%	222.9/sec	161.07	27.21

Korisnici	25	50	75	100	125	150	175	200	250	300
Intenzitet dolazaka	250	500	750	1000	1250	1500	1750	2000	2500	3000
Propusnost	223	408	464	487	416	444	475	450	447	460
Error (%)	0	0,00	0,00	0,00	0,30	0,64	0,91	1,44	2,63	2,39
Vrijeme odziva	6	16	54	97	192	227	260	318	437	513





# Paralelno preklapanje

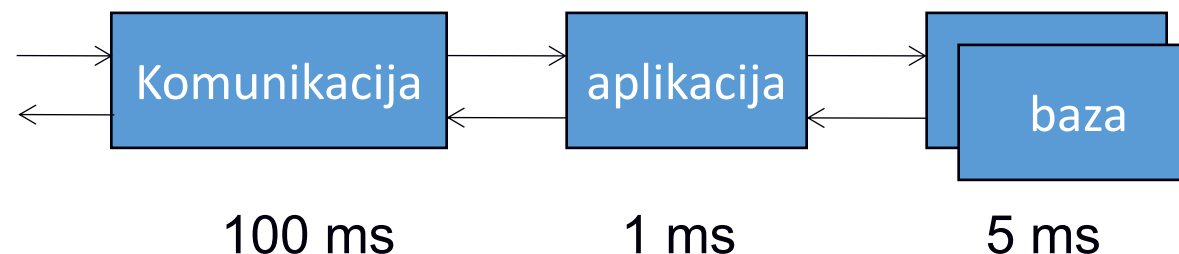


- Istodobno (paralelno) se mogu prihvatiti 2 zahtjeva
- Paralelnim preklapanjem komunikacije ne smanjuje se vrijeme odziva, a povećava se propusnost:

$$T_{\text{odziva}} = 100 \text{ ms} + 1 \text{ ms} + 10 \text{ ms} = 111 \text{ ms}$$

$$\text{Propusnost} = 2/0,1\text{s} = 20/\text{s}$$

# Paralelna obrada



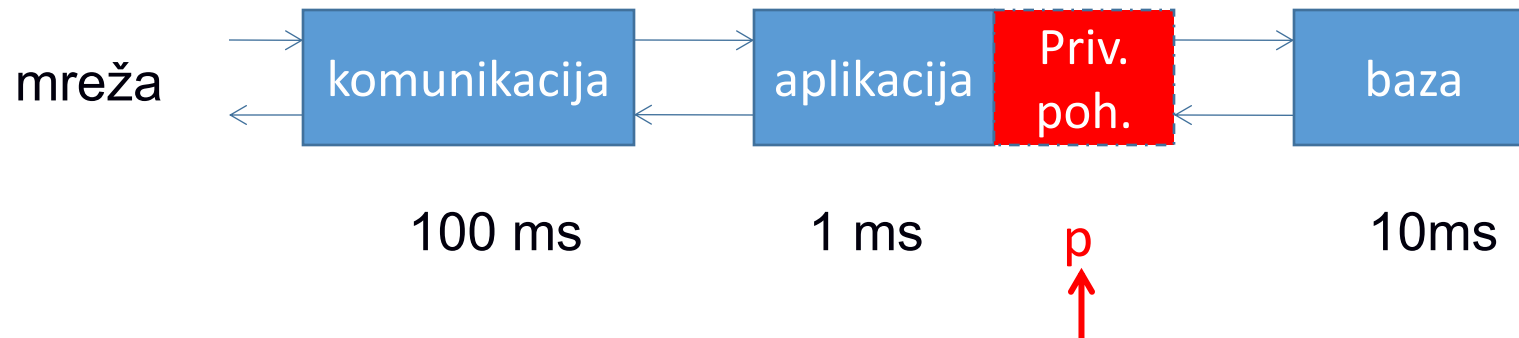
- Zapisi istog pretnica raspodijeljeni su na dva fizička diska pa se dohvaćaju paralelno
- Paralelnom obradom ne utječe se na propusnost, ali smanjuje se vrijeme odziva:

$$\text{Propusnost} = 1/0,1\text{s} = 10/\text{s}$$

$$T_{\text{odziva}} = 106 \text{ ms}$$

Ubrzanje!

