

UNIVERZITET U BEOGRADU
FAKULTET ORGANIZACIONIH NAUKA

Vladimir Đurica

**MODEL IZBORA VIRTUALIZACIONIH
TEHNOLOGIJA ZA IZGRADNJU CLOUD
SISTEMA**

doktorska disertacija

Beograd, 2018
UNIVERSITY OF BELGRADE

FACULTY OF ORGANISATIONAL SCIENCES

Vladimir Djurica

**MODEL OF CHOOSING VIRTUALISATION
TECHNOLOGIES FOR BUILDING CLOUD
SYSTEMS**

Doctoral Dissertation

Belgrade, 2018

Mentor:

dr Miroslav Minović, vanredni profesor Univerziteta u Beogradu, Fakulteta Organizacionih Nauka

Članovi komisije:

dr Miroslav Minović, vanredni profesor, Univerzitet u Beogradu, Fakultet Organizacionih Nauka

dr Dušan Starčević, redovni profesor, Univerzitet u Beogradu, Fakultet Organizacionih Nauka

dr Zoran Jovanović, redovni profesor, Univerzitet u Beogradu, Elektrotehnički Fakultet

dr Dejan Simić, redovni profesor, Univerzitet u Beogradu, Fakultet Organizacionih Nauka

dr Mladen Čudanov, vanredni profesor, Univerzitet u Beogradu, Fakultet Organizacionih Nauka

Datum odbrane: _____ 2018.

Ani Radovanović, na divnom sinu i nesebičnoj podršci.

Svom dragom mentoru, Miroslavu Minoviću na vođenju kroz put pun neizvesnosti.

Aleksandri Smiljanić, na odličnim savetima i korisnim komentarima.

*Zoranu Dimitrijeviću, na kafama koje su bile mnogo više od toga i gde sam takođe dobio puno saveta koji su pomogli
pisanju ovog rada.*

MODEL IZBORA VIRTUALIZACIONIH TEHNOLOGIJA ZA IZGRADNJU CLOUD SISTEMA

SAŽETAK

U okviru rapidno evolutivnog i konkurentnog poslovanja, kompanije i njihov menadžment, prinuđeni su na brzu adaptaciju tržišnih promena i usredsređenje na stvaranje nove vrednosti te ostvarenja maksimizacije povraćaja ulaganja u znanje i tehnologiju. Kompanije i njihov menadžment su u potrazi za tehnologijama i znanjem koji bi preoblikovali model poslovanja i povećali konkurentnost. Menadžment nastoji iskoristiti snagu savremenih tehnologija i konceptata poslovanja, te nalaženjem najbolje primene informacionih tehnologija i Interneta u poslovanju u cilju povećanja produktivnosti, efikasnosti i efektivnosti poslovanja.

Da bi kompanije postigle uspeh u novom okruženju, moraju restrukturirati postojeće organizacione strukture, redefinisati svoje strategije, poslovne procese, i izgraditi tehnološku infrastrukturu neophodnu radi efikasnijeg poslovanja i većih povraćaja investicija.

Cloud računarstvo se može posmatrati iz tri segmenta u okviru modela Softver-Platforma-Infrastruktura (SPI) koji se isporučuju kao servis. Koncept ide do tog nivoa apstrakcije da neko ko nudi softver kao uslugu može to da izgradi korišćenjem određene platforme ili infrastrukture na najam. Tako, dolazimo do cloud sistema na cloud sistemu. Pored javnih i privatnih cloud sistema, postoje i hibridni cloud sistemi. Ukoliko nemamo dovoljno resursa u svom privatnom, možemo da se proširimo na javni za onoliko vremena i resursa koliko nam je to potrebno.

Rad se bavi analizom i evaluacijom poslovnih i tehnoloških rešenja potrebnih za izgradnju cloud sistema. Glavni naučni doprinos rada je ekspertni sistem sa modelom izbora virtualizacionih tehnologija za pomoć u odlučivanju tokom izgradnje cloud sistema.

KLJUČNE REČI: CLOUD, VIRTUALIZACIJA, SAAS, PAAS, IAAS, SOFTVERSKI DEFINISANE MREŽE, SDN

NAUČNA OBLAST: Organizacione nauke i Tehničke nauke

UŽA NAUČNA OBLAST: Informacione Tehnologije

MODEL OF CHOOSING VIRTUALISATION TECHNOLOGIES FOR BUILDING CLOUD SYSTEMS

ABSTRACT

With rapidly evolving and competitive market situation, companies and their management are forced to a fast adaptation to the market changes and hence to focus on discovering ways of creating new value and maximizing the return on investments in knowledge and technology. Companies and their management are looking for technologies and knowledge that will change the way they are doing business and ensure increased competitiveness on the global markets. Management is trying to leverage the potential technologies available, aware of the necessity of implementing ICT and the Internet in their business in order to increase their productivity, efficiency and effectiveness.

If the companies could succeed in new surrounding they must change current organizational structures, redefine their strategies, business processes, build technology infrastructure necessary in order to support efficiently business processes and provide return on investments.

Cloud computing is most often divided in three segments within service, platform and infrastructure (SPI) model where each is being delivered as a service. This concept goes to the extent of someone offering software as a service can use platform or infrastructure as an underlying service. This is a cloud on top of a cloud system. Besides public and private cloud systems, there are hybrid cloud systems. If we have insufficient resources in the private cloud, we can expand to public cloud for as much resources are required.

This paper is focused on finding optimal cloud solution and performs analysis and evaluation of business solutions and underlying technologies. Main contribution of this thesis is a decision support expert system containing a model for selection of virtualization technologies for building cloud systems.

KEYWORDS: CLOUD, VIRTUALIZACIJA, SPI MODEL, SAAS, PAAS, IAAS, MAAS, XAAS

SCIENTIFIC FIELD: Organisational Sciences, Technical Sciences

SCIENTIFIC SUBFIELD: Information Technologies

SADRŽAJ

1. UVOD	1
1.1. CILJ RADA	1
1.2. METODE I HIPOTEZE	3
1.3. PREDMET ISTRAŽIVANJA	4
1.4. CILJ ISTRAŽIVANJA	4
1.5. KRATAK PREGLED DISERTACIJE	5
2. CLOUD SISTEMI	11
2.1. ISTORIJA CLOUD SISTEMA	11
2.2. KAKO DEFINIŠEMO CLOUD	14
2.3. CLOUD KONCEPT	15
2.4. RAZVOJ CLOUD SISTEMA	24
2.5. AKTUALIZACIJA CLOUD KONCEPTA	27
2.6. TEHNOLOŠKI TRENDОВI I POSLOVNI MODELI	29
2.7. POTENCIJALNE NIŠE PRIMENE	30
3. USLUŽNO NASPRAM SERVISNOG RAČUNARSTVA	32
3.1. KARAKTERISTIKE I MODEL ISPORUKE	34
4. SOFTVER-PLATFORMA-INFRASTRUKTURA (SPI) SERVISNI MODEL	36
4.1. SOFTVER KAO SERVIS (SAAS)	36
4.2. PLATFORMA KAO SERVIS (PAAS)	38
4.3. INFRASTRUKTURA KAO SERVIS (IAAS)	39
4.4. OSTALI CLOUD SERVISI (XAAS)	40
5. CLOUD MODELI I POSLOVNI KONCEPTI	41
5.1. KLASSE SERVISNOG RAČUNARSTVA	42
6. EKONOMIJA CLOUD RAČUNARSTVA	45
6.1. ELASTIČNOST: POMERANJE RIZIKA	45
6.2. COST-BENEFIT ANALIZA PRELASKA NA CLOUD	51
7. IZAZOVI I ŠANSE ZA CLOUD RAČUNARSTVO	54
7.1. DOSTUPNOST USLUGE	55

7.2.	ZAKLJUČAVANJE PODATAKA	56
7.3.	POVERLJIVOST PODATAKA I MOGUĆNOST EKSTERNE REVIZIJE PODATAKA	57
7.4.	USKA GRILA U PRENOSU PODATAKA	59
7.5.	NEPREDVIDIVOST PERFORMANSI	62
7.6.	SKALABILNO SKLADIŠTE PODATAKA (STORIDŽ)	63
7.7.	GREŠKE U VELIKIM DISTRIBUIRANIM SISTEMIMA	64
7.8.	BRZO SKALIRANJE	64
7.9.	DELJENJE REPUTACIJE	65
7.10.	LICENCIRANJE SOFTVERA	66
8.	<u>VIRTUALIZACIJA KAO OSNOVA CLOUD SISTEMA</u>	67
8.1.	HARDVERSKA VIRTUALIZACIJA	68
8.2.	PUNA VIRTUALIZACIJA	71
8.3.	DELIMIČNA VIRTUALIZACIJA	73
8.4.	PARAVIRTUALIZACIJA	74
8.5.	DESKTOP VIRTUALIZACIJA	75
8.6.	SOFTVERSKA VIRTUALIZACIJA	78
8.7.	VIRTUALIZACIJA MEMORIJE	80
8.8.	VIRTUALIZACIJA SISTEMA ZA SKLADIŠTENJA PODATAKA	82
8.9.	VIRTUALIZACIJA PODATAKA	85
8.10.	VIRTUALIZACIJA U OBLASTI RAČUNARSKIH MREŽA	86
9.	<u>HIPERVIZOR KAO OSNOVA CLOUD SISTEMA</u>	89
9.1.	XEN HIPERVIZOR	89
9.2.	HYPER-V HIPERVIZOR	93
9.3.	VSPHERE HIPERVIZOR	95
9.4.	POSLOVNI KRITERIJUMI UPOREDNOG PREGLEDA HIPERVIZORA KAO OSNOVE ZA CLOUD	97
9.4.1.	KRITERIJUM SPOSOBNOSTI IZVRŠAVANJA	99
9.4.2.	KRITERIJUM KOMPLETNOSTI VIZIJE	101
9.4.3.	UPOREDNI PREGLED TRŽIŠNE POZICIJE	103
9.4.4.	RAZMATRANJE KONKURENTSKIH PREDNOSTI PROIZVOĐAČA	104
9.5.	TEHNOLOŠKI KRITERIJUMI UPOREDNOG PREGLEDA HIPERVIZORA KAO OSNOVE ZA CLOUD	107
9.6.	ZAKLJUČAK I PROJEKCIJE VEZANE ZA BUDUĆNOST HIPERVIZORA U OKVIRU CLOUD SISTEMA	109
10.	<u>LINUX KONTEJNERI KAO OSNOVA CLOUD SISTEMA</u>	110
10.1.	LINUX KONTEJNERI (LXC)	110
10.2.	DOCKER KONTEJNERI	111
10.3.	ROKET KONTEJNERI	111

11. SOFVERSKI DEFINISANE MREŽE (SDM) KAO OSNOVA CLOUD SISTEMA	112
11.1. RANE PROGRAMABILNE MREŽE	114
11.1.1. PROGRAMABILNOST RAVNI PODATAKA	117
11.1.2. RAZDVAJANJE RAVNI KONTROLE OD RAVNI PODATAKA	121
11.1.3. MREŽNA VIRTUALIZACIJA	124
11.2. SOFVERSKI DEFINISANE MREŽE – FUNKCIJE I ARHITEKTURA	129
11.2.1. KLJUČNA SDM TERMINOLOGIJA	129
11.2.2. UKRATKO O SOFVERSKI DEFINISANIM MREŽAMA	130
11.2.3. PROGRAMABILNOST RAVNI PODATAKA	133
11.2.4. RAZDVAJANJE RAVNI KONTROLE OD RAVNI PODATAKA	135
11.2.5. MREŽNA VIRTUALIZACIJA	137
11.3. SDM KONTROLERI	140
11.4. MREŽNI SVIČ OPERATIVNI SISTEMI	142
11.5. MOGUĆNOSTI I BUDUĆI PRAVCI NA MREŽNOM SLOJU	145
12. PRISTUP IZBORA REŠENJA PRI IZGRADNJI CLOUD SISTEMA	151
12.1. PREDLOG PRISTUPU REŠENJA	151
12.2. ANALIZA REZULTATA ANKETE	153
12.3. PREDLOG REŠENJA	164
12.3.1. KAPITALNI TROŠKOVI	164
12.3.2. OPERATIVNI TROŠKOVI	165
12.3.3. POSLOVNO-TEHNOLOŠKI FAKTORI	166
12.3.4. EKSPERTNI SISTEM <i>KL</i> AUDIFIKATOR	169
12.3.5. EVALUACIJA – STUDIJA SLUČAJA	178
13. ZAKLJUČAK	181
13.1. CILJ RADA	181
13.2. PREDMET ISTRAŽIVANJA	183
13.3. CILJ ISTRAŽIVANJA	183
13.4. METODE I HIPOTEZE	184
13.5. RAZMATRANJE HIPOTEZA	185
13.6. REZIME DISERTACIJE	187
13.6.1. CLOUD	187
13.6.2. EKONOMIJA CLOUD RAČUNARSTVA	190
13.6.3. VIRTUALIZACIJA KAO OSNOVA CLOUD SISTEMA	192
13.6.4. KONTEJNERI	195
13.6.5. SOFVERSKI DEFINISANE MREŽE KAO OSNOVA CLOUD SISTEMA	196
13.6.6. MREŽNI SVIČ OPERATIVNI SISTEMI	199
13.6.7. MOGUĆNOSTI I PRAVCI NA MREŽNOM SLOJU	199

13.6.8.	PRISTUP IZBORA REŠENJA PRI IZGRADNJI CLOUD SISTEMA	201
13.7.	DOPRINOS I BUDUĆI PRAVCI ISTRAŽIVANJA	202
14.	<u>CITIRANI RADOVI</u>	203
15.	<u>PRILOG A – ANKETNA PITANJA</u>	210
16.	<u>PRILOG B – ŠIFRARNIK ZA SPSS</u>	215
17.	<u>PRILOG C – IMPLEMENTACIJA EKSPERTNOG SISTEMA <i>KLAUDIFIKATOR</i></u>	219
18.	<u>BIOGRAFSKI PODACI</u>	229

DODATNO:

1. Izjava o autorstvu
2. Izjava o istovetnosti štampane i elektronske verzije doktorskog rada
3. Izjava o korišćenju

INDEKS SLIKA

Slika 1 - Prednosti cloud računarstva.....	11
Slika 2 - Šta je cloud?	15
Slika 3 - Cloud računarstvo – dostupnost sa bilo kog uređaja i lokacije	16
Slika 4 – Korisnici i dobavljači cloud računarstva.....	18
Slika 5 - Tipovi cloud računarstva	32
Slika 6 - SPI Model (SaaS, PaaS, IaaS)	33
Slika 7 – SPI Model: Hostovanje-Izgradnja-Korišćenje (preuzeto sa Interneta)	36
Slika 8 - Mapiranje sloja na korisnike / operatere.....	37
Slika 9 – Primer SaaS aplikacija (slika preuzeta sa Interneta)	37
Slika 10 –PaaS su servisi bez instalacije i održavanja od strane korisnika.....	38
Slika 11 - IaaS, moguća arhitektura infrastrukture na zahtev	39
Slika 12 - Vrste cloud Sistema	42
Slika 13 – Obezbeđivanje resursa u periodu povišene tražnje.....	47
Slika 14 – Nedovoljno obezbeđenih resursa 1	47
Slika 15 – Nedovoljno obezbeđenih resursa 2	47
Slika 16 – STREAM test na 75 virtualnih mašina (a) memorije i (b) upisivanje na disk 1GB po mašini	61
Slika 17 - Okruženje bez virtualizacije (levo) i virtualizovano okruženje (desno)	68
Slika 18 – Puna virtualizacija, šematski prikaz.....	72
Slika 19 – Delimična virtualizacija, šematski prikaz[67]	73
Slika 20 – Detaljniji prikaz delimične virtualizacije[67]	74
Slika 21 – Paravirtualizacija - GuestOS je paravirtualizovano okruženje.....	75
Slika 22 - Desktop virtualizacija.....	76
Slika 23 - Desktop virtualizacija, arhitektura.....	76
Slika 24 - Desktop virtualizacija, šematski prikaz [68].....	77
Slika 25 - Virtualizacija memorije, arhitektura	80
Slika 26 - Skup svih memorija dostupnih operativnom sistemu.....	81
Slika 27 - Virtualizacija sistema za skladištenje podataka.....	83
Slika 28 - Virtualizacija podataka	85
Slika 29 - Garnterov kvadrant za x86 serversku virtualizaciju [71]	98
Slika 30 - Tradicionalna, vertikalno integrisana mrežna arhitektura	115

Slika 31 - Moguća reprezentacija SDM arhitekture	131
Slika 32 – Tehnologija kojom je cloud primarno izgrađen.....	153
Slika 33 - Pristup rešenju mrežnog sloja.....	154
Slika 34 - Tip 1 hipervizora koji se najviše koristi.....	154
Slika 35 - Tip Linux kontejnera u upotrebi.....	155
Slika 36 - Tehnologija virtualizacije mrežnog sloja.....	155
Slika 37 - Tip SDM kontrolera u upotrebi.....	156
Slika 38 - Linux kao NOS.....	156
Slika 39 - Upotreba starih i specifičnih sistema.....	157
Slika 40 – Uticaj specifičnih ili zastarelih sistema na arhitekturu cloud sistema.....	157
Slika 41 - Ostvarena automatizacija.....	158
Slika 42 - Ostvaren stepen automatizacije.....	158
Slika 43 - Mogućnost integracije u jedinstven sistem upravljanja.....	159
Slika 44 - Mogućnost upravljanja mrežnim slojem korišćenjem alata za automatizaciju.....	159
Slika 45 - Potreba za multi OS podrškom.....	160
Slika 46 - Primarna poslovno-tehnološka orijentacija clouda	160
Slika 47 - Ključni poslovni kriterijumi za arhitekturu sistema	161
Slika 48 - Uštede uvođenjem cloud tehnologija	162
Slika 49 - Povećanje poslovno-tehnološke agilnosti.....	162
Slika 50 - Strateško poravnanje (isporučivanje projekata i poverenje u IT odeljenje).....	163
Slika 51 - Stablo atributa ekspertnog sistema.....	169
Slika 52 - Struktura atributa.....	170
Slika 53 - Model ekspertnog sistema.....	171
Slika 54 – Funkcije odlučivanja za Cloud sa pravilima.....	173
Slika 55 - Funkcija odlučivanja za trošak.....	173
Slika 56 - Funkcija odlučivanja za specifikaciju servera	174
Slika 57 - Funkcija odlučivanja za kapacitet, snagu i efikasnost servera.....	175
Slika 58 – Pravila za osnovne attribute	176
Slika 59 - Rezultat evaluacije	178
Slika 60 - Aspekti opcije 1 (jaki i slabi)	179
Slika 61 - Aspekti opcije 2 (jaki i slabi)	180
Slika 62 - Aspekti opcije 3 (jaki i slabi)	180

INDEKS TABELA

Tabela 1 - Ekonomija obima za srednje i velike datacentre [43]	21
Tabela 2 - Cena kilovat sata po regionu USA [46]	21
Tabela 3 - Cene el. energije u Evropi za potrošnju od 2GWh/godini [48]	22
Tabela 4 - Pregled 10 prepreka i prilika za rast cloud računarstva (Ambrust M., 2009)	27
Tabela 5 - Na Grejeve podatke iz 2003. dodati su normalizovani podaci iz 2017	52
Tabela 6 - 10 glavnih izazova i šansi za prihvatanje i rast cloud računarstva	54
Tabela 7 - Prekidi rada AppEngine, Gmail i AWS (Ambrust M., 2009)	55
Tabela 8 – Tabela kompatibilnosti hostova [70]	79
Tabela 9 – Kriterijumi evaluacije za sposobnosti realizacije	99
Tabela 10 - Kriterijum evaluacije, kompletnost vizije	101
Tabela 11 – Uporedni tehnološki kriterijumi tip 1 hipervizora	107
Tabela 12 - Potencijal i uticaj SDM-a na poslovne operativne modele	146
Tabela 13 - Potencijal i uticaj SDM-a na poslovne operativne modele	200
Tabela 14 - Šifarnik za SPSS	215

1. UVOD

1.1. Cilj rada

Cilj rada jeste da: 1) prikaže kritički uporedni pregled aktuelnih pristupa uvođenju cloud rešenja u svakodnevno poslovanje preduzeća, 2) analizira aktuelne tehnologije po rešenjima i sloju kom pripadaju (vrste cloud sistema, virtualizacija, kontejneri, mreže) i da 3) predloži ekspertski sistem koji bi, za date kriterijume, poslužio u odlučivanju za gradivne komponente i izbor cloud rešenja.

Deo rada koji se odnosi na pregled ima za cilj da olakša donošenje kvalitetne poslovne odluke vezane za uvođenje cloud tehnologija. U okviru pregleda, rad primarno razmatra SPI model (eng. *Software, Platform, Infrastructure*).

Računarski centri današnjice poznaju različite vrste gradivnih blokova čiji je zajednički imenitelj često tehnologija virtualizacije. Dva domena na koje se rad fokusira u ovom smislu jesu server (operativni sistem) i računarska mreža. Primenjeno na sloj servera, tehnologija se nalazi u zreloj tehnološkoj fazi i omogućava apstrakciju fizičkih obradnih resursa, dok je donedavno, uglavnom tradicionalni mrežni sloj doživljavao transformaciju sa integrisanog pristupa na razdvojene ravni upravljanja i prosleđivanja podataka uz sve viši stepen programabilnosti i sa mogućnošću integrisanja sa ostalim slojevima. Ove paradigme promena na mrežnom sloju prepoznavamo kroz termine koji još uvek konvergiraju kroz industriju: (1) virtualizacija mrežnih funkcija i (2) softverski definisane mreže.

Tehnologije virtualizacije omogućavaju znatno viši stepen iskorišćenja fizičkih resursa i elastičnost resursa koji se dobavljaju na zahtev korisnika cloud sistema. Pored toga što omogućavaju dobavljanje na zahtev, cloud sistemi menjaju iskustvo interakcije korisnika koji popunjava web forme ili koristi aplikativni programski interfejs (API) i vrše dobavljanje svih vrsta traženih resursa bez ljudske interakcije.

Ovakav način upotrebe resursa omogućava vrstu optimizacije kakva ranije nije bila moguća. Osnova ovih mogućnosti je u transformaciji serverske infrastrukture u virtualne mašine zahvaljujući čemu se fizički resursi mnogo bolje optimizuju, migriraju i transformišu da odgovore na poslovne potrebe.

Na mrežnom sloju, imamo nove mogućnosti koje se odnose na migraciju, automatizaciju i upravljanje mrežnim resursima, kao i mogućnost prevazilaženja određenih tehnoloških prepreka poznatih u tradicionalnom okruženju.

Na ovaj način, počevši od najnižeg tehnološkog sloja, imamo konsolidaciju infrastrukture na nivou mrežnog sloja a zatim i virtualnih mašina na serverskom, to jest infrastrukturnom sloju, koje vrši obradu podataka. Za ovaj sloj vezujemo pojam infrastrukture na zahtev gde je ključna mogućnost dobavljanja i upravljanja slojem operativnog sistema na zahtev (eng. *Infrastructure as a Service, IaaS*). Sloj iznad, nalazi se sloj platforme na zahtev (eng. *Platform as a Service, PaaS*). Na ovom sloju korisnik gubi kontrolu nad operativnim sistemom ali dobija kontrolu nad platformom kao celinom. Konačno, najviši sloj apstrakcije cloud sistema podrazumeva dobavljanje softvera kao usluge, (eng. *Software as a Service, SaaS*).

Bilo da se radi o infrastrukturi, platformi ili softveru koji se dobavljaju kao usluga, ovakve sisteme odlikuju ranije nepostojeći i/ili bitno poboljšani kvaliteti poput elastičnosti i skalabilnosti. U novije vreme, sve više dolaze do izražaja mogućnosti horizontalne integracije sa javnim cloud sistemima.

Cloud sistemi pružaju mogućost postojanja dobavljača usluge (eng. *Service Provider, SP*) koji kao sloj ispod, ima ulogu infrastrukture ili platforme na zahtev. Pored ovog koncepta, formira se niz novih koncepata kako organizacije računarskih resursa, tako i poslovnih modela. S tim u vezi, postaje izazov u poslovnom odlučivanju da li i u kojoj meri graditi i oslanjati se na sopstvenu infrastrukturu naspram najma računarskih resursa kao servisa.

Rad razmatra pravce razvoja, šanse i niše za poslovne poduhvate i budući razvoj ovakvih sistema.

1.2. Metode i hipoteze

U radu se koriste opšte i posebne naučne metode:

- Od opštih metoda:
 - Sistemski pristup,
 - Analitičko-sintetičku metodu, kao postupak kojim se prilazi suštini problema, na način da se obrade međusobno uslovljeni odnosi, a potom i problem u celini,
 - Deduktivno-logičku metodu pomoću koje će se, na temelju spoznaje utvrđene praksom, utemeljiti vlastiti teorijski stavovi,
 - Komparativnu metodu.
- Od posebnih metoda:
 - Anketni upitnik za utvrđivanje uspešnosti primene cloud rešenja i ostvarenja konkurentskih prednosti u poslovanju kompanija,
 - Statističke metode radi obrade podataka i prezentacije rezultata.

Kroz rad ispitujemo sledeće hipoteze:

1. Opšta:

- a. Polazna hipoteza je da pri sadašnjem stanju tržišnih ponuda cloud tehnologija, ne postoji jedno optimalno rešenje za sve poslovne sisteme, već da se za konkretni poslovni sistem može iz datog skupa cloud sistema naći bar jedno rešenje koje je bolje od ostalih.

2. Pojedinačne:

- H1 - Postoji pozitivna povezanost između produktivnosti - konkurentnosti preduzeća i nivoa primene cloud tehnologija.
- H2 - Postoji mogućnost federacije različitih resursa – privatnih sa javnim ili drugim privatnim resursima.
- H3 - Postoji mogućnost izgradnje rešenja u slojevima – aplikativni servisi na platformi.
- H4 - Informacionim sistemima zasnovanim na cloud rešenjima postiže se ušteda u resursima u odnosu na tradicionalni pristup realizaciji infrastrukture.
- H5 - Informacionim sistemima zasnovanim na cloud rešenjima postiže se veća poslovna agilnost u izgradnji sistema u odnosu na tradicionalni pristup realizaciji infrastrukture.

- H6 - Opređenjem za cloud rešenja može se postići bolja skalabilnost sistema, odnosno jednostavnija alokacija ili dealokacija potrebnih resursa u odnosu na klasični pristup realizaciji informacione infrastrukture.
- H7 - Opređenjem za virtualizaciju na mrežnom sloju postiže se viši stepen automatizacije i bolje upravljanje celokupnim sistemom u odnosu na klasični pristup realizaciji informacione infrastrukture.

1.3. Predmet istraživanja

Predmet istraživanja predstavlja izrada kritičkog uporednog prikaza aktuelnih pristupa pri uvođenju cloud rešenja u preduzeće. Na osnovu pregleda tehnologija za virtualizaciju i raspoloživih rešenja, u radu su istraženi i obrađeni relevantni kriterijumi za upoređivanje postojećih rešenja i donošenje odluke o izboru rešenja. Na osnovu identifikovanih kriterijuma, vrši se komparacija tri reprezentativna rešenja za primenu cloud tehnologija u svakodnevnom poslovanju preduzeća.

U radu se daje i pregled izbora nekoliko vodećih softverski definisanih mreža (SDM) (eng. *Software Defined Networks, SDN*) u odnosu na tradicionalna, koja u ovom slučaju smatramo normalizovanim i stoga ne ulazimo u konkretne proizvođačke specifičnosti.

1.4. Cilj istraživanja

Osnovni cilj primene informacionih tehnologija jeste odgovor na poslovnu potrebu, praćenu povećanjem produktivnosti, efikasnosti i efektivnosti odvijanja poslovnih procesa. Važna stvar u dostizanju ovog cilja jeste izbor odgovarajućeg rešenja informacionog sistema koji pruža podršku poslovnim funkcijama na svim nivoima, od operativnog do strateškog.

Imajući u vidu da su poslednjih godina aktuelna rešenja zasnovana na korišćenju cloud tehnologija, cilj ovog rada jeste kritički uporedni pregled aktuelnih pristupa uvođenju cloud rešenja u svakodnevno poslovanje preduzeća. Ovakav pregled ima za cilj da olakša donošenje kvalitetne poslovne odluke o uvođenju cloud tehnologija. Glavni fokus rada biće na SPI modelu ali i ostalim cloud tehnologijama.

Rad se takođe bavi razmatranjem tehnologija serverske virtualizacije kao osnovom za cloud sisteme, i pregledom aktuelnih industrijskih rešenja od značaja. U radu su prikazani i mogući pravci razvoja, te i šanse i niše za poslovne poduhvate kroz budući razvoj cloud sistema.

1.5. Kratak pregled disertacije

U radu se razmatra definicija, koncept i početke cloud računarstva iz više uglova kroz poglavlje *Cloud sistemi*.

Počeci cloud računarstva se vezuju za Likliderovu viziju intergalaktičke računarske mreže dok je radio za ARPANET. Vizija je podrazumevala pristup aplikacijama sa bilo kog mesta, bilo kom mestu. Ovu ideju dalje nastavlja i unapređuje Džon Makarti [1], uvodeći koncept servisnog (eng. *utility*) računarstva.

Konstatacija je da su ključne tehnologije za razvoj cloud sistema bile web 2.0 tehnologije za isporuku do korisnika, kao i razne vrste tehnologija virtualizacije ali i sve brži razvoj mrežnih propusnih opsega i, na kraju, razvoj univerzalnih softverskih standarda interoperabilnosti.

Neke od ključnih prednosti cloud računarstva, kao servisnog računarstva su: (1) veća skalabilnost, (2) brži pristup infrastrukturi, (3) veća dostupnost servisa, (4) brža mogućnost izlaska na tržišta, (5) povećana efikasnost IT osoblja, (6) mogućnost bolje i šire geografske pokrivenosti servisa, (7) ušteda u ukupnim troškovima (uz bolji balans između kapitalnih i operativnih troškova), (8) mogućnost boljeg obezbeđenja poslovnog kontinuiteta, (9) ukupno više performanse celokupnog sistema.

Cloud računarstvo je isporuka računarskih servisa, nasuprot proizvodu, gde se deljeni resursi, softver i informacije isporučuju računarima i drugim uređajima u vidu merljivih servisa putem računarskih mreža, obično Interneta. Ovu temu rad razmatra kroz poglavlje *Uslužno naspram servisnog računarstva*.

Uvođenje clouda ne znači automatski da će aplikacije, servisi i procesi automatski biti migrirani na cloud rešenja. Mnoge kompanije su vrlo obazrive i analiziraju detaljno svoje strateške poslovne procese i intelektualnu svojinu kako bi utvrdile koja tehnološka infrastruktura treba da ostane pod kompanijskim direktnim nadzorom a koja će biti preseljena u cloud rešenje. Doprinos rada u ovom domenu je odlučivanje za ili protiv migracije na cloud rešenje.

Cloud servis je opšti termin za bilo šta što uključuje isporuku hostovanih servisa putem Interneta. Ovi servisi su razmatrani kroz poglavlje *Softver-Platforma-Infrastruktura (SPI) servisni model*. U okviru SPI modela, servisi su podeljeni u tri osnovne kategorije: (1) Softver kao servis (Software as a service, **SaaS**) - Klijentu se pruža mogućnost korišćenja dobavljačevih aplikacija koji se izvršavaju na cloud infrastrukturi, (2) Platforma kao servis (Platform as a service, **PaaS**) - Skup softvera i razvojnih alata hostovanih na infrastrukturi dobavljača usluge. Razvojni timovi razvijaju aplikacije na platformi koja se nalazi fizički u posedu dobavljača usluge. Za pristup ovakvoj infrastrukturi koristi se Internet, (3) Infrastruktura kao servis (Infrastructure as a service, **IaaS**) - Pruža mogućnost pokretanja, zaustavljanja, pristupanja i konfigurisanja virtualnih servera i sistema skladištenja podataka.

U okviru poglavlja *Cloud modeli i poslovni koncepti*, u radu se zad diskutuje javni, privatni i hibridni cloud. Javni cloud poseduje resurse koji su javno dostupni svima. Privatni cloud označava mrežne servise ili data centar hostovanih servisa u privatnom vlasništvu koji pruža usluge ograničenom broju ljudi. Kada isporučilac usluga koristi resurse javnog clouda da napravi svoj privatni cloud, rezultat se zove virtualni privatni cloud. Privatni ili javni, cilj cloud računarstva je da pruži jednostavan, skalabilan pristup računarskim resursima i ICT uslugama.

Osnovne karakteristike cloud sistema: (1) Samousluživanje na zahtev – Krajnji korisnik može jednostrano da obezbedi računarske usluge, poput serverskog obradnog vremena ili mrežnog sistema skladištenja podataka, i to po potrebi, a bez ljudske interakcije sa dobavljačem usluga, (2) Širok mrežni pristup – Mogućnosti su na raspolaganju kroz mrežu i pristupa im se kroz standardne mehanizme koji promovišu upotrebu od strane heterogenih punih ili tankih klijentskih platformi, (3) Udruživanje resursa – Računarski resursi dobavljača usluga se udružuju u službi upotrebe u višeklijentskom okruženju, sa različitim fizičkim i virtualnim resursima dinamički dodeljenim i raspoređivanim u skladu sa korisničkim zahtevima, (4) Rapidna elastičnost – Mogućnosti mogu biti elastično obezbeđene i oslobođene, u mnogim slučajevima automatski, sa skaliranjem ka spolja ili unutra, proporcionalno tražnji. Iz korisničke perspektive, resursi često deluju neograničeno i mogu biti prisvojeni u bilo kom obimu i vremenu, (5) merena usluga – Cloud sistemi vrše merenje upotrebe resursa a zatim i automatsku kontrolu i optimizaciju korišćenja istih resursa. Poglavlja *Ekonomija cloud računarstva* i *Izazovi i šanse za cloud računarstvo* govore o tački isplativosti prelaska na cloud kao i poslovno-tehnološkim izazovima te u isto vreme i poslovnim prilikama.

S obzirom na prepoznavanje tehnologija virtualizacije kao jednog od osnovnog gradivnog bloka cloud sistema, dalji doprinos rada u nastavku daje fokus na tehnologije virtualizacije i komparativnu analizu tri rešenja po tehnološkim i poslovnim funkcijama.

Uobičajeni cilj virtualizacije je centralizacija administrativnih zadataka uz poboljšanje skalabilnosti i opšte iskorišćenosti hardverskih resursa. Poglavlje *Virtualizacija kao osnova cloud sistema* i *Hipervizor kao osnova cloud sistema* detaljno daje poslovno i tehnološki uporedni pregled tri rešenja kao i pregled oblasti.

U konkretnom smislu virtualizacija se pojavljuje kao odgovor na neadekvatnu iskorišćenost hardverskih resursa u oba smera. To se odnosi pre svega na klasične scenarije u kojima se planira hardver za određenu namenu i sa tim u vezi jednom kapacitira i dobavi, te ostaje nepromenjen do kraja svog životnog veka uz minimalne ili nikakve mogućnosti proširivanja.

Pored gore navedenog, postoji i problem kompanijskog plana poslovnog kontinuiteta (eng. *contingency*) plana, koji podrazumeva izvršavanje procedure u slučaju nepredviđenih situacija u poslovanju, te raznih otkaza resursa i servisa. Business continuity plan podrazumeva dostupnost, zatim visoku dostupnost kao i skalabilnost u kontekstu obezbeđenja kontinuiteta poslovanja i mogućnosti njegovog proširivanja.

Razmatran odgovor na navedene potrebe je virtualizacija infrastrukture. Rad u nastavku diskutuje vrste virtualizacija, a u zatim primere virtualizacionih scenarija.

Sledi pregled vrsta virtualizacije: (1) hardverska virtualizaciju, (2) puna virtualizacija, (3) delimična virtualizacija, (4) paravirtualizacija, (5) desktop virtualizacija, (6) softverska virtualizacija, (7) virtualizacija memorije, (8) virtualizacija sistema za skladištenje podataka, (9) virtualizacija u oblasti računarskih mreža, (10) virtualizacija podataka.

Kao zaključak na temu serverske virtualizacije rad razmatra tri rešenja serverskog hipervizora tipa 1: vmware, xen i hyper-v.

Linux kontejneri omogućavaju virtualizaciju na nivou operativnog sistema. Ovakav sistem omogućava izvršavanje višestrukih izolovanih korisničkih domena na jednom sistemskom domenu (na nivou jednog kernela). Virtualizacija sistema na ovom nivou značajno je manje resursno zahtevna od pune virtualizacije

i nudi dosta veliku fleksibilnost. Takođe, kontejneri se mogu izvršavati na svakom operativnom sistemu koji ih podržava.

Kompletno upravljanje resursima radi kernel, u vidu usluge kontejnerima. Primeri upravljanja resursima su upravljanje memorijom, procesorskim vremenom, sistemskim prekidima i drugim resursima.

U okviru kontejnera rad razmatra LXC, Docker i ROKET uz zaključak da je Docker danas defakto standard u industriji. Kontejneri se razmatraju u okviru poglavlja *Linux kontejneri kao osnova cloud sistema*.

U nastavku, rad u okviru poglavlja *Softverski definisane mreže (SDM) kao osnova cloud sistema*, razmatra novi pristup mrežnom sloju. U tradicionalnim računarskim mrežama, ravan kontrole i ravan podataka čine nerazdvojivu celinu i obično se nalaze u mrežnom elementu nekog proizvođača. Mrežni elementi su nezavisni, distribuirani i konvergiraju ka određenom stanju, u zavisnosti od informacija koje razmenjuju. U SDM arhitekturi, ravan kontrole i ravan podataka su razdvojene i postoji centralni uvid u mrežne resurse. U tradicionalnim mrežama, mrežni elementi imaju u sebe ugrađene mnoge mrežne protokole, koji su često, zatvorena i patentirana kompanijska rešenja.

Neki od ključnih izazova u tradicionalnim računarskim mrežama su: (1) značajan broj korisnika ima potrebnu za manjim brojem mogućnosti koje su dostupne na uređajima, (2) ovako vertikalno integrisane opcije i funkcije čine da mrežni uređaji imaju višu cenu nego što bi mogli imati, (3) put za bilo kakvu promenu protokola je dugotrajan, često meren u godinama, čineći na ovaj način inovaciju veoma sporom i neefikasnom, (4) eksperimentisanje sa novim protokolima je teško ili čak nemoguće, (5) svaka promena postojećih funkcija u sistemu mora da prođe kroz proceduru prihvatanja specifičnog proizvođača i može da bude neprihvaćena, da implementacija traje (znatno) duže od očekivane ili željene, ne odgovori na potrebe i zahteve u potpunosti, ne odgovori kvalitetom ili nekim drugim željenim kriterijumom, (6) konfiguracioni interfejsi mogu da variraju, a obično i variraju u zavisnosti od proizvođača a često i (linije) proizvoda, (7) upravljanje ne-programabilnim mrežama je decentralizovano, stvarajući značajne izazove u automatizaciji, čineći operativne troškove visokim i sam proces podložnim greškama, (8) budući da je Internet kritična infrastruktura sveta, u čije je razvoj uloženo mnogo sredstava i opreme u velikim razmerama, otežan je razvoj i njegova evolucija. Ključni razlog za sporu evoluciju je potreba za usaglašavanjem između raznih organizacija, poput standardizacionih tela. Ova pojava naziva se okoštavanjem Interneta [2].

Osnovna definicija SDM-a bila bi separacija kontrolne ravni i ravni prosleđivanja podataka uz programabilnost funkcija. U tradicionalnim mrežama, ove dve ravni su vertikalno integrisane u uređaj, poput rutera ili sviča, i obično su proizvođačka rešenja zaštićena patentom.

SDM je pristup računarskim mrežama koji podrazumeva sledeće: (1) razdvajanje ravni prosleđivanja podataka od ravni kontrole, (2) definicija interakcija između prethodno navedenih ravni kroz jasno definisane apstrakcije (aplikativni programski interfejs, API), (3) logički centralizovan menadžment, (4) programabilnost, (5) otvoreni standard.

Kao odgovor na gore pomenute izazove, pojavila se ideja programabilnih mreža. Koncept softverski definisanih mreža (SDM) pojavio se sa težnjom da se inovacije u oblasti računarskih mreža odvijaju povećanom brzinom i uz isto iskustvo razvoja kao i u ostatku softverske industrije.

U prvom delu rad diskutuje rane programabilne mreže. Rad dalje razrađuje ključne primere koji su obeležili razvoj ove oblasti podeljene u podoblasti kao što su: programabilna ravan podataka, razdvajanje ravni kontrole od ravni podataka, odnosno hardvera, i na kraju mrežnu virtualizaciju. U pregledu programabilnih mreža iz 1999, budući razvoj predviđaju u tri pravca: (1) okruženja za izvršavanje aktivnih mreža, (2) mrežni kerneli sa otvorenim signalima, (3) node operativni sistemi, ili operativni sistemi na čvorovima.

U radu razmatramo samo prva dva jer su od ključnog značaja za razvoj oblasti. Iako je bilo pokušaja za otvaranjem, generalizacijom i korišćenjem node operativnih sistema, ovaj pristup, nije doživeo značajniji razvoj. Klasifikaciju istorije programabilnih mreža pre softverski definisanih mreža vršimo u sledećim kategorijama i sa sledećim primerima: (1) programabilnost ravni podataka: dodatak programabilnosti u tradicionalnim računarskim mrežama i aktivne mreže: (i) Xbind[3], (ii) Active Node Transfer system, ANTS[4], (iii) SwitchWare[5], (iv) NetScript[6]; (2) razdvajanje ravni kontrole od ravni podataka: razdvajanje dve ravni kontrole od ravni podataka i pristup otvorenih signala: (i) General Switch Management Protocol, GSMP[7], (ii) Tempest[8], (iii) Forward and Control Element Separation, ForCES[9], (iv) Routing Control Platform, RCP[10], (vi) SoftRouter[11]; (3) mrežna virtualizacija: apstrakcija mrežnih resursa kroz dodatak sloja mrežne virtualizacije i mrežnog overleja (eng. *network overlay*): (i) Tempest[8], (ii) Multicast Backbone, Mbone[12], (iii) Resilient Overlay Networks, RON[13], (iv) Global Environment for Network Innovation, GENI[14], (v) Virtual Network Infrastructure, VINI[15].

U narednom delu diskutujemo terminologiju i definiciju softverski definisanih mreža. Razmatramo šta je to SDM i kako se razlikuje ili dopunjuje u odnosu na koncepte poput virtualizacije mrežnih funkcija (*Network Function Virtualization*, NFV) i mrežna virtualizacija (*Network Virtualization*, NV).

Iza toga razmatramo pregled rešenja u kategorijama: programabilnost ravni podataka, razdvajanje kontrolne ravni od ravni podataka, mrežna virtualizacija (mrežni hipervizori) i završavamo sekciju sa SDM kontrolerima i mrežnim operativnim sistemima: (1) programabilnost ravni podataka: (i) OpenFlow [16], (2) razdvajanje ravni kontrole od ravni prosleđivanja podataka, (i) Secure Architecture for the Networked Enterprise, SANE [17], (ii) ETHANE [18]; Mrežna virtualizacija (mrežni hipervizori): (i) Open vSwitch [19], (ii) FlowVisor [20], (ii) Network Virtualization Platform (now vmWare NSX), NVP [21] [22] [23]; (3) SDM kontroleri: (i) NOX [24], (ii) ONIX [23], (iii) Yanc [25].

U poslednjem odeljku *Mogućnosti i budući pravci na mrežnom sloju* u okviru razmatranja SDM mreža diskutujemo potencijalne poslovne i tehnološke dobrobiti, uticaj i promene koje mogu desiti ili se već dešavaju sa uplivom raznih SDM rešenja.

Postoji rastući broj operativnih sistema za svičeve baziranih na Linux-u koji se mogu instalirati na poseban tip mrežnih uređaja koje nazivamo bare metal svičevima ili whitebox svičevima. Ovakvi uređaji se razlikuju od klasičnih, proizvođačkih, po tome što je na njih moguće instalirati više različitih operativnih sistema. Rad razmatra sledeća rešenja: (1) Cumulus Linux [26], (2) Open Network Linux, ONL [27] (3) Switch Light [28].

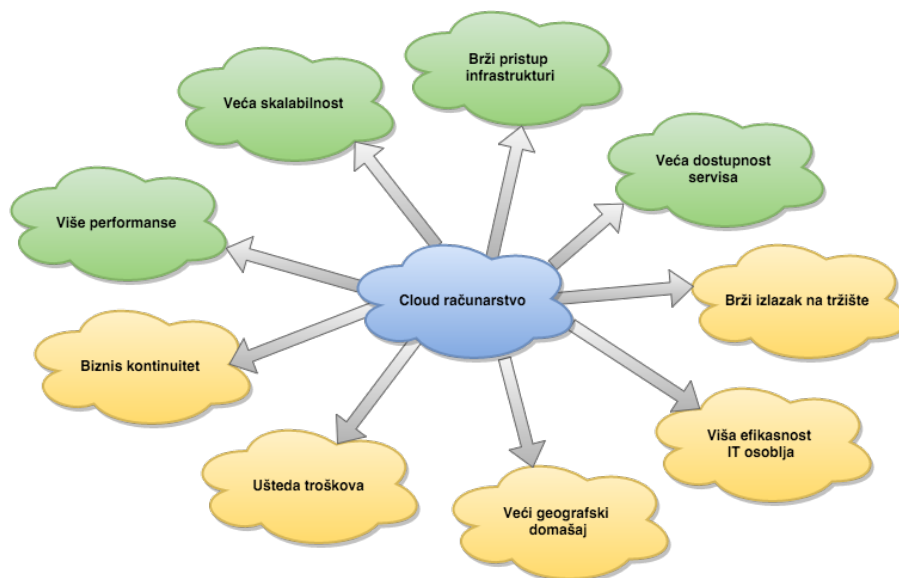
Dalje diskutujemo pravce razvoja i šanse za razvoj savremenih SDM rešenja od kojih biznis može potencijalno da ima koristi. Rad detaljno razrađuje svaku od sledećih poslovno-tehnoloških mogućnosti koje SDM potencijalno nosi i vršimo njihovu klasifikaciju na: (1) poslovne i (2) tehnološke: (i) istraživačke i razvojne i (ii) strateške i operativne kroz poglavlje *Pristup izbora rešenja pri izgradnji cloud sistema*. U istom poglavlju razmatraju se rezultati ankete, vrši se predlog rešenja i vrši prikaz ekspertnog sistema *Klaudifikator*.

2. Cloud sistemi

2.1. Istorija cloud sistema

Ideju „intergalaktičke računarske mreže“ predstavio je šezdesetih godina Liklider, [29] čovek koji je bio odgovoran za razvoj ARPANET-a (Advanced Research Projects Agency Network), 1969. godine [13].

Likliderova vizija bila je da svako na planeti bude povezan, da pristupa aplikacijama i podacima sa bilo kog mesta, bili oni na bilo kom mestu. Neki smatraju da je ovo bila vizija koja se danas može uporediti sa onim što poznajemo pod terminom cloud računarstva.



Slika 1 - Prednosti cloud računarstva

Druga struja eksperata cloud koncept pripisuje naučniku računarskog sveta Džonu Mekartiju [30] koji je predstavio ideju da se računarske usluge isporučuju poput servisa, nešto nalik na servisne biroe koji su postojali tokom šezdesetih godina dvadesetog veka [31].

Od šezdesetih na ovamo, cloud računarstvo se razvilo u više navrata, sa Web 2.0 budući najvišom i najskorijom evolucijom. S druge strane, s obziromna to da je pristup Internetu značajno unapređen u smislu propusnog opsega tek u devedesetim godinama, cloud računarstvo moralo je da sačeka sa realizacijom.

Jedan od prvih značajnijih trenutaka u cloud računarstvu bio je Salesforce.com, servis postavljen 1999, koji je pionirski poduhvat u isporuci aplikacije enterprajz nivoa putem veb-sajta. Ovaj primer [32] utro je put softverskim kompanijama u pogledu načina na koji se može isporučiti aplikacija krajnjem korisniku – putem interneta a preko običnog veb-pretraživača.

Naredni stepen u razvoju cloud sistema desio se sa pojavom Amazon veb-servisa 2002. godine (Amazon PR). Amazon je obezbedio svitu servisa baziranih na cloudu, uključujući tu i usluge skladištenja i obrade podataka, pa čak i usluge ljudske inteligencije kroz Amazon Mechanical Turk servis.

Nakon toga, Amazon 2006. godine lansira svoj Elastic Compute Cloud (EC2) kao komercijalni veb-servis koji omogućava malim kompanijama i pojedincima da unajme računare na kojima mogu da izvršavaju sopstvene aplikacije.

Naredni korak u razvoju cloud servisa desio se 2009. godine, sa pojavom Web 2.0 tehnologija, kada su Google i ostali počeli da nude enterprise aplikacije koje se izvršavaju putem veb-pretraživača. Primer jedne ovakve grupe aplikacija jesu Google Apps.

Ključni faktori koji su omogućili da cloud računarstvo evoluiraju jesu sazrevanje virtualizacionih tehnologija, razvoj univerzalnog mrežnog propusnog opsega velikih brzina i univerzalnih softverskih standarda koji se tiču interoperabilnosti.

Počeci onoga što je danas poznato pod nazivom cloud tehnologija datiraju iz vremena Mainframe računara šezdesetih godina prošlog veka kada je Džon Mekarti, predavač na Standfordu i MIT univerzitetu i dobitnik Tjuringove nagrade došao na ideju „uslužnog računarstva“ (eng. *Utility Computing*). Koncept je bio zasnovan na jednostavnom principu. Kao što se režijske usluge poput električne energije ili telefonske usluge naplaćuju, tako se i korišćenje računara može naplaćivati u zavisnosti od potrošnje. IBM je prepoznao i usvojio potencijal lukrativnog biznis modela u poslovanju zasnovanom na ovoj ideji i ubrzo zagospodario pružanjem računarskih usluga kompanijama. Ove usluge nisu bile nimalo jeftine. IBM je omogućavao kompanijama pristup i korišćenje velike snage Mainframe računara. Ti računari velike snage često su bili izuzetno skupi, velikih dimenzija i veoma složeni za održavanje, u toj meri da bi ih većina kompanija nije moglo priuštiti. Iz tog razloga su kompanije koje su imale potrebe za ovakvim računarskim resursima, radije plaćale usluge najma ovih, u to vreme, veoma skupih resursa.

Aktualizacijom personalnih računara koncept uslužnog računarstva postaje za većinu dotadašnjih klijenata stvar prestiža. Računari su generalno postali manji, brži, jeftiniji, te dostupniji i lakši za korišćenje. Strelovit razvoj mikroracunara je svakako doprineo deaktualizaciji skupih centralizovanih sistema. Sa druge strane, određeni faktori su bili od uticaja kada je u pitanju opstanak uslužnog računarstva. Kada su u pitanju brzina, snaga i prostor za skladištenje podataka, personalni računari su, u to vreme, imali ozbiljnih tehničkih nedostataka i nisu mogli da odgovore potrebama. Kompanije koje su imale veće zahteve su i dalje iznajmljivale Mainframe računare pa se koncept uslužnog računarstva održao profitabilnim još izvesno vreme. Ubrzani rast i razvoj Interneta promenio je način korišćenja računara i prenosa informacija. Sa davno nastalom idejom uslužnog računarstva kompanije kao što su *Google*, *Amazon*, kao i mnoge druge manje i više poznate kompanije, počele su da investiraju u svoju infrastrukturu kupujući farme servera kako bi svojim korisnicima ponudili što bolje i kvalitetnije servise i tako dali svoj veliki doprinos razvoju tehnologije oblaka.

2.2. Kako definišemo cloud

Gartner definiše cloud računarstvo kao: „vid računarstva gde se masivno skalabilni IT kapaciteti i mogućnosti nude u obliku usluge, korišćenjem internet tehnologija, velikom broju eksternih korisnika., a gde korisnici usluga treba da brinu samo o onome šta ta usluga može da učini za njih, a ne kako se ona implementira. Cloud nije arhitektura, platforma, alat, infrastruktura, internet sajt ili IT dobavljač” [33]. “To je novi računarski stil. Za podršku implementacije i korišćenje se mogu koristiti mnoge arhitekture. Na primer, moguće je koristiti cloud u privatnim preduzećima kako bi se konstruisali privatni cloud sistemi, ali postoji samo jedan javni cloud koji se bazira na internetu. “ [34].

ENISA (eng. *European Network and Information Security Agency*) definiše Cloud Computing [35]:

„Cloud računarstvo je „na zahtev“ model usluge za IT snabdevanje koji se najčešće zasniva na virtuelizaciji i distribuiranim računarskim tehnologijama. Cloud računarska arhitektura poseduje:

1. Visoko apstraktne resurse
2. Skoro trenutnu skalabilnost i fleksibilnost
3. Vršiti skoro trenutno rezervacije resursa
4. Zajedničke resurse (hardver, bazu podataka, memoriju...)
5. Uslugu na zahtev koju najčešće prati “plati prema korišćenju” sistem plaćanja
6. Programsko upravljanje putem aplikativnog programskog interfejsa (API)“

Zvanična definicija koju je objavio NIST (eng. *National Institute of Standards and Technology*), odnosno Odeljenje za računarsku bezbednost pri Ministarstvu trgovine u SAD-u i koja se smatra industrijskim standardom, počinje sa dve poruke koje upućuju na predostrožnost, a potom definišu cloud računarstvo nabranjem pet osnovnih karakteristika, tri modela usluge i četiri modela primene:

NIST definicija cloud računarstva [36]: “Cloud računarstvo je model koji omogućava pouzdan pristup na zahtev, mreži i zajedničkom skupu konfigurabilnih računarskih resursa (npr. mreže, serveri, skladišta, aplikacije i usluge), koji se mogu brzo nabaviti i osloboditi, uz minimalan napor ili interakciju davaoca usluge.“

Cloud računarstvo je još uvek pojam u razvoju. Njegova industrija predstavlja veliki ekosistem koga čine mnogi modeli, proizvođači i tržišta. Njegove definicije, primeri upotrebe, tehnologije koje ga podržavaju, problemi, rizici i prednosti biće još konstantno ažurirane kroz debate u javnom i privatnom sektoru. Postojeće definicije, obeležja i karakteristike će vremenom evoluirati i menjati se.

2.3. Cloud koncept

Cloud se odnosi na skup hardvera, mreža, sistema za skladištenje podataka, servisa i interfejsa koji omogućavaju isporuku računarskih usluga u vidu servisa. Cloud servisi omogućavaju isporuku softvera, infrastrukture i sistema za skladištenje podataka putem Interneta, bilo po komponentama, bilo u vidu celokupne platforme, u skladu sa korisničkim potrebama i/ili zahtevima.

Cloud računarstvo je isporuka računarskih servisa, nasuprot gotovog proizvoda, gde se deljeni resursi, softver i informacije isporučuju računarima i drugim uređajima u vidu merljivih servisa putem računarskih mreža, obično Interneta.



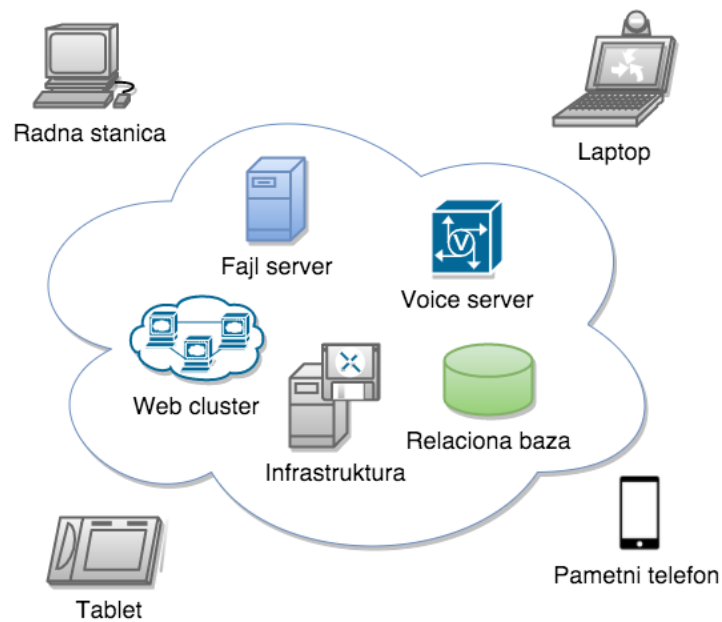
Slika 2 - Šta je cloud?

Cloud je apstrakcija računarskih resursa u toj meri da korisnik brine isključivo o usluzi koju dobavlja i o interfejsima koji mu služe u tu svrhu. Cloud računarstvo uključuje obradu podataka, softver, pristup podacima, i storidž resurse bez potrebe da korisnik cloud sistema zna i vodi računa o fizičkoj lokaciji podataka ili drugim detaljima računarskih resursa korišćenih u procesu.

Krajnji korisnici pristupaju cloud aplikacijama kroz veb-čitač ili kroz mobilnu aplikaciju napisanu za neku od mobilnih platformi, kao što su telefon i tablet, dok su poslovni softver i podaci skladišteni na serverima koji se nalaze na udaljenoj lokaciji (slike 2 i 3).

Cloud dobavljači teže da daju iste ili bolje usluge i performanse kao da su aplikacije instalirane na lokalnim računarima korisnika. S druge strane ono što je ključna odlika cloud servisa i ključni koncept ponude jesu iskustvo dobavljanja i korišćenja usluge kao i princip fleksibilnosti. Iako se često insistira na performansama čijem se dostignuću teži, najbolje stvari koje cloud sistemi nude su zapravo malopre pomenute.

U temeljima cloud računarstva je širi koncept konvergirane infrastrukture i deljenih resursa. Ovaj tip obradnog okruženja računarskog centra omogućava poslovnim entitetima da svoje aplikacije pokreću brzo i jednostavno uz jednostavnije upravljanje i manje održavanja. Ovo omogućava ICT službama da svoje resurse prilagođavaju rapidnom brzinom, uz jednostavniju manipulaciju i upravljanje istim. Postaje veoma lako dobiti potrebne skladišne, mrežne i računarske resurse kao i skoro jednako lako da se odgovori na fluktuacije i nepredvidivosti u tražnji za ovim resursima.



Slika 3 - Cloud računarstvo – dostupnost sa bilo kog uređaja i lokacije

U okviru cloud koncepta figurišu sledeći učesnici:

- **Krajnji korisnici sistema** ne moraju da znaju ništa o tehnologijama koje učestvuju u isporuci traženih servisa. Kada je manji biznis u pitanju, za korisnika, cloud dobavljač usluga deluje više kao data centar. Kod većih firmi, nadzire se ICT infrastruktura na nivou, kako internih resursa, tako i eksterno zakupljenih cloud servisa.
- **Biznis menadžment** mora da preuzme odgovornost za sveukupno upravljanje podacima i servisima koji se nalaze u cloud sistemu. Cloud servis dobavljači usluga moraju da pruže predvidiv i garantovan nivo usluge i bezbednosti svim stranama sa kojima saraduju.
- **Dobavljač cloud usluga** je odgovoran za celokupnu IT infrastrukturu i njeno održavanje [37].

Cloud koncept servisi su pravljani sa idejom da zadovolje princip višestrukog zakupa aplikacije/servisa (eng. Multitenancy). Ovaj princip se zasniva na konceptu višekorisničkog zakupa jedne instance aplikacije nasuprot konceptu multi-instance arhitekture gde se kreira posebna instanca za svakog korisnika aplikacije/usluge.

Cloud računarstvo se odnosi kako na aplikacije koje se isporučuju putem Interneta, tako i na hardverske sisteme koji obezbeđuju ove usluge višeg nivoa. Sami servisi se već dugo nazivaju softver kao usluga, SaaS, tako da ćemo koristiti ovaj termin. Hardver i softver u datacentrima je ono što nazivamo cloud sistemom.

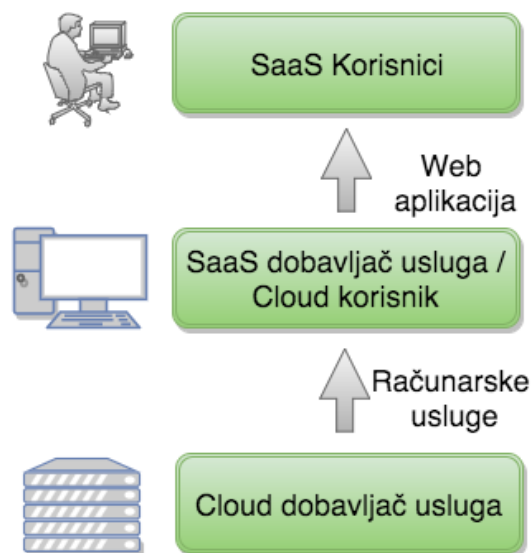
Kada je cloud sistem naplate baziran na principu “plati koliko koristiš” široj javnosti, takav sistem nazivamo javnim cloud sistemom. Sistem je prodavan kao utility servis. Aktuelni primeri javnog cloud sistema su Amazon Web Services, Google AppEngine i Microsoft Azure. Termin privatni cloud upotrebljavamo da označimo datacentre koji nisu dostupni široj javnosti. Cloud računarstvo je suma SaaS i utility računarstva, ali u principu ne uključuje privatni cloud.

Prednosti SaaS sistema, kako za krajnje korisnike, tako i za dobavljače usluga su vrlo jasne. Dobavljači usluga dobijaju znatno pojednostavljeniji način instalacije i održavanja softvera i centralizovanu kontrolu naspram verzioniranja. Krajnji korisnici imaju iskustvo dobavljanja servisa “bilo kad, bilo gde”, dele podatke i saraduju mnogo lakše držeći podatke u okviru bezbedne i održavane infrastrukture.

Cloud računarstvo ne menja ove stvari, međutim daje mogućnost dobavljačima SaaS usluga da bez formiranja sopstvenih datacentara nude ovakve servise, baš kao što je to slučaj sa ugovornom proizvodnjom u oblasti proizvodnje poluprovodničkih komponenti. Cloud računarstvo omogućava postavljanje SaaS i skaliranje na zahtev, bez da se gradi ili dobavlja datacenter. Analogno, kako korisnik može izvesnu kategoriju problema da prebaci na dobavljača SaaS usluga, tako SaaS dobavljač može da prebaci neke od svojih problema na cloud dobavljača (slika 4).

Sa hardverske tačke gledišta, postoji tri aspekta cloud računarstva [38]:

1. Iluzija o beskonačnim računarskim resursima raspoloživim na zahtev, te se na taj način eliminiše potreba korisnika cloud usluga da planira unapred resurse;
2. Eliminacija potrebe za unaprednim obavezivanjem cloud korisnika, što omogućava korisnicima da počnu kao veoma mali te da kasnije povećavaju računarske resurse, u skladu sa povećanjem obima potreba;
3. Sposobnost plaćanja upotrebljenih računarskih resursa za vrlo male vremenske jedinice (obrada podataka na sat i skladište podataka na dan) zajedno sa čuvanjem resursa, oslobađanjem kada njihova upotreba nije potrebna;



Slika 4 – Korisnici i dobavljači cloud računarstva

Razmotrićemo efekte cloud računarstva na cloud dobavljače usluga i SaaS dobavljače/cloud korisnike. Gornji sloj može biti rekurzivna, u tom smislu da SaaS dobavljači u isto vreme mogu biti i SaaS korisnici. Na primer, tzv. mashup dobavljač mapa može biti u isto vreme korisnik servisa Craigslist i Google maps.

Sve tri kategorije su vrlo bitne za tehnološke i ekonomske promene koje su omogućene cloud računarstvom. Svakako, u prošlosti su napori za dostizanjem koncepta uslužnog računarstva propali. Primećuje se da su jedna ili dve ključne karakteristike nedostajale. Na primer, Intel računarske usluge lansirane 2000-2001. godine, tražile su pregovore oko ugovora i obavezivanje na duži vremenski period, ne na sat.

Kao uspešan primer, imamo Elastic Compute Cloud (EC2) u okviru Amazon Web Servisa (AWS) gde se prodaje 1.0-GHz x86 ISA "delove" za 10 centi po satu, dok nova instanca ili "deo" može biti dodat za 2 do 5 minuta.

Amazonov Scalable Storage Service (S3) se naplaćuje \$0.12 do \$0.15 po gigabajt mesečno, sa troškom za dodatni propusni opseg od \$0.10 do \$0.15 po gigabajtu u i iz AWS preko Interneta. Amazonova strategija se zasniva na statističkom multipleksiranju višestrukih instanci u jednu fizičku mašinu, tako da fizička mašina može simultano biti korišćena od strane više mušterija koje se jedna drugoj neće uplitati u korišćenje.

Dok je privlačnost za cloud korisnika jasna (u ovom slučaju SaaS dobavljača usluga), ko bi želeo da postane cloud dobavljač i zašto? Za početak, shvatanje ekonomije skaliranja i priuštivog zahvaljujući statističkom multipleksiranju i velikim porudžbinama zahteva stvaranje izuzetno velikih računarskih centara.

Izgradnja, dobavljanje i lansiranje ovakvih centara je poduhvat koji se meri stotinama miliona dolara. S druge strane, zbog fenomena rasta veb-servisa tokom ranih 2000-tih, mnoge velike Internet kompanije, uključujući Amazon, eBay, Google, Microsoft i druge, već su izgradile ovakve sisteme. Takođe, ove firme su morale da izgrade skalabilnu softversku infrastrukturu (poput MapReduce, Google File Sistema, BigTable i Dynamo) i stvore operacionu ekspertizu da zaštite svoje računarske centre protiv potencijalnih fizičkih i elektronskih napada [39], [40], [41] i [42].

Prema tome, potreban ali ne i dovoljan uslov da se postane cloud dobavljač usluga podrazumeva veliku investiciju ne samo u datacentar, već i u softversku infrastrukturu i operacionu ekspertizu isto tako velikih dimenzija kako bi se uspešno upravljalo ovakvim centrom. Imajući u vidu uslove, veliki broj faktora može uticati na opredeljenje neke kompanije da postane dobavljač cloud usluga:

1. **Da zarade puno novca.** Iako 10 centi po server-satu izgleda malo, Tabela 1 sumira procenu Jamesa Hamiltona [43] da veliki data centri (desetine hiljada računara) mogu da kupe hardver, mrežni propusni opseg, i struju za 1/5 do 1/7 cene ponuđene datacentru srednje veličine (stotine ili hiljade računara). Zatim, fiksni troškovi razvoja softvera i razvoj mogu biti amortizovani kroz mnogo veći broj mašina/instanci. Drugi autori [44] i [45] procenjuju cenovnu prednost kao faktor 3 do 5. Na taj način, dovoljno velika kompanija bi mogla da iskoristi ekonomiju obima da ponudi uslugu daleko ispod cene datacentra srednje veličine i da i dalje ostvari dobar profit.
2. **Uticaj postojeće investicije.** Dodavanje cloud računarskih servisa povrh postojeće infrastrukture dodaje nov profit za (u idealnom slučaju) relativno malo uvećane troškove, pomažući na taj način amortizaciji velike investicije u datacentar. Doduše, po Werneru Vogelsu, CTO-u Amazona, mnogi Amazon Web Servisi su inicijalno razvijeni za interne potrebe.
3. **Odbrana franšize.** Kako konvencionalni serveri i enterprajz aplikacije budu prihvatali cloud računarstvo, proizvođači sa etabliranim franšizama u tim aplikacijama bi bili motivisani da ponude sopstvenu cloud opciju. Na primer, Microsoft Azure nudi odmah opciju migracije postojećih klijenata Microsoft enterprajz aplikacija u cloud okruženje.

Tabela 1 - Ekonomija obima za srednje i velike datacentre [43]

Tehnologija	Cena u datacentru srednje veličine	Cena u velikom data centru	Odnos
Mrežni resursi	\$95 za Mbit/sek/mesec	\$13 za Mbit/sek/mesec	7.1
Storidž	\$2.20 za GByte / mesec	\$0.40 za GByte / mesec	5.7
Administracija	cca 14 Servera / Administrator	> 1000 Servera / Administrator	7.1

Tabela 2 - Cena kilovat sata po regionu USA [46]

Cena po KWH	Lokacija	Najverovatniji razlog
3.6¢	Ajdaho, USA	Hidroelektrana, slanje na kratku razdaljinu
10.0¢	Kalifornija, USA	El. energija se transportuje na veliku daljinu kroz mrežu
18.0¢	Havaji	Neophodna dostava goriva kako bi se pravila el. energija

Vidimo da je faktor cene električne energije kao jedne od kompetitivnih prednosti, u slučaju srpske privrede, doduše sa subvencionisanom cenom, ujedno i potencijalno velik faktor privlačnosti greenfiled investicija za izgradnju datacentara/cloud dobavljača i generalno industrijskog biznisa koji se oslanja na potrošnju električne energije. Faktori pouzdanosti i redundantnosti sistema napajanja su van okvira razmatranja ovog rada.

Povodom održanja cena na trenutnom nivou, u Srbiji se predviđa usklađivanje cena električne energije sa evropskim cenama. Harmonizacija cena je inicijalno bila postavljena do 2012. godine, a zatim pomerena na 2015. Kako bilo, čak i tada, predviđa se da će Srbija imati cene koje su svakako među nižim u regionu, svakako i u Evropi [47].

Tabela 3 - Cene el. energije u Evropi za potrošnju od 2GWh/godini [48]

Zemlja	€ po kWh El. energija
Austrija	€ 0.1213
Belgija	€ 0.1182
Bugarska	€ 0.0746
Kipar	€ 0.1822
Republika Češka	€ 0.1195
Danska	€ 0.1091
Estonija	€ 0.0817
Finska	€ 0.0784
Francuska	€ 0.0763
Nemačka	€ 0.1340
Grčka	€ 0.1188
Mađarska	€ 0.1194
Irska	€ 0.1303
Italija	€ 0.1565
Letonija	€ 0.1015
Litvanija	€ 0.1185
Luksemburg	€ 0.1180
Malta	€ 0.1927
Holandija	€ 0.1181
Poljska	€ 0.1142
Portugal	€ 0.1064
Rumunija	€ 0.0925
Slovačka	€ 0.1327
Slovenija	€ 0.1162
Španija	€ 0.1271
Švedska	€ 0.0887
Ujedinjeno Kraljevstvo	€ 0.1149

Cene iz Tabele 3 date su za sektor industrije uz godišnju potrošnju od oko 2GWh po godini i bez PDV-a, koji se razlikuje od zemlje do zemlje.

- Napasti poziciju.** Kompanija koja ima datacentar i softverske resurse može želiti da postavi mostobran u ovom prostoru pre nego što se na tržištu pojavi „teškaš“. Google AppEngine

obezbeđuje alternativni put u cloud čija privlačnost leži u automatizaciji mnogih opcija vezanih za skalabilnost i balansiranje opterećenja koje bi razvojni timovi inače, morali sami da prave.

5. **Uticaj na odnose sa potrošačima.** IT servisno orijentisane organizacije poput IBM Globalnih Servisa imaju kroz svoju ponudu servisa i širok spektar odnosa sa korisnicima. Pod pretpostavkom da bi jedno ovakvo brendirano cloud rešenje dalo kompanijama umirujući osećaj sigurnosti to bi značilo investiciju u partnerske odnose za obe strane.

6. **Postati platforma.** Inicijativa kompanije Facebook za omogućavanje plug-in dodataka je odlična stvar za cloud sisteme. Kao što je poznato, jedan dobavljač plug-in usluga za Facebook je Joyent, cloud provajder. Ipak, Facebook je imao motivaciju da napravi svoju aplikaciju za društveno umrežavanje kao razvojnu platformu.

Nekoliko cloud računarskih (kao i konvencionalnih) centara se grade na lokacijama koje su dosta izmeštene. Motivacija za ovakve poteze nalazi se u ceni električne energije, hlađenja, cene rada, cene nekretnina kao i visine poreskih stopa. Svi pobrojani troškovi utiču sa oko trećinu u celokupnoj investiciji za projekat. Tabela 2 prikazuje cene na različitim lokacijama u SAD [44], dok Tabela 3 daje pregled cena u Evropi.

Na kraju, fizika kaže da je lakše transportovati fotone nego elektrone; drugim rečima, jeftinije je slati podatke putem optičkih kablova nego transportovati električnu energiju putem visoko naponskih prenosnika.

2.4. Razvoj cloud sistema

Cloud računarstvo ima potencijal da transformiše veliki deo ICT industrije, čineći softver još atraktivnijim, nudeći ga kao servis i oblikujući način na koji ICT hardver biva dizajniran i kupovan. Razvojni timovi sa inovativnim idejama ka razvoju Internet servisa više nemaju potrebu za kapitalnim investicijama za hardver ili ljudskim resursom kako bi upravljali tom infrastrukturom. Ne postoji potreba za brigom oko prevelike ili premale alokacije resursa za servisom čija popularnost ne bi bila usklađena sa predviđenom, te bi na taj način skupi resursi ostali neiskorišćeni. Drugi scenario je nedovoljna alokacija resursa za servis čija popularnost nenadano poraste, te se na taj način izgubi na broju klijenata i profitu. Štaviše, kompanije sa velikom količinom obrade podataka, mogu ostvariti rezultate onoliko brzo koliko su njihove aplikacije u mogućnosti da skaliraju, jer u cloudu, korišćenje 1,000 servera 1 sat ne košta više od korišćenja jednog servera 1,000 sati. Ova elastičnost resursa bez dodatnog plaćanja za skaliranje jeste presedan u istoriji informacionih tehnologija.

Cloud računarstvo se odnosi na aplikacije koje se isporučuju kao usluge putem Interneta i na hardver i sistemski softver u datacentrima koji pružaju ove usluge. Ovakve usluge se nazivaju softver kao usluga (eng. *Software as a Service*, SaaS). Datacentar, hardver i softver čine ono što nazivamo cloudom. Ukoliko je ovakav jedan cloud dostupan široj javnosti i zasnovan na principu plaćanja po osnovu korišćenja, tada ga nazivamo javnim cloudom. Servis se zapravo prodaje kao režijski trošak, poput struje, vode, gasa (eng. *Utility*). Termin Private cloud se koristi kada se misli na datacentre organizacija koje ih nisu stavile na raspolaganje široj javnosti. Tako, cloud računarstvo je suma SaaS i uslužnog računarstva, ali ne uključuje privatne cloudove. Ljudi mogu biti korisnici i dobavljači SaaS, ili korisnici ili dobavljači uslužnog računarstva.

Smatra se da je izgradnja i upravljanje izuzetno velikim data centrima na jeftinim lokacijama bila ključna stvar za omogućavanje cloud računarstva, jer su tokom eksploatacije ovakvih sistema otkrivene uštede na nivou faktora množenja od 5 do 7 puta u uštedama u troškovima struje, mrežnog saobraćaja, upravljanja, softvera i hardvera u okviru ovako velikih ekonomskih skala. Ovi faktori, kombinovani sa statističkim multipleksiranjem, kako bi uvećali iskorišćenje u poređenju sa privatnim cloudom, znače da bi cloud mogao da ponudi usluge ispod cene datacentra srednje veličine i da pritom i dalje ostvari dobar profit.

Svakoj aplikaciji je potreban model obrade, model skladištenja i model komunikacije. Statističko multipleksiranje potrebno da se dostigne elastičnost i iluzija beskonačnog kapaciteta zahteva da svaki od ovih resursa bude virtualizovan kako bi se sakrila implementacija (kako su multipleksirani i deljeni). Imajući prethodno u vidu, nameće se ideja da će se različite ponude utility računarstva bazirati na osnovu nivoa apstrakcije prezentovane programeru i u zavisnosti od nivoa upravljanja resursima.

Amazon EC2 je na jednoj strani spektra. EC2 instanca je u priličnoj meri nalik fizičkom hardveru, gde korisnik može da kontroliše skoro ceo softverski stek, od kernela na gore. Ovaj nizak nivo pristupa otežava Amazonu da ponudi automatsku skalabilnost i podršku u slučaju otkaza, jer su parametri u upotrebi za ovu vrstu usluge u direktnoj sprezi sa aplikacionim slojem. Na drugom kraju suprotnosti su domensko specifične aplikacije poput Google AppEngine. AppEngine cilja ekskluzivno na tradicionalne web aplikacije, zahtevajući aplikacionu strukturu čiste separacije između računarskog sloja koji ne pamti korake unazad i skladišnog sloja koji pamti korake unazad. AppEngine impresivno automatsko skaliranje i mehanizmi visoke dostupnosti, kao i Googleova tehnologija MegaStore skladište podataka na raspolaganju AppEngine aplikacijama. Aplikacije za Microsoftov Azure su pisane korišćenjem .NET biblioteka, i kompajlirane u Common Language Runtime, jezički nezavisno upravljanoj okruženju. Tako, Azure predstavlja sloj između aplikacionih frameworka poput AppEngine i hardverski virtualnih mašina poput EC2.

Kada je servisno, odnosno *utility* računarstvo, bolje rešenje od gradnje privatnog clouda? Prvi slučaj je kada servis varira sa vremenom. Dobavljanje resursa data centra u trenutku maksimalne tražnje za uslugama vremenom varira. Dobavljanje resursa datacentra i njegovo održavanje za trenutak maksimalne tražnje, koji se moraju održati par dana u mesecu, vodi neiskorišćenosti resursa. To potencijalno vodi do finansijskih ušteda čak i ukoliko je cena najma cloud mašina viša od gradnje sopstvene infrastrukture.

Drugi slučaj je kad je zahtev za resursima unapred nepoznat. Na primer, veb startup kompanija ima potrebu da podrži skok u tražnji u trenutku kada postaje popularna, gde bi se kasnije smanjila tražnja nakon što neki od posetilaca prestanu da koriste servis. Na kraju, organizacije koje vrše paketnu analitiku mogu da koriste uslugu cloud obrade podataka kako bi se posao obavio brže i to tako što bi umesto najma jedne mašine na 1,000 sati najmili 1,000 mašina na jedan sat.

Za prvi slučaj web startup kompanije sa varirajućim zahtevom kroz vreme i profitom proporcionalnom sa brojem korisničkih sati, jednačina izgleda kao jednačini ispod [49]:

$$UkorisniSat_{cloud} \times (\text{prihod} - \text{Cena}_{cloud}) \geq UserSat_{datacenter} \times (\text{prihod} - \text{Cena}_{datacenter} / \text{Iskorišćenje})$$

Leva strana množi neto prihod po korisnik času po broju sati, dajući očekivani profit od korišćenja cloud rešenja. Desna strana vrši istu kalkulaciju za datacenter sa fiksnim kapacitetom uz faktorisanje prosečnog korišćenja, uključujući normalne režime rada datacentra. Strana koja je veća predstavlja mogućnost većeg profita.

Predviđa se rast cloud računarstva, što bi razvojni timovi trebalo da uzmu u obzir. Svi nivoi bi trebalo da ciljaju na horizontalnu skalabilnost virtualnih mašina, naspram efikasnosti pojedinačne virtualne mašine.

S tim u vezi, još nekoliko napomena:

1. Aplikacioni softver treba da poseduje osobinu da bude sposoban da se skalira na gore i na dole rapidnim tempom, što je nov zahtev. Ovakav softver treba da poseduje sistem licenciranja baziran na upotrebi, "plati koliko koristiš" kako bi odgovarao konceptu cloud sistema.
2. Infrastrukturni softver se više ne nalazi/izvršava direktno na metalu (hardveru), već se nalazi u okviru virtualne mašine. Štaviše, treba da ima inkorporiran sistem za naplatu (eng. *billing software*).
3. Hardverski sistemi treba da budu dizajnirani/dimenzionisani kao kontejneri. Kontejner bi sadržao desetak računarskih ormara jer su to minimalne porudžbine koje su efikasne za ovakve obradne centre. Cena upotrebe ovakvog sistema će se tada izjednačiti sa performansama i kupovnom cenom po svojoj važnosti, nagrađujući u energetske proporcionalnosti poput stavljanja na stand by režim rada neupotrebljene memorijske, procesorske i skladišne resurse u režim niske potrošnje energije. Procesori bi trebalo da rade dobro sa virtualnim mašinama. Fleš memorija bi trebalo da bude dodata u memorijsku hijerarhiju a LAN switching i WAN ruting oprema se moraju poboljšati u smislu propusnog opsega i cene koštanja.

2.5. Aktualizacija cloud koncepta

Cloud računarstvo je novi termin za dugo čekanu osobinu računarstva da se usluge i servisi isporučuju na isti način kao i struja i voda i gde se naplata vrši na osnovu korišćenja usluge. Ovo se odnedavno obistinilo kao komercijalna stvarnost. Cloud računarstvo će verovatno imati isti uticaj na softver kao što su livnice imale na tešku industriju.

Tabela 4 - Pregled 10 prepreka i prilika za rast cloud računarstva (Ambrust M., 2009)

	Prepreka	Šansa
1	Dostupnost usluge	Koristiti više dobavljača cloud usluga; koristiti elastičnost u amortizovanju i suzbijanju DdoS napada
2	Zaključavanje podataka	Standardizovati API-je; Kompatibilan softver za surge computing/hibridni cloud
3	Poverljivost podataka i mogućnost revizije	Enkripcija podataka, VLANovi, fajerwol geografsko skladište podataka - storidž
4	Uska grla u prenosu podataka	Fizičko slanje diskova; backup/arhiviranje; mrežna oprema sa većim propusnim opsegom
5	Nepredvidivost performansi	Poboljšana podrška za virtualne mašine; fleš memorija; Gang scheduling virtualnih mašina
6	Skalabilno skladište podataka	Pronalazak skalabilnog storidža
7	Greške u velikim distribuiranim sistemima	Pronalazak debugger alata koji se oslanja na distribuirane virtualne mašine
8	Brzo skaliranje	Pronalazak auto-skalera koji se oslanja na mašinski jezik; snimanje statusa za čuvanje stanja sistema
9	Deljenje reputacije	Nuđenje servisa za zaštitu reputacije poput onih za elektronsku poštu
10	Softversko licenciranje	Naplata po korišćenju; prodaja količinske upotrebe

Pre izvesnog vremena, vodeći proizvođači hardvera su imali svoje pogone za proizvodnju poluprovodničkih komponenti. Trebalo je da proizvođači budu dovoljno veliki kako bi mogli da priušte izgradnju i operativne troškove ovakvog jednog pogona. Međutim, cena opreme se sa svakom novom generacijom udvostručavala. Danas su troškovi fabričke linije za proizvodnju poluprovodnika na nivou investicije od 3 milijarde dolara.

Zbog ovakve situacije, samo mali broj proizvođača sa izuzetno visokim obimom proizvodnje, poput Intela i Samsunga, može cenovno da opravda posedovanje i korišćenje ovakvih pogona.

Ovakva situacija je dovela do pojave poluprovodničkih livnica koje su pravile čipove za druge. Najpoznatiji primer je Taiwan Semiconductor Manufacturing Company (TSMC). Ovakve livnice omogućavaju postojanje firmi koje prave poluprovodničke komponente a da pritom nemaju sopstvene proizvodne pogone. Njihova vrednost je, pre svega, u inovativnom dizajnu. Firma poput nVidie danas može da bude uspešna na tržištu čipova a da pri tom nema investicione i operativne troškove kao i rizike pridružene uz posedovanje vrhunskog proizvodnog pogona za tu namenu.

Obrnuto, ovakve fabrike poluprovodnika mogu da vrše vremensko multipleksiranje ugovorene proizvodnje za više različitih firmi koje rade isključivo dizajn, ne i proizvodnju poluprovodničkih komponenti. To ovim ugovornim proizvođačima omogućava eliminaciju rizika proizvodnje neuspešnog proizvoda i uspešnu amortizaciju operacionih troškova. Slično tome, prednosti ekonomije obima i statističkog multipleksiranja vode izdvajanju nekolicine cloud dobavljača usluga koji će moći da amortizuju troškove svojih centara za obradu podataka na taj način što će se baviti isključivo dizajnom cloud rešenja, bez da imaju svoju infrastrukturu. Neka od pitanja koja se javljaju su:

- Šta je cloud računarstvo i kako je drugačije od prethodnih paradigmi kao što je SaaS?
- Zašto je cloud računarstvo spremno u ovom trenutku za izlazak na tržište dok to u prošlosti nije bilo tako?
- Šta je potrebno da se postane cloud dobavljač usluga i zbog čega bi kompanija želela da postane dobavljač cloud usluga?
- Koje su to potencijalne šanse ili potencijalni podsticaj za razvoj cloud računarstva?
- Kako bismo mogli da klasifikujemo tekuće ponude cloud usluga, kroz spektar ponude, i kako se tehnički i poslovni izazovi razlikuju u zavisnosti gde se data ponuda nalazi na spektru?
- Koji su, ako ih ima, novi ekonomski modeli u okviru cloud računarstva, i kako se dobavljač računarskih usluga odlučuje da li da pređe na cloud koncept ili da ostane u privatnom klasičnom data centru?
- Koje su prepreke za uspeh cloud računarstva i odgovarajuće šanse za prevazilaženje ovih prepreka?
- Na koji način bi trebalo izmeniti budući razvoj softvera da bi odgovorio adekvatno na zahteve cloud računarstva?

Iako se zagovara stanovište da je izgradnja i rad velikih datacentara bila ključni faktor za stvaranje cloud računarstva, dodatni tehnološki trendovi i novi poslovni modeli takođe su igrali značajnu ulogu u

ostvarivanju cloud koncepta. Jednom kada je cloud računarstvo “uzletelo”, nove poslovne mogućnosti i modeli upotrebe su otkriveni.

2.6. Tehnološki trendovi i poslovni modeli

Zajedno sa pojavom Web 2.0 desilo se pomeranje sa poslovnog modela koji podrazumeva visoku marginu profita i visoko obavezivanje na nisku marginu i nizak stepen obavezivanja, kao i kontekst samousluživanja, odnosno samodobavljanja resursa. Na primer, u Web 1.0, prihvatanje plaćanja kreditnom karticom od nepoznatog lica je zahtevalo ugovor sa servisom za procesiranje plaćanja poput VeriSign ili Authorize.net. Ovakav odnos bio je deo većeg poslovnog odnosa, čineći ceo proces tegobnim za pojedinca ili manji biznis u kontekstu prihvatanja plaćanja kreditnom karticom putem Interneta.

Sa pojavom PayPal servisa, svakoj osobi postaje omogućeno da prihvati plaćanje kreditnom karticom bez ugovora i bez dugoročnog obavezivanja, uz veoma simbolične provizije za online transakcije. Nivo dodira koji ovi servisi nude je minimalan do nepostojeći, ali izgleda da činjenica da su servisi na dohvat ruke pojedincima čini nedostatak dobre podrške manje bitnim.

Slično, individualne veb-strane sada mogu da koriste Google AdSense, kako bi generisale prihod od reklamiranja, pre nego da sklapaju ugovor sa recimo, kompanijom DoubleClick (sada kupljenom od strane Googla). Ove reklame mogu da obezbede poslovni model za Web 2.0 takođe. Pojedinci mogu da distribuiraju web sadržaj koristeći Amazon CloudFront pre nego da se oslanjaju na distributere sadržaja poput Akamai servisa.

Amazon Web Servisi su uveli “plati na osnovu korišćenja” bez potrebe za ugovorom. Sve što klijentu treba jeste kreditna kartica. Druga inovacija se odnosi na prodaju virtualnih mašina na hardverskom nivou, omogućavajući klijentu da izabere softver čije izvršavanje nema uticaja na aplikacije drugih klijenata, iako se deli ista hardverska infrastruktura. Sve ovo utiče na dalje sniženje cene.

2.7. Potencijalne niše primene

Iako tek treba da se pojave fundamentalno novi tipovi aplikacije omogućeni cloud računarstvom, veruje se da će nekoliko novih, bitnih, trenutno postojećih klasa aplikacija postati još značajnije. Kada je Džim Grej vršio istraživanje tehnoloških trendova u 2003. godini [50] [51], zaključio je da je ekonomska neophodnost postavljanje podataka blizu aplikacija, zbog toga što cena povezivanja WAN mreža pada sporije nego cena ostalog IT hardvera. Iako se cena hardvera od tada promenila, suština nije. S tim zapažanjem u vezi, razmotrićemo neke dobre prilike za ekspanziju cloud računarstva.

Mobilne interaktivne aplikacije. Tim O'Rajli veruje da „budućnost pripada servisima koji odgovaraju u realnom vremenu na informacije obezbeđene od strane njihovih korisnika ili ne-ljudskih senzora“ [52]. Takvi servisi bi bili privučeni u cloud ne samo zbog toga što moraju biti visoko dostupni, već takođe zbog toga što se ovi servisi generalno oslanjaju na velike količine podataka koje se najbolje hostuju u velikim datacentrima. Ovo je posebno slučaj za servise koji kombinuju dva ili više izvora informacija ili servisa, tzv. Mashup servisi. Iako nisu svi mobilni uređaji povezani na cloud 100%, izazov sa trenucima bez konekcije ka Internetu je adresiran uspešno u različitim aplikacionim domenima, te se ovo ne vidi kao poseban i velik problem.

Paralelna paketna obrada. Iako je do sada u razmatranju bio fokus na korišćenje cloud računarstva za interaktivne SaaS servise, cloud računarstvo pruža jedinstvenu mogućnost za paketnu obradu podataka i analitičke poslove koji analiziraju terabajte i čija obrada može trajati satima. Ukoliko postoji dovoljno paralelne obrade podataka u aplikaciji, korisnici mogu iskoristiti prednost cloud “cenovne asocijativnosti”. Koristeći stotine računara za kratak period košta kao i korišćenje nekoliko računara za duži vremenski period. Primera radi, Peter Harking, stariji inženjer u *The Washington Postu*, iskoristio je 200 Amazon EC2 instanci (1,407 server sati) da prebaci 17,481 strana putnih dokumenata Hilari Klinton u formu jednostavniju za prikaz na vebu, samo devet sati nakon što su objavljeni (AWS, 2009). Programske apstrakcije poput Google MapReduce i njegove alternative otvorenog izvornog koda Hadoop [39] omogućavaju programerima da izvedu ovakve operacije a da pri tom od njih sakriju operativnu kompleksnost upravljanja paralelnim izvršavanjem stotinama cloud servera. Svakako, Cloudera [53] krči put svojoj komercijalnoj verziji u ovom tržišnom segmentu. Još jednom, koristeći Grejevo shvatanje, analizom troškova i koristi potrebno je izmeriti cenu migracije velikih količina podataka u cloud sisteme nasuprot dobrobiti od potencijalnog ubrzanja analize podataka.

Kada se vratimo na ekonomski model od ranije, špekulišemo da deo Amazonove motivacije za hostovanje velike količine podataka besplatno [54] leži u ublažavanju cenovne strane ove analize, odnosno u privlačenju korisnika da kupe obradne računarske cikluse, uz podatke korisnika koji su već u cloud sistemu.

Porast analitike. Specijalni slučaj računarski intenzivne paketne obrade je poslovna analitika. Dok je ranije industrijom velikih baza podataka originalno dominirala transakciona obrada podataka, ovakvi zahtevi danas opadaju. Rastući udeo računarskih resursa se sada troši na razumevanje klijenata, lanaca nabavke, kupovnih navika, sistema rangiranja, itd. Otuda, iako će obim online transakcija nastaviti da raste lagano, podrška u odlučivanju raste rapidno, pomerajući balans resursa u obradi podataka sa transakcionog na poslovnu analitiku.

Proširenje računski intenzivnih desktop aplikacija. Poslednje verzije matematičkih softvera poput *Matlaba* i *Matematike* mogu da koriste cloud okruženje radi obrade skupih evaluacija. Slične aplikacije mogu identično da imaju koristi od proširenja u cloud. Naravno, razuman test je upoređivanje cene računarstva u cloudu plus cena migracije podataka u i iz cloud sistema naspram uštede u vremenu ukoliko se podaci već nalaze u cloud sistemu. Simbolična matematika uključuje veliku količinu obrade računarskih podataka po jedinici obrade podataka, čineći ga domenom vrednim istraživanja. Interesantan alternativni model mogao bi biti da se podaci drže u cloud sistemu a da se on oslanja na dovoljni propusni opseg kako bi se prezentacioni deo/GUI omogućio korisniku. Offline renderovanja slika ili 3D animacija može biti sličan primer. Imajući kompaktan opis objekata, 3D scena i karakteristika svetlosnih izvora, renderovanje slika predstavlja izuzetno paralelizovan zadatak sa visokim odnosom računanja po bajtu.