# Privremena pohrana



vjerojatnost da se pretinac ne nalazi  
u privremenom spremniku

- Korišteni pretinac se pohranjuje u privremeni spremnik
- Privremena pohrana ne utječe na propusnost, ali smanjuje vrijeme odziva:

$$\text{Propusnost} = 1/0,1\text{s} = 10/\text{s}$$

$$T_{\text{odziva}} = 101 + 10(1 - p) \text{ ms}$$

Ubrzanje!

# Ubrzanje obrade zahtjeva

## Ubrzanje (*speedup*):

- Skraćenje vremena odziva, može se postići paralelnom obradom zahtjeva
- Izražava se kao omjer vremena odziva jednog podsustava ( $T_1$ ) i vremena odziva  $p$  paralelnih podsustava ( $T_p$ ) uz jednaki teret:

$$S(p) = T_1/T_p$$

- Teorijski model ubrzanja: Amdahlov zakon

# Skaliranje donosi nove izazove...

- Raspoređivanje opterećenja u sustavu
  - Osigurava jednakomjerno opterećenje paralelnih podsustava – uravnoteženje opterećenja
- Raspoređivanje podataka
  - Osiguravanje koherencije (replikacija istovrsnih podataka)
  - Osiguravanje podjele (federacija odvojenih podataka )
- Protokoli za sinkronizaciju rada grupe
  - Osiguravanje vremenskog slijeda
  - Zajamčena dostava

# ...ali i nove mogućnosti

## Visoka raspoloživost sustava (*high availability*)

- Izvedba putem neosjetljivosti na pogreške (*fault tolerance*)
  - Neosjetljivost na pogreške omogućuje ostvarivanje dostupnosti web-aplikacija uz prisutnost pogrešaka u podsustavima
- Budući da povećanje kapaciteta obično zahtjeva ostvarenje replikacije, replikacija se može iskoristiti za ostvarivanje neosjetljivosti na pogreške
- Visoki stupanj eliminacije pogrešaka je veoma skup, često preskup

# Procjena potrebnog kapaciteta sustava

- Ako se kapacitet sustava  $C$  mjeri u pravilnim intervalima, dugoročna potreba za kapacitetom se može procijeniti podrazumijevajući model eksponencijalnog trenda:

$$C_{\text{budući}} = C_{\text{sadašnji}} \times e^{(LT)}$$

$L$  - trend rasta

$T$  - vrijeme kroz koje se trend aproksimira

- Vrijeme do udvostručenja kapaciteta može se izračunati:
  - $T_{\text{dvostruko}} = (\ln 2)/L$

# Literatura, ...

## Sadržaj ovog predavanja nastao je na temelju:

- N. J. Gunther: "**The practical performance analyst**", *Mcgraw Hill i Authors Choice Press*, 1998 i 2000. (poglavlja 2 i 3)
- D. A. Menasce, V.A.F. Almeida: "**Capacity planning for web services**", *Prentice Hall*, 2002 (poglavlja 1 i 5)
  - D. A. Menasce, V.A.F.Almeida, „Capacity Planning: An Essential Tool for Managing Web Services“, *IT professional*, Vol. 4, No. 4, 2002., pp. 33-38.”
- D. F. Vrsalovic, et. al: "**Performance prediction and calibration for a class of multiprocessors**", *IEEE Transactions on Computers*, Volume: 37 Issue: 11 , Nov. 1988, pp. 1353 -1365



## ... dodatno za one koji žele dalje istraživati

- A. O. Allen: "**Probability, Statistic, and Queueing Theory with Computer Science Applications**", Academic Press 1978.
- S. Joines, R. Willenborg, K. Hygh: "**Performance analysis for Java Web Sites**", Addison Wesley, 2003
- S. Sounders: "**High Performance Web Sites**", O'Reilly, 2007.
- T. Schlosssnagle: "**Scalable Internet Architectures**", Sams Publishing, 2007.
- D. A. Menasce, V.A.F. Almeida, L.W. Dowdy: "**Performance by Design**", Prentice Hall, 2004