Aplikacije “vezane za zemlju”. Neke aplikacije koje bi inače bile dobri kandidati za cloud elastičnost i paralelizam mogu biti osujećene troškom migracije, granicom kašnjenja pri seljenju u ili iz cloud sistema, ili oba. Na primer, dok je analitika povezana sa dugoročnim finansijskim odlukama vezanim za cloud, trgovanje na berzi koje zahteva obradu u realnom vremenu i preciznost na nivou mikrosekunde to nije. Dok cena (i verovatno kašnjenje) prenosa podataka preko WAN mreže ne padne, ovakve aplikacije mogu biti manje očigledni kandidati za cloud.

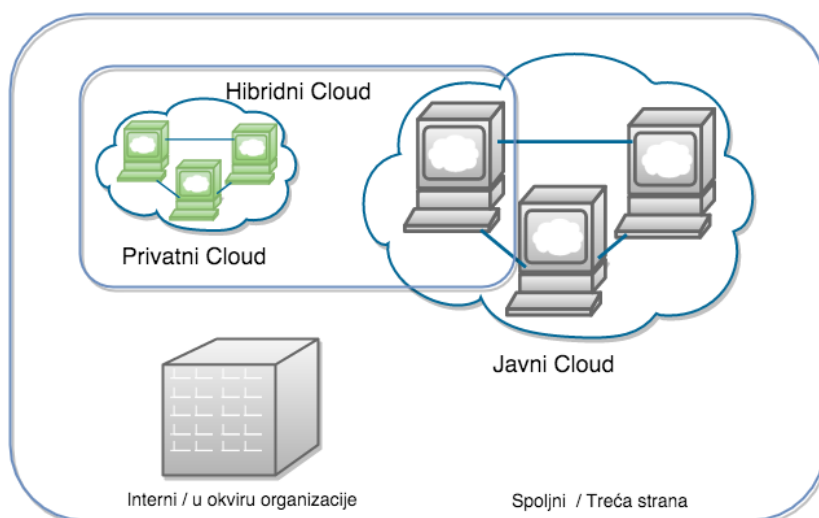
3. Uslužno naspram servisnog računarstva

Cloud računarstvo predstavlja drugačiji pristup od tradicionalnog u domenu isporuke informacionih tehnologija i usluga u okviru tih tehnologija. Cloud nudi novi način razmišljanja kako se određena poslovna potreba rešava i isporučuje kroz informacione tehnologije. On podrazumeva više različitih stvari kao što su novi interfejsi i način isporuke usluge. Kroz ovo poglavlje obrađujemo hipoteze H2 i H3.

Za cloud računarstvo se kaže da predstavlja narednu fazu u evoluciji Interneta. Pod delom *cloud*, se u cloud računarstvu podrazumeva način da se bilo šta iz sfere računarstva, od računarske snage, aplikacija, servisa pa do infrastrukture isporuči na zahtev, bez obzira na lokaciju korisnika i onda kada god je to korisniku potrebno. Ugrubo možemo podeliti cloud na tri različite forme (slika 5):

- Privatni cloud
- Javni cloud
- Hibridni cloud

Glavna zajednička odlika sva tri koncepta/forme clouda odnosi se na elastičnost - osobinu da u skladu sa potrebama za resursima/uslugama korisnik alokira ili oslobodi (eng. *Devision*) resurse. U isto vreme, ovo je jedan od ključnih razloga za opredeljenje korisnika za prelazak na cloud koncept.



Slika 5 - Tipovi cloud računarstva

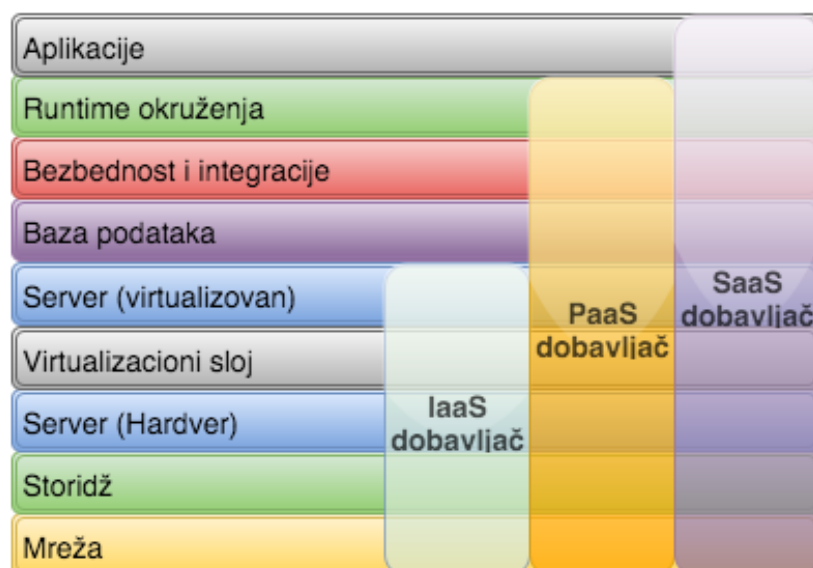
U okviru tradicionalnih datacentara, takođe je moguće alocirati i osloboditi resurse, međutim ne i na automatizovan način tako da se korisnik samooposluži.

Ova evolucija u cloud računarstvu, koja uveliko traje, može kompletno da promeni način na koji kompanije koriste tehnologiju da bi opsluživale klijente, partnere i dobavljače. Neke kompanije imaju svoju ICT infrastrukturu potpuno u cloudu. Ovakve kompanije veruju da dobijaju model koji je efikasniji, te da bolje odražava uloženu investiciju u isporuku ICT servisa.

Uvođenje clouda ne znači automatski da će aplikacije, servisi i procesi automatski biti migrirani na cloud rešenja. Mnoge kompanije su vrlo obazrive i analiziraju detaljno svoje strateške poslovne procese i intelektualnu svojinu kako bi utvrdile koja tehnološka infrastruktura treba da ostane pod kompanijskim direktnim nadzorom a koja će biti preseljena u cloud rešenje.

Cloud servis je opšti termin za bilo šta što uključuje isporuku hostovanih servisa putem Interneta. Ovi servisi su, šire gledano podeljeni u tri osnovne kategorije (slika 6):

- Infrastruktura kao servis (Infrastructure as a service, **IaaS**),
- Platforma kao servis (Platform as a service, **PaaS**) i
- Softver kao servis (Software as a service, **SaaS**).



Slika 6 - SPI Model (SaaS, PaaS, IaaS)

Naziv cloud potekao je od načina kako se Internet predstavlja na dijagramima i crtežima – kao oblak (eng. *cloud*).

Cloud ima tri karakteristike koje ga razdvajaju od tradicionalnih sistema u kojima se hostuje infrastruktura. Prvo, resursi se iznajmljuju na zahtev, neretko na sat ili čak minut. Drugo, vrlo je elastičan servis. Korisnik može da zakupi koliko god malo ili mnogo resursa da mu je potrebno i da količinsko angažovanje resursa menja kroz vreme po želji. Treće, upravljanjem (održavanjem) resursa se bavi dobavljač usluga, ne korisnik.

Faktori koji su uticali na stvaranje koncepta cloud computinga i njegovu popularizaciju su inovacije u domenu virtualizacije i distribuiranog računarstva, jeftiniji širokopolasni (eng. *broadband*) pristup Internetu, i na kraju, nepovoljna ekonomska situacija.

Cloud može biti javni, privatni i hibridni. Javan cloud poseduje resurse koji su javno dostupni svima. Privatni cloud označava mrežne servise ili datacenter hostovanih servisa u privatnom vlasništvu koji pruža usluge ograničenom broju ljudi. Kada isporučilac usluga (eng. *Service provider*) koristi resurse javnog clouda da napravi svoj privatni cloud, rezultat se zove virtualni privatni cloud. Privatni ili javni, cilj cloud računarstva je da pruži lak, skalabilan pristup računarskim resursima i ICT uslugama.

3.1. Karakteristike i model isporuke

Cloud računarstvo je model za omogućavanje pristupa sveprisutnom deljenom skupu računarskih resursa (npr. mreže, serveri, storidži, aplikacije i servisi) putem odgovarajućih mrežnih pristupnih resursa. Resursi kojima se pristupa mogu biti veoma obezbeđeni i pušteni uz minimalan napor i interakciju kako isporučioaca usluge, tako i naručioaca, što je možda i važnije.

Ovakav jedan cloud model je sastavljen of pet osnovnih karakteristika, tri servisna modela i četiri modela razvoja. Osnovne karakteristike:

Samousluživanje na zahtev – Krajnji korisnik može jednostrano da obezbedi računarske usluge, poput serverskog obradnog vremena ili mrežnog sistema skladištenja podataka, i to po potrebi, a bez ljudske interakcije sa dobavljačem usluga.

Širok mrežni pristup – Mogućnosti su na raspolaganju kroz mrežu i pristupa im se kroz standardne mehanizme koji promovišu upotrebu od strane heterogenih punih ili tankih klijentskih platformi poput mobilnih telefona, tableta, laptopova i radnih stanica.

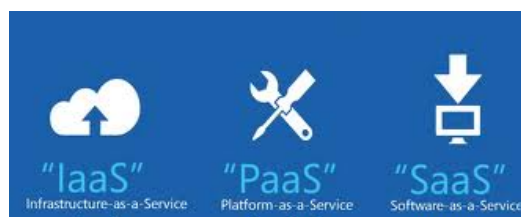
Udruživanje resursa – Računarski resursi dobavljača usluga se udružuju u službi upotrebe u višeklijentskom okruženju, sa različitim fizičkim i virtualnim resursima dinamički dodeljenim i raspoređivanim u skladu sa korisničkim zahtevima. Postoji vrsta nezavisnosti od lokacije u smislu da korisnička strana nema kontrolu niti je upoznata sa fizičkom lokacijom dodeljenih mu resursa ali može biti u mogućnosti da specificira lokaciju na višem nivou apstrakcije (npr. zemlja, država ili centar za obradu podataka). Primeri resursa uključuju storidž, obradu, memoriju i mrežni propusni opseg.

Rapidna elastičnost – Mogućnosti mogu biti elastično obezbeđene i oslobođene, u nekim slučajevima automatski, sa skaliranjem ka spolja ili unutra proporcionalnoj tražnji. Iz korisničke perspektive, resursi često deluju neogranično i mogu biti prisvojeni u bilo kom obimu i vremenu.

Merena usluga – Cloud sistemi vrše automatsku kontrolu i optimizaciju korišćenja resursa usklađivanjem mernih mogućnosti na nekom nivou apstrakcije adekvatnom za dati tip usluge (storidž, obrada, propusni opseg, aktivni korisnički nalozi). Upotreba resursa može biti praćena, kontrolisana, prijavljena, obezbeđujući transparentnost, kako za dobavljače, tako i za korisnike usluga.

4. Softver-Platforma-Infrastruktura (SPI) servisni model

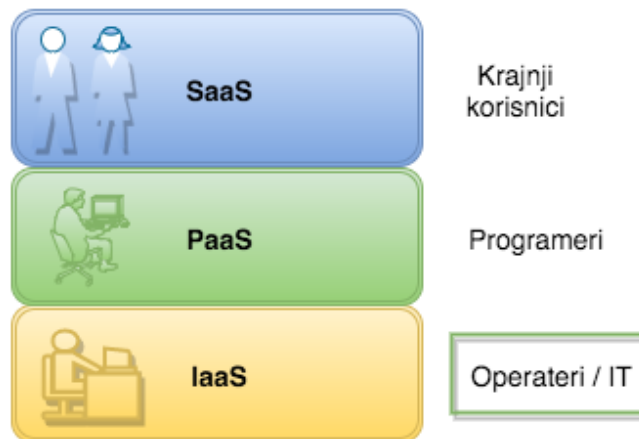
Cloud infrastruktura je kolekcija hardvera i softvera koji omogućavaju pet osnovnih karakteristika cloud računarstva. Cloud infrastruktura se može posmatrati kao nešto što sadrži fizički i apstraktni sloj. Fizički sloj sadrži hardverske resurse neophodne za obezbeđivanje cloud usluga i tipično sadrži server, sistem skladištenja podataka i mrežne komponente. Apstraktni sloj sadrži softver postavljen na fizičkom sloju, koji ispoljava osnovne cloud karakteristike. Konceptijski, cloud sloj se nalazi iznad fizičkog sloja. Kroz ovo poglavlje adresiramo hipotezu H3. Slika 7 predstavlja tri komponente SPI modela.



Slika 7 – SPI Model: Hostovanje-Izgradnja-Korišćenje (preuzeto sa Interneta)

4.1. Softver kao servis (SaaS)

U okviru isporuke **softvera kao usluge** (SaaS, eng. *Software as a Service*), dobavljač obezbeđuje hardversku infrastrukturu i potreban softver. Interakcija na relaciji korisnik – dobavljač odvija se putem portala. Spektar usluga iz domena isporuke softvera kao usluge je veoma širok. Kako dobavljač usluga hostuje, ne samo aplikaciju, već i podatke, korisnik je u mogućnosti da usluge sistema koristi sa bilo koje lokacije.



Slika 8 - Mapiranje sloja na korisnike / operatere

Klijentu se pruža mogućnost korišćenja dobavljačevih aplikacija koje se izvršavaju na cloud infrastrukturi (slika 8). Aplikacijama se pristupa kroz razne klijentske uređaje ili tanke klijentske interfejse, poput veb-pregledača (npr. web email), ili programski interfejs (slika 9). Korisnik ne upravlja niti kontroliše niže slojeve cloud infrastrukture uključujući mrežu, servere, operativne sisteme, storidže, ili čak pojedinačne mogućnosti aplikacije. Postoje mogući izuzeci ograničeni korisnički specifičnim konfiguracionim podešavanjima.

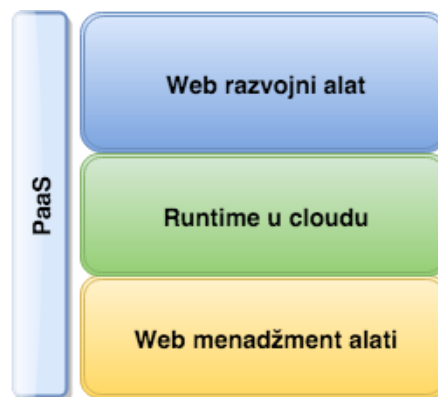


Slika 9 – Primer SaaS aplikacija (slika preuzeta sa Interneta)

4.2. Platforma kao servis (PaaS)

Platforma kao servis (PaaS, eng. *Platform as a Service*), se definiše kao skup softvera i razvojnih alata hostovanih na infrastrukturi dobavljača usluge. Razvojni timovi razvijaju aplikacije na platformi koja se nalazi fizički u posedu dobavljača usluge. Za pristup ovakvoj infrastrukturi koristi se Internet. PaaS provajderi mogu da koriste API (eng. *Application Programming Interface*) ili gateway softver instaliran na klijentovom računaru. GoogleApps je jedan primer PaaS servisa.

Korisniku je data mogućnost da na cloud infrastrukturu dobavljača, korišćenjem programskih jezika, biblioteka, servisa i alata podržanih od strane dobavljača, postavi korisnički kreirane ili drugačije dobavljene aplikacije. Ovo svakako ne znači da se podrška ne može obezbediti iz trećih izvora ukoliko dobavljač ne podržava željene tehnologije. Korisnik ne kontroliše niti upravlja slojem koji leži ispod cloud infrastrukture a sadrži mreže, servere, storidže, operativne sisteme. Međutim korisnik ima kontrolu nad dobavljenim aplikacijama i verovatno konfiguracionim podešavanjima za okruženje za hosting aplikacija (slika 10).

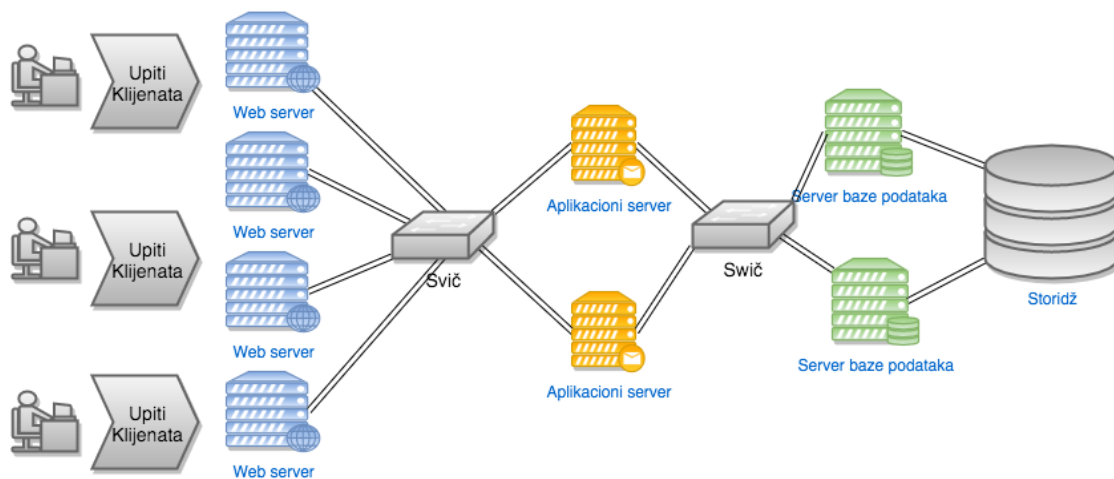


Slika 10 –PaaS su servisi bez instalacije i održavanja od strane korisnika

4.3. Infrastruktura kao servis (IaaS)

Infrastruktura kao servis (IaaS, eng. *Infrastructure as a Service*), pruža mogućnost pokretanja, zaustavljanja, pristupanja i konfigurisanja virtualnih servera i sistema skladištenja podataka. U okviru ovog poslovno računarskog koncepta, kompaniji koja najmi resurse se omogućava da plati isključivo resurse koje je sama zakupila, kao i mogućnost da zakupi onoliko kapaciteta koliko želi.

Mogućnost koja je data korisniku jeste da dobavi obradu podataka, skladištenje podataka, mrežne i ostale osnovne usluge računarskih resursa gde je on u mogućnosti da postavi i pokrene arbitrarni softver, koji može uključiti aplikacije i operativne sisteme. Korisnik ne upravlja niti kontroliše infrastrukturom koja se nalazi ispod ali ima kontrolu nad operativnim sistemom, sistemom skladištenja podatka i isporučenim aplikacijama. Pored toga, korisnik često ima i moguće ograničenu kontrolu određenih mrežnih komponenti (npr. hostovani fajervol, slika 11).



Slika 11 - IaaS, moguća arhitektura infrastrukture na zahtev

4.4. Ostali cloud servisi (XaaS)

Pored osnovnih SaaS, PaaS i IaaS cloud koncepata postoje i hibridne varijante, kao i neki derivati. Ide se dotle da se bilo šta može isporučiti kao servis te je otud nastao i naziv XaaS – bilo šta kao servis (eng. *'anything' as a Service*).

Hibridni cloud se sastoji od najmanje jednog privatnog i jednog javnog clouda. Hibridni cloud se obično nudi kroz jedan od dva načina: dobavljač ima privatni cloud i formirao je strateško partnerstvo sa dobavljačem javnog clouda, ili javni cloud dobavljač formira partnerstvo sa dobavljačem privatnog clouda.

Hibridni cloud je okruženje u kome organizacija upravlja nekim od resursa interno, zadržavajući ih u okviru sopstvene infrastrukture, dok druge resurse dobavlja eksterno, kroz dobavljača javnih cloud servisa. U idealnom slučaju, prednost hibridnog pristupa je da nudi prednosti jeftinijih i isplativijih resursa kroz javni cloud, bez izlaganja kritičnih i osetljivih aplikacije i podataka potencijalnim opasnostima. Ovaj tip hibridnog clouda se još naziva i hibridnim ICT-jem.

Cloud federacija ili federated cloud je uspostavljanje i menadžment više internih i eksternih cloud računarskih servisa kako bi se odgovorilo na određenu poslovnu potrebu. Federacija je skup više manjih delova koji služe zajedničkoj svrsi.

Hardver kao usluga (HaaS, eng. *Hardware as a Service*) je servis snabdevanja koji se definiše različito u kontekstu računarstva sa upravljanim servisima u odnosu na grid računarstvo. U prvom, HaaS je nešto nalik licenciranju. U grid računarstvu, HaaS se radi po sistemu naplate po osnovu korišćenja.

U okviru **računarstva sa upravljanim servisima** radi se zapravo o kupovini koja nalikuje na licenciranje. MSP (eng. *Managed Service Provider*) vrši monitoring i administraciju hardvera smeštenog kod klijenta sa udaljene lokacije gde je biznis model zasnovan na pretplati.

HaaS u okviru grid računarstva znači plaćanje po osnovu korišćenja dobavljačeve infrastrukture i procesorskih resursa. Grid računarstvo je tehnologija u kojoj je više računara spregnuto da radi kao jedan moćan računar.

XaaS (eng. *Anything/everything as a Service*) – XaaS je generički termin za nekoliko stvari uključujući X-kaousluga, gde se misli na „bilo šta kaousluga“/“sve kao usluga“. Akronim se odnosi na rastući broj usluga koje se isporučuju putem Interneta. XaaS je suština cloud računarstva.

Najčešći primeri XaaS koncepta su SaaS, PaaS, IaaS. Kombinovano, ova tri koncepta se nazivaju SPI model (SaaS, PaaS i IaaS). Drugi primeri XaaS koncepta su skladište podataka kao usluga (SaaS, eng. *Storage as a Service*), komunikacija kao usluga eng. (CaaS, eng. *Communication as a Service*), mreža kao usluga (NaaS, eng. *Network as a Service*) i nadzor kao usluga (MaaS, eng. *Monitoring as a Service*).

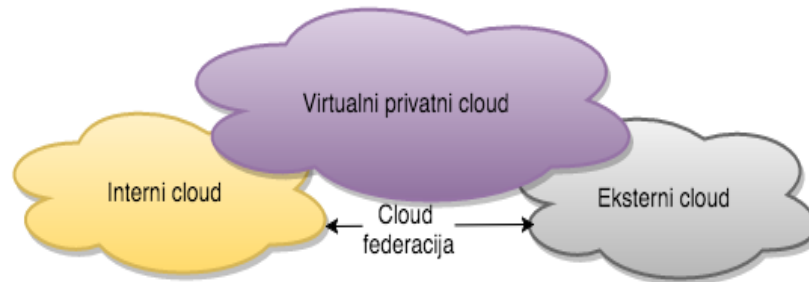
5. Cloud modeli i poslovni koncepti

Privatni cloud – Cloud infrastruktura se obezbeđuje za ekskluzivnu upotrebu jednog ili više poslovnih entiteta. Može biti u vlasništvu, upravljana i vođena od strane organizacije vlasnika, trećeg lica ili kombinacije prethodna dva. Takođe, ovakav cloud sistem se može nalaziti unutar ili van prostorija vlasnika te se može posmatrati kao interni (slika12).

Zajednički cloud (eng. *Community cloud*) – Cloud infrastruktura koja se obezbeđuje radi upotrebe od strane određene skupine korisnika iz organizacija koje dele zajedničke ciljeve (npr. misiju, bezbednosne zahteve, politiku ili usaglašenost sa određenim standardima). Može biti u vlasništvu, upravljana i vođena od strane jedne ili više organizacija u zajednici, trećeg lica, ili kombinacije pomenutog i može se nalaziti u ili van prostorija tih organizacija.

Javni cloud – Javna cloud infrastruktura obezbeđena za slobodnu upotrebu najšire javnosti. Može biti u vlasništvu, upravljana i vođena od strane poslovne, akademske, vladine organizacije ili kombinacije pomenutih organizacija. Nalazi se u okviru prostorija dobavljača.

Hibridni cloud – Cloud infrastruktura je kompozicija dve ili više različite cloud infrastrukture (privatne, zajedničke, javne) koje ostaju nezavisni entiteti, ali ostaju povezane standardizovanom ili vlasničkom (proprietary) tehnologijom koja omogućava portabilnost podataka i aplikacija, kao na slici 12 (npr. cloud bursting radi balansiranja opterećenja) [55].



Slika 12 - Vrste cloud Sistema

5.1. Klase servisnog računarstva

Svakoj aplikaciji treba model obrade podataka, model skladištenja i, pod pretpostavkom čak i najjednostavnijeg oblika distribuiranosti, model komunikacije. Statističko multipleksiranje neophodno da se postigne elastičnost i iluzija neograničenih kapaciteta zahteva virtualizovane resurse tako da je implementacija multipleksiranja i deljenja sakrivena od programera. Postoji mogućnost da različiti oblici ponude uslužnog računarstva mogu biti razlikovani po osnovu nivoa apstrakcije prezentovanog ka programeru i nivou upravljanja resursima.

Amazon EC2 je jedan kraj spektra ponude. EC2 instance izgledaju u priličnoj meri kao i fizički hardver, i korisnici mogu da kontrolišu skoro ceo softverski stek, od kernela na gore. API koji je na raspolaganju je “tanak” nekoliko desetina API poziva za zahtev za konfiguracijom i virtualizacijom hardvera. Ne postoji apriori ograničenje po vrsti aplikacije koja može da bude hostovana. Nizak nivo virtualizacije – sirovi procesorski ciklusi, blokovski sistem skladištenja i IP nivo umrežavanja omogućavaju programerskim timovima da razvijaju šta god žele. S druge strane, to otežava Amazonu da ponudi automatsku skalabilnost i toleranciju na otkaze jer je semantika povezana sa replikacijom i upravljanjem drugim problemima u vezi sa stanjem sistema visoko zavisna od aplikacionog sloja.

AWS nudi određen broj servisa kojima se može upravljati na višem nivou, uključujući nekoliko različitih storidž servisa koji se mogu koristiti uz EC2 uslugu, nešto poput SimpleDB. S druge strane, ove ponude imaju veće kašnjenje kao servis i nestandardizovane API interfejsje, i nisu u toj meri u širokoj upotrebi poput drugih delova AWS.

Na drugoj strani spektra nalazi se platforme poput Google AppEngine i Force.com, poslovna razvojna platforma Salesforce-a. AppEngine je ciljan isključivo na tradicionalne web aplikacije, namećući strukturu čiste separacije između dela za obradu podataka i skladišta podataka. Štaviše, od AppEngine aplikacije se očekuje da bude koncipirana tako da servisira zahteve, i kao takva da bude vrlo racionalna u korišćenju procesorskog vremena na nivou zadatka. AppEnginovo impresivno automatsko skaliranje i mehanizmi visoke dostupnosti kao i MegStore (zasnova na BigTable) storiž podataka koji služi AppEngine aplikacijama, oslanjaju se na ova ograničenja. Zbog toga, AppEngine nije pogodan za opštu namenu. Slično, Force.com je dizajniran da podrži poslovne aplikacije koje se izvršavaju na salesforce.com bazi i ništa drugo.

Microsoft Azure se nalazi između prethodno pomenutih tehnologija, negde na pola spektra fleksibilnosti naspram programerske udobnosti. Azure aplikacije su pisane korišćenjem .NET biblioteka, i kompajlirane u Common Language Runtime okruženju. Sistem podržava opšti tip namene, pre nego specifične. Korisnici sistema dobijaju izbor jezika, međutim ne mogu da upravljaju strukturom ispod, poput operativnog sistema i izvršnog okruženja. Biblioteke omogućavaju stepen automatske mrežne konfiguracije i tolerancije na otkaz, ali zahtevaju od programera da deklarativno specificira neke osobine aplikacije da bi se ova funkcionalnost implementirala. Imajući u vidu izložene činjenice, Azure je međurešenje između kompletnog aplikacionog frameworka poput AppEnginea, sa jedne strane, i hardverskih virtualnih mašina, poput EC2, sa druge strane.

Tabela 4 sumarije kako ove tri klase virtualizuju obradu podataka, skladište podataka i umrežavanje. Raštrkanost ponude skalabilnog sistema skladištenja podataka sugerije da pitanje skalabilnosti ovakvih sistema sa API interfejsom i sa uslugama SQL-a ostaje i dalje otvoreno. Amazon je počeo da nudi Oracle baze hostovane na AWS, ali se ekonomija i licencni modeli ovih proizvoda čine manje prirodnim za okruženje cloud računarstva.

Da li će jedan model cloud računarstva preovladati i potisnuti ostale? Možemo povući analogiju između programskih jezika i frameworka. Programski jezici niskog nivoa poput C-a ili assemblera dozvoljavaju finu kontrolu i blisku komunikaciju sa hardverom, međutim, ukoliko programer piše web aplikaciju, upravljanje komunikacionim portovima, socketima, slanjem zahteva, itd. upravljanje ovakvim procesima je veoma zamorno, čak i sa veoma dobrim gotovim bibliotekama. S druge strane, framework visokog nivoa, poput Ruby on Rails-a, čini ove stvari nevidljivim za programera, međutim, korisne su isključivo ukoliko aplikacija odgovara po zahtev/odgovor strukturi i apstrakciji koju Rails omogućava. Svako odstupanje zahteva zaranja u framework, u najboljem slučaju, može biti vrlo težak zadatak za implementaciju. Nijedan razuman Ruby programer se ne bi usprotivio superiornosti jezika C za određene zadatke, kao i obrnuto. S tim u vezi, postoji osnova za verovanje da će određeni zadaci zahtevati različite klase servisnog računarstva.

Nastavljajući analogiju programskog jezika, baš kao što jezici višeg nivoa mogu biti implementirani u niže, tako i visoko automatizovani cloud može biti hostovan povrh cloud sistema koji je u manjoj meri automatizovan. Na primer, AppEngine može biti hostovan povrh Azure ili EC2 clouda; Azure može biti hostovan povrh EC2. Naravno, AppEngine i Azure svaki nudi određene opcije (AppEngine skaliranje, toleranciju na otkaz, MegaStore storidž sistem) ili velike, kompleksne API-je (Azure .NET biblioteke) koje nemaju besplatnu implementaciju, tako da bi bilo koji pokušaj da se “klonira” AppEngine ili Azure zahtevao implementaciju-reimplementaciju, bilo funkcionalnosti, bilo API poziva što predstavlja (pre)velik izazov.

6. Ekonomija cloud računarstva

Nekoliko napomena u vezi sa ekonomskim modelima cloud računarstva:

- U okviru odlučivanja da li se držanje servisa u cloudu isplati dugoročno gledano, čini se da fina granularnost ekonomskog modela u okviru cloud ponude čini ovo odlučivanje fluidnijim. Tako, na primer, osobina elastičnosti cloud sistema omogućava da se prelazak izvede uz prenos rizika.
- Takođe, iako cena hardverskih resursa ima tendenciju stalnog pada, to se takođe dešava u nepravilnom rasporedu. Na primer, cena obrade i cena skladišta podataka brže padaju od cene WAN konekcija. Cloud računarstvo može da prati ove promene – i potencijalno ih prosledi do klijenta – efikasnije u odnosu na samostalni centar za obradu podataka koji bi taj isti klijent napravio, rezultujući bližoj ceni stvarnih troškova upotrebe tih resursa.
- Prilikom odlučivanja za pomeranje servisa u cloud, potrebno je dodatno utvrditi očekivan prosečni skok korišćenja resursa, posebno ako aplikacija može imati veoma promenljive skokove u zahtevima. Praktični limiti u realnom svetu se nalaze na nivou instalirane opreme. Različiti operacioni troškovi su varijabilni i korelisani sa tipom cloud okruženja.

6.1. Elastičnost: pomeranje rizika

Iako se ekonomska privlačnost cloud računarstva često opisuje kao pretvaranje kapitalnih u operativne troškove (CapEx to OpEx, eng. *Capital expenses to operative expenses*), čini se da je “plati na osnovu korišćenja” dosta precizniji način da se opiše ekonomska dobrobit za zakupca cloud sistema. Ovako zakupljeni sati mogu biti neuniformno korišćeni tokom vremena a da se i dalje plaća po osnovu korišćenja (npr. koristi se 100 server sati danas a sutradan ništa, i da se i dalje plati samo korišćeno serversko vreme). U zajednici koja se bavi računarskim mrežama ovakav način prodaje je već poznat pod nazivom naplata po osnovu korišćenja. Dodatno, nepostojanjem kapitalnih izdvajanja za ovu namenu, kapital se može usmeriti na sam poslovni poduhvat.

Tako, iako na primer, Amazonov sistem naplate na kraju može biti skuplji od kupovine i upotrebe uporedivog servera za isti period, cenovna razlika prevaže u korist cloud koncepta koji nudi ekonomske prednosti iz kategorije elastičnosti sistema i transfera rizika, naročito rizika od prevelike ili nedovoljne alokacije resursa.

Prvo se razmatra elastičnost cloud sistema u okviru primera #1. Ključno zapažanje je da osobina cloud računarskog sistema da se resursi dodaju ili uklanjaju uz finu granularnost (po jedan server u slučaju EC2) i vremenom uvođenja merenim u minutima, u poređenju sa nekoliko nedelja, omogućava poklapanje alociranih resursa sa potrebom za poslom koji ti serveri treba da obave. U praktičnoj upotrebi, iskorišćenje servera u datacentrima je u rasponu od 5% do 20% [45]. Iako ovakav stepen iskorišćenost deluje nisko, ovo proističe iz uočavanja da u odnosu na standardnu upotrebu, prilikom povećanih zahteva potrebe skaču sa faktorom množenja u rasponu od 2 do 10. Pojedini korisnici namerno obezbede manje resursa nego što je potrebno u periodu maksimalne potražnje i dozvoljavaju resursima da budu neupotrebljeni u vreme kada za njima nema potražnje. Što su varijacije izraženije to ima više neupotrebljenih resursa. Jednostavan primer objašnjava kako elastičnost jednog ovakvog sistema omogućava više nego dovoljnu kompenzaciju više cene po server/satu sistema naplate po korišćenju.

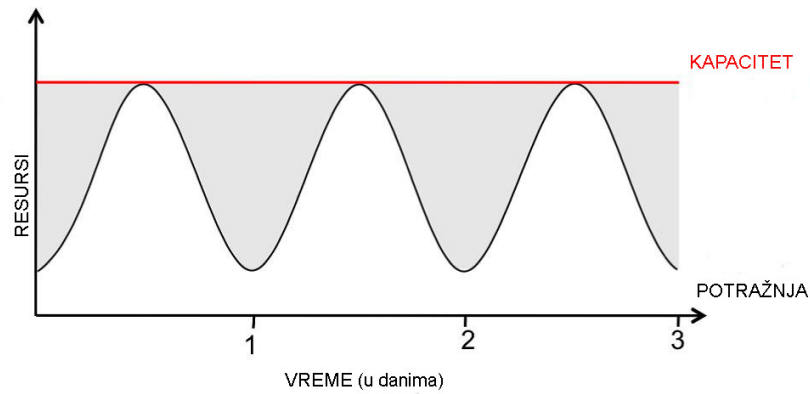
PRIMER #1: Elastičnost. Pretpostavimo da imamo servis koji ima predvidivu dnevnu tražnju gde maksimum zahteva 500 servera a samo 100 tokom noći, kao što je prikazano u Slici 13. Dok god je prosečno iskorišćenje tokom dana 300 servera, stvarno iskorišćenje $300 \times 24 = 7200$ server sati; međutim, pošto moramo obezbediti 500 server sati u periodima maksimalnog iskorišćenja, plaćamo zapravo $500 \times 24 = 12\,000$ server sati, faktor 1.7 više nego što nam je potrebno. Zaključujemo da dok god je cena sistema naplate po korišćenju tokom perioda od 3 godine manja od 1.7 puta cena koštanja servera, možemo ostvariti uštede korišćenjem servisnog računarstva.

Zapravo, prethodno naveden primer potcenjuje dobrobit elastičnost, jer kao dodatak na jednostavne dnevne obrasce upotrebe računarskih resursa, mnogi netrivialni resursi takođe imaju sezonske ili druge periodične varijacije u potrebi za resursima (npr. e-commerce skokovi u decembru i usluge deljenja slika nakon praznika i perioda godišnjih odmora) kao i neka neočekivana iskakanja zbog eksternih događaja (npr. vesti). Obzirom da dobavljanje nove opreme može trajati nedeljama, jedini način da se ovakva potreba reši je da se resursi obezbede unapred.

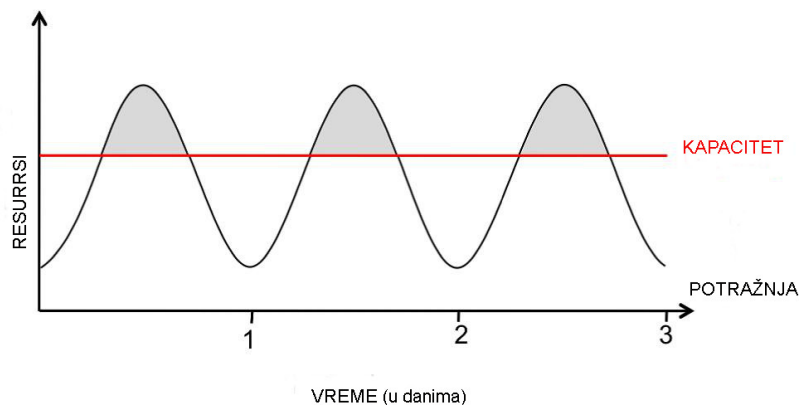
Videli smo da, čak i kada dobavljači usluga dobro predvide ovakve skokove, kapaciteti svakako ostaju neiskorišćeni, i ukoliko se greškom preceni količina potrebnih resursa, tim gore.

Druga mogućnost, je svakako, da se podceni tražnja (Slika 14) što bi moglo dovesti do odbijanja usluge za korisnike koji su preko kapaciteta sistema. Dok su finansijski efekti prebacivanja preko kapaciteta vrlo merljivi, efekte nedovoljne alokacije je vrlo teško meriti. Ipak ono što je vrlo ozbiljno: ne samo da je prinos na odbijenog korisnika nula, već je vrlo moguće da je takav korisnik trajno odbijen.

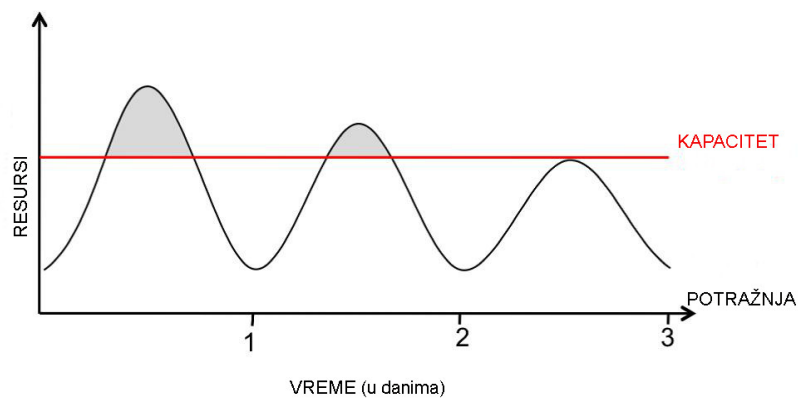
Slika 15 ima za cilj da predstavi sledeće ponašanje: korisnici će napustiti sistem servisa koji nema dovoljno alociranih resursa u datom trenutku do momenta do kog se ne smanji tražnja i resursi postanu ponovo slobodni. U tom momentu će korisnici početi da se vraćaju sistemu, ali u manjem broju.



Slika 13 – Obezbeđivanje resursa u periodu povišene tražnje



Slika 14 – Nedovoljno obezbeđenih resursa 1



Slika 15 – Nedovoljno obezbeđenih resursa 2

PRIMER #2: Transfer rizika. Pretpostavimo da je 10% korisnika koji imaju iskustvo malopre opisane loše usluge trajno izgubljeno, a da bi ti isti korisnici da su imali uslugu sve vreme ostali korisnici Sistema. Sistem je inicijalno napravljen da pruža usluge za 400,000 korisnika (1000 korisnika po serveru X 400 servera), međutim zbog pozitivnog novinskog članka tražnja poraste na 500,000 u prvom satu. Od 100,000 korisnika koji su odbijeni u smislu usluge (neusluženi), pretpostavka je da će njih 10,000 biti trajno izgubljeni kao klijenti, ostavljajući aktivnu korisničku bazu od 390,000. U narednom satu će sajt posetiti 250,000 novih korisnika ali sajt je i dalje za 240,000 korisnika preko svojih projektovanih kapaciteta. Ova situacija rezultira u dodatnih 24,000 odbijanja, ostavljajući bazu od 376,000 stalnih korisnika. Ukoliko se ovaj trend nastavi, posle 500,000 ili 19 sati, broj novih korisnika će se približiti nuli i sajt će biti u okviru kapaciteta. Jasno je da će ovako projektovan sistem sakupiti manje od 400,000 korisnika vrednih stabilnog prihoda tokom ovih 19 sati, međutim, da još jednom ilustrujemo argument nedovoljne alokacije resursa, da ne govorimo o lošoj reputaciji od strane nezadovoljnih korisnika.

Postavlja se pitanje da li se ovakvi scenariji zaista dešavaju u praksi? Kada je Animoto [56] učinio svoj servis dostupan putem Fejsbuka, desio se porast tražnje od 50 servera na 3500 servera za samo tri dana. Čak i da je prosečno iskorišćenje pojedinačnog servera bilo relativno malo, niko nije mogao predvideti dupliranje resursa na svakih 12 sati tokom tri dana.

Nakon što je ovakva potražnja utihnula, saobraćaj se sveo na nivo koji je znatno niži od maksimuma. Tako, u ovom primeru iz prakse, skaliranje sistema na gore i potreba za elastičnošću nije bila samo optimizacija troškova već i operativna potreba, i skaliranje na dole je omogućilo da se iskorišćenje resursa na čekanju tačnije podudara sa realnim potrebama. Elastičnost je od vrednosti kako postojećim firmama, tako i novopokrenutim biznisima. Na primer, firma Target, američki drugi po veličini gigant maloprodaje, koristi AWS za Target.com web sajt. Dok su drugi dobavljači imali znatne probleme sa performansama, povremenu nedostupnost na “crni petak” (28. novembar), Target.com i Amazon.com sajtovi su bili sporiji za samo 50%. Koristeći isti princip elastičnosti, Salesforce.com hostuje klijente od dva do 40,000 korisnika.

Čak i manje dramatični slučajevi sasvim su dovoljni radi ilustracije ključnih dobrobiti od cloud računarstva: rizik od pogrešne procene zahteva ka sistemu se pomera od dobavljača usluge ka vlasniku cloud sistema. Cloud vlasnik može da naplati dodatatak za preuzimanje ovog rizika koji bi zapravo reflektovao višu cenu upotrebe server sata u poređenju sa trogodišnjom cenom kupovine istog sistema. Postoji prosta jednačina koja uopštava sve prethodno pomenute slučajeve.

Pretpostavlja se da cloud dobavljač usluga uposli sistem koji se plaćaju po osnovu korišćenja, u okviru kojih korisnik plaća proporcionalno vreme i količinu resursa koje koristi. Dok Abramson, Hosanagar i Stuer [57], [58], [59] smatraju da je potreban sofisticiraniji model plaćanja za usluge infrastrukture, izgleda da će princip postavljanja cene po osnovu korišćenja biti preovlađujući jer je jednostavniji i transparentniji, kao što je slučaj sa širokom primenom “pravih” uslužnih servisa poput struje, gasa, itd. Slično tome, pretpostavlja se da je klijentov profit direktno proporcionalan broju upotrebljenih sati. Ova pretpostavka je konzistentna sa online oglašivačkom politikom generisanja profita u kojoj je broj prikaza oglasa direktno proporcionalan ukupnom vremenu poseta provedenom od strane krajnjih korisnika na servisu [49].

$$\text{KorisnikSat}_{\text{cloud}} \times (\text{prihod} - \text{Cena}_{\text{cloud}}) \geq \text{KorisnikSat}_{\text{datacenter}} \times (\text{prihod} - \text{Cena}_{\text{datacenter}} / \text{Iskorišćenje})$$

Leva strana množi neto prihod po korisnik satu (prihod realizovan po korisnik satu umanjen za cenu plaćanja cloud računara po korisnik satu) po korisnik satu, dajući očekivani profit od korišćenja cloud sistema. Desna strana izvršava istu kalkulaciju samo za datacenter fiksno kapaciteta množenjem prosečnog korišćenja, uključujući i periode bez maksimalne tražnje. Koja god strana je veća, predstavlja potencijalni profit. Očigledno, ako je iskorišćenje = 1:0 (oprema datacentra ima 100% iskorišćenja), u kom slučaju obe strane jednačine izgledaju isto. Ipak, teorija redova čekanja nam govori da kako se približavamo stanju 1:0, vreme odziva sistema se približava beskonačnosti. U praksi, korisni kapacitet datacentra (bez kompromitovanja usluge) je obično između 0.6 i 0.8. Dok datacenter mora da dobavi više resursa radi obezbeđivanja usluge u momentima maksimalne tražnje, cloud dobavljač jednostavno može da uključi ovu vrstu rizika u cenu najma resursa. Ovo praktično znači da ukoliko iznajmite Internet link od 100 Mbita/sekundi, moći ćete da koristite svega 60 do 80 Mbita/sekundi u praksi.

Jednačina jasno pokazuje da je zajednički element u svim primerima sposobnost kontrolisanja cene po korisnik satu operativnog servisa. U primeru 1, cena po korisnik satu bez elastičnosti je visoka zbog resursa koji stoje neiskorišćeni – viša cena dok je isti broj korisnik sati. Ista stvar se dešava ukoliko se preceni zahtev, te alociraju resursi koji kasnije ostaju neiskorišćeni. U drugom primeru, cena po korisnik satu se povećava kao rezultat podcene maksimalnog skoka zahteva koji rezultira odbijanjem korisnika: Obzirom da se neki procenat korisnika trajno gubi, fiksni troškovi ostaju isti ali se sada amortizuju manjim brojem korisnik sati. Ovo ilustruje fundamentalna ograničenja sistema “kupovnog” modela u kontekstu bilo kakvog netrivialnog povećanja zahteva.

Na kraju, postoje dve dodatne koristi iz cloud sistema za korisnika iz mogućnosti da promeni svoju upotrebu resursa u satima, naspram godina. Prvo, neočekivano skaliranje na dole (oslobađanje trenutno neupotrebljenih resursa) – na primer, tokom poslovnog usporavanja, ili ironično tokom poboljšane softverske efikasnosti – standardno ovakve situacije nose finansijsku kaznu sa sobom. Uz trogodišnju amortizaciju, server od 2,100 USD, ukoliko se iz navedenih razloga izopšti iz sistema posle samo godinu predstavlja “kaznu” od 1,400 USD. Cloud računarstvo eliminiše ovu kaznu.

Drugo, tehnološki trendovi sugerišu da se tokom korisnog dela životnog ciklusa neke opreme, cena hardvera pada i novi hardver i softver postaju dostupni. Cloud provajderi, koji već uživaju u dobrobiti zahvaljujući velikom obimu nabavke, potencijalno mogu preneti neke od ovih ušteda na svoje klijente. Zaista, ozbiljni korisnici AWS sistema su imali prilike da osete pad cena za 20% u domenu sistema skladištenja podataka dok je u oblasti mreža pad bio 50% tokom proteklih 2.5 godine, uz dodatak devet novih servisa tokom perioda od manje od jedne godine. Ukoliko nove tehnologije ili cenovni planovi postanu raspoloživi dobavljaču cloud usluga, postojeće aplikacije i klijenti mogu potencijalno imati koristi od ove vrste inovacije momentalno, bez potrebe za kapitalnim ulaganjima u cilju uštede. Za manje od dve godine, Amazon Web servisi su povećali broj različitih tipova servera (“instanci”) sa jedne na pet. Za manje od jedne godine je dodato sedam novih infrastrukturnih servisa i dva nova načina za podršku.

6.2. Cost-benefit analiza prelaska na cloud

Dok je prethodna tema bila okrenuta kvantifikaciji ekonomske vrednosti specifičnih vrednosti cloud računarstva kao što je elastičnost, u ovom odeljku ćemo obraditi pitanje da li je ekonomičnije preći sa datacentra na cloud hostovano rešenje, ili se pak zadržati na postojećem rešenju?

Tabela 5 sadrži ažuriranu novu Grejevu tabelu iz 2003. godine, na podatke iz 2016., omogućavajući nam da pratimo stopu promene cloud tehnologija tokom ovih pet godina. Primetimo da su cene širokopojasnog pristupa Internetu ponajmanje opale, ispod faktora 3. Troškovi računarske obrade opali su najviše, dok se sposobnost iskorišćenja dodatne računarske snage zasniva na pretpostavci da program može da iskoristi sva procesorska jezgra u okviru računarskog sistema. Ova pretpostavka je pre tačna za servisno računarstvo, sa putno virtualnih računara koji opslužuju hiljade ili milione korisnika, nego za programe koji se izvršavaju unutar datacentra jedne kompanije.

Da bi olakšao izračunavanje, Grej je izračunao šta je 1 USD mogao da kupi u 2003. Godini. Tabela 5 pokazuje njegove brojeke naspram 2008. godine. i upoređuje sa EC2/S3 troškovima. Na prvi pogled, izgleda da će dolar više dati ukoliko se upotrebi za kupovinu hardvera u 2008. godini nego da se plati za korišćenje istog hardvera. S druge strane, ova jednostavna analiza prenebreže nekoliko bitnih faktora [50].

Posebno plaćanje po osnovu korišćenja resursa. Aplikacije nejednako koriste resurse za obradu, skladištenje i mrežni saobraćaj. Neke više koriste procesorsku snagu, neke mrežne resurse, i tako dalje, I samim tim mogu zasititi te resurse dok druge ne koriste u dovoljnoj meri. Model naplate na osnovu korišćenja svojstven cloud servisima omogućava posebno plaćanje po osnovi upotrebe datog resursa. Na ovaj način se smanjuje količina neiskorišćenosti resursa koji su na raspolaganju. Dok tačne uštede zavise od aplikacije, pretpostavimo da se za određen slučaj procesorska snaga koristi u obimu od oko 50%, dok su mrežni resursi u potpunosti iskorišćeni; u tom slučaju imamo za scenario datacentra plaćanje dvostruko više u odnosu na realno iskorišćenje procesorskih resursa. Tako, umesto da kažemo da 2.65 USD košta da se iznajmi procesor vredan 1 USD, bilo bi tačnije reći da košta 2.65 USD da se iznajmi procesor vredan 2 USD. Kao dodatna napomena, AWS cene za mrežni saobraćaj su zapravo konkurentnije od cene koju bi kompanija srednje veličine platila za isti saobraćaj.

Tabela 5 - Na Grejeve podatke iz 2003. dodati su normalizovani podaci iz 2017

	WAN propusni opseg	CPU sati (sva jezgra)	Disk storidž
Resurs u 2003	1 Mbps WAN link	2 GHz CPU, 2 GB DRAM	200 GB disk, 50 Mb/s transfer rate
Cena u 2003	100 USD/mesečno	2000 USD	200 USD
1 USD kupuje u 2003	1 GB	8 CPU sati	1GB
Resurs u 2016	100 Mbps WAN link	2 GHz, 2 socket-a, 4 jezgra/socket-u, 4 GB DRAM	1 TB disk, 115 MB/s kontinualnog transfera
Cena u 2016	3600 USD/mesečno	1000 USD	100 USD
1 USD kupuje u 2016	3 GB	145 CPU sati	
Cena/perform. Pobjedj.	2.7x	16x	10x
Cena najma 1 USD na AWS	0.27–0.40 USD	\$2.56	1.20–1.50 USD

Struja, hlađenje i fizički troškovi fabrike. Cena struje, hlađenja i amortizacije zgrade su za sada isključeni iz ovog pojednostavljenog modela. Hamilton procenjuje da se cena procesora, storidža i mrežnog propusnog opsega okvirno duplira ukoliko se uračuna amortizacija zgrade [60]. Koristeći procenu, kupovinom 128 sati procesorskog vremena u 2008. stvarno košta 2 USD pre nego 1 USD, u poređenju sa 2.56 na EC2. Slično, 10 GB prostora na disku košta 2 USD pre nego 1 USD u poređenju sa 1.20 – 1.50 USD na mesečnom nivou na S3 Amazonovom servisu. Na kraju, S3 zapravo replicira podatke bar 3 puta radi trajnosti i performansi, obezbeđuje trajnost, i repliciraće podatke dalje ukoliko postoji visoka tražnja za podacima. To zapravo znači da je cena 6.0 USD kada se kupuje naspram 1.20 do 1.50 USD mesečnog najma na S3.

Operativni troškovi (održavanje). Danas su troškovi upravljanja računarskim resursima relativno niski. Ponovno pokretanje servera je jednostavno te i minimalno obučeno osoblje može da vrši zamenu komponenti u serverskim rek ormarima. S jedne strane, pošto servisno računarstvo koristi virtualne mašine umesto fizičkih, sa tačke gledišta cloud korisnika, ovakvi zadaci mogu biti povereni cloud dobavljaču usluga. S druge strane, zavisno od nivoua virtualizacije, mnogi troškovi upravljanja softverom mogu ostati – nadogradnje, primena softverskih zakrpa, itd.

Vraćanje na temu “upravljane naspram neupravljanom” infrastrukturom, veruje se da će cene biti niže u okvirima upravljane infrastrukture (Microsoft Azure, Google AppEngine, Force.com) nego u domenu hardverskog nivoa isporuke infrastrukture poput Amazon EC2 servisa. Ipak, izgleda veoma teško kvantifikovati dobrobiti na takav način da se stvori preovlađujuće mišljenje o ovim stvarima.

PRIMER #3: Migracija u cloud. Pretpostavimo da biološka laboratorija generiše 500 GB novih podataka prilikom svakog eksperimenta. Računaru EC2 instanciranom je potrebno 2 sata po GB da obradi te podatke. Laboratorija ima ekvivalent 20 instanci lokalno, tako da vreme da se evaluiira eksperiment je $500 \times 2 / 20$ ili 50 sati. Mogli bi da obrade sve podatke za sat vremena na 1000 instanci AWS EC2 clouda. Cena ovakvog eksperimenta bila bi 1000×0.10 USD ili 100 USD u obradi podataka i još 500×0.10 USD = 100 USD za prenos podataka preko mreže. Dalje, mereno je vreme prenosa podataka od laboratorije do AWS-a pri brzini od 20 Mbita/sec (Garfinkel, 2007). Vreme prenosa je bilo $500 \text{ GB} \times 1000 \text{ MB/GB} \times 8 \text{ bita/byte} / 20 \text{ Mbita/sec} = 4,000,000 / 20 = 200,000$ sekundi ili više od 55 sati. Kada se to uporedi sa 50 sati lokalne obrade podataka, laboratorija je zaključila da im se ne isplati prelazak na cloud rešenje.

Povezan problem je softverska složenost i cena delimične ili pune migracije podataka sa tradicionalnog sistema na cloud. Pored toga što je migracija podataka zadatak koji se obavlja jednom, može predstavljati velik izazov i trud. Ovaj zadatak predstavlja nove poslovne prilike za kompanije koje obezbeđuju integraciju podataka između javnog i privatnog cloud sistema.

7. Izazovi i šanse za cloud računarstvo

Razmatraćemo probleme i šanse u cloud računarstvu. Svaki problem uparićemo sa šansom – idejom kako bi se problem mogao prevazići. Prve tri su tehničke prepreke prihvatanju cloud računarstva, narednih pet su tehničke prepreke rastu cloud računarstva nakon prihvatanja, i poslednje dve se tiču politike i poslovne prepreke prihvatanju cloud računarstva.

Tabela 6 - 10 glavnih izazova i šansi za prihvatanje i rast cloud računarstva

RBr.	Izazov	Šansa
1	Dostupnost usluge	Korišćenje više cloud provajdera za obezbeđenje poslovnog kontinuiteta; korišćenje elastičnosti radi odbrane od DDOS napada
2	“Zaključavanje” podataka	Standardizovan API; Pravljenje softvera koji bi omogućio pravljenje tzv. “surge computing”
3	Privatnost podataka i mogućnost revizije	Deployment enkripcije, VLAN-ova, fajervola; poštovanje nacionalnih zakona putem geografskih data storidža
4	Uska grla prenosa podataka	Slanje poštom hard diskova; data backup arhiviranje; niži troškovi WAN rutera; LAN svičevi sa većom propusnom moći
5	Nepredvidivost performansi	Poboljšana podrška za virtuelne mašine; fleš memorija; gang scheduling algoritam za paralelnu obradu podataka
6	Skalabilno skladištenje podataka	Pronalazak skalabilnog sistema za skladištenje
7	Greške u velikim distribuiranim sistemima	Pronalazak debugera koji se oslanja na distribuirane VM
8	Brzo skaliranje	Pronalazak brzog skaliranja koje se oslanja na učenje sistema; snapshoti sistema radi očuvanja cloud sistema
9	Deljenje reputacije	Ponuda servisa za zaštitu servisa (npr. email)
10	Licenciranje softvera	Plaćanje po osnovu korišćenja; Količinska prodaja

7.1. Dostupnost usluge

Organizacije brinu da li će servisne računarske usluge imati adekvatnu dostupnost, i ovo stvara oprez vezan za cloud računarstvo. Ironično, postojeći SaaS proizvodi imaju set visokih standarda u ovom smislu. Google Search zaista odgovara telefonskom tonu pre biranja broja u kontekstu Interneta: da su ljudi kojim slučajem otišli da potraže bilo šta na Internetu i da Google pretraga nije dostupna, zaista bi većina pomislila da sam Internet nije dostupan. Korisnici imaju slična očekivanja od novih servisa, što je teško postići. Naredna tabela prikazuje dostupnost Amazon Simple Storage Servisa (S3), AppEngine i Gmail u 2008 godini i pruža objašnjenja za prekid rada. Uprkos negativnom publicitetu zbog ovih prekida u radu, malo enterprajz IT infrastrukturnih servisa je jednako dobro.

Tabela 7 - Prekidi rada AppEngine, Gmail i AWS (Ambrust M., 2009)

Servis, vrsta prekida - razlog	Trajanje	Datum
S3 servis, delimičan prekid rada - autentifikacija [39]	2 h	15-02-2008
S3 servis, prekid rada - gossip protokol [41]	6-8 h	20-07-2008
AppEngine, delimičan prekid rada - programska greška [43]	5 h	17-06-2008
Gmail, prekid rada - kontakt modul [29]	1.5 h	11-08-2008

Baš kao što veliki Internet dobavljači koriste više mrežnih dobavljača kako ih ispad iz sistema ili propadanje jedne od takvih kompanija ne bi dovelo u nefunkcionalno stanje, pretpostavlja se da bi se na sličan način mogla adresirati dostupnost cloud servisa – izborom više od jednog dobavljača. Filozofija visoke dostupnosti u cloud računarstvu je vođena principom da ne postoji jedna tačka oslonca čijim bi padom krahirao celokupan sistem. S druge strane, upravljanje cloud sistemom od strane samo jedne kompanije jeste zapravo takva tačka. Čak i ukoliko kompanija ima višestruke data centre na različitim geografskim lokacijama i koristi više mrežnih dobavljača, može imati jedinstvenu softversku infrastrukturu i sistem za računovodstvo ili, jednostavno, takva firma može da prestane da posluje. Veliki klijenti će oklevati da migriraju na cloud bez strategije poslovnog kontinuiteta (eng. *Business Continuity*) za ovakve situacije. Izgleda da je ovde šansa za nezavisnim softverskim stekovima koje bi pružale različite kompanije, jer bi za jednu kompaniju bilo teško da ekonomski opravda pravljenje i održavanje dva ili više nezavisna softverska steka a sve to u ime softverske nezavisnosti.

Još jedna prepreka dostupnosti su distribuirani napadi odbijanja usluga (DDoS). Kriminalci prete smanjenju profita SaaS provajdera čineći njihov servis nedostupnim, iznuđujući novac u rasponu od 10 do 50 hiljada dolara, kako ne bi stupili u DDoS napad. Ovakvi napadi obično podrazumevaju upotrebu velikih tzv. botnetova, koji se mogu naći i iznajmiti na crnoj berzi po ceni od 0.03 USD po 14 botova (simuliranih nepostojećih korisnika) za nedelju dana. Cloud sistem omogućava dobavljaču da se brani brzim skaliranjem kao prvim odgovorom na ovakve situacije. Pretpostavimo da EC2 instanca može da izdrži 500 botova, i napad koji je u toku generiše dodatnih 1 GB/sekundi lažnog mrežnog saobraćaja i 500,000 botova. Po ceni od 0,03 USD, ovakav napad bi napadača koštao 15,000 USD investiranih unapred. Po cenama iz februara 2009. godine, žrtvu napada bi napad koštao 360 USD po satu u mrežnom saobraćaju i dodatnih 100 USD po satu (1,000 instanci) računarske obrade. Napad bi dakle, morao da traje 32 sata da bi koštao više potencijalnu žrtvu nego ucenjivača. Napad zasnovan na upotrebi botnetova koji bi bi trebalo da traje ovoliko dugo bi bio teško održati, jer što duže napad traje, to ga je lakše otkriti i braniti se protiv njega a botovi koji se koriste u ovom napadu ne bi mogli skorije biti korišćeni u novom napadu na istog dobavljača cloud usluga. Kao i sa elastičnošću, cloud računarstvo pomera cilj napada sa SaaS dobavljača na dobavljača servisnih računarskih usluga, koji mogu spremnije da dočekaju i bolje apsorbuju i gde je vrlo moguće da će već postojati zaštitni mehanizam od DDoS napada kao deo osnovne kompetencije dobavljača.

7.2. Zaključavanje podataka

Softverski stekovi su pomerili interoperabilnost među platformama, ali API-ji za cloud sisteme su još uvek uglavnom u privatnom posedu dobavljača usluga ili barem nisi bili predmet aktivne standardizacije. Iz tog razloga, klijenti nisu u mogućnosti da lako izvuku podatke iz cloud sistema kako bi te iste podatke upotreбили u okviru drugog sistema. Briga oko težine dolaženja do podataka čini jedan veliki broj organizacija koje razmatraju cloud opreznim i dosta njih ne želi da izvrši prelazak na cloud rešenja. Zaključavanje klijenta može biti privlačno sa stanovništva dobavljača cloud usluge, međutim ne i sa stanovništva korisnika iste usluge. To pre svega, iz nekoliko praktičnih razloga kao što su promene cenovne politike, dostupnost i pouzdanost i na kraju, izlazak dobavljača iz posla.

Na primer, online servis za skladištenje podataka koji se zove The Linkup se zatvorio 8. avgusta 2008. godine, nakon što je izgubljen pristup za oko 45% korisničkih podataka [61]. *The Linkup* se oslanjao na online servis za skladištenje podataka - Nirvanix radi skladištenja korisničkih podataka, te je ostalo nejasno iz kog razloga su podaci izgubljeni jer se ove dve organizacije međusobno optužuju. U međuvremenu je korisnicima ovog servisa savetovano da pređu na drugog dobavljača usluga skladištenja.

Očigledno rešenje sastoji se u standardizaciji API-ja tako da SaaS razvojni timovi mogu da obezbede raspoređivanje podataka na više storidž sistema postižući efekat da ispadom jedne kompanije iz poslovanja ne bude doveden u pitanje kompletan integritet sistema. Negativna strana ovakvog koncepta bi mogla da dovede do „trke do dna“ – spuštanje cena do nestanka margine profita za dobavljače. Iznećemo dva argumenta koji bi pojasnili zašto strah od ove stvari ne treba da postoji.

Prvo, kvalitet servisa je bitan jednako kao i cena, tako klijentova odluka za prelazak na drugi sistem ne mora nužno, i najverovatnije neće (uvek) biti vođena isključivo finansijskim parametrima. Tako, imamo danas situaciju da su izvesni dobavljači cloud računarskih usluga skuplji od drugih sa cenovnim faktorom množenja 10 jer su znatno pouzdaniji i nude dodatne servise u cilju poboljšanja načina na koji se sistem koristi.

Drugo, kao dodatak na ublažavanje zabrinutosti oko zaključavanju podataka, standardizacija API-ja omogućava nov upotrebn model u okviru koga identična softverska infrastruktura može biti upotrebljena u privatnom ali i u javnom cloud sistemu. Ovakva jedna opcija čini moguće ono što nazivamo „eng. *surge computing*“ kada zapravo imamo nadošli talas tražnje koji će se prelići u javni cloud za vreme trajanja povišene tražnje.

7.3. Poverljivost podataka i mogućnost eksterne revizije podataka

“Moji poverljivi kompanijski podaci nikada neće biti u cloud sistemu.” Ovo je često izjavljivana rečenica. Trenutne ponude cloud sistema su u osnovi javne, pre nego privatne mreže, izlažući sistem većem broju mogućih napada. Postoji takođe zahtev za revizijom, u smislu Sarbanes-Oxley i HIPPA američkog zakona (eng. *Health and Human Services Health Insurance Portability and Accountability*) koji moraju biti ispoštovani pre nego što kompanijski podaci budu pomereni u neki od cloud sistema.

Izgleda da nema fundamentalnih prepreka pravljenju cloud računarskog okruženja bezbednim poput većine privatnih računarskih sistema u trenutnoj upotrebi i da mnoge prepreke mogu biti prevaziđene dobrim razumevanjem tehnologija poput enkripcije podataka na sistemima skladištenja podataka, virtualne lokalne računarske mreže (VLAN-ovi) i mrežni uređaji poput fajervola, filtera IP paketa). Na primer, enkripcija podataka pre stavljanja u cloud sistem može biti bezbedniji način da se upravlja podacima nego da su ti isti podaci pohranjeni u privatni datacenter bez enkripcije. Ovaj pristup je uspešno koristila firma TC3 koja se bavi uslugama iz oblasti zdravstva sa pristupom osetljivim ličnim i zdravstvenim podacima pacijenata prilikom migracije svojih HIPAA-compliant aplikacija u AWS.

Slično, mogućnost revizije mogla bi biti dodata kao još jedan sloj van domašaja virtualizovanog operativnog sistema koji se izvršava na fizičkom sloju (ili virtualizovana aplikacija), dajući viši stepen bezbednosti u odnosu na mehanizme ugrađene u same aplikacije kao i obezbeđujući na taj način centralizujući softverske odgovornosti po pitanju bezbednosti i mogućnosti revizije u jedinstven logički sloj. Ovakva nova mogućnost podupire perspektivu cloud računarstva u kontekstu pomeranja fokusa sa hardvera, na kom se gradi sistem, na mogućnosti koje su dostupne kroz ovakav jedan sistem.

U vezi sa tim postoji briga koje mnoge zemlje imaju i čiji zakoni od SaaS dobavljača traže zadržavanje podataka unutar geografskih granica zemlje. Slično, izvesnim poslovnim sistemima neće možda odgovarati mogućnost da neka zemlja ima pristup njihovim podacima kroz sudstvo. Na primer, evropski klijent može biti zabrinut u vezi sa korišćenjem SaaS usluga, imajući u vidu USA PATRIOT akt.

Cloud računarstvo daje SaaS dobavljaču i SaaS korisniku veću slobodu u vezi sa postavljanjem svog sistema skladištenja podataka. Na primer, Amazon ima na raspolaganju u zakup usluge koje su locirane fizički u SAD i u Evropi, omogućujući dobavljačima da drže podatke gde god oni to žele. Sa AWS regionima, jednostavna konfiguraciona promena eliminiše potrebu za pregovore sa dobavljačem usluga preko okeana.

7.4. Uska grla u prenosu podataka

Aplikacije nastavljaju da bivaju sve više intenzivne po pitanju obrade podataka. Ukoliko pretpostavimo da one mogu biti “rastavljene” radi prenosa kroz cloud sistem i među cloud sistemima, ovo može komplikovati situaciju na mnogo načina. Pri ceni od 100 USD do 150 USD po terabajtu prenetih podataka, ovi troškovi mogu brzo rasti, čineći prenos podataka značajnim problemom. Cloud korisnici i cloud dobavljači usluga treba da razmisle o mogućim implikacijama dodavanja mogućnosti postavke i saobraćaja na svakom nivou sistema ukoliko žele da minimizuju troškove. Ovaj način rezonovanja možemo sresti u novom Amazonovom servisu koji je u razvoju i zove se Cloudfront servis.

Jedna mogućnost za prevazilaženjem visokih troškova Internet prenosa je fizičko slanje diskova. Džim Grej nalazi da je najjeftiniji način slanja velike količine podataka upravo slanje noćnom poštom hard diskova [51]. Iako nema garancija od strane proizvođača diskova ili računara da je ovaj način isporuke pouzdan, iskustva govore da od 400 slanja zabeležen samo jedna slučaj otkaza diska. Čak i ovakve slučajeve je moguće pokriti slanjem dodatnih diskova u redundantnom nizu sličnom RAID tehnologijama.

Radi kvantifikacije, pretpostavimo da želimo da pošaljemo 10 TB sa lokacije A u Amazonov cloud centar. Pretpostavimo dalje da nam je na raspolaganju kapacitet propusnog opsega od 20 Mbit/sekundi. Bilo bi potrebno [49]:

$$10 \times 10^{12} \text{ Bajtova} / (20 \times 10^6 \text{ bits/sekundi}) = (8 \times 10^{13}) / (2 \times 10^7) \text{ sekundi} = 4,000,000 \text{ sekundi}$$

Ovo je više od 45 dana. Amazon bi takođe naplatio 1,000 USD na ime mrežnog saobraćaja za prijem podataka.

Ukoliko bismo poslali 10 x 1 TB diskova noćnom poštom, bilo bi potrebno manje od jednog dana da se prenese 10 TB podataka i cena bi u USA bila oko 400 USD, a saobraćaj bi bio 1500 Mbit/sekundi. Na ovaj, lateralni način, se kašnjenje (eng. *Latency*) skraćuje za faktor množenja, 45.

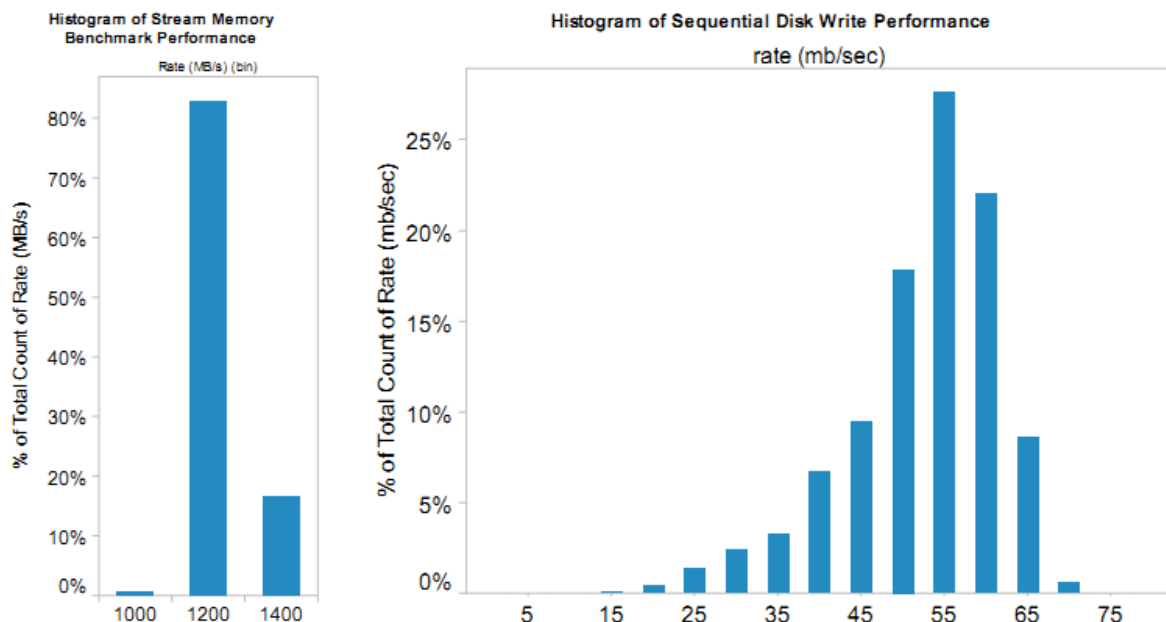
Ako se vratimo prethodno izloženom primeru biološke laboratorije, bilo bi potrebno oko 1 sat da se podaci upišu na diskove, 16 sati da se prenesu poštom, oko 1 sat da se iščita 500 GB podataka, i zatim 1 sat da se obrade podaci. Na ovaj način, vreme potrebno da se obrade podaci vezani za eksperiment bili bi 20 sati naspram 50, a cena bi bila oko 200 USD po eksperimentu, te bi odluka bila da se pređe na cloud. Kako kapacitet diskova sa sve većom cenovnom dostupnošću raste, znatno brže od mrežnog saobraćaja – nekih 10X naspram 3X u poslednjih 5 godina sudeći po tabeli 5 – prenos podataka će postajati sve atraktivniji svake godine.

Druga prilika je naći druge razloge da se cloud napravi privlačnijim za čuvanje podataka, jer jednom kada su podaci u cloudu, usko grlo više nije problem i na ovaj način se mogu stvoriti novi servisi. Amazon je, primera radi, počeo besplatno da hostuje velike setove podataka (npr. SAD popisne podatke) na svom S3 servisu; pošto nema naplate za migracije podataka između S3 i EC2 sistema. Na ovaj način može doći do privlačenja interesovanja za EC2 servisima. Kao još jedan primer, razmotrimo off site arhiviranje i backup usluge. Pošto kompanije poput Amazona, Googla i Microsofta vrlo verovatno šalju znatno više podataka nego što primaju, cena ulaznog propusnog opsega bi mogla biti znatno niža. S tim u vezi, na primer, ukoliko bi nedeljna puna rezervna kopija (eng. *Backup*) podataka funkcionisala fizičkim slanjem diskova a kompresovani dnevni inkrementalni bekapi išli preko mreže, cloud računarstvo bi moglo da ponudi dostupne bekap sisteme van prostorijske kompanije. Jednom kada se arhivirani podaci nađu u cloud sistemu, novi servisi mogu rezultovati većom prodajom cloud računarskih servisa, kao što je pravljenje indeksa za pretraživanje arhiviranih podataka ili prepoznavanje slika na svim arhiviranim slikama prema grupi osoba koje se pojavljuju na svakoj od slika.

Treća, radikalnija prilika je pokušati da se redukuje cena WAN propusnog opsega bržim tempom. Jedna pretpostavka je da se 2/3 cene WAN saobraćaja odnosi na cenu rutera visokih performansi, gde je svega trećina cena optičkog vlakna [62]. Radi se na istraživanju jednostavnih rutera od osnovnih komponenti sa centralizovanom i jeftinom kontrolom kao alternative distribuiranim ruterima visokih performansi [16]. Ukoliko bi se ovakva tehnologija postavila na strani WAN dobavljača usluga, mogli bismo prisustvovati do sada nezabeleženom padu troškova WAN resursa.

Uz dodatak činjenici da je WAN propusni opseg usko grlo, intra-cloud mrežne tehnologije mogu bit takođe usko grlo. Danas, u okviru datacentra, obično 20-80 obradnih čvorova unutar računarskog ormara (eng. *Rack*) su povezani svičem (eng. *switch*) koji se nalazi na vrhu ormara na svič drugog ili agregacionog nivoa. Svič oprema drugog nivoa, ili agregacioni svičevi, su povezani preko rutera na sisteme za skladištenje podataka i WAN mreže, kao što je Internet ili inter-datacentar WAN mreža. Jeftin 1 Gbit ethernet uglavnom se nalazi kao osnova na nižim nivoima agregacije. Ovaj propusni opseg može predstavljati usko grlo u komunikaciji među čvorovima (eng. *nodes*). Još jedan set aplikacija sa paketnom obradom podataka koja ima potrebu za većim propusnim opsegom su aplikacije visokih performansi; nedostatak propusnog opsega je jedan razlog iz kog naučnici koriste cloud računarstvo.

Ethernet od 10 gigabita se obično koristi za agregacioni nivo računarskih mreža u okviru cloud sistema, ali je u ovom trenutku previše skup da bi se spustio na nivo individualnih servera (oko 1,000 USD za 10 GbE serversku konekciju, naspram 100 USD za 1 GbE konekciju - podatak iz 2008). Godine 2011, kada 10 Gbic serverska kartica može da se nađe već od 35 USD stvaraju se uslovi da se nekadašnji agregacioni nivo preseli na nivo hosta u mreži jer bi ovo umnogome smanjilo kašnjenje (eng. *latency*) na mrežnim konekcijama. Ovo bi zauzvrat omogućilo više jezgara i virtualnih mašina po fizičkom server čvoru i na taj način skaliranje mreže. Ovo dalje omogućava širu upotrebu 40 GbE i 100 GbE na višim nivoima agregacije.



Slika 16 – STREAM test na 75 virtualnih mašina (a) memorije i (b) upisivanje na disk 1GB po mašini

7.5. *Nepredvidivost performansi*

Sudeći po dosadašnjem iskustvu, virtualne mašine mogu vrlo uspešno da dele radnu memoriju i procesorsku snagu, međutim ulazno/izlazne (I/O) operacije su znatno problematičnije za deljenje. Slika 15(a) pokazuje prosečan memorijski propusni opseg za 75 EC2 instance na kojima se izvršava STREAM aplikacija za merenje performansi memorije [63]. Prosečan propusni opseg je 1355 MB po sekundi, sa standardnom devijacijom od samo 53 MB/sekundi, manje od 4% srednje vrednosti (mean). Slika 15(b) prikazuje prosečni propusni opseg za 75 EC2 instance pri simultanom upisu na 1 GB lokalne diskove. Prosečna vrednost upisivanja je blizu 55 MB po sekundi uz standardnu devijaciju sa malo preko 9 MB/sekundi, više od 16% srednje vrednosti. Ovo ukazuje na problem ulazno izlaznog interfejsa među virtualnim mašinama.

Jedan od načina na koji bi mogla da se poboljša arhitektura i operativni sistemi kako bi efikasno virtualizovali prekide (eng. *interrupts*) i ulazno/izlazne kanale. Tehnologije poput PCIeExpress je dosta teško virtualizovati, međutim one su kritične za cloud infrastrukturu. IBM mainframe računari i operativni sistemi su uglavnom prevazišli ove vrste uskih grla još osamdesetih godina dvadesetog veka, te imamo uspešne slučajeve iz kojih možemo da učimo.

Još jedna mogućnost je da će fleš memorija smanjiti broj ulazno/izlaznih interfejsa. Fleš memorija je poluprovodnička memorija koja čuva podatke i kada ostane bez napajanja, dakle na isti način na koji to rade i hard diskovi, međutim, pošto nema pokretnih delova, znatno je brža u smislu pristupa podacima (mikrosekunde naspram milisekundi), i na kraju, troši manje energije. Fleš memorija može da održi mnogo veći protok ulazno/izlazne operacija u odnosu na hard disk, gde više virtualnih mašina sa konfliktnim nasumičnim ulazno/izlaznim opterećenjima mogu da koegzistiraju bolje na istom fizičkom računaru bez interfejsa koji vidimo kod hard diskova. Nedostatak interferencije koji vidimo sa poluprovodničkom glavnom memorijom u Slici 3(a) može da se proširi na poluprovodničko skladište takođe, te na taj način povećavajući broj aplikacija koje se mogu izvršavati na virtualnim mašinama i tako deliti isti računar. Ovaj napredak bi mogao spustiti cenu eksploatacije sistema cloud dobavljačima usluga, koja bi na kraju bila povoljnije i za korisnike ovakvih sistema.

Još jedna nepredvidiva prepreka tiče se planiranja izvršavanja (eng. *scheduling*) virtualnih mašina za neke od klasa programa paketne obrade, naročito za računarstvo visokih performansi. Imajući u vidu da je ovakav vid računarstva u upotrebi da bi se opravdale vladine kupovine superračunarskih centara vrednih 100 miliona USD u kojima radi od 10,000 do 1,000,000 procesora, svakako postoje mnogi zadaci sa potrebom za paralelizmom koji bi mogli profitirati od elastičnog računarstva. Cenovna asocijativnost znači da ne postoji cenovna kažnjivost za korišćenje 20 puta više računarskih resursa za 1/20 deo vremena. Potencijalne aplikacije koje bi mogle imati koristi uključuju kategoriju koja u sebi sadrži visok stepen finansijskog povraćaja na uloženo – finansijska analiza, istraživanje nafte, filmske animacije – te bi lako mogle opravdati skromnu premiju za 20x ubrzanje. Jedna od procena je da trećinu današnjeg serverskog tržišta čini računarstvo visokih performansi (HPC, eng. *High Performance Computing*).

Prepreka u privlačenju HPC nije korišćenje klastera; većina paralelne obrade se danas obavlja u klasterima korišćenjem interfejsa za razmenu poruka (MPI, eng. *Message Passing Interface*). Problem je što mnoge HPC aplikacije moraju da obezbede da sve niti izvršavaju simultano, međutim današnje virtualne mašine i operativni sistemi ne omogućavaju način vidljiv sa programerske strane da se ovo obezbedi. Iz tog razloga, prilika da se ovakva prepreka prevaziđe jeste da se ponudi nešto nalik na gang scheduling algoritam za cloud računarstvo.

7.6. Skalabilno skladište podataka (Storidž)

Ranije u okviru ovog rada, identifikovali smo svojstva koja cloud računarstvu daju privlačnost: mogućnost kratkoročnog najma resursa (što implicira skaliranje na dole, kako i na gore u zavisnosti od potrebe za resursima), nema troškova unapred, i na raspolaganju su beskonačni kapaciteti na zahtev. Iako je ovo direktno jasno na šta se odnosi u kontekstu obrade podataka, manje je očigledno kako može da se primeni na stalno skladište.

Kao što se vidi iz tabele 4, bilo je mnogo pokušaja da se da odgovor na ovo pitanje, varirajući od načina na koji se vrše upiti i storidž API-ja, nuđenja garancija u vezi sa performansama do kompleksnosti strukture podataka direktno podržane od strane sistema skladištenja podataka. Prilika, koja je i dalje otvoren problem za istraživanje, jeste stvoriti storidž sistem koji bi, ne samo izašao u susret ovim potrebama, već bi ih kombinovao sa prednostima cloud sistema kao što su arbitralno skaliranje na zahtev.

On bi omogućavao programeru upravljanje resursima u smislu skalabilnosti, dugotrajnosti podataka i visoke dostupnosti.

7.7. Greške u velikim distribuiranim sistemima

Jedan od većih izazova u cloud računarstvu je uklanjanje grešaka u veoma velikim distribuiranim sistemima. Česta pojava je da se ove greške ne mogu reprodukovati na manjim konfiguracijama, te uklanjanje mora da se izvrši direktno na produkciji, bez mogućnosti da se izvrši testiranje/ispitivanje.

Jedna mogućnost može biti oslanjanje virtualne mašine na cloud računarstvo. Mnogi tradicionalni SaaS dobavljači usluga su razvili svoju infrastrukturu bez korišćenja virtualnih mašina, bilo zbog toga što su prethodili gradnja sistema pojavi virtualnih mašina, bilo da su smatrali da ne mogu da priušte opadanje performansi sa uvođenjem virtualnih mašina. Pošto su virtualne mašine *de rigueur* u servisnom računarstvu, ovaj nivo virtualizacije može učiniti mogućim dohvatanje informacija na načine koji nisu mogući bez virtualizacije.

7.8. Brzo skaliranje

Naplata po osnovu korišćenja svakako se primenjuje i na storidž i na mrežni propusni opseg, gde se meri protok podataka. Obrada podataka je unekoliko drugačija, u zavisnosti od nivoa virtualizacije. Google AppEngine automatski skalira kao odgovor na povećanje i smanjenje opterećenja, a korisnicima se naplaćuje u skladu sa iskorišćenim ciklusima. AWS naplaćuje po satu za broj instanci koje su zauzete, čak i ukoliko mašina ništa ne obrađuje.

Prilika je zatim da se automatski skalira na gore i na dole kao odgovor na opterećenje da bi se ostvarila ušteda u novcu, ali bez da se prekrši ugovor o nivou usluga (eng. *Service Level Agreement*). Svakako, jedan od pristupa bi bio prožimajuća i agresivna upotreba učenja mašine na osnovu statističkih podataka o korišćenju sistema, gde se ovaj pristup koristi pre svega kao statistički i alat za predikciju, dinamičko skaliranje, automatsku reakciju na probleme sa performansama i ispravnošću rada, kao i generalno automatsko upravljanje mnogim aspektima sistema.

Još jedan razlog za skaliranje je da se sačuvaju resursi, baš kao i novac. Pošto računar koji je uključen a ne obavlja neki zadatak koristi oko dve trećine resursa koje bi koristio pri punom angažovanju, pažljiva upotreba resursa bi mogla da redukuje uticaj na okruženje datacentra, koje u ovom trenutku dobija dosta negativne pažnje. Cloud računarski dobavljači već vrše pažljivo i sa niskim dodatnim angažovanjem praćenje upotrebe resursa. Uvođenjem po protoku i po vremenu način naplate, servisno računarstvo ohrabruje programere da obrate pažnju na efikasnost (oslobađajući resurse kada više nisu potrebni), i na taj način omoguće direktnije merenje operativnih i razvojnih neefikasnosti.

Biti svestan troškova predstavlja prvi korak u njihovom održavanju te eventualno i smanjenju, ali napor potreban za konfigurisanje čini izazov da se mašine ostave upaljene iako ne obavljaju nikakav posao preko noći kako bi razvojni tim nastavio odmah ujutru da radi na njima. Brz i jednostavan za korišćenje, alat za snimanje stanja mašina i njihovo ponovo pokretanje (eng. *Snapshot/restart*), ohrabrio bi čuvanje računarskih resursa.

7.9. Deljenje reputacije

Reputacije se ne virtualizuju dobro. Loše ponašanje jednog klijenta može uticati na reputaciju cloud sistema kao celine. Na primer, stavljanje na nepoželjnu listu EC2 IP adrese [64] od strane sistema za zaštitu od neželjene pošte može ograničiti aplikacije koje se aktivno izvršavaju na sistemu. Pravljenjem sistema za zaštitu reputacije, nešto nalik na „email od poverenja“ bila bi jedna od mogućih solucija.

Još jedan problem je pitanje prenosa pravne odgovornosti – cloud dobavljači usluga bi želeli da pravna odgovornost ostane na klijentima i da ne bude prenesena na njih (npr. firme koje šalju neželjenu poštu bi trebalo da budu smatrane odgovornim, ne Amazon).

7.10. *Licenciranje softvera*

Aktuelni modeli licenciranja često ograničavaju računare na kojima softver može da se izvršava. Korisnici plaćaju za softver, a onda plaćaju godišnje održavanje. Svakako, SAP je najavio da će povećati svoje godišnje održavanje licence za barem 22% od kupovne cene softvera, što će u tom slučaju postati uporedivo sa cenama Oracle-a [52]. S tim u vezi, mnogi cloud dobavljači su se oslonili na softver otvorenog izvornog koda (eng. *Open source*) barem delimično jer su licencni modeli komercijalnih softvera neadekvatni za uslužno računarstvo.

Glavna prilika je da, ili softver otvorenog izvornog koda (p)ostane popularan, ili da softverske kompanije iz domena komercijalnog softvera prilagode svoje modele licenciranja kako bi u većoj meri odgovarao cloud računarstvu. Na primer, Microsoft i Amazon nude “naplatu na osnovu korišćenja” Windows Servera i Windows SQL Servera na EC2. Windows koji se izvršava na EC2 platformi košta 0.15 USD po satu naspram 0.10 USD mašine koja otvorenog koda.

S tim u vezi uočava se prepreka gde je potrebno animirati prodajne strukture proizvođača softvera da prodaju proizvode u cloud sisteme. “Naplata na osnovu korišćenja” deluje nekompatibilno sa tromesečnim prodajnim praćenjem kako bi se postavila metrika i izmerila efektivnost, bazirana na jednovremenoj kupovini. Prilika za dobavljače cloud sistema je da jednostavno ponude popuste za grupnu kupovinu licenci za korišćenje. Na primer, Oracle-ovi prodavci bi mogli prodati 100,000 instanca-sati direktno preko Oracle-a, koje bi bile iskorišćene tokom perioda od 2 godine po ceni nižoj od one koju bi dobio pojedinačni kupac van ovog distributivnog kanala.

8. Virtualizacija kao osnova cloud sistema

Virtualizacije podrazumeva kreiranje virtualnih, pre nego 'stvarnih' verzija nečega nalik hardverskoj platformi, operativnom sistemu, storidž uređaju ili mrežnom resursu.

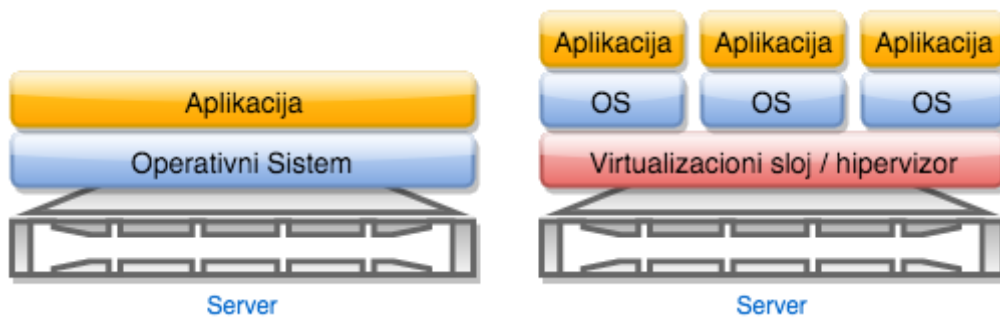
Virtualizacija se može posmatrati kao deo sveukupnih trendova u enterprise IT okruženjima koja uključuju autonomno računarstvo, scenario u kom će ICT okruženje biti u prilici da upravlja samim sobom baziranom na percipiranim aktivnostima, i utility računarstvom, u okviru koga se računarska obradna moć percipira kao alat koji klijent plaća samo kada mu je potreban.

Uobičajen cilj virtualizacije je centralizacija administrativnih zadataka uz poboljšanje skalabilnosti i opšte iskorišćenosti hardverskih resursa.

U konkretnom smislu virtualizacija se pojavljuje kao odgovor na neadekvatnu iskorišćenost hardverskih resursa u oba smera. To se odnosi pre svega na klasične scenarije u kojima se planira hardver za određenu namenu te jednom kapacitira i dobavi, te ostaje nepromenjen do kraja svog životnog veka uz ograničene ili nikakve mogućnosti proširivanja. Mogućnost promene, zapravo elastičnosti ovakvog resursa, se gotovo isključivo, odnosi jednosmerno i to u smeru proširivanja. Ukoliko se ispostavi da je dobavljeni resurs veći ili prosto vremenom to postao, u odnosu na realne potrebe, sav potencijal koji nije u upotrebi ostaje neiskorišćen. Proširivanje hardverskog resursa je moguće samo u okvirima proizvođačkih specifikacija i poseduje čak i faktor vremenske oročenosti. Često se nakon nekog vremena teško, ili uopšte ne može ni izvršiti nadogradnja hardverske konfiguracije.

Pored gore navedenog postoji i problem kompanijskog contingency i business continuity plana. Contingency plan podrazumeva izvršavanje procedure u slučaju nepredviđenih situacija u poslovanju te raznih otkaza resursa i servisa. Business continuity plan podrazumeva dostupnost, zatim visoku dostupnost kao i skalabilnost u kontekstu obezbeđenja kontinuiteta poslovanja i mogućnosti njegovog proširivanja.

Kako bi se uspešno adresirale sve navedene potrebe potrebno je mnogo toga uraditi, ali prvi korak podrazumeva virtualizaciju infrastrukture. Nadalje ćemo istražiti vrste virtualizacija.