# Rekapitulacija (1)

- Koji su elementi životnog ciklusa raspodijeljenog sustava?
  - Definicija zahtjeva, analiza rješenja, sinteza, ispitivanje, rad, mjerenja i modifikacija zahtjeva
- Koji su najvažniji nefunkcijski zahtjevi?
  - Performance, raspoloživost i ukupna cijena vlasništva
- Kako se oni skupno zovu?
  - Kvaliteta usluge (QoS)
- Što je to ugovor o razini usluge (SLA)?
  - Ugovor između korisnika i davatelja usluge koji definira razinu usluge
- Zašto je vrednovanje performansi važno?
  - Zbog planiranja kapaciteta koji je potreban za uspješan rad
- Koje metode analize se koriste u praksi?
  - Iskustvo, modeliranje i simulacija
- Kakve vrste modela tereta postoje u praksi?
  - Prirodne aplikacije, umjetne aplikacije (*benchmarks*) i neizvodivi modeli opisani intenzitetom zahtjeva, prosječnim vremenom obrade i sl.

## Rekapitulacija (2)

- Koji su koraci pri razvoju modela sustava?
  - Razumijevanje funkcioniranja, modeliranje tereta, mjerenje sustava u pogonu radi utvrđivanja parametara tereta, razvoj modela, verifikacija i validacija, analiza mogućih scenarija promjena, prognoza promjena tereta u budućnosti, prognoza performansi sustava nakon puštanja u pogon te u budućnosti
- Koje su najčešće greške kod modeliranja?
  - Presložena analiza, nema specifičnog cilja, prejudiciranje, nedovoljno razumijevanje sustava, neadekvatne mjere nereprezentativni teret, neuključivanje važnih parametara, promatranje u krivom intervalu vrijednosti parametara, krivo baratanje ekstremima, nedovoljno promatranje evolucije sustava i tereta, kriva interpretacija rezultata
- Što definiraju pojmovi MTBF i MTTR?
  - Srednje vrijeme između pogrešaka i srednje vrijeme popravka

## Rekapitulacija (3)

- Kako je definirana raspoloživost sustava?

$$D = \text{MTBF} / (\text{MTBF} + \text{MTTR})$$

Postotak vremena u kojemu je sustav na raspolaganju korisnicima

- Kako se računa raspoloživost paralelno ( $D_p$ ) i serijskih ( $D_s$ ) povezanih sustava?

$$D_p = (1 - D_1) * D_2 + (1 - D_2) * D_1 + D_1 D_2$$

$$D_s = D_1 * D_2$$

- Koje su dvije osnovne grupe troškova za gradnju i pogon web-sustava i kako su obično raspodijeljeni?

Kapitalni i operativni troškovi, grubo raspodijeljeni 50/50 kroz tri godine.

- Objasnite pojam vremena odziva i kapaciteta

Vrijeme odziva određuje odziv na jedan zahtjev

Kapacitet odgovara maksimalnom broju zahtjeva koji se mogu obraditi u jedinici vremena

## Rekapitulacija (4)

- Koje se metode za poboljšanje performansi koriste u arhitekturi raspodijeljenih sustava?  
Serijsko preklapanje, paralelno preklapanje, paralelno izvođenje i privremena pohrana
- Kako je definirano ubrzanje?  
Ubrzanje je omjer vremena izvođenja na jednom i više paralelnih podsustava
- Što su tipični uzroci natjecanja za sredstva (*contention*) ili potrebe za usklađivanjem podataka (*coherence*)?
  - Zajedničke funkcije i varijable u operacijskom sustavu
  - Izmjena zajedničkih podataka koji se mijenjaju u privremenim spremnicima (*cache*)
  - Promet podataka u/iz glavne memorije
  - Čekanje na ulaz/izlaz
  - Sinkronizacijski primitivi

## Rekapitulacija (5)

- Koliko razina protokola uključuju web-aplikacije kojima se pristupa putem lokalnih mreža *Ethernet* spojenih na Internet?
  - Četiri: *Ethernet*, IP, TCP i HTTP
- Što uključuje vrijeme odziva web-aplikacije?
  - Vrijeme pristupa klijenta (klijent – ISP), vrijeme prijenosa paketa od klijentovog do poslužiteljevog ISP-a (ISP – ISP), vrijeme dostupa do poslužitelja (ISP – poslužitelj), ISP – davatelj internetske usluge (*Internet Service Provider*)
- Kako se može ostvariti razmjeran rast aplikacije?
  - Vertikalno ili horizontalno
- Koje probleme donosi horizontalan rast?
  - Usklađivanje, raspodjela tereta i raspodjela podataka