Slika 17 - Okruženje bez virtualizacije (levo) i virtualizovano okruženje (desno)

8.1. Hardverska virtualizacija

Hardverska virtualizacija ili virtualizacija platforme se odnosi na kreiranje virtualnih mašina koje se ponašaju kao pravi računari sa operativnim sistemom. Softver koji se izvršava na ovim virtualnim mašinama je odvojen od hardverskih resursa koji se nalaze ispod. Primera radi, računar na kome se nalazi Microsoft Windows može da hostuje virtualnu mašinu koja izgleda kao (i jeste) računar sa Ubuntu Linux operativnim sistemom; Ubuntu softver se može izvršavati na ovakvoj mašini.

U hardverskoj virtualizaciji, *host* mašina je prava mašina na kojoj se radi virtualizacija, a *guest* mašina je virtualna mašina. Termini *guest* i *host* se koriste radi distinkcije na softver koji se izvršava na fizičkoj i softver koji se izvršava na virtualnoj mašini. Softver ili firmware koji kreira virtualnu mašinu na hardverskom hostu naziva se *hipervizor* ili *Virtual Machine Monitor*. Različiti tipovi hardverske virtualizacije uključuju:

1. **Punu virtualizaciju:** Skoro kompletnu simulaciju stvarnog hardvera kako bi se softveru omogućilo, a koji se tipično sastoji od guest operativnog sistema, da se izvršava nepromenjen i time praktično u svom native okruženju
2. **Delimičnu virtualizaciju:** Nešto, ali ne sve od ciljnog, host, okruženja je simulirano. Kao posledica toga, neke od guest aplikacija na istim operativnim sistemima će morati da budu u nekoj meri modifikovane kako bi se izvršavale.
3. **Paravirtualizaciju:** Hardversko okruženje nije simulirano; s druge strane, guest aplikacije se izvršavaju u okviru svojih, izolovanih domena, kao da se izvršavaju na posebnom sistemu. Guest aplikacije je potrebno prilagoditi kako bi se izvršavale u ovakvim okruženjima.

Hardware-assisted virtualizacija je način da se unapredi efikasnost hardverske virtualizacije. Uvodi specijalno dizajnirane procesore i hardverske komponente koje unapređuju performanse guest okruženja.

Hardverska virtualizacija nije isto što i hardverska emulacija: u hardverskoj emulaciji, deo hardvera imitira drugi deo, dok u hardverskoj virtualizaciji, hipervizor imitira određeni deo računarskog hardvera ili ceo računar. Dalje, hipervizor nije isto što i emulator; oba su računarski programi koji imitiraju hardver, ali njihovi domeni su različiti.

Razlozi za hardversku virtualizaciju:

- U slučaju hardverske konsolidacije, mnogi manji fizički serveri se zamenjuju jednim većim fizičkim serverom da bi se povećalo iskorišćenje skupih hardverskih resursa poput procesora. Iako je u tom slučaju hardver konsolidovan, operativni sistem(i) obično nisu. Umesto toga, svaki operativni sistem koji se izvršava na fizičkom serveru postaje konvertovan u poseban operativni sistem koji se izvršava unutar virtualne mašine. Veliki server može da hostuje mnogo takvih guest virtualnih mašina. Ovo se zove i Physical-to-Virtual (P2V) transformacija.
- Virtualna mašina se može lako kontrolisati i ispitivati i nadzirati spolja u odnosu na fizičku mašinu. Njeno konfigurisanje je znatno fleksibilnije. Ovo je veoma korisno prilikom razvoja kernela i prilikom držanja kurseva iz operativnih sistema.
- Nova virtualna mašina može biti dobavljena po potrebi, bez da to bude ispraćeno prethodnom nabavkom (dodatnog) hardvera.

- Virtualna mašina se može lako relocirati sa jedne na drugu fizičku mašinu, po potrebi. Na primer, prodavac koji ide kod klijenta može da iskopira virtualnu mašinu sa demo softverom i ponese je na svom laptop računaru, bez potrebe da u te svrhe nosi hardver. Slično tome, greška unutar virtualne mašine ne šteti host sistemu, te ne postoji rizik da se sruši sistem na laptopu.
- Zahvaljujući osobini lake relokacije, virtualne mašine se mogu koristiti u kontekstu disaster recovery scenarija. S druge strane, kada se više virtualnih mašina konkurentno izvršava na istom fizičkom hostu, svaka virtualna mašina može da varira i ponaša se nestabilno po pitanju performansi. Ovo je u direktnoj zavisnosti od poslova koji obavljaju ostale virtualne mašine koje se izvršavaju na istom fizičkom hostu, osim ukoliko se uposle tehnike temporalne izolacije među virtualnim mašinama.

Primeri virtualizacionih scenarija:

- Izvršavanje jedne ili više aplikacija koje nisu podržane od strane host operativnog sistema: Virtualna mašina koja izvršava potreban guest operativni sistem bi mogla da omogućiti aplikaciji da se izvršava, bez ikakvih intervencija na host operativnom sistemu.
- Evaluacija alternativnog operativnog sistema: Novi operativni sistem može da se izvršava unutar virtualne mašine, bez uticaja na host operativni sistem.
- Server virtualizacija: Višestruki serveri bi mogli da se izvršavaju na jednoj fizičkoj platformi (serveru), kako bi efikasnije koristili raspoložive hardverske resurse tog fizičkog servera.
- Duplikacija specifičnih okruženja: Virtualna mašina bi mogla, zavisno od toga koji se virtualizacioni softver koristi, da bude duplicirana na više hostova, ili da bude ponovo uspostavljena (eng. *restored*) pod pretpostavkom da je prethodno izvršen backup iste virtualne mašine.
- Formiranje zaštićenog okruženja: Ukoliko se guest operativni sistem koji se izvršava na virtuelnoj mašini ošteti na način koji je teško popraviti, primera radi takav slučaj može nastati prilikom izučavanja ponašanja zlonamernog koda (eng. *Malware*) ili instalacijom softvera koji kompromituje stabilnost sistema iz bilo kog razloga, virtualna mašina se može jednostavno odbaciti bez ikakvih negativnih posledica po host sistem, te se može naredni put koristiti nova, identična kopija (instanca) ove iste virtualne mašine.

8.2. Puna virtualizacija

U računarskoj nauci, pod punom virtualizacijom [65] podrazumeva se virtualizaciona tehnika koja se koristi da se dobavi određena vrsta virtualnog okruženja, takva koja simulira u potpunosti hardver koji se nalazi ispod virtualizacionog sloja. Puna virtualizacija zahteva da svaka bitna osobina hardvera bude u potpunosti podržan u okviru jedne ili više virtualnih mašina, uključujući pri tome i skup instrukcija, ulazno izlaznih operacija, sistemske prekide (eng. *Interrupts*), pristup memoriji, i koji god drugi elementi da su korišćeni od strane softvera koji se izvršava na samom hardveru, a koji je namenjen da se izvršava i na virtualnoj mašini. U ovakvom okruženju, svaki softver sposoban da se izvršava na samom hardveru, gde najčešće biva operativni sistem, može biti izvršavan u virtualnom okruženju. Očigledan test virtualizacije je da li se operativni sistem koji je namenjen samostalnom izvršavanju direktno na hardveru može da se izvršava unutar ovakve virtualne mašine.

Drugi oblici platformske virtualizacije dozvoljavaju samo izvesnim ili modifikovanim softverima da se izvršavaju na virtualnim mašinama. Koncept pune virtualizacije ima čvrsto uporište u literaturi, ali se ne referencira uvek ovim terminom.

Važan primer pune virtualizacije je prvi put urađen za IBM CP/CMS operativni sistem. Demonstracija je urađena na IBM CP-40 sistemu 1967. godine, a zatim distribuirana kao koncept otvorenog koda u okviru CP/CMS 1967-1972, i ponovo implementirana kao IBM VM family od 1972. do danas. Svaki CP/CMS korisnik je imao simuliranu, nezavisnu instancu računara. Svaka takva virtualna mašina je imala pun skup mogućnosti hardvera na kom se izvršava i kao takva je za korisnika istog sistema bila identična svojoj fizičkoj instanci. Simulacija je bila potpuna i bila je bazirana na priručniku za hardver - *Principles of Operation*. Tako, uključivala je elemente poput skupa instrukcija, glavne memorije, sistemskih prekida, izuzetaka (eng. *Exceptions*), i pristupa uređajima. Rezultat je bio jedna mašina koja je mogla biti multipleksirana između korisnika.

Puna virtualizacija je moguća jedino pravilnim kombinovanjem hardverskih i softverskih elemenata. Na primer, nije bilo moguće sa mnogim drugim IBM hardverskim platformama izvesti sličnu stvar. Primera radi, to nije bilo moguće izvesti sa System/360 ili sa 370 serijama do 1972. godine kada je dodat hardverski modul za virtualizaciju.

Slično tome, virtualizacija nije bila moguća na x86 platformama do 2005-2006 te dodataka na hardverskom nivou od strane AMD-a i Intela u vidu AMD-V i Intel VT-x ekstenzija. Mnoge platforme za virtualizaciju su tvrdile i pre ovoga da su postigle željeni cilj. Neki od njih su: Adeos, Mac-on-Linux, Parallels Desktop for Mac, Parallels Workstation, VMware Workstation, VMware Server (ranije GSX Server), VirtualBox, Win4BSD i Win4Lin Pro. VMware, na primer, koristi tehniku binarne translacije da automatski modifikuje x86 softver u letu i prekodira u odgovarajući skup instrukcija. Ovakva tehnika se može smatrati punom virtualizacijom.

Ključni izazov za punu virtualizaciju je presretanje i simulacija privilegovanih operacija, poput ulazno/izlaznih instrukcija. Efekti svake operacije moraju biti čuvani u okviru date virtualne mašine – virtualnim operacijama ne sme biti dozvoljeno da utiču na operacije drugih virtualnih mašina, kontrolnog programa ili hardvera. Neke instrukcije se mogu izvršiti direktno na hardveru, pošto su njihovi efekti u potpunosti sastavljeni od elemenata kojima upravlja kontrolni program, poput memorijskih lokacija i aritmetičkih registara. Ali druge instrukcije koje bi uticale na virtualne mašine ne smeju biti direktno izvršavane; one moraju biti uhvaćene i simulirane. Takve instrukcije ili očitavaju ili utiču na stanje informacija koje je van virtualne mašine.

Puna virtualizacija se pokazala visoko uspešnom za:

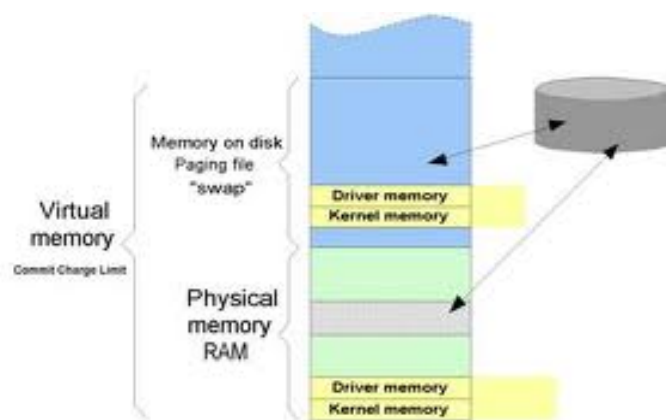
- Deljenje računarskih sistema među korisnicima;
- Izolovanjem korisnika međusobno (i od kontrolnog programa);
- Emulacijom novog hardvera da se postigne unapređena pouzdanost, bezbednost i produktivnost;



Slika 18 – Puna virtualizacija, šematski prikaz

8.3. Delimična virtualizacija

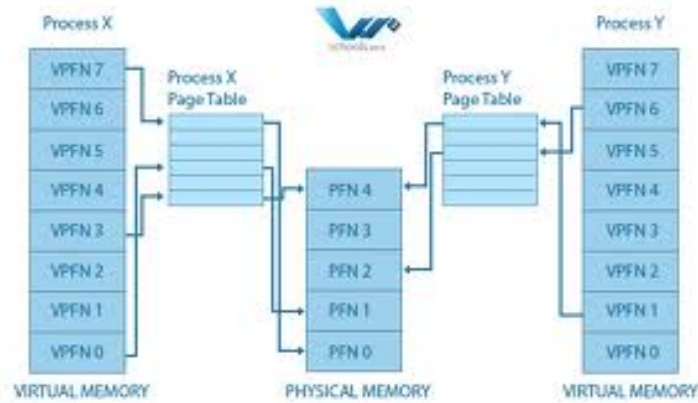
U okviru delimične ili parcijalne virtualizacije [66], uključujući i virtualizaciju adresnog prostora, virtualne mašine simuliraju višestruke instance hardvera koji se nalazi ispod njih, naročito adresnog prostora. Obično, ovo znači da ceo operativni sistem ne može da se izvršava unutar virtualne mašine jer bi to bila zapravo potpuna virtualizacija, ali to znači da mnoge aplikacije mogu da se izvršavaju. Ključni deo svake parcijalne virtualizacije je virtualizacija adresnog prostora, u kom svaka virtualna mašina ima svoj nezavisni adresni prostor. Da bi se realizovala ova mogućnost, zahteva se proces koji se bavi relociranjem adresnog prostora i njegovim mapiranjem na fizičku adresu.



Slika 19 – Delimična virtualizacija, šematski prikaz[67]

Parcijalna virtualizacija bila je bitan istorijski događaj na putu ka punoj virtualizaciji. Korišćena je u prvoj generaciji time sharing sistema CTSS, u IBM M44/44X sistemu kao i sistemu poput MVS i Commodore 64. Ovaj termin se takođe može koristiti da se opiše bilo koji operativni sistem koji omogućava poseban adresni prostor za pojedinačne korisnike ili procese, uključujući mnoge koji se danas ne bi razmatrali kao sistemi sa virtualnim mašinama. Iskustva sa parcijalnom virtualizacijom i njenim ograničenjima dovelo je do pravljenja prvih sistema sa potpunom virtualizacijom (IBM-ov CP-40, prva iteracija CP/CMS koji kasnije postali IBM-ova VM Family). Mnogi noviji sistemi poput Microsoft Windowsa i Linuxa, koriste ove mehanizme.

Delimičnu virtualizaciju je znatno jednostavnije implementirati u odnosu na punu virtualizaciju koja je često obezbeđivala korisne, robusne virtualne mašine, sposobne da podrže važne aplikacije. Parcijalne virtualizacije su se pokazale visoko uspešnim za deljenje računarskih resursa među korisnicima.



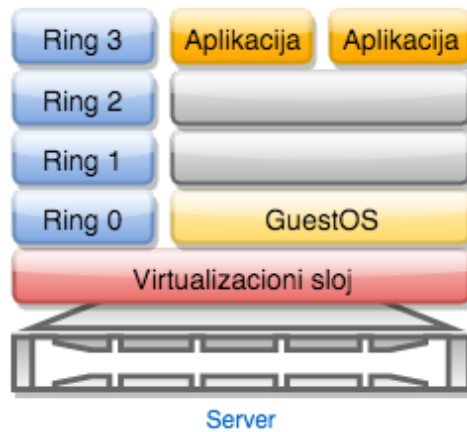
Slika 20 – Detaljniji prikaz delimične virtualizacije[67]

S druge strane, u poređenju sa punom virtualizacijom, negativna strana je u situacijama koje zahtevaju kompatibilnost unazad ili portabilnost. Teško je anticipirati precizno koje opcije koristi koja aplikacija. Ukoliko se određeni hardverski zahtevi ne simuliraju, u tom slučaju softver koji koristi te funkcije neće moći da se izvršava.

8.4. Paravirtualizacija

Paravirtualizacija [66] je virtualizaciona tehnika koja prezentira softverski interfejs virtualnoj mašini koja je slična ali ne i identična hardveru koji je pogoni.

Namena modifikovanog interfejsa je smanjenje dela izvršnog vremena guest mašine koji se troši na operacije koje je značajno teže izvršavati u virtualnom okruženju u poređenju sa nevirtualizovanim okruženjem. Virtualizacija obezbeđuje specijalno definisane interfejse (eng. *Hooks*) kako bi se obezbedilo guest i host operativnim sistemima da upute zahteve i razmenjuju informacije vezano za zadatke, koji bi se inače izvršavali u virtualnom domenu (gde su performanse izvršavanja lošije). Uspešno paravirtualizovana platforma može dozvoliti VMM menadžeru (eng. *virtual machine monitor*) da bude jednostavniji (prebacivanjem kritičnih zadataka iz virtualnog u host domen) i/ili da smanji sveukupnu degradaciju performansi mašinskog izvršavanja unutar virtualne guest mašine.

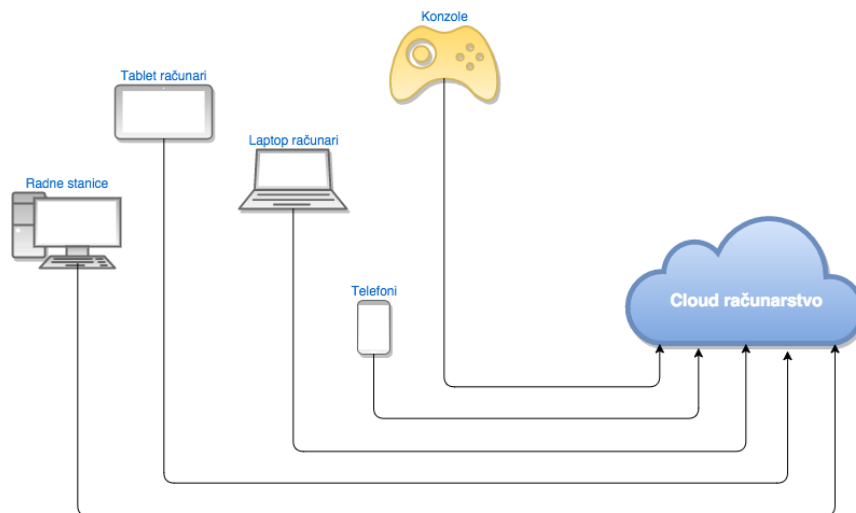


Slika 21 – Paravirtualizacija - GuestOS je paravirtualizovano okruženje

Paravirtualizacija zahteva da na guest operativni sistem bude eksplicitno portovan para API – konvencionalna distribucija operativnog sistema koja nije svesna paravirtualizacije ne može se izvršavati povrh paravirtualizacionog VMM. S druge strane, čak i u slučajevima gde operativni sistem ne može biti modifikovan, i dalje komponente mogu biti raspoložive. One će omogućiti mnoge od značajnih prednosti u performansama paravirtualizacije. Na primer, XenWindowsGplPv projekat sadrži skup paravirtualizaciono svesnih drajvera pod GPL licencom koji su namenjeni da budu instalirani na Microsoft Windows virtualnu guest mašinu koja se izvršava na Xen hipervizoru.

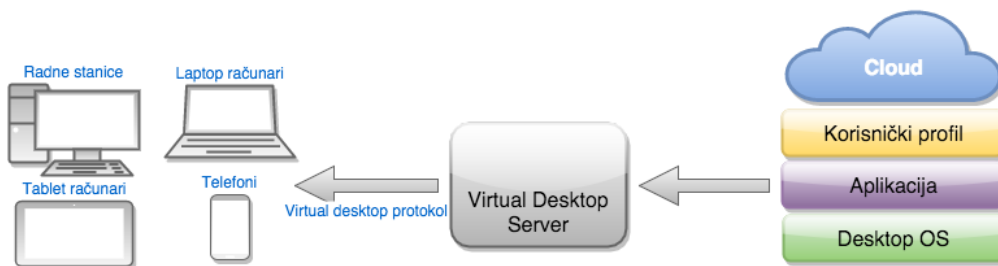
8.5. Desktop virtualizacija

Desktop virtualizacija [68] ili klijentska virtualizacija je koncept separacije logičke desktop mašine od fizičke. Jedan od oblika desktop virtualizacije, virtualna desktop infrastruktura (VDI), može biti posmatrana kao napredniji oblik hardverske virtualizacije. Umesto putem direktne interakcije na relaciji računar - ulazno/izlazni uređaji poput tastature, miša i monitora, korisnik stupa u interakciju sa host računarom putem računarske mreže kao što je LAN, WLAN ili Internet koristeći drugi desktop računar ili mobilni uređaj. Dodatno, host računar u ovom scenariju postaje serverski računar sposoban da hostuje više virtualnih mašina u isto vreme za više korisnika.



Slika 22 - Desktop virtualizacija

Kako organizacije nastavljaju da virtualizuju infrastrukturu i konvergiraju svoja data centar okruženja, klijentska arhitektura nastavlja da evoluiru kako bi se iskoristila prednost predvidivosti, kontinuiteta i kvaliteta usluga isporučenih rešenja u oblasti infrastrukture. Primera radi, kompanije poput HP-a, IBM-a, obezbeđuju hibridni VDI model uz opseg virtualizacionog softvera i modela isporuke kako bi unapredili ograničenja distribuiranog klijentskog računarstva. Odabrana klijentska okruženja pomeraju obradni posao sa PC platforme na obradne centre i njihove servere, kreirajući lako upravljive virtualne klijente, sa aplikacijama i klijent operativnim okruženjima koja se hostuju u tim data centrima.



Slika 23 - Desktop virtualizacija, arhitektura

Za korisnike, ovo praktično znači da mogu pristupiti svojim desktopima sa bilo koje lokacije, bez da budu vezani za pojedinačni (klijentski) uređaj. Obzirom da su resursi centralizovani, korisnici koji se kreću, menjaju lokaciju sa koje rade, mogu pristupiti istom okruženju i svojim podacima i aplikacijama. Za ICT administratore, ovo znači više centralizovano, efikasno klijentsko okruženje koje je lakše za održavanje i sposobno da agilnije odgovori na promene i potrebe svojih korisnika i njihovog biznisa.

Još jedan oblik, virtualizacija sesije (eng. *Session virtualization*), omogućava više korisnika da se priključi i prijavi na deljen ali moćan računar preko mreže i da svi korisnici ovaj sistem koriste simultano. Svakom korisniku je dat desktop i folder za čuvanje podataka. Sa takozvanom multiseat konfiguracijom, virtualizacija sesije se može postići korišćenjem jednog PC računara sa višestrukim monitorima, tastaturama i miševima.

Tanki klijenti, koji se sreću kod desktop virtualizacije, su jednostavni i jeftini računari dizajnirani za namenu priključenja na mrežu. Njima može nedostajati značajan prostor na hard disku, RAM memorija pa čak i procesorska snaga.

Korišćenjem tehnologije desktop virtualizacije kompaniji se omogućava fleksibilno korišćenje resursa i raste joj kompetitivnost na tržištu. Virtualni desktopi omogućavaju da se implementira brže i sa višim stepenom ekspertize. Ispravno testiranje se može izvesti bez potrebe da se uznemiri krajnji korisnik. Pomeranje desktop okruženja u cloud istovremeno omogućava manje tačaka na kojima sistem može da krahira, ukoliko se omogući trećoj strani da upravlja bezbednošću i infrastrukturom.



Slika 24 - Desktop virtualizacija, šematski prikaz [68]

8.6. Softverska virtualizacija

Softverska virtualizacija [69], ili virtualizacija na nivou operativnog sistema, hostuje višestruka okruženja operativnog sistema u okviru jedne instance operativnog sistema.

Ovo je virtualizacioni metod gde jezgro (eng. *Kernel*) operativnog sistema dozvoljava višestruke instance izolovanog korisničkog prostora, umesto samo jedne. Ovakve instance, često nazivane kontejnerima, VE, VPS ili jails) mogu izgledati kao pravi serveri, sa stanovništva njihovih korisnika. Na Unix sistemima, o ovoj tehnologiji se može razmišljati kao o naprednoj implementaciji standardnog chroot mehanizma. Dodatno na izolacione mehanizme, kernel često obezbeđuje opcije upravljanja resursima kako bi se ograničio međusobni uticaj kontejnera.

Virtualizacija na nivou operativnog sistema se često koristi u scenarijima virtualnog hostinga gde je korisno da se bezbedno alociraju ograničeni hardverski resursi među velikim brojem međusobno nepoverljivim korisnicima. Sistem administratori takođe mogu koristiti ovaj koncept, u manjem obimu, prilikom konsolidacije serverskog hardvera migracijom servisa na odvojene kontejnere istog fizičkog servera.

Ostali često korišćeni scenariji uključuju odvajanje aplikacija radi povećane bezbednosti, hardverske nezavisnosti ili zbog boljeg upravljanja datim resursom/aplikacijom.

Implementacije na nivou operativnog sistema omogućavaju migraciju na živo (bez spuštanja virtualnih mašina) u svrhe dinamičkog upravljanja i raspoređivanja opterećenjem među kontejnerima u okviru klastera.

Ovaj oblik virtualizacije je dosta nizak sa nivoom overheda jer aplikacije u virtualnim particijama koriste interfejs za standardne sistemske pozive i ne postoji potreba za emulacijom ili među-virtualnom mašinom, kao što je to slučaj sa sistemima za virtualizaciju poput Vmware i QEMU ili sistemima za paravirtualizaciju kao što su Xen i UML. Takođe ne zahteva hardversku podršku da bi se efikasno izvršavalo.

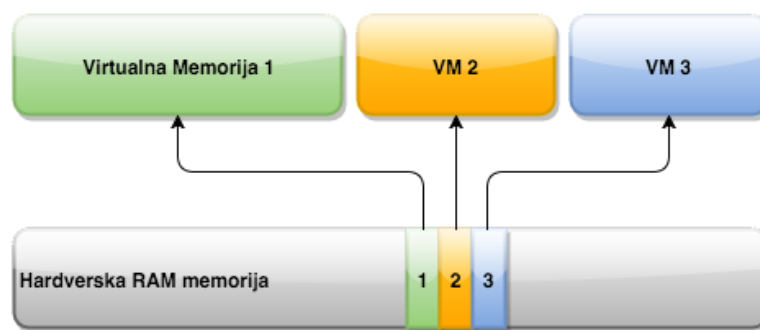
Softverska virtualizacija nije fleksibilna u onoj meri u kojoj su to druge tehnologije virtualizacije jer ne može da hostuje različit guest operativni sistem u odnosu na host, ili drugi guest kernel. Primera radi, u okviru Linuxa, druge distribucije su podržane, međutim drugi operativni sistemi kao što je Windows se ne mogu hostovati. Ovo ograničenje je delimično prevaziđeno kod Solarisa u okviru funkcionalnosti koja se naziva brendirane zone. Kod nje je moguća izvesna modifikacija okruženja na takav način da se omogući okruženje u okviru kontejnera koje emulira starije verzije Solarisa, 8 i 9, na host verziji 10. Ovakva funkcionalnost je bila planirana i za Linux ali je napušten razvoj.

Tabela 8 – Tabela kompatibilnosti hostova [70]

Mehanizam	Operativni sistem	Tip licence
chroot	većia Linux/Unix sistema	Privatna, BSD, GNU, GPL, CDDL
iCore Virtual Accounts	Windows	Privatna, Freeware
Linux/Vserver	Linux	GNU GPL v.2
LXC	Linux	GNU GPL v.2
OpenVZ	Linux	GNU GPL v.2
Parallels Virtuozzo kontejneri	Linux, Windows, Mac	Privatna
Container/Zone	Solaris, OpenSolaris	CDDL
FreeBSD Jail	FreeBSD	BSD
sysjail	OpenBSD, NetBSD	BSD
WPARs	AIX	Privatna
HP-UX kontejneri	HPUX	Privatna
Sandboxie	Windows	Shareware

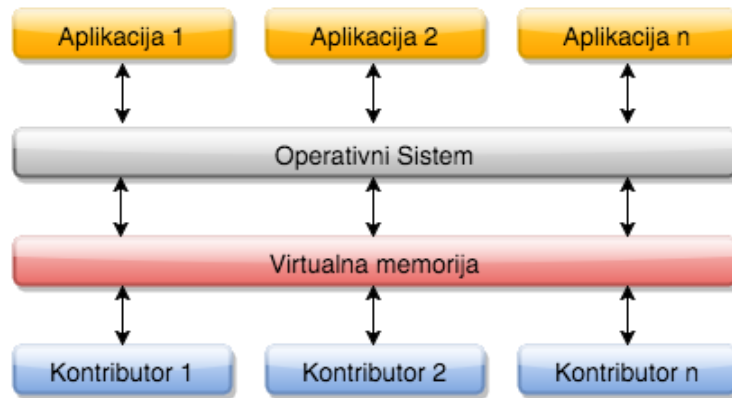
8.7. Virtualizacija memorije

Virtualizacijom memorije [69] se postiže separacija ili distinkcija fizičkog resursa promenljive memorije sa slučajnim pristupom (RAM) od pojedinačnog sistema u okviru data centra, u cilju agregacije potrebnih memorijskih resursa u okviru virtualizovanih memorijskih resursa raspoloživih računarima u okviru datacentra. Ovako virtualizovani memorijski resursi su dostupni aplikacijama koje se izvršavaju pomoću operativnih sistema. Ovako distribuirano organizovan memorijski resurs se može koristiti kao keš memorija visokih performansi, sloj za razmenu poruka ili veliki memorijski resurs za CPU ili GPU aplikacije (centralni ili grafički procesor, respektivno).



Slika 25 - Virtualizacija memorije, arhitektura

Tehnologije virtualizacije memorije omogućavaju umrežene, samim tim i distribuirane, servere koji dele ove memorijske bazene kako bi se prevazišla fizička ograničenja memorijskih resursa. Uz ovu funkcionalnost ugrađenu u mrežu, aplikacije mogu da iskoriste prednosti veoma velikih memorijskih resursa kako bi se poboljšale sveukupne performanse, sistemsko iskorišćenje, povećanje efikasnosti memorijskog iskorišćenja kao i novi načini iskorišćenja ovako organizovanih memorijskih resursa. Softver na memorijskim čvorovima (eng. *Nodes*) tj. serverima, omogućava čvorovima da se povežu na memorijski bazen u cilju njegovog stvaranja, upisivanja i čitanja iz njega. Softver za upravljanje ovakvim strukturama obavlja unos podataka, izvoz podataka, definiše pravila i politiku dobavljanja memorijskih resursa, dodeljuje zadatke čvorovima koji čine ovaj resurs i upravlja zahtevima klijentskih čvorova. Memorijskom bazenu se može pristupiti na nivou aplikacije ili operativnog sistema. Na aplikativnom nivou, bazenu se pristupa kroz API ili preko mrežnog fajl sistema kako bi se ostvario deljeni memorijski resurs visokih performansi. Na nivou operativnog sistema, page keš može koristiti memorijski bazen kao velik resurs koji je mnogo brži od lokalnog ili umreženog memorijskog prostora.



Slika 26 - Skup svih memorija dostupnih operativnom sistemu

Virtualizacije memorije se razlikuju od deljenih memorijskih sistema. Deljeni memorijski sistemi ne dozvoljavaju apstrakciju memorijskih resursa, zahtevajući implementaciju unutar jedne instance operativnog sistema, naspram klasterovanog aplikacijskog okruženja.

Virtualizacija memorije se takođe razlikuje od virtualizacija skladišnih (eng. *Storage*) medija – flash, solid state hard diskova (SSD) i sličnih tehnologija. Ova vrsta medija tehnologije se odnosi na hard diskove i/ili medije za permanentno skladištenje podataka, dok se tehnologija virtualizacije memorije odnosi na zamenu ili dopunu tradicionalne RAM memorije.

Virtualizacija memorije je tehnologija nastala iz upravljanja memorijom i tehnika upravljanja virtualnom memorijom. U oba polja, putanja inovacije se pomerila od čvrsto spregnutih odnosa između logičkih i fizičkih resursa ka znatno fleksibilnijim, abstraktnim odnosima gde su fizički resursi alocirani po potrebi.

Virtualni memorijski sistemi nude sloj apstrakcije između fizičke i logičke adrese u RAM memoriji, na znatno fleksibilniji način gde se fizički resursi alociraju po potrebi.

Sistemi sa virtualnom memorijom apstrahuju između fizičkog RAM-a i virtualnih adresa, dodeljujući virtualnoj adresnoj memoriji adrese fizičke RAM memorije ali i skladišnog prostora na diskovima, te na taj način proširujući operativnu memoriju preko kapaciteta na račun brzine. NUMA i SMP arhitekture optimizuju memorijsku alokaciju unutar višeprosorskih sistema. Dok ove tehnologije dinamički upravljaju memorijom u okviru individualnih računara, virtualizacija memorije upravlja agregiranim memorijskim resursima više umreženih računara i tretira ih kao jedinstven memorijski entitet.

U tandemu sa inovacijama u oblasti upravljanja memorijom, pojavio se veći broj tehnika virtualizacije kako bi se hardverski resursi iskoristili na najbolji način. Virtualizacija aplikacija je prvi put urađena na mainframe računarima. Naredni korak je bio virtualizacija skladišne infrastrukture – storidž sistema, ka resursima kao što su diskovi na serveru, NAS ili SAN uređaji. Server virtualizacija, ili puna virtualizacija, partitionisanje jednog fizičkog servera na više virtualnih mašina, konsolidacija višestrukih instanci operativnog sistema u okviru iste mašine u svrhu više efikasnosti i fleksibilnosti. Kako kod virtualizacije servera, tako i kod virtualizacije storidža, aplikacija nije svesna resursa koji koristi, kao ni njegove distinkcije na logičku, odnosno fizičku prirodu. Na ovaj način, efikasnost i fleksibilnost se postižu bez potrebe za rekonfiguracijom aplikacija. Na isti način, virtualizacija memorije alocira memoriju celokupnog umreženog klastera servera među računarima u tom klasteru.

8.8. Virtualizacija sistema za skladištenja podataka

Virtualizacija storidža [69] je koncept i termin korišćen u okviru računarskih nauka. Specifično, storidž sistemi mogu koristiti virtualizacione koncepte kao alate kako bi obezbedili bolju funkcionalnost i naprednije opcije unutar storidž sistema.

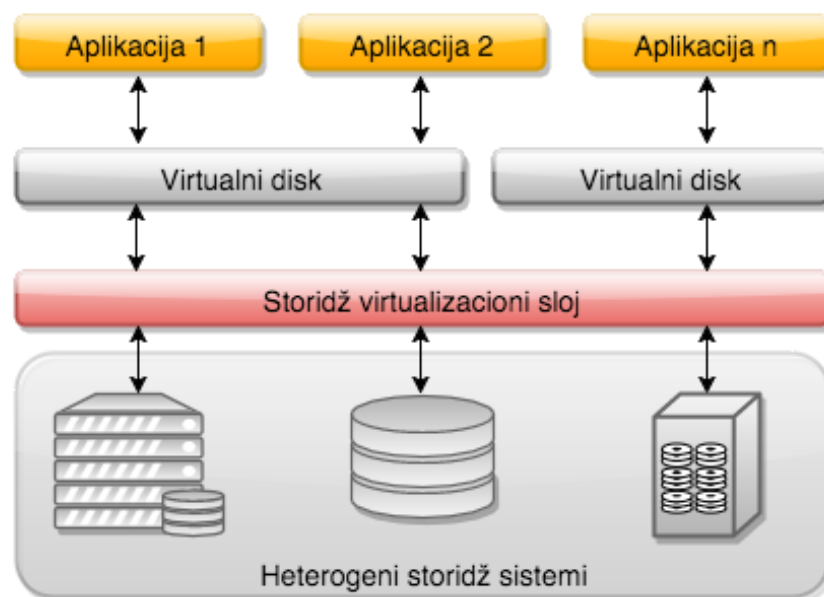
Šire govoreći, storidž sistem je takođe poznat kao *storage array*, *disk array* ili *filer*. Storiđž sistemi tipično koriste poseban hardver i softver zajedno sa diskovima kako bi obezbedili veoma brzo i pouzdano skladište za obradu podataka. Storiđž sistemi su dosta kompleksni, te se mogu smatrati računarima sa posebnom namenom dizajniranim da obezbede skladišne kapacitete zajedno sa naprednom zaštitom podataka. Hard diskovi unutar storidža su samo jedan element ovakvog sistema, zajedno sa hardverom i softverom sa posebnom namenom.

Storiđž sistemi mogu da obezbede ili pristup na osnovu blokova podataka, ili na osnovu fajlova. Prvi pristup se obično isporučuje putem nekog od sledećih protokola: Fibre Channel, iSCSI, SAS, FICON. Pristup baziran na fajlovima se obezbeđuje putem NFS ili CIFS protokola.

Unutar konteksta storidž sistema, postoje dva primarna tipa virtualizacije:

- **Blokovska virtualizacija** u ovom kontekstu se odnosi na sloj apstrakcije (separacije) logičkog storidža (particije) u odnosu na fizički storidž na taj način da mu se može prići bez obzira na fizičku strukturu i heterogenu strukturu. Ova separacija omogućava administratorima storidž sistema veću fleksibilnost u smislu upravljanja krajnjim korisnicima.
- **Virtualizacija na nivou fajlova** adresira NAS izazove eliminišući zavisnosti između podataka kojima se pristupa na nivou fajla i lokacije gde se fajlovi fizički nalaze. Ovo omogućava optimizaciju storidža i konsolidaciju servera kako bi se izvršila migracija fajlova bez prekida u radu.

Virtualizacija storidža omogućava da se postigne nezavisnost od lokacije apstrakcijom fizičke lokacije podataka. Virtualizacioni sistemi vrše prezentaciju ka korisniku logičkog prostora za skladištenje podataka i upravljaju procesom mapiranja fizičke lokacije tih podataka.



Slika 27 - Virtualizacija sistema za skladištenje podataka

Moguće je imati višestruke slojeve virtualizacionog sloja ili mapiranja. Zatim je moguće izlaz jednog sloja virtualizacije koristiti kao ulaz u naredni, viši sloj virtualizacije. Virtualizacija mapira prostor sa *back-end* resursa na *front-end* resurse. U ovoj instanci, *back-end* se odnosi na logičku jedinicu (Logical Unit Identifier, ili LUN) koja se ne prezentuje računaru ili host sistemu za direktno korišćenje. *Front-end* LUN ili drugi referentni prikaz (npr. *volume*) se prezentuje host računaru za korišćenje.

Stvarni oblik mapiranja će zavisiti od izabranih implementacija. Neke implementacije mogu ograničiti granularnost mapiranja što može ograničiti mogućnosti uređaja. Obično se granularnosti kreću od jednog fizičkog diska do manjeg skupa fizičkih diskova.

U storidž okruženju baziranom na blokovima, pojedinačni blok informacija se adresira korišćenjem LUN identifikatora i ofseta u okviru tog LUN-a, poznatog kao logičko blokovsko adresiranje (eng. *Logical Block Address, LBA*)

Virtualizacioni softver ili uređaj je odgovoran za održavanje konzistentnog pogleda (prezentacije) svih informacija u vezi sa mapiranjem virtualizovanog storidža. Ovo mapiranje informacija često se naziva metapodacima i skladišti se u tabeli mapiranja.

Adresni prostor može biti ograničen kapacitetom potrebnim za tabelu mapiranja. Nivo granularnosti i ukupan adresni prostor mogu uticati direktno na veličinu metapodataka, te na taj način i na tabelu mapiranja. Iz ovog razloga, često se prave kompromisi, između količine adresnog kapaciteta i granularnosti ili pristupne granularnosti.

Jedna često korišćena metoda da se adresiraju ovi limiti je višestruki nivoi virtualizacije. U nekoliko storidž sistema koji se danas koriste, čest slučaj je da se primenjuju tri sloja virtualizacije.

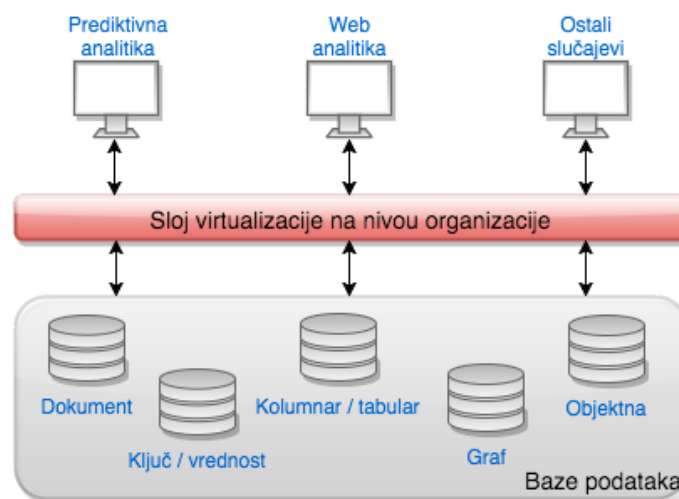
Pojedine implementacije ne koriste tabelu mapiranja i umesto toga koriste algoritam za računanje lokacije. Ove implementacije koriste dinamičke metode da izračunaju lokaciju pristupa, pre nego da čuvaju informaciju u tabeli mapiranja.

8.9. Virtualizacija podataka

Virtualizacija podataka opisuje proces apstrakcije različitih izvora podataka (baze podataka, aplikacije, repozitorijumi fajlova, veb-sajtovi, itd.) kroz jedinstven pristupni sloj podataka (koji se može sastojati od nekoliko pristupnih mehanizama).

Apstrakcija omogućava pristup klijentima putem jednog pristupnog sloja, serijalizacije, formata, strukture, itd. kako svaki klijent ne bi morao da upravlja sa više ili svim ovim slojevima.

Ovaj koncept i softver se često koriste u okviru integracije podataka, poslovne inteligencije, servisno orijentisane arhitekture, cloud računarstva, enterprajz pretraživanja, migraciji master podataka i virtualnim upravljanjem master podacima.



Slika 28 - Virtualizacija podataka

Softver za virtualizaciju podataka može uključivati funkcije razvoja, operacije i/ili upravljanja. Softver za virtualizaciju je tehnologija koja obezbeđuje neke od sledećih funkcionalnosti:

- **Apstrakciju** – Apstrahuje tehnički aspekt skladištenja podataka kao što su lokacija, struktura skladišta, API, pristupni jezik i tehnologija skladištenja.
- **Virtualizovani pristup podacima** – Povezivanje sa različitim izvorima podataka i ostvarivanje pristupa ka istim tim podacima iz jedne logičke tačke.
- **Transformacija / Integracija** – Transformacija, poboljšanje kvaliteta i integrisanje podataka baziranih na upotrebi iz više izvora.

- **Federacija podataka** – Kombinovanje skupa rezultata sa različitih izvora.
- **Fleksibilno isporučivanje podataka** – Objavljivanje skupa rezultata kao pogleda i/ili usluge isporuke podataka aplikacijama ili korisnicima na zahtev.

8.10. Virtualizacija u oblasti računarskih mreža

Virtualizacija mreža [69] je proces kombinovanja hardverskih i softverskih mrežnih resursa i mrežne funkcionalnosti u jedan, softverski baziran administrativni entitet, virtuelnu mrežu. Mrežna virtualizacija uključuje virtualizaciju platforme, često kombinovanu sa virtualizacijom resursa.

Mrežna virtualizacija se može kategorizovati kao eksterna, kombinujući mnoge mreže ili delove mreže u virtualnu jedinicu, ili interna, omogućujući funkcionalnost nalik mrežnim softverskim kontejnerima na jedinstvenom sistemu. Bilo da je virtualizacija interna ili eksterna, na implementacionom nivou se razlikuju proizvođačka rešenja.

Komponente virtualne mreže

Različiti prodavci (eng. *vendors*) hardverskih i softverskih komponenti nude virtualizaciju mrežne infrastrukture kombinacijom sledećeg:

4. Mrežni hardver, kao što su svičevi i mrežni adapteri, takođe poznati kao mrežni interfejs karte (eng. *Network Interface Card*, NIC)
5. Mrežni elementi poput fajervola, load balansera
6. Mreže, kao što su virtualne lokalne mreže (eng. *Virtual Local Area Network*, VLAN) i kontejneri poput virtualnih mašina VM i Solaris kontejnera
7. Mrežni storage uređaji
8. Mrežni M2M elementi poput telekomunikacionih 4G HLR i SLR uređaja
9. Mrežni mobilni elementi poput laptopova, tableta i mobilnih telefona
10. Mrežni medij, poput ethernet ili fibre channel tehnologija

U nastavku sledi pregled jednog čestog scenarija mrežne virtualizacije i primer proizvođačke implementacije tih scenarija.

Neki proizvođači nude **eksternu mrežnu virtualizaciju**, u okviru koje jedna ili više logičkih mreža bivaju kombinovane ili deljene u poddomene u okviru virtualnih mreža, sa ciljem da unaprede efikasnost velikih kompanijskih mreža ili datacentara. Ključna komponenta ovakve eksterne mreže su VLAN-ovi i mrežni svičevi. Korišćenjem VLAN i svičing tehnologija, sistem administrator može da konfigurira sisteme fizički zakačene za istu mrežu a koji se nalaze u okviru druge virtualne mreže. Na taj način, VLAN tehnologije omogućavaju sistem administratorima da kombinuju sisteme na odvojenim lokalnim mrežama u VLAN spanning tree segmente velikih korporativnih mreža.

Drugi prodavci nude **virtualizaciju internih mreža**. Ovde se jedan sistem konfigurira u okviru kontejnera poput Xen domena, kombinuje sa hipervizor kontrolnim aplikacijama ili pseudo interfejsima poput VNIC interfejsa sa ciljem da se stvori „mreža u kutiji“. Ovakvo rešenje unapređuje opštu efikasnost na nivou pojedinačnog sistema izolacijom aplikacija u posebne kontejnere i/ili pseudo interfejse.

Primeri interne mrežne virtualizacije

Citrix i Vyatta su napravili Virtual Network stack kombinovanjem Vyatta rutiranja, fajervol usluga i Ipsec VPN funkcionalnosti sa Citrix Netscaler load balanserom, Branch Repeater WAN optimizacijom i Access Gateway SSL VPNom. vNetworkStack projekat definiše celokupnu arhitekturu virtualne mreže za ekspozitive / udaljene kancelarije, datacentre i cloud računarsko okruženje.

OpenSolaris network virtualization funkcionalnosti omogućuju scenario tipa „mreža u kutiji“. Funkcionalnosti OpenSolaris Crossbow projekta daju mogućnosti u okviru kontejnera poput zona ili virtualnih mašina na jednom sistemu da dele resurse i razmenjuju podatke. Glavne funkcionalnosti Crossbow-a uključuju VNIC pseudo-interfejse i virtualne svičeve, koji emuliraju mrežnu povezanost omogućavajući kontejnerima da razmenjuju podatke bez potrebe za prosleđivanjem podataka u eksternu mrežu.

Microsoft Virtual Server koristi virtualne mašine, nalik Xen-ovim, da napravi „mrežu u kutiji“ scenario za x86 platformu. Ovi kontejneri mogu da pogone različite operativne sisteme, kao što su Windows i Linux i mogu biti povezani sa ili nezavisni od systemske mrežne karte (NIC-a).

Kombinovana interna i eksterna mrežna virtualizacija

Pojedini prodavci nude eksterna i interna rešenja mrežne virtualizacije kao softverske proizvode. Na primer, **Machine-To-Machine Intelligence (M2MI)** tehnologija pokriva interne, eksterne, softverske mrežne elemente više prodavaca, ali i hardverske mrežne elemente. M2MI je jedinstvena po svom pristupu primene “bele liste” blokiranja pristupa po svim rešenjima mrežnih komponenti više prodavaca. Ovaj pristup omogućava da virtualna mašina ne može da bude napadnuta tehnikom “ARP spoof”, tehnikom koja se koristi da se kompromituju virtualne mašine na mrežnom nivou. Ako bismo upoređivali sa OSI referentnim modelom, to bi onda pripadalo sloju 2. **VMware** nudi proizvode koji nude samo internu i eksternu mrežnu virtualizaciju. VMwareov osnovni pristup je “mreža u kutiji” na jedinstvenom sistemu, korišćenjem virtualne mašine kojom se upravlja hipervizorskim softverom. VMware dalje nudi VMware Infrastructure softver kako bi povezo i kombinovao mreže u višestruke mrežne kutije i dalje u eksterni virtualizacioni scenario.

9. Hipervizor kao osnova cloud sistema

9.1. Xen Hipervizor

Xen hipervizor je sloj softvera koji se izvršava direktno na računarskom hardveru preuzimajući ulogu operativnog sistema te na taj način omogućavajući računarskom hardveru da hostuje i izvršava konkurentno više operativnih sistema na istoj platformi.

Xen je alat industrijskog nivoa i takođe je alat otvorenog izvornog koda. Nudi efikasnu i bezbednu virtualizaciju za razne arhitekture procesora. Neke od njih su x86, x86_64, IA64, ARM ali podržava i druge. Xenov hipervizor podržava razne host operativne sisteme kao što su Windows, Linux, Solaris i razne verzije BSD operativnog sistema.

Xen pokreće mnoge javne cloud servise i mnoge hosting servise, poput Amazon Web Services, Rackspace Hosting i Linode. Komercijalni proizvodi za virtualizaciju poput Oracle VM i XenServer su napravljeni na Xen osnovi, kao i rešenja za desktop virtualizaciju poput QubesOS i XenClient. Od verzije 3.0, sav kod potreban da se pogoni Linux kao Xen menadžment operativni sistem ili guest, je deo Linux jezgra.

Računar sa Xen hipervizorom sadrži sledeće tri komponente:

- **Xen Hipervizor** – virtualizacioni sloj direktno do hardvera.
- **Domain 0**, privilegovan domen (Dom0) – Privilegovan guest sistem koji se izvršava sa direktnim pristupom hardveru i služi za menadžment guest virtualnih mašina.
- Višestruki **DomainU**, neprivegovani guest operativni (domenski) sistemi, DomU – Neprivilegovani guest operativni sistemi koji se izvršavaju na hipervizoru nemaju direktan pristup hardveru poput memorije, diska, itd.

Dom0 je obično modifikovana verzija Linux, NetBSD ili Solaris operativnih sistema. Korisnički domeni mogu biti bilo nemodifikovane kopije operativnih sistema pod licencom otvorenog izvornog koda ili proizvođačko rešenje, poput Microsoft Windows sistema, ukoliko host procesor podržava x86 virtualizaciju (npr. Intel VT-x ili AMD-v) ili ukoliko sam host sistem poseduje posebne drajvere za podršku Xen sistema za paravirtualizaciju.

Virtualizacija serverske infrastrukture može doneti sledeće dobrobiti:

- Konsolidaciju,
- Povećan stepen iskorišćenja,
- Rapidno dobavljanje resursa,
- Dinamička otpornost na kvarove na najnižem nivou naspram softverskog (eng. *rapid bootstrapping* ili *rebooting*),
- Hardverska otpornost na kvarove (migracijom virtualne mašine na drugačiji hardver),
- Mogućnost da se bezbedno odvoje virtualni operativni sistemi,
- Mogućnost da se podrži nasleđen nekompatibilan i/ili specifičan softver kao i nove instance operativnog sistema na istom računaru.

Xenova podrška za migraciju na živo sa jednog na drugi host omogućava upravljanjem količinom opterećenja i omogućava prevenciju potrebe za spuštanjem virtualnih mašina.

Virtualizacija takođe ima prednosti kada se radi na razvoju, uključujući i sam razvoj operativnih sistema. Izvršavanje novog sistema kao kao guest operativnog sistema eliminiše potrebu za hardverskim restartovanjem računara kad god nastane greška koja to zahteva. Posebno zatvoreni guest sistemi mogu biti veoma korisni u proučavanju ponašanja računarskih virusa ili crva bez mogućnosti kompromitovanja van kontrolisanog okruženja. Ovo se postiže takozvanim Sandbox sistemima.

Konačno, proizvođači hardverskih komponenti mogu doneti odluku da svoje sisteme isporučuju sa nekoliko virtualnih guest sistema, kako bi njihovi proizvodi ili rešenja mogli da izvršavaju aplikacije koje se nalaze na više različitih operativnih sistema.

Xen može da se izvršava kao virtualizaciona platforma, kao što je to Citrix XenServer Enterprise Edition (prethodno poznat kao XenEnterprise), ili ugrađen u host operativni sistem. Kao primer druge kategorije, Xen je sastavni deo sledećeg:

- SUSE Linux Enterprise 10 distribucije,
- Red Hat RHEL 5 i Fedora 7,
- Sun Microsystems, Solaris,
- Debian Etch.

Na većini procesora, Xen koristi oblik virtualizacije poznat kao paravirtualizacija, što znači da guest operativni sistem izvršava specijalni hypercall ABI umesto da se oslanja na određenu arhitekturu i njene funkcionalnosti. Kroz proces paravirtualizacije, Xen postiže odlične performanse čak i na x86 arhitekturi koja ima reputaciju nekooperativne arhitekture za tradicionalne virtualizacione tehnologije.

Na x86, Xen host kernel kod se izvršava u okviru Prstena 0, dok se hostovani domeni izvršavaju u okviru Prstena 1 ili Prstena 3 nivoa.

Intel i AMD su doprineli modifikaciji Xen platforme kako bi podržala njihove ekstenzije u okviru arhitekture, Intel VT-x i AMD-V, respektivno. Iako se ove tehnologije implementaciono veoma razlikuju, kao i po skupu instrukcija, Xen upravlja njima putem zajedničkog sloja apstrakcije, omogućavajući nemodifikovanom guest operativnom sistemu da se izvršava unutar Xen virtualne mašine. Ovo je moguće još od verzije Xen 3.0.

Ovaj, malopre pomenut razvoj, omogućava virtualizaciju proizvođačkih sistema poput Microsoft Windows, obzirom da guest sistemski kernel ne zahteva modifikacije kada se host izvršava na procesorskoj arhitekturi koja poseduje skup instrukcija za virtualizaciju, tačnije Intel VT-x ili AMD-V.

Hardver assisted virtualizacija (HVM) nudi novi skup instrukcija koje podržavaju direktne pozive od strane paravirtualizovanog guest sistema ili drajvera u hipervizor i to se obično koristi za ulazno izlazne operacije ili neke druge takozvane hiperpozive (eng. *Hypercalls*). Takođe omogućavaju dodatne režime izvršavanja: “root mode” i “non-root mode”. Oba ova režima se izvršavaju u okviru prstenova 0-3. Xen host se izvršava u *root* režimu i ima pristup pravom hardveru, dok se nemodifikovani guest izvršava u prstenovima od 0-3 u *non root* režimu, sa svojim “hardverom” kome pristupa pod punom kontrolom hipervizora.

Xen-HVM poseduje hardversku emulaciju baziranu na QEMU projektu kako bi obezbedio ulazno izlazne procese u virtualnim mašinama. Sistem emulira hardver putem preko QEMU menadžera za uređaje (*qemu-dm*), daemon koji se izvršava kao back-end proces u okviru dom0. Ovo znači da virtualne mašine ovo vide kao hardver. Xen 3.0.2 podržava sledeće nemodifikovane guest sisteme:

- Generalno Windows familiju operativnih Sistema,
- Linux.

Što se tiče ograničenja koja Xen ima, spomenućemo da ne može da iskoristi više od 100 paravirtualizovanih blok uređaja po hostu. Dodatno, neki operativni sistemi poput OpenSuse imaju podržane limite. Xen 4.1 na OpenSuse može da radi sa do 64 fizička procesora, 2TB RAM i 64 virtualne mašine po server. Komercijalne implementacije Xen servera su:

- Citrix XenServer,
- Oracle VM,
- Sun Xvm,
- Virtual Iron,
- Thinsy Corporation.

9.2. Hyper-V hipervizor

Hyper-V postoji u dve varijante: kao samostalan proizvod pod nazivom Microsoft Hyper-V Server 2008 i kao dodatna funkcionalnost koja se instalira po potrebi uz Windows Server 2008 i R2 verzijom istog.

Samostalna verzije Hyper-V hipervizora je besplatna. To je zapravo varijanta Windows 2008 Core koja uključuje Hyper-V funkcionalnost. Ostale Windows 2008 funkcionalnosti su onemogućene a Windows usluge su takođe ograničene. Besplatni Hyper-V Server 2008 je limitiran na interfejs sa komandnom linijom (eng. *Command Line Interface*), gde konfiguracija “host” ili “parent” (Hyper-V Server 2008) operativnog sistema fizičkog hardvera i softvera radi korišćenjem shell komandi. Novi CLI je baziran na meniju i to čini početnu konfiguraciju dosta jednostavnijom, a neke besplatno dostupne skripte to dalje proširuju. Administracija i konfiguracija “host” (Hyper-V Server 2008 OS) i guest virtualnih sistema se uglavnom rade tako što se skine sa Internet mreže i instalira dodatno dodatak Microsoft Management Console (MMC). Ova MMC konzola, tačnije snap in za menadžment konzolu se instalira na, npr. Windows 7 PC računar ili drugi Windows 2008 Server, ili čak i na System Center Virtual Machine.

Alternativno, drugi Windows Server 2008 računar, sa Hyper-V instaliranom funkcionalnošću se može instalirati kako bi se upravljalo Hyper-V Server 2008 mašinom uz pomoć mehanizma redirekcije konzole. Dalja konfiguracija i upravljanje Hyper-V Server 2008 se može ostvariti korišćenjem Remote Desktop, odnosno RDP sesije ili redirektovanom standardnom MMC konzolom kao što je *Computer Management* ili *Group Policy (Local)*. Takođe, moguće je administrirati i sa pune instalacije Windows 2008 Server sistema. Od Hyper-V Server R2 Microsoft je isporučio i Windows Powershell v2, sa značajnim unapređenjima za bolju CLI kontrolu, i ažuriranu bazu koda Windows Server 2008 R2.

Hyper-V podržava izolaciju u smislu particije. Particija je logička jedinica izolacije, podržana od hipervizora u okviru koje se izvršava operativni sistem. Hipervizor instanca mora imati bar jednu *parent* particiju, na kojoj se izvršava Windows Server 2008. Virtualizacioni stek se izvršava na toj particiji i ima direktan pristup hardverskim uređajima. Parent particija zatim kreira *child* particije koje hostuju guest operativne sisteme. *Parent* particija to radi korišćenjem hypercall API poziva, koji su aplikativni programski interfejs Hyper-V platforme.

Virtualizovana particija nema pristup fizičkom procesoru, niti upravlja njegovim pravim prekidima (eng. *Interrupts*). Umesto toga, ima virtualni pogled na procesor i izvršava se u *Guest Virtual Address* prostoru, koji, zavisno od konfiguracije hipervizora, može biti deo ili ceo virtualni adresni prostor. Hipervizor može da izabere da asocira samo podskup procesora svakoj od particija. Hipervizor upravlja prekidima ka procesoru, i preusmerava ih ka odgovarajućoj particiji korišćenjem logičkog *Synthetic Interrupt Controller* (SynIC). Hyper-V može da vrši hardversko ubrzanje procesa prevođenja adresa (eng. *Address translation*) guest sistema korišćenjem drugog nivoa translacije ili prevođenja koju obezbeđuje procesor, a koja se naziva EPT na Intel platformi i RVI (bivša NPT) na AMD platformi.

Child particije nemaju direktan pristup hardverskim resursima, ali umesto toga imaju virtualni pogled na resurse, u smislu virtualnih uređaja. Bilo kakav zahtev ka virtualnim uređajima se preusmerava putem VMBus mehanizma ka uređajima na *parent* particiji koja će obrađivati zahteve. VMBus je logički kanal koji omogućava međuparticionu komunikaciju. Odgovor se takođe šalje posredstvom VMBus mehanizma. Ukoliko su uređaji u okviru *parent* particije takođe virtualni uređaji, biće dalje preusmerena komunikacija dok se ne stigne do *parent* particije koja ima direktnu komunikaciju sa fizičkim hardverom. *Parent* particije izvršavaju i *Virtualization Service Provider* (VSP). VSP se povezuju na VMBus i upravljaju pristupom ovoj magistrali i prosleđuje na nju zahteve iz *child* particija. virtualni uređaji irtualni uređaji *child* particije interno se izvršavaju kao virtualizacioni servisni klijenti (eng. *Virtualization Service Client, VSC*), koji redirektuju zahteve ka VSPovima u *parent* particiji preko VMBus-a. Ceo proces je transparentan za guest operativni sistem.

Virtualni uređaji takođe mogu da iskoriste prednost Windows Server virtualizacionih funkcionalnosti, koje se nazivaju *Enlightened I/O*, za storidž, mrežne i grafičke podsisteme, između ostalih. Ova *Enlightened I/O* tehnologija je specijalizovana implementacija svesna virtualizovanog okruženja te sposobna da iskoristi VMBus direktno, što omogućava zaobilazanje bilo kakvog sloja emulacije uređaja. Ovo čini komunikaciju efikasnijom, ali zahteva od guest sistema da podržava *Enlightened I/O*. Windows Server 2008, Windows 7, Windows Vista, Red Hat enterprise Linux su jedini operativni sistemi koji spadaju u tu grupu koji podržavaju *Enlightened I/O*, što im omogućava da se brže izvršavaju kao guest operativni sistemi pod Hyper-V platformom u odnosu na druge sisteme koji moraju da koriste emulaciju hardvera.

9.3. vSphere hipervizor

VMWare spada u kategoriju hipervizora koji se izvršava direktno na hardveru i odvaja guest mašine potpuno od hardvera. Iz tog razloga ga nazivaju “bare metal hypervisor”.

VMWare softver omogućava kompletno virtualizovan set hardvera ka guest operativnim sistemima. Vmware softver virtualizuje hardver za video adapter, mrežni adapter i adapter ka hard disku. Host omogućava prosleđivanje drajvera za guest operativne sisteme po pitanju USB, serijskih i paralelnih uređaja. Na ovaj način, VMWare virtualne mašine postaju visoko portabilne između računara, jer svaki host izgleda gotovo potpuno identično sa stanovišta guest mašine. U praksi, sistem administrator može da pauzira izvršavanje guest mašine, pomeri je ili kopira taj guest na drugi fizički računar, pomeri ili kopira tu mašinu na drugi fizički računar i nakon toga nastavi izvršavanje tačno na tački na kojoj je izvršavanje zaustavljeno. Alternativno, za enterprise servere, postoji rešenje koje se zove vMotion koje omogućava migraciju operativnih guest virtualnih mašina među različitim ali sličnim hardverskim hostovima uz mogućnost da pritom dele identični storidž. Svaka od ovih tranzicija je transparentna korisnicima virtualnih mašina tokom vremena trajanja migracije.

Vmware hipervizor ima dosta bolji i optimizovaniji način izvršavanja guest operativnih sistema (takođe se nazivaju još i target sistemima) u odnosu na emulatore poput Bochs koji simulira funkciju svake procesorske instrukcije na target mašini jedan na jedan, ili vrši dinamičku rekompilaciju koja kompajlira blok mašinskih instrukcija prvi put kad se izvrše a iza toga koriste preveden kod direktno kada se kod izvršava. Ovaj pristup koristi Microsoft Virtual PC za Mac OS X. VMWare softver ne emulira skup instrukcija za različit hardver koji nije fizički prisutan. Ova bitna razlika doprinosi značajno performansama ali u isto vreme stvara probleme prilikom migracija virtualnih mašina na drugi fizički host koji koristi drugi skup instrukcija, recimo sa Intel na AMD procesora ili čak i među sistemima sa različitim brojem procesora. Zaustavljanje guest mašine pre migracije na drugu arhitekturu ne stvara probleme u nastavku.

VMWare proizvodi korene svojih virtualizacionh ekstenzija vuku iz x86 instrukcijskog skupa, i ne zahtevaju procesore koji podržavaju virtualizaciju. Na takvim starijim procesorima, koriste procesor za direktno izvršavanje koda kada god je to moguće. Primer je tzv. user-režim (eng. *User-mode*), kao i virtualni 8086 režim koji se izvršava na x86 platformi.

Kada nije moguće izvršavati direktno, kao što je sa kernel nivo i kod u takozvanom realnom modu, Vmware proizvodi prepisuju kod dinamički, a proces Vmware poziva funkciju binarne translacije ili BT. Preveden kod biva uskladišten u rezervnu memoriju, obično na kraju adresnog prostora, čiji mehanizmi segmentacije štite i obično čine nevidljivim ovaj deo memorije. Iz ovih razloga, VMware radi znatno brže od emulatora, izvršavajući se sa više od 80% performansi brzine kojom bi se virtualni guest operativni sistem izvršavao bez hipervizora na istom tom hardveru. U jednoj od svojih studija, VMware tvrdi da se virtualizovan host izvršava 0-6% sporije pod VMware ESX serverom u odnosu na direktno izvršavanje na hardveru.

VMware pristup izbegava određene poteškoće vezane za virtualizaciju x86 platforme. Virtualne mašine mogu da se izbore sa određenim spornim instrukcijama tako što ih zamene drugim, ili jednostavno izvršavanjem kernel koda u korisničkom modu. Zamenom skupa instrukcija raste rizik da kod ne uspe u pronalaženju očekivanog sadržaja kojim menja datu instrukciju. Ukoliko sam sebe čita; nije moguće zaštititi kod od čitanja za vreme dok se omogućava normalno izvršavanje, te zamena u letu instrukcija postaje komplikovana. Izvršavanje koda nemodifikovanog u korisničkom modu takođe neće uspeti jer većina instrukcija koje čita stanje mašine i ne pozivaju sistemski izuzetke (eng. *exceptions*). Onda dolazi do greške usled nepoznavanja stanja u kome se sistem nalazi, dok izvesne instrukcije bez objave menjaju ponašanje u korisničkom modu. Iz tog razloga je potrebno prepisivati, izvršavajući simulaciju tekućeg programskog brojača, na originalnoj lokaciji. Kada god je to potrebno, treba primeniti i remapiranje hardverskih tačaka relaska (eng. *breakpoint*).

VMware ima dva proizvoda iz segmenta bare metal hipervizora, ono što industrija najčešće prepoznaje kao VMware vSphere ESX i VMware vSphere ESXi. Ova dva proizvoda su slična, međutim, u slučaju ESXi verzije je uklonjena servisna konzola i zamenjena minimalnom BusyBox instalacijom. Dalje ESXi je manje zahtevan što se tiče skladišnog prostora na disku kao i memorijskog footprinta. ESXi je namenjen da se izvršava sa fleš diska na serveru ali se takođe može pokretati i sa hard diska. ESXi hostovima se ne može upravljati direktno sa konzole. Svo upravljanje se dešava kroz VirtualCenter Server.

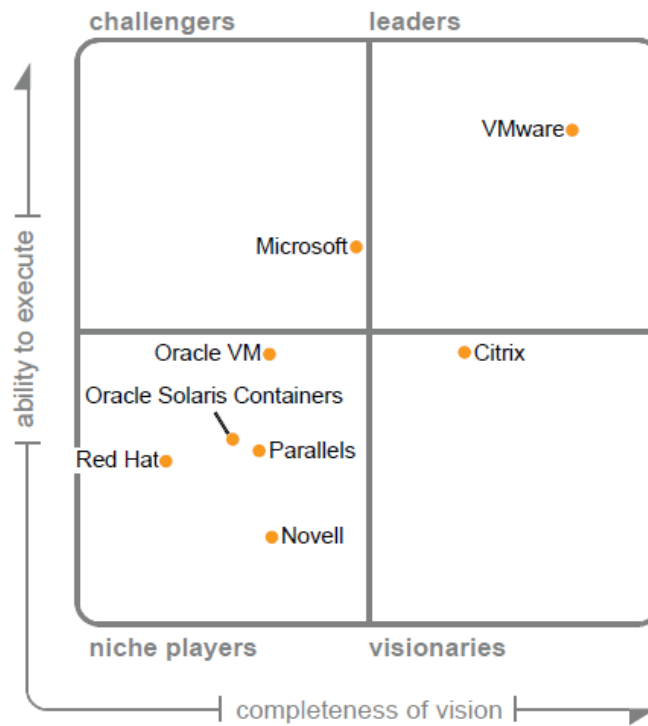
9.4. Poslovni kriterijumi uporednog pregleda hipervizora kao osnove za cloud

Virtualizacija serverske infrastrukture omogućava osnovu za nov način upravljanja serverske platforme, kao i skup alata za automatizaciju, nove bezbednosne arhitekture i nove procese. Iako su virtualizacione tehnologije sa aspekta cloud sistema tek tehnologije koje omogućavaju ove koncepte, činjenicu da ih nose ne treba zanemarivati. Svakako, ove tehnologije omogućavaju da se klijenti usmere ka višim servisima poput upravljanja i automatizacijama procesa a da ne brinu o infrastrukturi. S tim u vezi, tehnološki izbori koji nose celokupnu platformu su veoma bitni.

Tržište je u poslednje vreme postalo veoma konkurentno u domenu hipervizora. Dok Vmware, budući pionir u ovoj oblasti, i dalje suvereno vlada, očekuje se da će tržište rasti u narednih bar još godinu do dve. Preuzimanja ili spajanja kompanija ali i investicije su dovele velike proizvođače poput Microsofta i Oracle-a na ovo tržište. Veliki broj kompanija iz tržišnog segmenta malih i srednjih preduzeća tek treba da virtualizuje svoju infrastrukturu. Rešenja koja su ovakvim kompanijama na raspolaganju do pre samo nekoliko godina nisu postojala. Nadalje, ovo tržište će nastaviti da raste kroz tehnološke inovacije, investicije i preuzimanje firmi, odnosno kupovine. Kao domeni rasta u ovoj oblasti percipiraju se dodatni alati za upravljanje ovakvim infrastrukturama, automatizaciju procesa i uspešno korišćenje tehnologije virtualizacije u okviru cloud sistema. Nesumnjivo, virtualizacione platforme će ostati osnova za ovakve sisteme i sve više će dobijati na značaju.

Tehnološki pioniri virtualizacionih tehnologija su kompanije Vmware, kao najpoznatija po rešenjima za velika preduzeća, SWsoft, danas poznatija pod imenom Parallels, i projekat otvorenog izvornog koda, Xen, koji je namenjen dobavljačima usluga.

S druge strane, tržište se dosta izmenilo. Gartnerov magični kvadrant daje prikaz trenutnog stanja na tržištu, odnos snaga po proizvođačima i uvid u stratešku orijentaciju. Fokus ovog prikaza je na x86 platformi.



Slika 29 - Garnterov kvadrant za x86 serversku virtualizaciju [71]

Iz Garnterovog magičnog kvadranta vidimo da su VMware, Citrix i Microsoft lideri na tržištu. Microsoft ima poziciju izazivača preostala dva lidera. Gartner je dao pregled isključivo proizvođačkih tehnologija.

9.4.1. Kriterijum sposobnosti izvršavanja

Kriterijum sposobnosti izvršavanja bazira se na evaluaciji kvaliteta i efikasnosti procesa, sistema, metoda ili procedura koje omogućavaju ICT dobavljačima usluga performanse kako bi bili kompetitivni, efikasni i efektivni, sa sveukupno pozitivnim uticajem na prihod, zadržavanje i reputaciju. Na kraju, tehnološki dobavljači su evaluirani po osnovu sposobnosti da kapitalizuju svoju viziju.

Tabela 9 – Kriterijumi evaluacije za sposobnosti realizacije

Kriterijum evaluacije	Težina
Proizvod / Usluga	Visoka
Opšta vitalnost (Poslovnih jedinica, Finansije, Strategija, Organizacija)	Visoka
Realizovanih prodaja / Politika cena	Visoka
Tržišna responzivnost i praćenje	Niska
Marketing	Visoka
Korisničko iskustvo	Standardno
Operacije	Nisko

Sposobnost izvršavanja u domenu virtualizacije serverske infrastrukture nije vezano isključivo za funkcionalnosti proizvoda, već u velikoj meri za održavanje konstantno promenljivog poslovnog modela sa veoma dinamičnim trendom. Odlični proizvodi mogu propasti, ali i loši proizvodi mogu biti veoma uspešni, zavisno od efektivne proizvođačke realizacije.

Proizvod ili usluga: ključni proizvodi i usluge u ponudi tehnološkog ponuđača koji zaokružuju određeno tržište. Ovo uključuje trenutne proizvodne i uslužne kapacitete, kvalitet, skup funkcionalnosti, veštine, itd., bilo da su u ponudi direktno ili kroz originalne proizvođače opreme (OEM) i partnerske ugovore. Ključni faktori koji se evaluiraju uključuju: opseg operativnih sistema, skalabilnost i efikasnost, elastičnost, zrelost, upravljanje embedded resursima, upravljanje opcijama za efikasniju administraciju, mogućnost administriranja sistema kao celine, virtualizovani ekosistemi, administrativna skalabilnost i integracija sa drugim dobavljačima usluge upravljanja i održavanja ovakvih sistema.

Opšta vitalnost (poslovnih jedinica, finansije, strategija, organizacija): procena celokupne finansijske vitalnosti organizacije, finansijski i praktičan uspeh poslovnih jedinica i verovatnoća

individualnih poslovnih jedinica kako bi se nastavilo sa investicijom u proizvod, što podrazumeva nastavak nuđenja i podrške za proizvod, kao i nastavak ponude tehnološki najsofisticiranijeg proizvoda u okviru tehnološkog proizvodnog portfolia kompanije.

Realizovanih prodaja / politika cena: mogućnosti tehnološkog dobavljača usluga u svim pretprodajnim aktivnostima, kao i struktura koja ih podržava. Ovo uključuje upravljanje sklapanjem poslova, cenovnom politikom i pregovaranjem, pretprodajnom podrškom i celokupnom efektivnošću prodajnih kanala.

Tržišna responzivnost i praćenje: sposobnost odgovora, promene pravca, fleksibilnosti i konkurentnosti uz razvoj mogućnosti, konkurentnog delovanja, evolucije korisničkih zahteva i tržišne dinamičke promene. Ovaj kriterijum takođe razmatra dobavljačevu istoriju i responzivnost.

Marketing: jasnost, kvalitet, kreativnost i efikasnost programa dizajniranih da isporuče poruku koju organizacija želi da iskomunicira i na taj način utiče na tržište, promocija brenda i posla, povećanje svesti vezane za proizvod, i uspostavljanje pozitivne identifikacije sa proizvodom i/ili brendom od strane kupaca. Ovo je moguće postići kombinacijom publikovanja, promotivnih aktivnosti, liderskim nastupom, ličnom preporukom i proizvodnih aktivnosti.

Korisničko iskustvo: odnosi, proizvodi i usluge/programi koji omogućuju klijentima da budu uspešni sa proizvodima koji se evaluiraju. Specifično, ovo uključuje načine na koji korisnici primaju tehničku i ostalu podršku. Ovo takođe može uključivati pomoćne alate, programe korisničke podrške (te i kvalitet), dostupnost korisničkih grupa, ugovore servisne podrške, itd.

Operacije: sposobnost organizacije da ispuni svoje ciljeve i obaveze. Faktori uključuju kvalitet organizacione strukture poput veština, iskustava, programa, sistema i drugih elemenata potrebnih organizaciji kako bi efektivno i efikasno nastavila da funkcioniše.

Tabela 10 - Kriterijum evaluacije, kompletnost vizije

Kriterijum evaluacije	Težina
Razumevanje tržišta	Visoka
Marketing strategija	Visoka
Prodajna strategija	Standardna
Ponuda (proizvodne) strategije	Standardna
Bizis Model	Standardna
Vertikalna / Industrijska strategija	Standardna
Inovacija	Standardna
Geografska strategija	Niska

9.4.2. Kriterijum kompletnosti vizije

Evaluirani su tehnološki dobavljači po osnovu svoje sposobnosti da ubedljivo artikuliraju logičke izjave povodom trenutnih i budućih tržišnih pravaca, inovacija, korisničkih potreba i kompetitivnih snaga. Na kraju, tehnološki dobavljači usluga se rangiraju po svom razumevanju pojma kako se tržišne snage mogu iskoristiti da kreiraju nove mogućnosti za dobavljače.

U domenu serverske virtualizacije, razumevanje dobavljača tehnoloških usluga i njegova artikulacija strateških tehnoloških putanja vezanih za virtualizaciju jesu naročito bitni. Ovo se odnosi na širenje osnova buduće infrastructure. Njena dalja moguća ekspanzija ka cloud računarstvu je od velikog značaja i to ga diferencira.

Razumevanje tržišta: sposobnost tehnološkog dobavljača da razume potrebu svojih klijenata i da ih prevede u proizvode i usluge. Dobavljači usluga koji pokazuju nivo vizije slušaju i shvataju želje i potrebe klijenata, mogu oblikovati ili unaprediti te iste želje i potrebe sopstvenim dodatkom vizije.

Marketing strategija: jasan i diferenciran skup poruka konzistentno komuniciran kroz organizaciju i slat kroz veb-sajt, reklame, korisničke programe i izjave za pozicioniranje.

Prodajna strategija: strategija prodaje proizvoda koji koriste odgovarajuću mrežu direktne i indirektno prodaje, marketinga, usluga i komunikacionih posrednika koji šire okvire dosezanja tržišta, veština, ekspertize, tehnologije, usluga i baze korisnika.

Ponuda (proizvodne) strategije: pristup tehnološkog dobavljača razvoju proizvoda i isporuci koja uključuje strategiju diferencijacije, funkcionalnosti, metodologije i skupa funkcionalnosti, kako se mapirao na trenutne i buduće zahteve.

Biznis model: čvrstoća i logika fundamentalne logike poslovne ponude tehnološkog dobavljača.

Vertikalna / industrijska strategija: strategija tehnološkog dobavljača da upravlja resursima, veštinama i ponudama kako bi se odgovorilo na specifične potrebe ili pojedinačne tržišne segmente, uključujući i vertikalne.

Inovacija: direktni, povezani, komplementarni i sinergijski raspored resursa, ekspertiza ili kapitalnih investicija, konsolidacija, odbrambena ili preventivna svrha.

Geografska strategija: strategija tehnološkog dobavljača da upravlja resursima, veštinama i ponudama izvan "kuće" ili svoje matične geografije, bilo direktno ili preko partnera, kanala i predstavništava na odgovarajući način, u kontekstu geografije i tržišta.

9.4.3. Uporedni pregled tržišne pozicije

Lideri - VMware je jedini prepoznat kao lider. Dok se izazivači trude da ugroze VMwareovu leadersku poziciju, kako smo videli na slici 30, VMware ima veliku prednost na tržištu. Jasno je ispred po razumevanja tržišta, proizvodnoj strategiji, poslovnom modelu, tehnološkim inovacijama, proizvodnim sposobnostima i realizovanim prodajama. Proizvođački izazov je održati momentum u svim ovim oblastima, što će biti izuzetno teško uz veoma jaku konkurenciju koja uveliko stasava i intenzivno investira i samim tim pojačava pritisak na VMwareov poslovni model.

Izazivači – Microsoft je jedini u kvadrantu izazivača. Microsoft još uvek uči vezano za ovo tržište i njegove potrebe. S tim u vezi, treba da nastavi da unapređuje svoju proizvodnu strategiju i tehnološke inovacije, kako bi postao lider. U ovako rapidnom rastu tržišta, poboljšanje sopstvenih proizvodnih ponuda i realizacija prodaje (naročito u kontekstu velikih klijenata) će ostati veoma važne.

Vizionari – Citrix se nalazi u vizionarskom kvadrantu. Kombinuje duboko razumevanje tržišta, solidnu inovaciju i dobru proizvodnu strategiju uz izazove u realizaciji prodaje, marketinga i pitanja vezanih za dugoročno održanje svog XenServer proizvoda (nasuprot mogućnosti da tržište više prihvati Hyper-V). Pokazatelji rasta Citrix rešenja na tržištu su dobri i Gartner smatra rešenja ovog proizvođača pozicioniranim odmah do rešenja VMwarea. Pitanje druge prirode odnosi se pre na to da li će Citrix moći da ovaploti, odnosno monetizuje taj momentum, pored VMwarea, koji se koncentrisao na enterprajz okruženje, Microsofta, koji se usmerio na manja i srednja preduzeća, i Red Hata, koji se usmerio na zajednicu otvorenog koda.

9.4.4. Razmatranje konkurentskih prednosti proizvođača

CITRIX – Nakon preuzimanja XenSource, Citrix je postao vizionar u domenu virtualizacije, između VMware kao lidera i Microsofta koji se trudi da osvoji leadersku poziciju. Citrix ima viziju, i solidnu proizvodnu ponudu. Izazov za Citrix bio je da nađe tržišnu nišu koja je ili neosvojena ili delimično osvojena od strane Microsofta. Marketing je takođe bio segment koji nije bio u dovoljnoj meri razvijen. Citrix je bio uspešniji sa svojim trenutnim klijentima. Citrixov smeo korak u realizaciji XenServer-a (uključujući XenCenter i XenMotion) rezultovao je velikim poletom aktivacije proizvoda. Uprkos tome, proizvođač nije još uvek uspeo da naplati ugovore o održavanju ili dodacima kroz *Essentials* ponudu. Citrix se takođe trudi da nađe komfornu i komplementarnu ulogu sa Microsoft Hyper-V, dodajući proširene alate za upravljanje na Microsoft ponudu (što može uticati na smanjenje oslanjanja na Xen platformu). Izazov koji se ovde pojavljuje sastoji se u tome da Microsoft još uvek nije definisao tržište server virtualizacije koje će ostaviti Citrix-u. Imajući u vidu oba konkurenta, Microsoft smatra da treba široko da pokrije tržište svojom ponudom. Citrixovi direktori su objavili da Xen hipervizor nije obavezno i strateško opredeljenje. Štaviše, Citrix kaže da će budućnost menadžment virtualizacije ove firme biti zasnovan na Xen ili Hyper-V hipervizorima. Ukoliko Citrix može uspešno da izgradi jak skup alata za upravljanje i poslovni model, njegova strategija svakako može imati smisla.

Snage:

- Vizija za korišćenje virtualizacije - da se omogući cloud računarstvo,
- Proizvodna ponuda – sposobnost inovacije,
- Odnos sa Microsoftom (naspram konkurencije),
- Lider u ponudi proizvoda baziranih na OSS,
- Poslovni model – konverzija aktivacija besplatnog proizvoda ka prihodu za održavanje i alate za upravljanje,
- Realizacija marketinga i penetracija na tržište,
- Dugoročna vitalnost Xen virtualizacione strategije,
- Konkurencija, posebno Red Hat sa KVM proizvodom,
- Izbegavanje tržišnog sukoba u okviru partnerstava sa Microsoftom.

Microsoft – Strategija Microsofta konsolidovana je odnedavno. Do pre šest godina Microsoftov odgovor VMware je bio Virtual Server 2005. Hyper-V je dosta snažnije konkurentsko rešenje, i sa migracijom “na živo” u verziji R2 koja je izašla 2009. godine. Microsoft ima odličan prolaz u manja i srednja preduzeća koje tek stupaju u domen virtualizacije. Microsoftov najveći problem je da prevaziđe VMware-ovu duboku penetraciju tržišta osim kod klijenata koji to nisu pokrenuli kao proces. Na mnogo načina, Microsoft je zauzeo tržišnu nišu sa segmentom klijenata koji su dosta kasno krenuli da usvajaju ove tehnologije. Iako su Microsoftove strategije počele da uključuju koncepte privatnog cloud računarstva (slično poput VMware strategije), manje je verovatno da će Microsoftova korisnička baza ići ka pravljenju privatnih cloud sistema. To su pre svega manje kompanije koje vide manje vrednosti u implementaciji privatnih cloud koncepata i sporije usvajaju tehnološke promene te su i manje spremne na dramatične promene. Microsoft treba da nađe način da privuče postojeće VMware klijente – što nije lako dok god VMware isporučuje kvalitetne proizvode i uslugu. Microsoft će biti veoma ugrožen ukoliko VMware nastavi da širi svoj arhitekturni uticaj u oblasti infrastrukture. Microsoftov najveći adut je da virtualizacija ne treba da bude nezavisan poslovni proces, te će biti u prilici da nastavi da nudi cenovnu prednost u odnosu na VMware.

Snage:

- Administrativno okruženje poznato Windows administratorima,
- Instalirana baza Windows sistema u firmama srednje veličine,
- Snaga rešenja (i cena) za preduzeća srednje veličine,
- Komplementarne snage u aplikativnoj arhitekturi koje mogu proširiti virtualizacioni deo,
- Kompanijska finansijska snaga.

Slabosti:

- Ograničena inovaciona i reaktivna strategija na rapidno evolutivnom tržištu,
- Bitka sa snažnom i postojećom bazom VMware korisnika, posebno u enterprajz okruženjima,
- Microsoftova evolutivna strategija i putanja u vezi sa cloud računarstvom,
- Jedna tačka oslonca host operativnog sistema te i postojanje potrebe za spuštanjem i restartom sistema. Ovo se eventualno može izbeći korišćenjem Server Core verzije.

VMware –nesumnjivo jeste lider na tržištu. Iako je bio pionir na polju x86 serverske virtualizacije, u isto vreme je bilo malo konkurentskog pritiska. Razlog izlaska VMware na tržište bio je vođen studijom slučaja koja se bazirala na potrebi za konsolidacijom serverske infrastrukture, te za uštedom troškova i maksimizacijom iskorišćenja postojećih resursa. Sada kada alternative otvorenog izvornog koda sve više dobijaju na značaju, Microsoft dosta investira na ovom polju, kao i Oracle. VMwareova strategija uspeha mora nastaviti da evoluirá. Kako se penetracija virtualizacije sve više širi, sa sve manjim brojem servera za konsolidaciju, racionalizacija u investiciju u VMware će se pomerati ka poboljšanim operativnim i poslovnim dobitima. VMwareov izazov je da zaštiti i nastavi da gradi svoju bazu korisnika i bude tehnološki lider kako se bude širio u komplementarna tržišta koja se oslanjaju na ove tehnologije, poput cloud računarstva. Kako se tržište bude menjalo, tako će i poslovni model morati da ga prati. Primer ovakvih kretanja je to što je VMwareova ponuda za tržište firmi srednje veličine bila dosta skupa do nedavnog ulaska Microsofta na tržište. Kako konkurentske ponude bivaju sve bolje, VMware će biti u konstantnom izazovu da održi ovu razliku i da nastavi da opravdava relativno visoke cene.

Snage:

- Strategija koja seže sve do infrastrukture za cloud sisteme, nova aplikativna arhitektura i bolji menadžment,
- Tehnološko vođstvo i inovacije,
- Visok stepen zadovoljstva klijenata,
- Velika baza postojećih klijenata (posebno među globalnih 2000), i rapidan rast broja dobavljača usluga koji planiraju da koriste VMware kao rešenje – vCloud inicijativa.

Slabosti:

- Strategija održavanje rasta sa porastom poslovnih potreba,
- Slabija realizacija u segmentu preduzeća srednje veličine,
- Zavisnost od ekspanzije u nova i izazovna komplementarna tržišta (npr. IT i servisna automatizacija).

9.5. Tehnološki kriterijumi uporednog pregleda hipervizora kao osnove za cloud

U tabeli 11 dat je pregled sva tri hipervizora uz poređenje njihovih ključnih karakteristika koje se mogu koristiti prilikom odlučivanja za tehnologiju koja će pokretati neki cloud sistem.

Tabela 11 – Uporedni tehnološki kriterijumi tip 1 hipervizora

Tehnološki kriterijumi	vSphere	Hyper-V	XenServer
Zauzeće prostora na disku (Disk Footprint)	144 MB (VMware vSphere Hypervisor)	>3GB sa Server Core instalacijom ~10GB sa punom instalacijom Windows Servera	>1GB
Nezavisnost od operativnog sistema	Ne postoji oslanjanje na operativni sistem (VMware vSphere Hypervisor)	Oslanja se na Windows 2008 partciju	Oslanja se na Linux u Dom0 menadžment partiji
Drajveri (Hardened Drivers)	Optimizovani od strane proizvođača hardvera	Generički Windows drajveri	Generički Linux drajveri
Napredno upravljanje memorijom	Sposobnost povraćaja nekorišćene alocirane memorije, deduplikacija podataka, kompresija memorijskih stranica	Koristi se tehnika “ballooning”. Ne poseduje sposobnost deduplikacije ili kompresije.	Koristi se tehnika “ballooning”. Ne poseduje sposobnost deduplikacije ili kompresije. Ne prilagođava alokaciju memorije na osnovu upotrebe od strane virtualne mašine.
Napredno upravljanje skladištem (storage)	VMware vStorage VMFS, Storage vMotion, Storage DRS	Nema integrisan klasterovan fajl system, bez “živog” sistema skladištenja podataka	Nema integrisan klaster fajl system, bez mogućnosti migracije na živo u okviru sistema skladištenja podataka, system podržava veoma mali broj “nizova”
Visoka I/O skalabilnost	Direktan drajver model	I/O usko grlo u okviru “roditelj” OS-a	I/O usko grlo u okviru Dom0 menadžment OS-a
Upravljanje resursima hostova	Network traffic shaping, deljenje resursa po VM, postavljen kvalitet servisnih prioriteta za sistem skladištenja i I/O	Nedostaju slične mogućnosti	Nedostaju slične mogućnosti

Unapređenje performansi	AMD RVI, Intel EPT large memory pages, universal 32-way vSMP, VMI paravirtualizacija, VMDirectPath I/O, PV guest SCSI drajver	Large memory pages, 4-way vSMP samo na Windows 2008 i Windows 7 VM-a	Nema large memory pages, nema paravirtualizaciju za guest SCSI uređaje, zahteva nefleksibilan SR-IOV
Tehnologija bezbednosti u okviru virtualne infrastrukture	VMware VMsafe™ Omogućeno na nivou hipervizora	Ništa uporedivo	Ništa uporedivo
Fleksibilna alokacija resursa	Pridruživanje “na živo” virtualnim mašinama vCPU i memorija, VMFS rast kapaciteta, proširivanje diskova, dodavanje virtualnih diskova	Ništa uporedivo	Ništa uporedivo
Kreiranje i upravljanje prilagođenom slikom (image)	VMware image builder omogućava administratorima da naprave prilagođene ESXi slike za različite potrebe kao što je ISO bazirana instalacija, PXE bazirana instalacija i automatski deployment	Ništa uporedivo	Ništa uporedivo
Auto Deployment	vSphere Auto Deploy omogućava brže dobavljanje više hostova. Novi hostovi su automatski dobavljeni bazirano na korisnički definisanim pravilima.	Zahteva detaljno podešavanje systems center configuration menadžera	Ništa uporedivo
Upravljanje fajervol interfejsom	ESXi firewall je service-orijentisan i tzv. “stateless” fajervol koji štiti ESXi 5.0 menadžment interfejs. Konfiguriše se korišćenjem vSphere klijenta ili putem komandne linije esxcli interfejsa.	Ništa uporedivo	Ništa uporedivo
Unapređeni virtualni hardver	32-way virtualno SMP, 1TB virtual RAM / VM, Ne-hardverski ubrzana 3D grafika, USB 3.0 podrška, Unified Extended Firmware Interface (UEFI).	4-way virtualno SMP, 64 GB RAM po VM	8-strano virtualno SMP, 32 GB RAM po VM

Možemo zaključiti da VMware još uvek ima najkompletniju ponudu a da ostala dva proizvođača tek treba da razviju alate, pre svega iz oblasti menadžmenta, skalabilnosti i poslovnog kontinuiteta.

9.6. Zaključak i projekcije vezane za budućnost hipervizora u okviru cloud sistema

Napokon se ostvaruje dugo očekivana situacija da se računarski resursi mogu najmiti na osnovu njihove upotrebe. Princip elastičnosti se upoređuje sa poslovnom potrebom i služi direktno klijentima putem Interneta, kao što količina posla može da naraste (ili se smanji) znatno brže nego pre 20 godina. Nekada je bilo potrebno mnogo godina da bi posao narastao do nekoliko miliona klijenata – sada je to moguće mereno u mesecima.

Sa tačke gledišta cloud dobavljača, pravljenje veoma velikog centra za obradu podataka, cloud centra, na mestima gde je gradnja jeftina, korišćenjem jeftinih komponenti, storidža, i njihovim umrežavanjem, stvara se mogućnost prodavanja istih resursa baziranih na njihovom korišćenju ispod cene datacentra prosečne veličine, dok se profit multipleksira u okviru velikih grupa klijenata. Iz perspektive korisnika cloud sistema, interesantno je novoosnovanim, tzv. startup softverskim firmama, da prave sopstvene datacentre kao što bi to startup biznis uradio za fabričku liniju. Pored startup biznisa, mnoge postojeće organizacije koriste osobinu cloud elastičnosti sistema redovno. Takvi primeri su Washington Post, Pixar, kao i različiti univerziteti.

Cloud sistemi, iako nisu revolucionarni po prepoznavanju tehnologija koje se u njima koriste, jesu revolucionarni po poslovnom konceptu kojim se vrši isporuka bilo krajnjem korisniku, bilo drugom dobavljaču rešenja kao međusloju.

Ovakav pristup će, nadamo se, omogućiti brži i kvalitetniji a pogotovo jednostavniji način na koji se sistemi svih veličina grade ili će se u budućnosti graditi. Pri tome ne treba gubiti iz vida mnoge već predložene prednosti, nove mogućnosti koje postoje i koje će se tek uočavati i realizovati. Cloud je, pre svega, način da se optimizuju način izgradnje sistema i troškovi, kako izgradnje, tako i upotrebe.

Po pitanju tehnologije koja pogoni cloud sisteme, VMware za sada pruža najviše a prati ga Xen. Microsoft je najkraće na ovom tržišnom segmentu i ima ulogu napadača pozicije, ne ugrožava VMware ali cilja na još nepopunjenu nišu, kao i na korisnike male do srednje veličine.

10. Linux kontejneri kao osnova cloud sistema

Bez obzira na implementaciono rešenje, rešenja u oblasti Linux kontejnera se vrlo malo razlikuju. Uglavnom su to unapređenja osnovne verzije kontejnera (eng. *Linux Containers*, LXC).

Svi kontejneri obezbeđuju dve ključne stvari: (1) izolaciju procesa i (2) kontrolu pristupa resursima. Pod izolacijom podrazumevamo mogućnost da izvršavanje poslova na jednom podsistemu ne može da utiče na izvršavanje nekog drugog podsistema u okviru istog sistema, na primer u okviru istog operativnog sistema. Kontrola resursa se odnosi na sposobnost zadržavanja resursa podsistema u okviru željenog i unapred definisanog skupa resursa [72].

10.1. *Linux kontejneri (LXC)*

Linux kontejneri omogućavaju virtualizaciju na nivou operativnog sistema. Ovakav sistem omogućava izvršavanje višestrukih izolovanih (korisničkih) domena na jednom sistemskom domenu (na nivou jednog kernela). Virtualizacija sistema na ovom nivou značajno je manje resursno zahtevna od pune virtualizacije i nudi dosta veliku fleksibilnost. Takođe, kontejneri se mogu izvršavati na svakom operativnom sistemu koji ih podržava.

Kompletno upravljanje resursima radi kernel, kao uslugu kontejnerima. Neki od primera upravljanja resursa su upravljanje memorijom, procesorskim vremenom, sistemskim prekidima i svim drugim resursima.

LXC koristi kontrolne grupe (eng. *cgroups*) u okviru Linux kernela [73]. Cgroups podsistem je sistem za upravljanje resursima koji obezbeđuje okvir za generičko grupisanje procesa. Projekat je započela kompanija Google 2006. godine.

10.2. Docker kontejneri

Doker [74] je omotač oko standardnih Linux kontejnera. Sadrži kompletan fajl sistem, kao i sve ostalo potrebno za izvršavanje: kod, izvršno okruženje, sistemske alate, sistemske biblioteke i sve ostalo potrebno za nezavisno izvršavanje softvera koji se izvršava u okviru Docker (izovanog) Linux kontejnera. Ovo garantuje uvek istovetno izvršavanje softvera, bez obzira na njegovo okruženje [73].

Doker omogućava izvršavanje kontejnera na svim Linux platformama koje se oslanjaju na kernel verzije 2.6.46 ili noviju, kao i na Microsoft Windows-u.

Doker endžin se izvršava na sloju iznad host operativnog sistema i njegova ključna uloga je da reši problem pakovanja raznih alata i biblioteka u nezavisnu celinu koja se izvršava u izolovanom aplikacijskom domenu. Za razliku od virtualnih mašina, koje su izuzetno glomazne, Docker fajl sadrži običan tekst fajl sa izlistanom konfiguracijom čime značajno smanjuje potrebu za resursima. Ovaj pristup naročito je pogodan za razvoj softvera jer se ovakvi fajlovi lako ubacuju u sisteme za kontrolu koda.

10.3. Rokat kontejneri

Rokat dolazi od firme koja je napravila CoreOS i predstavlja alternativu Docker kontejnerima. Isto kao i Docker, predstavlja izvršno okruženje za aplikacije sadržeći sve relevantne podsisteme potrebne za izvršavanje celine. CoreOS je minimalna verzija Linux baziranog operativnog sistema namenjenog izgradnji skalabilnih sistema velikih razmera.

Specifičnost Rokat rešenja se ogleda u drugačijem pristupu organizacije procesa [75] [74], tako što ih grupiše pod istim glavnim brojem procesa. Primera radi, ako imamo proces čiji je broj procesa 400, i ako dalje izvršavanje neke zavisne podkomponente zahteva otvaranje zavisnog procesa, novi proces će biti kreiran sa brojem 40x, sekvencijalno, počevši od 1.

Takođe, arhitekturno, Rokat sadrži u okviru kontejnera takozvani root kontejner i podkontejnere što omogućava restartovanje procesa u slučaju potrebe.

11. Sofverski definisane mreže (SDM) kao osnova cloud Sistema

Računarske mreže su bitan i nezaobilazan deo svakog cloud sistema. Iako već duže vreme u istoriji računarskih mreža postoji inicijativa da se mreže učine upravljivijim, programabilnijim, lakšim za proširenje i manje zavisnim od proizvođača i konkretnog hardvera, tek nedavno se pojavljuju prva rešenja koja industrija počinje da prihvata. Razlog za to, pre svega je, što je dosta pristupa podrazumevalo novu mrežnu arhitekturu bez mogućnosti bezbolnog prelaska ili uz određena ograničenja. U prethodnim poglavljima smo diskutovali o mrežnoj virtualizaciji i ostalim vidovima virtualizacije. U ovoj sekciji ćemo se baviti evolucijom programabilnih računarskih mreža koje danas poznajemo pod izrazom softverski definisane mreže. Ostale slične i prateće koncepte poput virtualizacije mrežnih funkcija ćemo pominjati u smislu razgraničenja i pojašnjenja značenja teme koju obrađujemo u ovoj sekciji.

Računarske mreže sastoje se od mrežnih elemenata koji obavljaju različite funkcije. Primeri nekih od tih funkcija su funkcija sviča, rutera, fajervola, itd. Po funkcionalnosti, računarske mreže delimo na dve ravni: ravan upravljanja i ravan (prosleđivanja) podataka. Kontrolna ravan ima ulogu odlučivanja u pogledu saobraćaja i smatra se *mozgom* mreže, s obzirom na to da je tu smeštena logika. Ravan podataka ima izvršnu funkciju prosleđivanja saobraćaja na osnovu instrukcija dobijenih od kontrolne ravni [76].

U tradicionalnim računarskim mrežama, ravan kontrole i ravan podataka čine nerazdvojivu celinu i obično se nalaze u mrežnom elementu nekog proizvođača. Mrežni elementi su nezavisni, distribuirani i konvergiraju ka određenom stanju, u zavisnosti od informacija koje razmenjuju. U SDM arhitekturi, ravan kontrole i ravan podataka su razdvojene i postoji centralni uvid u mrežne resurse. U tradicionalnim mrežama, mrežni elementi imaju u sebe ugrađene mnoge mrežne protokole, koji su često zatvorena i patentirana kompanijska rešenja. Neki od ključnih izazova u tradicionalnim računarskim mrežama su:

- (1) Značajan broj korisnika ima potrebu za manjim brojem mogućnosti koje su dostupne na uređajima,
- (2) Ovako vertikalno integrisane opcije i funkcije čine da mrežni uređaji imaju visoku cenu,
- (3) Put za bilo kakvu promenu protokola je dugotrajan, često meren godinama, na ovaj način čineći inovaciju veoma sporom,
- (4) Eksperimentisanje sa novim protokolima je teško ili čak nemoguće,

- (5) Svaka promena postojećih funkcija u sistemu mora da prođe kroz proceduru prihvatanja specifičnog proizvođača. On može da bude neprihvaćen, da implementacija traje (znatno) duže od očekivane ili željene, da ne odgovori na potrebe zahteva u potpunosti, da ne odgovori kvalitetom ili nekim drugim željenim kriterijumom,
- (6) Konfiguracioni interfejsi mogu i obično variraju u zavisnosti od proizvođača a često i proizvoda
- (7) Upravljanje ne-programabilnim mrežama je decentralizovano, stvarajući značajne izazove u automatizaciji, čineći operativne troškove visokim i sam proces podložnim greškama
- (8) Budući da je Internet kritična infrastruktura sveta, u čiji je razvoj uloženo mnogo sredstava i opreme u velikim razmerama, otežani su njegov razvoj i evolucija. Ključni razlog za sporu evoluciju je potreba za usaglašavanjem između raznih organizacija, poput standardizacionih tela. Ova pojava se naziva okoštavanjem Interneta [2].

Kao odgovor na gore pomenute izazove, pojavila se ideja programabilnih mreža. Koncept softverski definisanih mreža (SDM) pojavio se sa željom da se inovacija u oblasti računarskih mreža odvija većom brzinom, uz isto iskustvo razvoja kao i u ostatku softverske industrije. Svakako, koncept SDM-a, nije potpuno nov već je evoluirao kroz veliki broj projekata i više različitih pravaca. Na kraju, sinergijski efekat više različitih pravaca omogućio je konvergiranje ka softverski definisanim mrežama. Tako, u nastavku razmatramo pristup aktivnih mreža, koje su donele programabilne mrežne interfejse i pravac otvorenih signala (eng. *open signaling*) koji je doneo odvajanje kontrolne ravni od ravni prosleđivanja podataka.

U prvom delu ćemo govoriti o ranim programabilnim mrežama i objasniti neke ključne primere koji su obeležili razvoj ove oblasti podeljene u podoblasti kao što su: programabilna ravan podataka, razdvajanje ravni kontrole od ravni podataka, odnosno hardvera, i na kraju, mrežna virtualizacija.

U drugom delu diskutujemo terminologiju i definiciju softverski definisanih mreža. Razmatramo šta je to SDM i kako se razlikuje ili dopunjuje u odnosu na koncepte poput virtualizacije mrežnih funkcija (*Network Function Virtualization*, NFV) i mrežna virtualizacija (*Network Virtualization*, NV).

Iza toga razmatramo pregled rešenja u kategorijama: programabilnost ravni podataka, razdvajanje kontrolne ravni od ravni podataka, mrežna virtualizacija (mrežni hipervizori) i završavamo sekciju sa SDM kontrolerima i mrežnim operativnim sistemima.

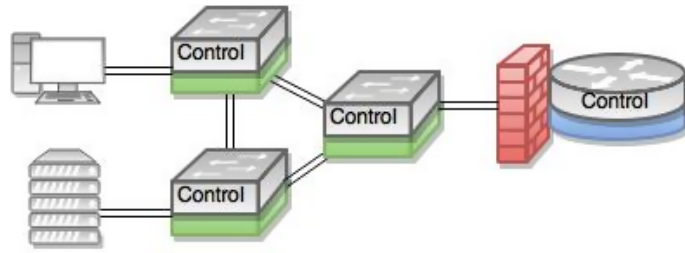
U poslednjem delu diskutujemo potencijalne poslovne i tehnološke dobrobiti, uticaj i promene koje mogu ili se već dešavaju sa uplivom raznih SDM rešenja.

11.1. Rane programabilne mreže

Početakom 1990-ih Internet je doživeo veliki rast potreba za mrežnim resursima. Ovaj rast je inicirao značajan broj istraživanja kako bi se mrežni resursi unapredili u svakom aspektu. Najveći napor bio je u oblasti i načinu na koji se mreže grade i kako se njima upravlja. U to vreme, svaka promena je morala da prođe proces standardizacije kroz zvanične organizacije, što je činilo ovaj proces veoma sporim, u proseku 5 do 10 godina. Kao što vidimo, proces inovacije bio je dosta spor, što je bila motivacija da se krene sa inovacijama u oblasti računarskih mreža. Počecemo objašnjenjem kako tradicionalne mreže funkcionišu.

Na fizičkom nivou, računarske mreže se sastoje od različitih tipova mrežnih uređaja ili elemenata koji mogu biti fizički i virtualni. Bez obzira na to, ovakvi uređaji obavljaju istu funkciju. Tipični primeri mrežnih uređaja su svičevi, ruteri i midlboks sistemi. Midlebox sistem može biti bilo koji posredni uređaj koji obavlja funkciju drugu u odnosu na uobičajenu funkciju IP rutera na datagram putu između izvornog i krajnjeg hosta [77]. Primeri midlboks sistema su: fajervol, bezbednosni sistemi za otkrivanje upada, load balanseri, adresni prevodioci, WAN optimizatori, itd.

Logički, računarske mreže se sastoje od kontrolne ravni koja donosi odluke vezane za saobraćaj i ravni podataka koja je zadužena za prosleđivanje podataka na osnovu ovih odluka. U tradicionalnim računarskim mrežama ove dve ravni su vertikalno integrisane (slika 30). SDM, ali i neke ranije arhitekture i fizički razdvajaju ove dve logičke ravni. U jednom ovako vertikalno integrisanom okruženju, svaki protokol ili značajnije unapređenje funkcija, zahtevaju proces standardizacije kroz zvanične organizacije, efektivno čineći ceo proces sporim i otežavajućim u pogledu inovacija.



Slika 30 - Tradicionalna, vertikalno integrisana mrežna arhitektura

Tradicionalne računarske mreže su distribuirane po svojoj prirodi. Njihovi nezavisni mrežni elementi vrše razmenu informacija promenama mrežnih stanja na osnovu kojih konvergiraju u određena stanja. Prisutan je nedostatak centralne topologije i mogućnosti centralizovanog upravljanja mrežom.

Ovi izazovi su postali motivacija za pojavu programabilnog mrežnog pristupa, ali pre svega, prvi i najvažniji poriv bio je da se mreže učine proširivijim, po pitanju svojih funkcionalnosti i programabilnim a u smislu upravljanja.

U ovoj sekciji diskutujemo o ranim programabilnim računarskim mrežama, pre pojave SDM tehnologija. Neki od radova postali su inspiracija za SDM dok su neki ostali pokušaji koji nisu imali veću praktičnu primenu. Ipak, interesantno je videti evoluciju misli u ovoj oblasti.

Programabilne mreže imale su dva glavna pravca: aktivne mreže i pristup otvorenih signala [78]. Pristup aktivnih mreža se zasnivao na programabilnosti ravni prosleđivanja podataka, dok je pristup otvorenih mrežnih signala predlagao odvajanje kontrolne ravni od ravni podataka. Takođe, mrežna virtualizacija je igrala (i još uvek igra) važnu ulogu u vršenju mrežnih eksperimenata. Njena ključna uloga i doprinos u SDM domenu su SDM hipervizori. Proći ćemo kroz ove pristupe i diskutovati ključna rešenja iz svake od oblasti.

Evo kako definišemo programabilne računarske mreže: programabilne mrežne arhitekture su realizovane kroz skup mrežnih algoritama koji uzimaju u obzir mrežno stanje i odgovaraju vremenskim skalama po kojima ovi algoritmi operišu [78].

U pregledu programabilnih mreža iz 1999. godine, budući razvoj predviđen je u tri pravca:

- (1) Okruženja za izvršavanje aktivnih mreža,
- (2) Mrežni kerneli sa otvorenim signalima,
- (3) Node operativni sistemi, ili operativni sistemi na čvorovima.

Razmatramo prva dva jer su od ključnog značaja za razvoj oblasti. Iako je bilo pokušaja za otvaranjem, generalizacijom i korišćenjem operativnih sistema sa čvorovima, ovaj pristup, nije doživeo značajniji razvoj. Klasifikaciju istorije programabilnih mreža pre SDM-a vršimo u sledećim kategorijama i sa sledećim primerima:

1. **Programabilnost ravni podataka:** Dodatak programabilnosti u tradicionalnim računarskim mrežama i aktivne mreže:
 - a. Xbind,
 - b. ANTS,
 - c. SwitchWare,
 - d. NetScript.
2. **Razdvajanje ravni kontrole od ravni podataka:** Razdvajanje dve ravni kontrole od ravni podataka i pristup otvorenih signala:
 - a. GSMP,
 - b. Tempest ,
 - c. ForCES,
 - d. RCP,
 - e. SoftRouter.
3. **Mrežna virtualizacija:** Apstrakcija mrežnih resursa kroz dodatak sloja mrežne virtualizacije i mrežnog overleja (eng. *network overlay*):
 - a. Tempest,
 - b. MBone,
 - c. RON,
 - d. GENI,
 - e. VINI.

11.1.1. Programabilnost ravni podataka

Programabilne mreže pružaju mrežne usluge putem programabilnog interfejsa. Programabilna mreža je mreža koja ima resurse i funkcionalnosti izložene kroz aplikativni programski interfejs (API). Ovakav API se može iskoristiti za pravljenje usluga višeg nivoa.

Pristup aktivnih mreža pojavio se sredinom devedesetih godina dvadesetog veka. Ovaj pristup predstavlja prvu apstrakciju mrežnih resursa dostupnih preko programabilnog interfejsa (API). Obzirom da je ovakav pristup prvi te vrste u mrežama, nazvan je pristupom iznova (eng. *clean slate*) u mrežnoj arhitekturi [79]. Aktivne mreže podrazumevale su dva modela:

- (1) Model kapsule - U ovom modelu, mrežni programi se ugrađuju u zaglavlje paketa podataka, zapravo ga zamenjuju i zatim se izvršavaju na svojim krajnjim destinacijama [4].
- (2) Model programabilnog rutera ili sviča - U ovom modelu, putanjom prosleđivanja podataka može biti upravljano putem posebnog kanalom. U ovom modelu korisnik može da programira mrežu pružajući program koji izvršava obradu u posebnom, *out of band*, kanalu, dok prenos podataka i dalje vrši u okviru primarnog kanala, *in band*.

Aktivna mreža omogućava ruteru da vrši obradu podataka sve do aplikacionog sloja OSI referentnog modela, dok u tradicionalnim mrežama, ruteri vrše obradu na trećem, odnosno mrežnom sloju [80]. U aktivnim mrežama, umesto kontrolne ravni, prilagođavanje mrežnih usluga odvija se na nivou transporta. Iako ovaj metod pruža visok stepen prilagođavanja usluga, stepen složenosti je visok. Pristup aktivnih mreža imao je dosta nerešenih aspekata, pre svega u domenu performansi i bezbednosti [5], što je sprečilo šire prihvatanje. Pogledajmo u nastavku neka od bitnijih rešenja.

Xbind (1996): Model vezivanja (od engleske reči *bind*) predstavlja konceptualni frejmwork za kreiranje, deployment i upravljanje multimedijalnim uslugama na ATM mrežama. Model vezivanja je omogućio vezivanje mrežnih resursa na distribuiran način, podržavajući različita mrežna okruženja i protokole [81].

Arhitektura je sadržala dve ključne komponente:

- (1) Bazu interfejsa za vezivanje (kolekciju interfejsa) i
- (2) Algoritme koji se izvršavaju na tim vezujućim interfejsima, u isto vreme pružajući garancije kvaliteta usluge (quality of service, QoS).

Konkretna implementacija vezujuće arhitekture, xbind, je skup alata srednjeg sloja (eng. *middleware*), sposoban za komunikaciju preko velikog broja različitih arhitektura i operativnih sistema.

Glavni doprinos vezujućeg modela je frejmwork za:

- (1) Upravljanje stanjima,
- (2) Apstrakcijama stanja koje ovaj srednji sloj pruža i
- (3) Generalizovanu arhitekturu servisa.

Active Nodes Transfer System, ANTS (1998): ANTS je razvijen na univerzitetu MIT i predstavljao je skup alata baziranih na aktivnim mrežama za kreiranje i dinamički deployment mrežnih protokola. U ovom pristupu, mreža postaje programski sistem na kome se operacije izvršavaju na mrežnim čvorovima.

Ciljevi dizajna ANTS arhitekture bili su:

- (1) Podrška za paralelno izvršavanje raznih mrežnih protokola i servisa,
- (2) Omogućavanje uvođenja novog protokola kroz dogovor između servisa, naspram registracije protokola kroz centralizovan katalog, što je za cilj imalo mogućnost integrisanja raznih rešenja razvijenih od različitih proizvođača i
- (3) Dinamički deployment servisa, obzirom da se očekivalo da servis bude stalno dostupan, čak i za vreme trajanja ažuriranja servisa.

U ANTS arhitekturi:

- (1) Kapsule zamenjuju pakete i postaju odgovorne za obradu paketa,
- (2) Aktivni čvorovi ili nodovi, zamenjuju rutere i krajnje čvorove. Ovi čvorovi su izvršavali upravljanje stanjem kapsula i izvršavanje programskih rutina koje su se nalazile u kapsulama,
- (3) Distributivni mehanizam je bio zadužen za isporuku rutina, kad i gde su potrebne.

Sa svim gore nabrojanim, ovaj skup alata je postao samoregulišući mehanizam za deljene informacije u okviru protokola, baziran na kapsulama. ANTS je deo svoje bezbednosti bazirao, odnosno nasleđivao od programskih jezika koji su imali jasno i čvrsto definisane tipove promenljivih [4].

SwitchWare (1998): Ovaj rad predstavila je grupa autora sa Univerziteta u Pensilvaniji [5]. Motivacija je bila nedostatak mogućnosti da se doda novi ili modifikuje postojeći protokol. SwitchWare je adresirao ovu potrebu dodajući aktivne pakete koji su postali mobilni kod koji se izvršavao na mrežnim čvorovima. Ovo, ne samo da je omogućilo inovacije u mrežama, već je povećalo i performanse.

SwitchWare aktivna mrežna arhitektura se sastojala od:

- (1) Aktivnih paketa koji su sadržali programski kod koji se izvršavao na čvorovima,
- (2) Aktivnih ekstenzija, svičleta (eng. *switchlets*) koji su obezbeđivali servise i
- (3) Infrastrukture aktivnih rutera koji su obezbeđivali ključne bezbednosne servise.

Za razliku od aktivnih paketa koji su se kretali svuda kroz mrežu, aktivne ekstenzije su se izvršavale samo na određenim čvorovima i smatrani su dinamičkim funkcionalnim dodacima. Pod dinamičkim podrazumevamo mogućnost da se ekstenzija aktivira i deaktivira po potrebi, gde se na ovaj način optimizuje izvršno okruženje (eng. *run-time*). U SwitchWare-u, umesto mrežnih paketa korišćeni su aktivni paketi koji su sadržali:

- (1) Podatke, koji predstavljaju tzv. *payload*,
- (2) Mobilne programe, koji predstavljaju kontrolnu logiku tradicionalnog zaglavlja paketa uz mogućnost dodatnog prilagođavanja.

Bezbednosni model korišćen u SwitchWare arhitekturi koristio je za uspostavljanje relacija kriptografiju baziranu na poverenju između krajnjih entiteta. On se oslanjao, kao i ANTS na prednosti programskog jezika sa čvrsto definisanim tipom promenljivih koji se zvao programski jezik za aktivne mreže (eng. *Programming Language for Active Networks, PLAN*). PLAN jezik je bio posebno namenjen za SwitchWare arhitekturu, međutim, bilo je moguće koristiti i druge generalne programske jezike poput Caml-a.

NetScript project (2001): NetScript je sistem jezički sistem za dinamičko programiranje i deployment protokol softvera u aktivnim mrežama [6]. Ovaj koncept je takođe stvorio softversku apstrakciju mrežnih resursa. NetScript programi sadrže primitivne funkcije koje se primenjuju na pakete. Primeri su: rutiranje, inspekcija paketa, multipleksiranje i demultipleksiranje, upravljanje redovima, itd. Ove primitivne funkcije formiraju graf međusobno povezanih uređaja koji procesiraju pakete koji teku kroz ovaj graf.

NetScript paketi se razlikuju od drugih rešenja iz domena aktivnih mreža po tome što ih odabiraju za procesiranje aktivni mrežni čvorovi. Ovaj operativni model omogućava implementaciju kako postojećih, tako i budućih, novih protokola.

Neke od kategorija mrežnih sistema koje je moguće implementirati u okviru NetScrip-a su: ruteri, fajervoli, analizatori protokola, load balanseri, sistemi za detekciju upada u mreže, itd.

11.1.2. Razdvajanje ravni kontrole od ravni podataka

Razdvajanje ravni kontrole od ravni prosleđivanja podataka je počelo 1995. godine, serijom radionica i pokretom koji se zvao otvoreni (mrežni) signali (eng. *Open Signaling*). Grupa koja je ovo pokrenula to se zvala *Open Signaling Working Group* (OPENSIG). Ova radna grupa je postavila za cilj da *ATM, Internet i mobilne mreže učini otvorenim, proširivim i programabilnim* [82]. Grupa je istraživala prethodni rad na tu temu iz 60-tih i došla do zaključka da model integrisanog hardvera i softvera postavlja izvesna ograničenja na rast postojećih i dizajn novih mrežnih servisa. Ideja je bila da se mrežni resurs učini programabilnim, proširivim i fleksibilnim, na način sličan računarskim resursima. Ovo je postignuto kroz fizičko razdvajanje ravni kontrole i ravni komunikacionog hardvera. Kontrolni softver je upravljao komunikacionim hardverom preko jasno definisanog programskog interfejsa (API-ja). Ovaj pristup je bio dizajniran da omogući deployment većih razmera za nove i postojeće servise na distribuiran način. OPENSIG grupa je verovala da otvoreni i programabilni interfejsi ka komunikacionom hardveru mogu da obezbede mogućnost za razvoj novih aplikacija.

Ove radionice su se organizovale dva puta godišnje, od 1995. do 1997. godine, i okupljale su učesnike iz industrije i akademije. Mnogi članovi OPENSIG-a su osnovali nove firme koje su napravile značajnu količinu, takođe značajnog softvera otvorenog izvornog koda. U isto vreme, odigravao se paralelno forum koji se zvao *Multi Switching Forum* koji je imao za cilj da podrži višestruke kontrolne mehanizme. Na ovom forumu je između ostalih, prezentirana tehnologija GSMP koju objašnjavamo u daljem tekstu.

Jedna od zanimljivosti je da je uočena sličnost između OPENSIG pokreta i *Open Networking Foundation* (ONF) [83], osnovane u Martu 2011. godine [84]. Slično OPENSIG-u, ONF je ima za cilj da promoviše usvajanje SDM tehnologija kroz razvoj i podršku otvorenih standarda.

Rešenja

GSMP (1996): Član *Internet Engineering Task Force* (IETF) i bivši član OPENSIG-a, Peter Newman podneo je predlog za standardizaciju generalnog protokola za kontrolu i upravljanje ATM svičem, (*General Switch Management and Control Protocol, GSMP*). GSMP je unapređivan do verzije 3 koja je obuhvaćena predlogom dokumenta RFC3292.

GSMP je omogućavao kontroleru da uspostavi i raskine konekcije, doda i obriše listove na “jedan ka više” konekcijama, upravlja portovima sviča, zahteva informaciju o konfiguraciji i zahteva statistiku [85]. U svojoj poslednjoj verziji 3, funkcionalnost je dozvoljavala da više eksternih svič kontrolera kontroliše stanje sviča, ATM-a, frejm relaja ili MPLS-a, kako *unicast* tako i *multicast* [7].

Tempest (1998): Tempest frejmvork [8] napravljen je na Univerzitetu Kembridž. Motivacija za Tempest istraživanje bila je mogućnost da se uradi deployment različitih kontrolnih arhitektura na istu fizičku mrežu. Prezentiran je kao frejmvork koji je omogućio fleksibilne i programabilne mrežne funkcije na nivou kontrolne ravni.

U Tempest mreži više različitih kontrolera istovremeno postoje i kontrolišu virtualne delove fizičkog sviča, tretirajući ih kao logičke celine. Ovi logički kontroleri su nazivani svičletima (eng. *switchlets*). Skup logički povezanih svičleta čini virtuelnu mrežu.

Tempest je omogućio definiciju kontrolnih polisa na nivou: (1) paketa, (2) konekcije i (3) pojedinačnom nivou virtuelne mreže. U ovom frejmvorku, dešava se promena od generalne arhitekture, koja pokušava da zadovolji sve kontrolne zahteve, ka arhitekturi okrenutom ka servisima, u kojoj je moguće adresirati i buduće potrebe. Frejmvork takođe omogućava prilagođavanja, pritom zadržavajući interoperabilnost sa drugim operaterima i dinamičko kreiranje kontrolnih arhitektura čije trajanje odgovara trajanju potrebe za datim servisom. Na kraju, obzirom da postoji koncept upravljanja konekcijama, modifikacija servisa je moguća na aplikativnom nivou.

Forwarding and Control Element Separation, ForCES frejmvork (2004): Mrežna radna grupa IETF-a je izašla sa predlogom za ForCES mrežne elemente, opisane u dokumentu RFC 3746 [9].

ForCES koncept prepoznaje monolitne, fizički vertikalno integrisane mrežne arhitekture u mrežnim elementima, kao i njihove podelu na logičke delove za kontrolu i prosleđivanje podataka. Kontrolna ravan je obično implementirana bazirana na arhitekturi procesora opšte namene, dok je ravan prosleđivanja podataka bazirana na arhitekturi mrežnih procesora. ForCES je definisao frejmvork i standardizovao komunikaciju između kontrolne ravni i ravni podataka. Ovo je omogućilo sledeće:

- (1) Fizičku separaciju logičkih celina ravni kontrole i ravni podataka,
- (2) Proizvođačku specijalizaciju,
- (3) Proizvođačku međuoperativnost i sistem integraciju (uz mogućnost povećanog izbora i fleksibilnosti u dizajnu mrežnih arhitektura),
- (4) Znatno bržu inovaciju i
- (5) Skalabilnost.

Uz implementaciju ForCES-a na svim mrežnim blejdovima (eng. *network blade*), bilo je moguće koristiti kontrolnu ravan jednog proizvođača i ravan prosleđivanja podataka drugog proizvođača u istoj šasiji.

Routing Control Platform, RCP (2005): RCP je logički centralizovana kontrolna platforma koja vrši selekciju putanje u ime rutera i šalje im odabrane rute koristeći standardni iBGP protokol (eng. *Interior Border Gateway Protocol*) [10].

U okviru istog mrežnog administrativnog domena, koji nazivamo još i autonomnim sistemom (AS), ruteri razmenjuju informacije vezane za rute ka eksternim destinacijama. U ovakvim okruženjima, za iBGP implementacije je poznato da imaju različite izazove, kao što su: skalabilnost mrežnih (mesh) topologija, refleksija putanja (eng. *route reflections*), itd. RCP je dizajniran sa idejom da adresira ove izazove i predložena je platforma sa funkcijom da obezbedi sakupljanje eksternih ruta, centralizovano čuvanje topologije i odabir BGP putanja u ime rutera a u okviru istih autonomnih sistema.

Doprinos RCP platforme je smanjeno vreme i overhead konvergencije bez izmena na postojećoj bazi instaliranih rutera.

SoftRouter arhitektura (2004): Još jedan primer separacije ravni kontrole i podataka je SoftRouter arhitektura [11]. Ovaj koncept je vrlo sličan prethodno diskutovanom ForCES-u. Kontrolna ravan je implementirana na serverima opšte namene i naziva se kontrolnim elementom, koji je fizički odvojen od hardvera, odnosno elemenata za prosleđivanje podataka. Ova dva elementa međusobno dinamički stupaju u interakciju. Svaki element ravni prosleđivanja podataka, nakon inicijalizacije, vrši otkrivanje raspoloživih kontrolnih elemenata i uspostavlja konekciju sa jednim od njih kroz jasno definisan interfejs, na taj način formirajući logički ruter.

Ključni doprinos SoftRouter arhitekture su unapređena:

- (1) Pouzdanost,
- (2) Skalabilnost,
- (3) Bezbednost,
- (4) Jednostavnost dodavanja novog servisa ili modifikacija postojećeg,
- (5) Smanjenje cene.

11.1.3. Mrežna Virtualizacija

Iako smo ranije u radu govorili o mrežnoj virtualizaciju, u nastavku obrađujemo mrežnu virtualizaciju u kratkom evolutivnom kontekstu koji je prethodio programabilnim mrežama.

Kao i u svakom drugom aspektu računarskih nauka, motivacija za virtualizaciju je:

- (1) Povećana i efikasnija upotreba hardverskih resursa koje je moguće *deliti*,
- (2) *Izolacija*, pre svega u funkciji bezbednosti, privatnosti i upravljanja,
- (3) *Agregacija* fizičkih resursa radi kreiranja virtualnih, većih od pojedinačnih fizičkih,
- (4) *Dinamička alokacija* resursa koje je moguće dobiti po potrebi,
- (5) *Pojednostavljeno upravljanje* kroz uopštene programske apstrakcije.

Virtualizacija u oblasti mreža postojala je kao koncept i pre programabilnih mreža. U telekomunikacijama, postojao je X.25 protokol koji je omogućavao deljenje fizičkog voda na logičke komunikacione kanale. Nakon toga, pojavio se koncept virtualnih lokalnih računarskih mreža (eng. *Virtual Local Area Network*, *VLAN*) koji je omogućavao punu logičku separaciju različitih računarskih mreža, ekvivalentnu fizičkoj.

VLAN koncept je osmišljen kako bi kroz istu fizičku opremu moglo da koegzistira nekoliko logičkih mreža te da nije neophodno duplirati opremu. Pripadnost određenom VLAN-u se definiše kroz asocijaciju porta na mrežnom uređaju a VLAN 0 se obično koristi za upravljanje. Zatim, tu su virtualne privatne mreže (eng. *virtual private networks, VPN*). Ova tehnologija se pojavila kao odgovor na potrebu da se saobraćaj koji putuje preko javne infrastrukture, poput Interneta, sačuva privatnim. VPN omogućava kreiranje *point-to-point* konekcije bilo kroz uspostavljanje tunela, bilo kroz korišćenje enkripcije nad saobraćajem.

Kasnije, mrežna virtualizacija je procvetala u nekoliko kategorija: virtualizacija mrežnih karti (eng. *network interface cards, NIC*), u vidu vNIC i pNIC rešenja, virtualizaciju svičeva (vSwitch), i virtualizaciju geografski distribuiranih lokalnih računarskih mreža sa rešenjima poput VXLAN, NVGRE, STT. Ovakva rešenja uspostavljaju tunele na transportnom nivou i enkapsuliraju L3 saobraćaj.

U nastavku se fokusiramo na mrežnu virtualizaciju i rešenja iz programabilnih mreža.

Rešenja

Multicast Backbone, MBone (1994): MBone je rani primer overlej mreže preko Internet infrastrukture. Razvijen 1992. godine, MBone je bio pokušaj da se multicast putem povežu privatne mreže u istraživačke svrhe. Smatran je prvim primerom virtualne mreže jer deli isti fizički medij sa Internetom [12].

U ovom istraživačkom projektu, modifikovani čvorovi koriste prilagođeni protokol kontrolne ravni i vrše enkapsulaciju na komunikacionim krajevima. Koncept koristi podskup multikast adresnog prostora u okviru adresnog prostora klase D. Modifikovani čvorovi, zapravo računari sa softverom, služe kao MBone ruteri koji su povezani kroz IP enkapsulirane tunele, formirajući mrežnu kičmu mornaričke škole. Overlej je zapravo omogućavao ono što danas poznajemo pod multicast tehnologijom, preko Interneta, čiji čvorovi nisu nudili ovaj vid komunikacije. Danas, multikast je podržan i široko dostupan od strane više proizvođača.

MBone i još nekoliko drugih primera koji su sledili bili su motivacija za ono što su kasnije postale overlej mreže, što diskutujemo dalje u radu.

Tempest (1998): Tempest projekat koji smo prethodno diskutovali je primer, kako za mrežnu virtualizaciju, tako i za razdvajanje kontrolne ravni od ravni prosleđivanja podataka.

Resilient Overlay Networks, RON (2001): RON koncept [86] je bio Internet overlej za *end-to-end* komunikacione sisteme. Obezbeđivao je redundansu fizičke putanje korišćenjem mehanizma za otkrivanje i oporavljanje prekida rada na dostupnim putanjama. Predstavljao je mehanizam rutiranja koji aktivno nadzire stanje i kvalitet svih putanja, i vrši korektivne akcije u slučajevima opadanja kvaliteta usluge ili nedostupnosti putanje. Radi skaliranja, RON čvorove je trebalo postaviti kroz različite autonomne sisteme.

Zbog kompromisa sumarijacija putanja, koji je neophodan zarad skaliranja, protokoli za rutiranje (*routing*, ne *routed*) nisu uvek svesni svih dostupnih putanja. Obzirom da vrši održavanje spiska svih raspoloživih putanja u okviru autonomnog sistema, RON čvor može brzo da ponudi alternativu. Ova informacija je čuvana u centralnoj bazi podataka. RON overlej daje bolje rezultate u odnosu na tipičan proces konvergencije BGP mehanizma za oporavak u slučaju gubitka dostupnosti putanje. RON čvorovi formiraju meš mrežnu strukturu, stvarajući broj virtualnih linkova po formuli $n(n-1)$. Selekcija putanja u RON mreži se obavlja na osnovu metrika kao što su: protok, kašnjenje, procenat gubitka paketa, itd.

Ključni doprinosi RON-a:

- (1) Detekcija grešaka,
- (2) Povećana pouzdanost,
- (3) Pобољšanje performansi (procenat gubitka, kašnjenje i protok),
- (4) Mogućnost aplikativno-svesnog rutiranja (eng. *application aware routing*).

Global Environment for Networking Innovations, GENI (2006): Još jedan primer projekta iz kategorije pristupa iznova (*clean slate design*) je GENI [14]. Početak ovog projekta podržala je Američka naučna nacionalna fondacija (eng. *National Science Foundation, NSF*) sa ciljem da pomogne eksperimentalno istraživanje u oblasti mreža i kreiranje konkurentnih servisa. Projekat i danas privlači istraživače. GENI je deljeno test okruženje koje omogućava privatne, izolovane, delove virtualnih mreža i distribuiranih sistemskih resursa. Omogućava:

- (1) Dobavljanje,
- (2) Povezivanje na nivou L2,
- (3) Puno prilagođavanje čvorova nivoa L3 ili više,
- (4) Kontrolu protoka saobraćaja i
- (5) Deployment bilo kakvog prilagođenog protokola nivoa L3.

Ima puno načina da se GENI resursi iskoriste i sa njima radi. Mi ističemo aspekt programabilnosti L2 nivoa u službi isporuke bilo kakvog L3 overlej protokola. Svaka virtuelna kriška je potpuno izolovana i operiše bez uticaja od strane drugih kriški u okviru GENI mrežnih resursa. GENI takođe ima nešto što se zovu mrežni agregati. To su, bilo mrežni, bilo računarski resursi, koje obezbeđuju razne organizacije. Agregati, odnosno skupine resursa, su izloženi preko programabilnih interfejsa koji se zove *Aggregate Manager API*. Funkcije koje su dostupne preko ovog programskog interfejsa su:

- (1) Listanje,
- (2) Upit statusa,
- (3) Dobavljanje i
- (4) Uklanjanje resursa iz privatne kriške.

Sa uvođenjem GENI koncepta, deo istraživačke javnosti verovao je da mrežna virtualizacija predstavlja put ka Internetu budućnosti jer obezbeđuje okruženje u kome više mrežnih protokola može da koegzistira a da svakim od njih upravlja druga organizacija [15].

Virtual Network Infrastructure, VINI (2006): VINI bio je još jedno protokol test okruženje. Ovo test okruženje je bilo korisno u svrhe dizajna, testiranja performansi, merenja i vršenja eksperimenata. Takođe je koristio deljene hardverske resurse, obezbeđujući realističnost i visok stepen kontrolisanosti mrežnog eksperimentisanja na nivou resursa korišćenih u okviru virtualne mrežne kriške.

VINI dizajn ciljevi bili su:

- (1) Mogućnost izvršavanja pravog softvera za rutiranje,
- (2) Okruženje sa realnim mrežnim okruženjem,
- (3) Kontrolisani mrežni događaji,
- (4) Prenos pravog saobraćaja [15].

Najznačajniji doprinos VINI okruženja napravljen je na bazi činjenice da je vrlo teško bilo sprovesti eksperiment u razmerama realnog produkcionog okruženja a neretko i nemoguće. Naročito, ako se ticalo puštanja u rad novih protokola. Pre VINI, jedina alternativa su bili simulatori ili mnogo manja produkciona mreža, koja je zahtevala pregovaranje sa proizvođačima kako bi se izvršilo puštanje eksperimentalnih protokola.

11.2. Softverski definisane mreže – funkcije i arhitektura

11.2.1. Ključna SDM terminologija

Mrežna kriška (eng. *network slice*): Potpuno izolovan virtualni mrežni overlej, kriška fizičke mreže. Predstavlja podskup hardvera i zaglavlja paketa kroz L2 mrežne elemente bez menjanja topologije ispod.

Mrežna virtualizacija, mrežni hipervizor (eng. *network virtualization, network hypervisor*): Apstrakcija mreže, odvojena od hardvera ispod, koja omogućava da se kreira nova topologija kao overlej na postojeću topologiju.

SDM kontroler ili mrežni operativni sistem (eng. *SDN controller or network operating system - NOS*): Ključna komponenta u SDM mrežnoj arhitekturi. Obavlja funkciju centralizovane kontrole i donosi odluke vezane za mrežni saobraćaj.

Uređaj za prosleđivanje (eng. *forwarding device*): Hardver na L2 nivou, najčešće svič. U kontekstu SDM-a, uređaj koji upoređuje pakete i primenjuje neko od akcija kao što su prosledi, ispusti, prosledi kontroleru i prepishi zaglavlje. Ovakav uređaj radi na ravni podataka i komunicira sa kontrolerom kroz programabilne apstrakcije.

Tok ili mrežni tok (eng. *flow*): Struja mrežnih paketa između izvora i destinacije. Tok je definisan skupom pravila i akcija koje se primenjuju na ta pravila. Flow logika je zamenjuje logiku destinacije (kao što je IP adresa).

Ravan podataka (eng. *data plane*): Skup hardverski povezanih uređaja u logičku celinu. Još jedan način da se objasni je skup uređaja za prosleđivanje podataka koji čine logičku celinu.

Kontrolna ravan (eng. *control plane*): Kontrolna logika, obično SDM kontroler koji donosi odluke.

Ravan upravljanja (eng. *management plane*): SDM aplikacije koje uključuju implementacije mrežnih funkcionalnosti.

Severni interfejs ili severni API (eng. *northbound interface*): Interfejs između aplikacionog i kontrolnog sloja, tj. aplikacija i kontrolera. Omogućava apstrakciju za razvoj SDM aplikacija.

Južni interfejs ili južni API (eng. *southbound interface*): Interfejs između kontrolnog sloja i sloja za prosleđivanje podataka. Predstavlja apstrakciju logike prosleđivanja podataka.

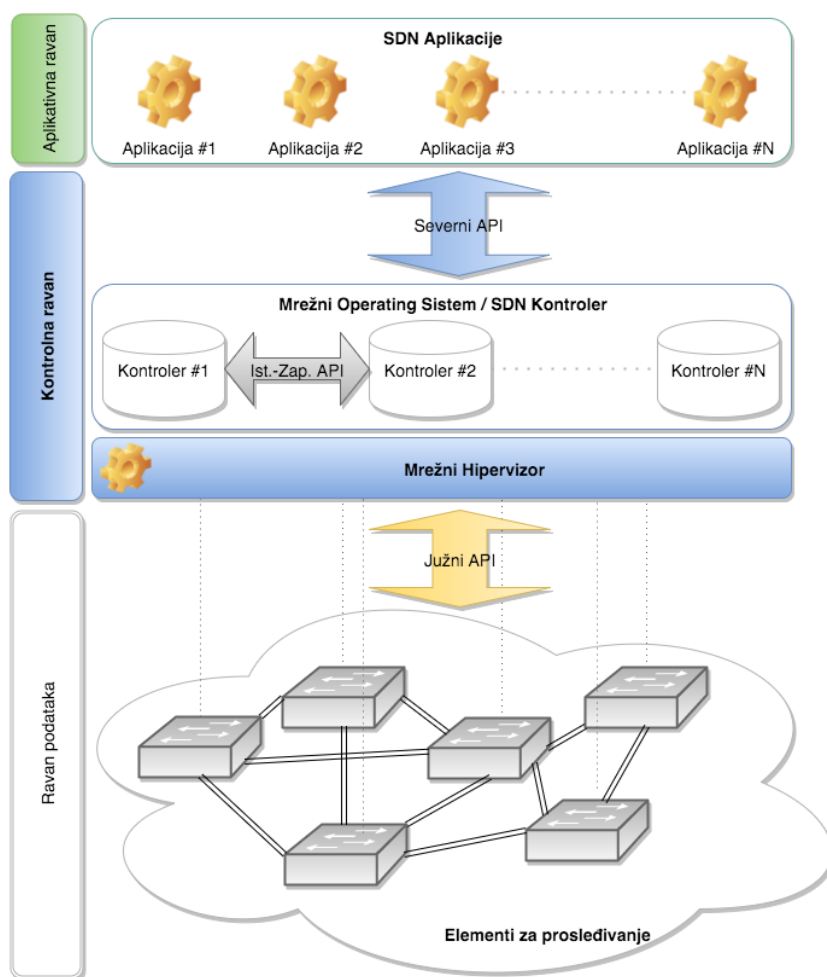
Istočno-zapadni interfejs, ili istočno-zapadni API (eng. *east-west interface*): Interfejs između SDM kontrolera. U slučaju kada arhitektura konkretnog rešenja podrazumeva postojanje više kontrolera, oni mogu ostvariti *peer-to-peer* komunikaciju gde svaki od kontrolera može da zatraži informaciju od ostalih.

11.2.2. Ukratko o softverski definisanim mrežama

Najkraća definicija SDM-a bila bi odvajanje kontrolne ravni i ravni prosleđivanja podataka. U tradicionalnim mrežama, ove dve ravni su vertikalno integrisane u uređaj poput rutera ili sviča i obično su proizvođačka rešenja zaštićena patentom. Šire, SDM je pristup računarskim mrežama koji podrazumeva sledeće:

- (1) Razdvajanje ravni prosleđivanja podataka od ravni kontrole,
- (2) Definiciju interakcija između gore navedenih ravni kroz jasno definisane apstrakcije (aplikativni programski interfejs, API),
- (3) Logički centralizovan menadžment,
- (4) Programabilnost,
- (5) Otvoreni standard.

Ravan podataka je kolekcija jednostavnih hardverskih elemenata za prosleđivanje podataka koja obavlja funkcije prosleđivanja na osnovu odluka ravni kontrole. Neke od ovih odluka su prosledi, ispusti, prosledi kontroleru i prepisi zaglavlje. Kontrolna ravan donosi odluke vezane za saobraćaj i daje nalog ravni prosleđivanja podataka (hardveru) kroz južni API interfejs, najčešće OpenFlow [16]. Kontrolna ravan ima pregled kompletne mrežne topologije u okviru svog domena, kao i kompletnu kontrolu nad višestrukim elementima ravni prosleđivanja podataka i njihovim stanjima. Ova kontrolna logika nalazi se u SDM kontroleru koji se naziva još i mrežnim operativnim sistemom. Zavisno od konkretnog dizajna arhitekture, može biti jedan ili više kontrolera. U slučaju više kontrolera, kontroleri mogu da uspostave *peer-to-peer* komunikaciju kroz istočno-zapadni API. Kontroler omogućava usluge SDM aplikacijama preko severnog API-ja. Slika 31 predstavlja jednu od mogućih arhitektura SDM.



Slika 31 - Moguća reprezentacija SDM arhitekture

Takođe, sve više se aktualizuje element za prosleđivanje podataka, odnosno svič u okviru SDM arhitekture, koji se naziva svič bela kutija (eng. *whitebox*) ili svič ogoljeni metal (eng. *bare-metal*) - koristićemo izraz u originalu *bare-metal*. To je zapravo hardver koji dolazi bez operativnog sistema (samo goli metal, otuda i izraz). Na ovakve svičeve mogu da se instaliraju različiti operativni sistemi, najčešće bazirani na Linux-u. Ovo su hardverski agnostični operativni sistemi. Kombinovan sa *bare-metal* svičevima, ovakav pristup omogućava fleksibilnost izbora i optimizaciju okruženja, i sprečava proizvođačko zaključavanje ili zaključavanje u okviru jednog rešenja. Neki od primera svičing operativnih sistema za *bare-metal* svičeve su: Cumulus Linux [26], XORPlus [87], PicOS [88], Switch Light [28] i FBOSS [89].

U nastavku definišemo virtualizaciju mrežne funkcije (eng. *network function virtualization, NFV*) i mrežnu virtualizaciju (eng. *network virtualization, NV*) i diskutujemo njihove relacije sa softverski definisanim

mrežama. Fokusiraćemo se na osnovne definicije ovih termina, jer je svaka tema opširna i zaslužuje poseban pregledni rad.

Funkcija mrežne virtualizacije

Virtualizacija mrežnih funkcija (eng. *network function virtualization, NFV*, i *virtual network function, VNF*) je još jedan često pominjan termin koji ponekad unosi zabunu u SDM kontekst. NFV se se fokusira na određenu funkciju i njeno iznošenje iz hardvera u virtualnu mašinu. Primeri mrežnih funkcija su: prevodioci adresa, fajervoli, sistemi za detekciju i prevenciju provala u računarskim mrežama, itd.

Mrežna virtualizacija

Mrežna virtualizacija je igrala važnu ulogu u razvoju SDM-a. Predstavlja abstrakciju hardvera koji se nalazi ispod. Omogućava paralelno postojanje virtualnih mreža na deljenim fizičkim resursima.

Diskutovali smo korene mrežne virtualizacije u kontekstu overlej mreža kao što su RON i MBone. Proširićemo ovo dalje kroz Open vSwitch, FlowVisor i NVP.

SDM, NFV i NV i odnos među njima

SDM: Odvaja kontrolnu od ravni prosleđivanja podataka, pružajući logički centralizovan menadžment, orkestraciju i automatizaciju mrežnih resursa.

NFV: Optimizuje mrežne servise, odvajajući mrežnu funkcionalnost iz hardverskog uređaja, prenoseći je u softver sa ciljem da se postigne bolja i brža inovacija i dobavljanje za date mrežne funkcionalnosti.

NV: Omogućava postojanje više različitih korisnika kroz podršku za koegzistenciju različitih mrežnih arhitektura. Mrežna virtualizacija takođe omogućava koegzistenciju mnogostrukih SDM kontrolera.

11.2.3. Programabilnost ravni podataka

Slično prethodno diskutovanim konceptima pre pojave SDM tehnologija, klasifikujemo istoriju SDM mreža u sledeće kategorije i prezentiramo njihove primere od značaja:

1. Programabilnost ravni podataka:
 - a. OpenFlow.
2. Razdvajanje ravni kontrole od ravni prosleđivanja podataka:
 - a. SANE,
 - b. ETHANE.
3. Mrežna virtualizacija (mrežni hipervizori):
 - a. Open vSwitch,
 - b. FlowVisor,
 - c. NVP.
4. SDM kontroleri:
 - a. NOX,
 - b. ONIX,
 - c. Yanc.
5. Mrežni operativni sistemi:
 - a. Cumulus Linux,
 - b. Open Network Linux,
 - c. Switch Light.

Rešenja

OpenFlow (2008): OpenFlow je napravljen na Stanford univerzitetu i inicijalno je implementiran u Univerzitetskom kampusu [16]. Smatra se prvim standardnim komunikacionim interfejsom između kontrolne ravni i ravni podataka. To je otvoren protokol za programiranje mrežnog toka (*flow*) u mrežnim uređajima i u isto vreme podržava izolaciju produkcionog od eksperimentalnog saobraćaja. Ova funkcionalnost postaje ključna za vršenje mrežnih eksperimenata. Postaje moguće testirati nove mrežne protokole (trećeg nivoa) koji nisu bazirani na IP protokolu. Ključne komponente OpenFlow sviča su:

- (1) Tabela mrežnog toka (*flow table*) - nad svakim tokom se primenjuje akcija tipa: prosledi, ispusti, prosledi na sve portove, prepisi zaglavlje, prosledi kontroleru,
- (2) Bezbedni kanal - služi za komunikaciju između sviča i kontrolera i
- (3) OpenFlow protokol koji omogućava komunikacioni standard između sviča i kontrolera.

Motivacija za OpenFlow je potekla sa idejom da se napravi kompatibilan prelaz između tehnologija u upotrebi i SDM koncepta kao što su to bili SANE i ETHANE. S obzirom da je protokol doneo standardizovnje API kontrolne ravni, mreže su postale spremne da budu više programabilne i više softverski definisane. Ono što je OpenFlow ponudio jeste da se postojeći hardver iskoristi uz unapređenje firmvera na uređajima, kako bi podržavali OpenFlow. Naravno, bilo je moguće napraviti nove i jednostavnije OpenFlow svičeve, te tako razlikujemo dve vrste svičeva:

- (1) Namenski (eng. *dedicated*) koji predstavlja jednostavni datapath element koji ne podržava L2 ili L3 procesiranje i
- (2) OpenFlow omogućen svič koji predstavlja bilo koji proizvođački uređaj kome je firmver unapređen tako da podržava ovaj protocol.

Dodavanje novog protokola u OpenFlow omogućenom mrežnom okruženju se postiže implementacijom u SDM kontroleru. Implementacioni koraci su sledeći:

- (1) Definisane novog mrežnog toka,
- (2) Vezivanje portova mrežnom toku,
- (3) Podešavanje prosleđivanja svih paketa za dati mrežni tok ka kontroleru,
- (4) Novi protokol vrši konfiguraciju ostalih mrežnih elemenata na svom datapathu a okviru svoje mrežne kriške,

11.2.4. Razdvajanje ravni kontrole od ravni podataka

SANE (2006): SANE i ETHANE projekti su smatrani pretečama onog što danas podrazumevamo pod softverski definisanim mrežama. SANE [17] arhitektura pojavila se kao odgovor na rastući izazov za upravljanjem mrežnim okruženjem, i povećanom kompleksnošću ovakvim sistema. Ponudila je logički centralizovano upravljanje mrežnim bezbednosnim i drugim mrežnim polisama koje piše mrežni operater. Ove polise su pisane jezikom koji je više korisnički orijentisan i bliži prirodnom (u ovom slučaju engleskom) jeziku. Jednom kad je polisa napisana i puštena, odluke vezane za kontrolu donosi server. Centralizovan autoritet za donošenje odluka i server se zajedno zovu domen kontroler. Domen kontroler takođe prima informacije vezane za promenu stanja veze i ima centralizovan pregled topologije.

Ciljevi dizajna ovog rešenja su bili: (1) polise pisane bliže prirodnom jeziku, topološki nezavisne i takve da apstrahuju hardverske razlike među proizvođačima čija oprema se koristi, (2) pomeranje sa L3 na L2 nivo upravljanja, radi sprečavanja smanjenja nivoa bezbednosti na nižim nivoima, (3) korišćenje bezbednosnog principa posedovanja samo neophodnih informacija, pojačavanjem sistema da na taj način kompromitovani sistemi ne otkriju kritične informacije i (4) uvođenje centralizovanog upravljanja bezbednošću sistema i time, ne samo pojednostavljenja bezbednosnog upravljanja, već i smanjenja stepena izloženosti servisa koji nude kritične informacije.

SANE je sloj koji se umeće između L2 i L3 OSI nivoa, na primer, između ethernet i IP sloja, predstavljajući još jedan sloj zaštite. Ovaj sloj upravlja eksternim enkriptovanim putanjama između dve krajnje tačke koje nazivamo mogućnostima. SANE bezbednosni tok rada je podeljen u dve faze: (1) korisnička registracija na domen kontroleru i (2) zahtev za mogućnost. Sledi primer SANE mrežne polise.

Zadatak polise: dodeli korisniku iz specificirane grupe da komunicira sa fajl serverom iz prodaje.

Generalna sintaksa polise: *users in group group-name to access sales file server.*

ETHANE (2007): ETHANE [18] je još jedan primer centralizovanog mrežnog upravljanja sa željom da se izbegnu greške u konfiguraciji kroz skup različitih alata kao što su pristupne kontrolne liste, paketni filteri, itd. Smatra se proširenjem Sane koncepta, gde je omogućen lakši deployment. ETHANE povezuje ethernet svičeve koji rade na principu mrežnog toka (flow bazirane) sa centralnim kontrolerom koji upravlja mrežnim tokovima. Svičevi su jednostavni elementi za prosleđivanje podataka i povezani su sa kontrolerom preko bezbednog kanala. U svakoj ETHANE mreži, svaka komunikacija između dva krajnja mrežna čvora mora da bude eksplicitno odobrena kroz: (1) kontroler i (2) elemente za prosleđivanje podataka, ETHANE svičeve. Polisa se nalazi u kontroleru koji donosi odluke vezano za mrežne tokove i polise. Za tokove koji su odobreni, kontroler obavlja proračune i pronalazi optimalnu putanju i na odabranoj putanji omogućava mrežni tok, konfigurisući svičeve.

Dizajn principi uključuju:

- (1) Deklarativnu mrežnu polisu višeg nivoa - pojednostavljujući upravljanje,
- (2) Mogućnost da se definiše mrežna putanja - dozvoljavajući preusmeravanje mrežnog saobraćaja,
- (3) Povećanu bezbednost - uz jaku spregu između paketa i njihovog porekla, omogućavajući praćenje korisnikove mobilnosti, hardvera i korisnički generisanih paketa.

11.2.5. Mrežna virtualizacija

Open vSwitch (2009): Open vSwitch je mrežni virtualni svič otvorenog izvornog koda, pod Apači licencom. Open vSwitch omogućava sledeće:

- (1) *Centralizovano upravljanje* koje može da se implementira koristeći sistem za konfiguraciju i sistem za asinhronu poruke,
- (2) *Virtualne privatne mreže* (VPN) koje mogu da se formiraju iznad hardverskih resursa i u višekorisničkom okruženju. Ovaj overlej sistem omogućava konekciju sa distribuiranim fizičkim resursima podržavajući protokole za uspostavljanje komunikacionih tunela,
- (3) *Mobilnost mreža* baziranih na IP protokolu. Mnoga rešenja u oblasti virtualizacije ne mogu da održe aktivnu konekciju na nivou transporta tokom migracije virtuelne mašine, iz kog razloga mobilnost (bez prekida rada) nije uvek moguća.

Open vSwitch je podržan u hipervizorskim okruženjima kao što su Xen, KVM, Proxmox VE, QEMU i VirtualBox. Takođe, dolazi uz mnoge Linux, odnosno Unix distribucije kao što su FreeBSD, NetBSD, Debian, Ubuntu (baziran na Debian-u) i Fedora. Nalazi se u okviru privilegovanog domena povezujući hardverske interfejsa sa virtualnim resursima. Posедуje mehanizam za prosleđivanje baziran na tabelama koji može biti upotrebljen za logičko particionisanje ravni za prosleđivanje. Open vSwitch se razlikuje od tradicionalnih rešenja u tome što eksportuje granularne interfejsa za kontrolu prosleđivanja, koji mogu biti korišćeni da podrže stvari kao što su: QoS, tunele, filtriranje paketa [19].

FlowVisor (2009): FlowVisor je mrežni rezač (eng. *slice*), koji omogućava virtualizaciju na nivou L2. Svaka, ovako isečena mrežna kriška se zove FlowSpace. Ono što je virtualna mašina u računarskim resursima opšte namene, to je FlowSpace u okviru FlowVisor domena.

Motivacija za FlowVisor je deljenje hardverskih resursa odgovornih za prosleđivanje podataka uz obezbeđivanje izolacije. Pristup omogućava korišćenje postojećih resursa da se isporuče produkcione brzine uz minimalni dodatno angažovanje resursa (eng. *overhead*). Može da se instalira na OpenFlow omogućenom hardveru, kao i u na hardveru koji izvršava operativne sisteme opšte namene. Budući da je mrežni hipervizor, FlowVizor se ugnježduje kao sloj između ravni podataka i kontrole, i zajedno sa OpenFlow omogućenim svičevima, omogućava particionisanje mreže.

Mrežu deli i kontroliše na mrežne kriške u smislu: protoka, topologije, saobraćaja, procesora i tabela za prosleđivanje. Što se tiče gornjih slojeva, FlowVisor omogućava veze ka kontroleru po izboru, jednom ili više. Cilj dizajna uključuje:

- (1) Poboljšanu alokaciju resursa,
- (2) Mogućnost testiranja promene pre puštanja u produkciju,
- (3) Izolovano višekorisničko okruženje uz deljenje hardvera,
- (4) Povećanu bezbednost,
- (5) Mogućnost eksperimentisanja sa novim protokolima,
- (6) Povećanu mobilnost resursa,
- (7) Povećanu brzinu puštanja novih mogućnosti u upotrebu, itd.

Ovo su zapravo odlike tipične za virtualizaciju opštih računarskih resursa, ali su primenjive na mrežnom nivou, pružajući apstrakciju hardvera i jasne interfejsne nivou iznad.

Network Virtualization Platform, NVP (2014): NVP [21] je inicijalno napravila Nicira networks koju je kasnije kupio VMWare i promenio naziv proizvoda u VMware NSX [22]. Motivacija za ovo rešenje je porast očekivanja u pogledu podržavanja mobilnosti na mrežnom nivou, kako po pitanju jednostavnosti, tako i po pitanju brzine, nalik na ono što je već bilo postignuto sa računarskim resursima. Pod mobilnošću podrazumevamo migraciju mrežnih resursa sa privatne na javnu višekorisničku infrastrukturu uz zadržavanje originalne mrežne konfiguracije. NVP je dizajniran da ispuni ove potrebe koristeći virtualizacione tehnološke principe korišćene u računarskim resursima opšte namene. Izazovi i dizajn principi NVP platforme bili su:

- (1) Dizajn datapath-a,
- (2) Deklarativno programiranje i
- (3) Skalabilnost.

NVP je kompletno rešenje iz oblasti mrežne virtualizacije, pravljeno oko Onix kontrolerske platforme [23]. Mrežni virtualizacioni sloj je dozvoljavao pravljenje i koegzistenciju virtualnih mreža sa različitim i odvojenim arhitekturama, topologijama i servisnim modelima na deljenoj fizičkoj mrežnoj infrastrukturi. Platforma pruža punu podršku za aplikacije kao što su fajervoli, ruteri, sistemi za otkrivanje i prevenciju upada, itd. Upravljanje i programabilnost virtualnih mreža se odvijaju preko API-ja, a ne više kroz pojedinačne konfiguracione interfejsne (eng. *command line interface, CLI*) koji variraju u zavisnosti od

proizvođača i modela [90]. Ovo je ključni doprinos NVP platforme. NVP mrežni hipervizor apstrahuje hardverski substrat i prevodi korisnički zadate konfiguracione promene na nizak nivo instrukcija za konkretan hardverski uređaj. Open vSwitch (OVS) je korišćen za prosleđivanje paketa na sve čvorove bazirane na x86 uniformom hardveru. Kontroler je zadužen za upravljanje svim virtualnim mrežama na programski način. Za *point-to-point* komunikaciju, NVP koristi tunele u virtualnim svičevima, logičke putanje podataka (eng. *datapath*) između svakog para hosta i hipervizora. Za komunikaciju između više tačaka (eng. *multipoint*), koriste se servisni čvorovi.

11.3. *SDM kontroleri*

Počecemo sa prvim SDM kontrolerom, NOX-om i nastaviti sa prvim primerom distribuiranog kontrolera, ONIX. Pomenućemo i rešenje na koje smo naišli a koje je bazirano na ekstenziji Linux operativnog sistema.

Rešenja

NOX (2008): Razvijen i napravljen otvorenim kodom kroz Apači licencu od strane Nicira Networks, NOX je prvi SDM kontroler odnosno mrežni operativni sistem baziran na OpenFlow-u. Motivacija je da se napravi mrežni operativni sistem koji bi omogućio uniforman i centralizovan programski interfejs ka celokupnoj mreži [24], dok aplikacije koje se izvršavaju povrh kontrolera pružaju potrebne mrežne funkcionalnosti. Inspiracija za NOX je došla od SANE, ETHANE i 4D projekata. NOX se izvršava u korisničkom prostoru (eng. *user space*) i implementiran je u C++ programskom jeziku. Programer ima mogućnost izbora dva jezika za razvoj mrežnih aplikacija, Python i C++. Tako, Python može biti korišćen za pravljenje brzih prototipova (tzv. *proof of concept*, POC rešenja), dok se C++ može koristiti za dobijanje visokih performansi i optimizaciju u produkciji, ako prototip rešenja bude usvojen. NOX arhitektura sadrži:

- (1) Open Flow svičeve,
- (2) Server na kome se izvršava NOX kontroler,
- (3) Bazu podataka koja sadrži informacije o mreži koje aplikacije koriste da bi donele upravljačke odluke.

NOX takođe koristi mrežne tokove. Ovo, naravno, implicira inspekciju paketa i njegovo uparivanje sa akcijom koju treba izvršiti nad datim paketom toka. Ukoliko nema poklapanja, a isto kao i svaki prvi paket u mrežnom toku, paket se prosleđuje kontroleru radi donošenja odluke. U posebnim slučajevima mrežnog toka, kao što je novi protokol, saobraćaj se usmerava ka kontroleru.

ONIX (2010): ONIX je prvi distribuirani SDM kontroler [23]. Motivacija vezana za stvaranje ONIX-a je da se primarno targetiraju velika, distribuirana, višekorisnička okruženja, sa primarnim fokusom na skaliranu, redundansu i dalju generalizaciju apstrakcija. Vrlo bitan doprinos ONIX-a je mogućnost da se pišu aplikacije uz apstrakciju distributivnih mehanizama. ONIX je mrežna platforma koja kontroliše stanje mrežnih svičeva, nastavljajući se na rad prethodnih rešenja poput RCP, SANE, ETHANE, 4D i NOX-a. Može biti instaliran na jednom ili više servera. Za aplikacionog programera omogućava API, dok na nižim nivoima prikuplja informacije o stanju vezane za jednostavne uređaje za prosleđivanje podataka i propagira kontrolno stanje između ovih uređaja, efektivno upravljajući mrežnim stanjem. Dizajn uključuje:

- (1) Generalizovan API koji zadržava mogućnost da se isporuči specifična funkcionalnost i omogućava dovoljno prostora za granularnost,
- (2) Skalabilnost koja nije narušena funkcionalnostima kontrolne ravni,
- (3) Pouzdanost, koja se postiže upravljanjem otkaza opreme,
- (4) Jednostavnost, omogućujući pisanje aplikacija na jednostavan i lak način i
- (5) Performanse, što se postiže održavanjem balansa između kontrolne ravni i optimalnih performansi.

Arhitektura ONIX platforme se sastoji od:

- (1) Mrežnih elemenata, odnosno hardvera,
- (2) Kontrolnog kanala, koji obezbeđuje kontrolu fizičkih uređaja, bilo *in-band* ili *out-of-band*,
- (3) ONIX distribuirani kontroler, koji se izvršava preko višestrukih čvorova klastera, obezbeđujući programsku i generalizovanu apstrakciju kontrole, skalabilnost i otpornost na kvarove za programere i
- (4) Kontrola, koja se nastavlja povrh apstrakcije koju obezbeđuje kontroler.

Skalabilnost se postiže kroz: particionisanje, agregaciju i trajnost. Pouzdanost se obezbeđuje kroz detekciju otkaza, preuzimanje posla od instance koja je otkazala ili redistribucijom posla na preostale ONIX instance.

Yanc (2014): Yanc SDM kontroler [25] koristi Linux operativni sistem kao osnovu i proširuje je tako da postane mrežni operativni sistem. Ključni izazov kod SDM kontrolera koji Yanc pokušava da reši je monolitni dizajn kontrolera koji je tipično vezan za programski jezik koji kontroler podržava. Ovo ograničava brzinu i diverzitet razvoja SDM aplikacija. Dalje, većina kontrolera ima problema sa portabilnošću platformi kroz različite hardverske platforme. Budući da je Linux operativni sistem opšte namene, široko je rasprostranjen i ima dobru podršku za različite arhitekture. Može da se izvršava na uniformnom a i zastarelom hardveru ali i obezbedi podršku za buduće hardverske arhitekture, bez potrebe za lobiranjem za podršku kroz industriju.

U Yanc arhitekturi, dizajn mrežnih aplikacija nije ograničen od strane kontrolera. Apstrakcija kontrolne ravni, kao što je OpenFlow je reprezentovana drajverom. Pružanje podrške za novi protokol svodi se na pisanje novog drajvera, koji može jednostavno biti dodat, nadograđen, zamenjen i uklonjen. Administracija aplikativnog ponašanja, prava i drugih interakcija može da se obavlja od strane sistem administratora. Manipulacija mrežnog stanja takođe može da se obavlja korišćenjem standardnih Linux fajl sistem komandi.

11.4. Mrežni svič operativni sistemi

Postoji rastući broj operativnih sistema za svičeve baziranih na Linux-u koji se mogu instalirati na poseban tip mrežnih uređaja koje nazivamo *bare metal* svičevima ili *whitebox* svičevima. Ovakvi uređaji se razlikuju od klasičnih, proizvođačkih, po tome što je na njih moguće instalirati više različitih operativnih sistema. Slična analogija su mikroracunari opšte namene, bazirani na x86 arhitekturi, na koje je moguće instalirati operativni sistem po želji kao što su Windows, Linux ili neki drugi sistem.

Iako u pivoju, raste broj ovih sistema pa anticipiramo nastavak pozitivnog trenda i sve veći uticaj ovakvih rešenja.

Rešenja

Cumulus Linux (2014): Cumulus Networks je napravio Cumulus Linux, mrežni operativni sistem baziran na Linux-u [26]. Motivacija za ovo rešenje je omogućiti da Linux pogoni što veći dijapazon hardvera, pokrivajući razne procesorske arhitekture. Cumulus Linux cilja da reši sledeće izazove:

- (1) Ograničena skalabilnost u tradicionalnim okruženjima,
- (2) Kompleksnost konfiguracionih promena i njihova propagacija kroz mrežne elemente,
- (3) Visoka cena opreme i održavanja.

Cumulus Linux je baziran na Debian distribuciji. Mnogi postojeći Linux alati, mogu potencijalno da se izvršavaju na ovom operativnom sistemu. Bash, ili druga skripting okruženja, mogu da se izvršavaju u svom prirodnom okruženju i pomognu u poslovima automatizacije zadataka. Povrh toga, današnji defakto standardni alati za upravljanje, automatizaciju i dobavljanje resursa, Chef, Puppet i CFEngine su podržani. Bilo koji standardni programski jezik koji Debian distribucija podržava ili koji na nju može da se instalira takođe može da se upotrebi. Kako je svič drajver deo Linux kernela, pisanjem ili dodavanjem novog drajvera se obezbeđuje podrška za dati hardver. Pomenuti alati za automatizaciju i upravljanje se nalaze u korisničkom delu sistema i predstavljaju deo hardverskog sloja za apstrakciju (eng. *hardware abstraction layer*). Konačno, cena može biti redukovana korišćenjem zajedničke softverske platforme za postojeći hardver i omogućujući portabilnost sistema kroz različite proizvođače i procesorske arhitekture (x86, PowerPC i ARM). Cumulus Linux rešenje se sastoji iz tri sloja:

- (1) Hardver, uglavnom svič,
- (2) Mrežni operativni sistem, u ovom slučaju Cumulus Linux i
- (3) Linux aplikacije iz neke od sledećih kategorija: rutiranje, monitoring, mrežno upravljanje, automatizacija, mrežni overlejš.

Open Network Linux (2014): Open Network Linux (ONL) je Linux distribucija za *bare-metal* svičeve. Nastala je kao rezultat saradnje između kompanije Big Switch, Facebook i NTT [27]. ONL je baziran na Debian distribuciji i predstavlja osnovni mrežni operativni sistem sa ciljem da postane platforma za kreiranje prilagođenih algoritama za prosleđivanje podataka (eng. *forwarding*). ONL dva implementaciona primera, kao nadgradnja na osnovu su Facebookov FBOSS [91] i Big Switchev Switch Light OS [28].

ONL je inicijalno napravljen sa idejom da podrži svičeve otvorenog dizajna (eng. *open source*), odnosno *bare-metal* svičeve pod kapom Open Compute Project-a (OCP). Međutim, sve veći broj proizvođača počinje da podržava ONL, čak i oni koji prave sopstvena rešenja zatvorenog tipa. ONL je viđen kao potencijalni alat za učenje u akademiji. Ono što Mininet predstavlja u oblasti učenja i eksperimentisanja u oblasti virtualizacije mreža, to ONL potencijalno može da postane u oblasti prosleđivanja paketa (eng. *packet forwarding*). Najvažnije, viđen je kao zajednički blok za izgradnju različitih proizvođačkih rešenja gde donosi dobrobiti poput:

- (1) Upravljanja mrežnim resursima na isti način kao serverima,
- (2) Apstrakcija platforme i podrška različitim arhitekturama i
- (3) Podrška za hardver otvorenog dizajna.

Switch Light (2013): Switch Light je baziran na Open Network Linux u a napravio ga je Big Switch Networks [28], [92]. Potekao je iz projekta Indigo koji je inicijativa otvorenog izvornog koda u okviru projekta Floodlight. Dolazi u vidu dva rešenja, Switch Light OS i Switch Light VX. Verzija OS je rešenje za *bare-metal* svičeve, dok je VX rešenje za virtualne svičeve u okviru KVM hipervizora koji se izvršavaju na generalnim procesorskim arhitekturama. Switch Light sadrži API koji apstrahuje ravan podataka i za L2 i za L3 nivo, kako za OS verziju, tako i za VX. Takođe, ovo rešenje sadrži u sebi podršku za OpenFlow protokol koju koristi da se upari sa SDM kontrolerom. Pošto je baziran na ONL-u, ima svu kompatibilnost i podršku za hardver automatski nasleđenu. Switch Light podržava:

- (1) Bezbedno podizanje i mehanizam otkrivanja,
- (2) Centralizovano upravljanje firmwareom,
- (3) Centralizovanu programabilnost i kontrolu,
- (4) Multi-kontroler otkrivanje i kontrolu.

Zaključujemo: (1) da se većina rešenja u oblastima bazira na debian distribucijama, (2) da je arhitektura troslojna: (i) hardver, (ii) operativni sistem baziran na Linux-u i (iii) aplikacije, (3) da je upravljanje mrežnim resursima potencijalno moguće na sličan ili isti sistem kao sa serverima, (4) da zbog generalizacije hardvera kroz operativni sistem i dobre podrške drajverima, postoji dobra podrška različitim arhitekturama i uređajima, (5) da postoji dobra podrška za hardver otvorenog dizajna, (6) da većina rešenja omogućava centralizovano upravljanje firmverom.

11.5. *Mogućnosti i budući pravci na mrežnom sloju*

Rani rad na programabilnim mrežama se fokusirao na pristup aktivnih mreža, koji je imao dosta praktičnih izazova. Enkapsulacija mrežnih programa i njihovo procesiranje na aktivnim čvorovima nije doživela razvoj. Iako su mreže postale programabilne, način kako se promene puštaju u produkciju, alati za razvoj i ceo pristup nije adresirao fundamentalne potrebe istraživača u oblasti mreža i industrije. Drugi, pristup otvorenih mrežnih signala razmatrao je razdvajanje ravni kontrole i ravni prosleđivanja podataka. Ovaj pristup je naišao na bolji odziv i dokazao se kao više primenjiv za potrebe pojednostavljenog upravljanja i izazova povećanja kompleksnosti. Dalje, programabilne apstrakcije su bile ključni izvor za omogućavanje daljeg napretka koji je pratila ekspanzija mrežnih kontrolera i platformi. Ovo je omogućilo dalje apstrakcije, kako za one koji upravljaju mrežama, tako i za one koji projektuju nove arhitekture.

Softverski definisane mreže nisu jedno rešenje. To je pristup arhitekturi izgradnje računarskih mreža na modularan način i uz punu programabilnost. Još jedan vrlo važan aspekt evaluacije računarskih mreža je inicijativa da se koristi hardver opšteg tipa koji je jeftiniji. Veruje se da će ovakva promena na hardverskoj strani otvoriti put ka većem rastu i širem usvajanju novog pristupa ravni podataka u računarskim centrima budućnosti, ali i u manjim firmama. SDM nije ekskluzivno arhitektura za velike.

Dalje diskutujemo pravce razvoja i šanse za razvoj savremenih SDM rešenja od kojih posao potencijalno može da ima koristi.

Tabela 12 - Potencijal i uticaj SDM-a na poslovne operativne modele

Poslovni	Tehnološki	
	Razvoj i straživanje	Strateški i Operativni
<ul style="list-style-type: none"> ▪ Porast konkurencije ▪ Ugovoreno održavanje mreža ▪ Povećana agilnost ▪ Redukovana cena uređaja ▪ Produžen životni ciklus opreme ▪ Pojednostavljenje poslova mrežnih operatera 	<ul style="list-style-type: none"> ▪ Povećana brzina inovacije u mrežama ▪ Povećana brzina inovacije uređaja ▪ Omogućeno eksperimentisanje ▪ Porast analitike 	<ul style="list-style-type: none"> ▪ Otvoren pristup tehnologiji ▪ Povećan izbor dizajna mrežne arhitekture ▪ Dobavljanje mrežnih resursa kao usluge ▪ Centralizacija upravljanja mrežama ▪ Centralizacija upravljana kompletnom infrastrukturom ▪ Smanjen faktor ljudske greške ▪ Pojednostavljenje poslova mrežnih operatera ▪ Upravljanje isključivo serverima ▪ Razdvojene mrežne funkcionalnosti ▪ Povećane performanse prosleđivanja paketa

Povećana agilnost: Postaje jednostavnije i brže napisati mrežnu politiku ili konfiguracionu promenu i implementirati je kroz ceo datapath. Postaje takođe brže razviti mrežnu funkcionalnost i pustiti je u produkciju. SDM donosi veću i bržu mogućnost prilagođavanja.

Otvoren pristup tehnologiji: Tehnološko zaključavanje je situacija kada korisnik ne može da promeni proizvođača, odnosno dobavljača usluge iz razloga nepostojanja interoperabilnosti ili visokih troškova. Pružajući jasne, otvorene i od industrije podržane apstrakcije i standarde poput OpenFlow-a, tehnološko zaključavanje se može izbeći.

Još jedan način da se izbegne zaključavanje je korišćenje Linux operativnog sistema, kao zajedničke platforme, bilo u izvornom obliku, bilo uz određena prilagođavanja. Diskutovali smo upotrebu Linuxa kao SDM kontrolera ali i kao otvorenog operativnog sistema za *bare-metal* svičeve.

Porast konkurencije: Konkurencija će se verovatno povećati sa povećanjem broja proizvođača koji podržavaju otvorene komunikacione apstrakcije poput OpenFlow-a. Ovo potencijalno vodi ka promeni od diferencijacije u smislu funkcionalnosti ka preusmeravanju proizvođača da se diferenciraju po kriterijumima cene, kvaliteta, pouzdanosti, performansi, kapaciteta protoka, itd.

Ugovoreno održavanje mreža: Diskutovali smo o mrežnim overlejima koji se izvršavaju na deljenoj fizičkoj infrastrukturi. Vertikalni deo ili kriška ovakve virtualne mreže može da bude dobavljena na korišćenje korisniku. Ovakva mrežna kriška može da se održava unutar firme ali je u isto vreme moguće i ugovoreno održavanje mreža. Na taj način, firma može da posluje, potencijalno čak i bez mrežnih inženjera a da u isto vreme ima punu uslugu i kapacitet korišćenja. Ovakav koncept pomaže i agilnosti poslovanja firme koja se opredeljuje za ugovorno održavanje mreža jer omogućava: (1) bolju skalabilnost, odnosno rast, (2) veću elastičnost u oba smera, (3) mogućnost izmene i zamene jednog ugovornog održavanja drugim, (4) povećanje likvidnosti uz redukovanja operativnih troškova.

Povećan izbor dizajna mrežnih arhitektura: Sa otvorenim apstakcijama ka hardveru, svaki proizvođač koji podržava date abstrakcije, postaje moguć izbor. Ravan podataka koja se sastoji od hardvera, može da bude sastavljena od više različitih proizvođača. Ako razmatramo kontrolnu ravan, imamo sve veći izbor SDM kontrolera koji dodatno mogu da koegzistiraju ukoliko ispod imamo mrežni hipervizorim i međusobno sarađuju putem istočno-zapadnog API-ja. Konačno, mnogi alati i okruženja za automatizaciju, orkestraciju i dobavljanje servera mogu biti od koristi u SDM okruženju.

Dobavljanje mreže kao usluge (NaaS): Mogućnost dobavljanja mrežne kriške kroz programabilne apstrakcije može da pruži slično iskustvo kao dobavljanje računarskih resursa. Dobavljanje mreže kao usluge i iskustvo upravljanja može biti analogno dobavljanju infrastrukture kao servisa (IaaS) na Amazonu i Microsoft Azuru.

Centralizovano upravljanje mrežom: Tradicionalne mreže su dizajnirane kao distribuiran sistem. Njihovi nezavisni elementi vrše razmenu informacija, pregovaraju i konvergiraju kad god dođe do promene stanja na mreži. Ne postoji centralizovana topologija ili upravljanje. U softverski definisanim mrežama, upravljanje mrežama se vrši na logički centralizovan način. Fizički, zbog raznih redundansi, visoke dostupnosti i podrške distribuiranim modelima, imamo heterogeno okruženje.

Centralizovano upravljanje kompletnom infrastrukturom: Severni interfejs može potencijalno da se kombinuje sa ostatkom infrastrukture i prezentira kroz jedinstvenu abstrakciju, naročito u rešenjima zasnovanim na Linux operativnom sistemu.

Upravljanje samo serverima: Uz povećanje broja mrežnih operativnih sistema upravljanje mrežnim uređajima za prosleđivanje podataka postaje zapravo upravljanje serverima. Ako tome dodatom SDM

kontroler baziran na Linuxu, dobijamo unifikaciju klad resursa gde sve postaje upravljanje operativnim sistemima.

Pojednostavljenje poslova mrežnih operatera: Izvestan broj SDM rešenja podržava programske jezike za pisanje mrežnih polisa koji su jednostavni i bliži prirodnom izražavanju. Sa ovakvim jezicima u okviru mrežnih ekosistema, kompanije mogu da zaposle tehničare umesto mrežnih inženjera, odnosno da spuste kriterijume tehničkog znanja potrebnih za obavljanje ovog posla. Na ovaj način mogu da zaposle jeftiniju radnu snagu i na taj način uštede. Smanjenjem broja konfiguracionih komandi koje je potrebno napisati, broj radnih mesta koji je potreban za obavljanje ove funkcije će takođe potencijalno biti manji. Ovo sve je moguće jer su mrežne polise na višem nivo apstrakcije, centralizovano se njima upravlja i kada se jednom napišu, kontroler vrši konfiguraciju. U tradicionalnom okruženju više mrežnih operatera bi moralo da radi na svakom uređaju konfiguracione promene. Iako postoje alati za automatizaciju i u tradicionalnom pristupu, kompletan koncept SDM-a je efikasniji i elegantniji.

Smanjen faktor ljudske greške: Centralizovano upravljanje mrežom značajno smanjuje nastanak ljudske greške na konfiguracionom nivou. Neki od primera su greške u konfiguraciji pristupnih kontrolnih listi, paket filtera, itd. Nekoliko mehanizama omogućava smanjenje grešaka: (1) kontroler preuzima konfiguraciju kompletnog datapath-a eliminišući potrebu za ručnom konfiguracijom po uređaju, (2) mrežne polise i jezici u kojima se one pišu su visokog nivoa, znatno bliže ljudskom prirodnom izražavanju i zamenjuju nekoliko pojedinačnih alata kao što su pristupne kontrolne liste, paket filteri, itd., (3) generalne apstrakcije se koriste umesto proizvođački specifičnih konfiguracionih komandi i interfejsa, tako stvarajući novi unificirani programabilni ekosistem u kome je iskustvo konfiguracije proizvođački agnostičko.

Redukovana cena uređaja: Velik broj mrežnih funkcionalnosti i protokola su implementirani u tradicionalnim mrežnim uređajima, čineći ove uređaje skupima. Odvajanjem ravni kontrole i ravni prosleđivanja podataka, hardver može biti jednostavniji i samim tim i jeftiniji.

Još jedna mogućnost smanjenja cene hardvera je prelazak sa specijalizovanih proizvođačkih mrežnih procesora, tzv. ASIC kola, na korišćenje: (1) trgovačkog silicijuma ili, što bi imalo još veći uticaj (2) opštih računarskih arhitektura (x86, ARM, MIPS, PowerPC). Na ovaj način bi bilo moguće optimalno izabrati arhitekturu procesora, zavisno od primene i odnosa performansi naspram cene.

Produžen životni ciklus opreme: S obzirom da elementi ravni podataka postaju jednostavni elementi za prosleđivanje podataka dok je logika smeštena u kontrolnu ravan, logika se ažurira kao i bilo koji softver, često bez potrebe da se hardver zameni dok god API ka hardveru nije prevaziđen. Čak i u tim situacijama, moguće je ažurirati *firmware* gde izvesna SDM rešenja nude tu mogućnost kroz centralizovano upravljanje.

Povećana brzina inovacije uređaja: Hardverske inovacije se često oslanjaju na proizvođače i njihova specijalizovane procesore, takozvana ASIC kola. Brzina inovacije je time vezana za sposobnost i brzinu konkretnog proizvođača da isporuči inovaciju. Sa prelaskom sa specijalizovanih prilagođenih ASIC procesora na generalne procesorske arhitekture, viši stepen inovacije i često kraći ciklus ovakvih proizvođača može pomoći da se inovira brže.

Razdvojene mrežne funkcionalnosti: Po mrežnom uređaju, proizvođači nude velik broj funkcija, opcija i protokola. Mnogi korisnici imaju potrebu za manjim podskupom ovog broja funkcija. Tradicionalni pristup kreira veliki instalacioni otisak, dodajući na kompleksnosti okruženja i zahtevima za resursima. Obzirom da je kontrolnu logiku moguće izmestiti gotovo bilo gde van uređaja (server, virtuelnu mašinu, cloud, itd.), a SDM aplikacije se razvijaju i instaliraju po potrebi, postaje moguće na sistemu imati samo neophodan skup mrežnih funkcija.

Povećane performanse prosleđivanja paketa: Funkcija fleksibilnosti i upravljivosti mreža stavlja akcenat na programabilnost ravni podataka. Često se percipira da programabilnost ima cenu gledano kroz performanse. Zadržavanje programabilnosti uz poboljšanje performansi stavlja u perspektivu rešenja iz oblasti brzih ravni koja se mogu integrisati u deo SDM, NFV ali i klasičnih rešenja zasnovanih recimo na Quagga ruteru i slično. Ova rešenja se takođe mogu koristiti u virtualizovanim datacentrima kako bi olakšalo i prebacilo prosleđivanje paketa sa guest operativnih sistema.

Povećana brzina inovacije u mrežama: Diskutovali smo o tradicionalnim proizvođačkim i vertikalno integrisanom pristupu mrežnim elementima. U ovakvim ekosistemima inovacija zavisi isključivo od volje i mogućnosti proizvođača da adresira zahteve svojih klijenata. Čak i u slučaju prihvatanja zahteva, proces je dugotrajan i može uključivati standardizaciona tela. U tom slučaju kompleksnost i potrebno vreme za implementaciju se još više povećavaju. Sa SDM modularnom arhitekturom koja sadrži različite apstrakcije između komponenti, jednostavnije je unapređivati delove sistema brzinom razvoja softvera. Razvoj rešenja se vrši standardnim alatima, programskim jezicima i tehnikama. Sa povećanjem podržanih

programskih jezika u SDM arhitekturama, pisanje prototipa može da se uradi u jednom jeziku a puno optimizovano rešenje može da se implementira u za to pogodnijem programskom jeziku.

Omogućeno eksperimentisanje: Istraživači u oblasti računarskih mreža dugo nisu imali pravu mogućnost realnog testiranja i vršenja eksperimenata. Često nije bilo moguće probati nov protokol ili neku novu funkcionalnost. Za ovakve potrebe dugo vremena su isključivo korišćeni mrežni simulatori i eventualno manje produkcijske instalacije uz saradnju sa proizvođačima i prilagođavanje uređaja eksperimentu. SDM sada omogućava ovakve eksperimente. Pruža mogućnost šta-ako analize, predviđanje događaja ili simulaciju mrežnog ponašanja. Takođe omogućava razdvajanje proizvodnog od eksperimentalnog saobraćaja i omogućava razvoj i upotrebu novih protokola.

Porast analitike: Softverski definisane mreže donose veliki broj informacija kao što je stanje hardvera, aplikacija i korisničkih informacija. Ovakve informacije, spregnute sa konkretnom logikom, mogu da pomognu u donošenju raznih odluka. Jedan od ključnih ciljeva bi mogao biti unapređenje infrastrukture u realnom vremenu ali u isto vreme može biti korišćeno za uvećanu bezbednost, samoregulaciju i samopopravljanje sistema, mehanizme redundantnosti i visoke dostupnosti. Sa poslovne strane, može doprineti da se mrežama bolje i efikasnije upravlja, da se mreže bolje i brže razumeju i analiziraju, kao i da se novi servisi prilagode, specijalizuju i kreiraju [93].

12. Pristup izbora rešenja pri izgradnji cloud sistema

Pri sadašnjem stanju tržišnih ponuda cloud tehnologija, ne postoji jedinstveno rešenje za sve poslovne sisteme, već se za konkretni poslovni sistem iz datog skupa tehnologija može izabrati bar jedno rešenje bolje od ostalih. S tim u vezi definisali smo nekoliko hipoteza koje ispituje anketom. Anketa je izvršena nad 100 kompanija.

12.1. *Predlog pristupu rešenja*

U radu koristimo opšte i posebne naučne metode kao ispod.

- Od opštih metoda:
 - Sistemski pristup,
 - Analitičko-sintetičku metodu, kao postupak kojim se prilazi suštini problema, na način da se obrade međusobno uslovljeni odnosi, a potom i problem u celini,
 - Deduktivno-logička metoda pomoću koje će se, na temelju spoznaje utvrđene praksom, utemeljiti vlastiti teorijski stavovi,
 - Komparativna metoda.
- Od posebnih metoda:
 - Anketni upitnik za utvrđivanje uspešnosti primene cloud rešenja i ostvarenja konkurentskih prednosti u poslovanju kompanija,
 - Statističke metode radi obrade podataka i prezentacije rezultata.

Anketnim upitnikom, kao jednim od posebnih naučnih metoda ispitujeemo hipoteze:

1. Opšte:

- a. Polazna hipoteza je da pri sadašnjem stanju tržišnih ponuda cloud tehnologija, ne postoji jedno optimalno rešenje za sve poslovne sisteme, već da se za konkretni poslovni sistem može iz datog skupa cloud Sistema naći bar jedno rešenje koje je bolje od ostalih.

2. Pojedinačne:

- H1 - Postoji pozitivna povezanost između produktivnosti - konkurentnosti preduzeća i nivoa primene cloud tehnologija.
- H4 - Informacionim sistemima zasnovanim na cloud rešenjima postiže se ušteda u resursima u odnosu na tradicionalni pristup realizaciji infrastructure.
- H5 – Informacionim sistemima zasnovanim na cloud rešenjima postiže se veća poslovna agilnost u izgradnji Sistema u odnosu na tradicionalni pristup realizaciji infrastructure.
- H6 – Opredeljenjem za cloud rešenja može se postići bolja skalabilnost Sistema, odnosno jednostavnija alokacija ili dealokacija potrebnih resursa u odnosu na klasični pristup realizaciji informacione infrastructure.
- H7 - Opredeljenjem za virtualizaciju na mrežnom sloju postiže se viši stepen automatizacije i bolje upravljanje celokupnim sistemom u odnosu na klasični pristup realizaciji informacione infrastructure.

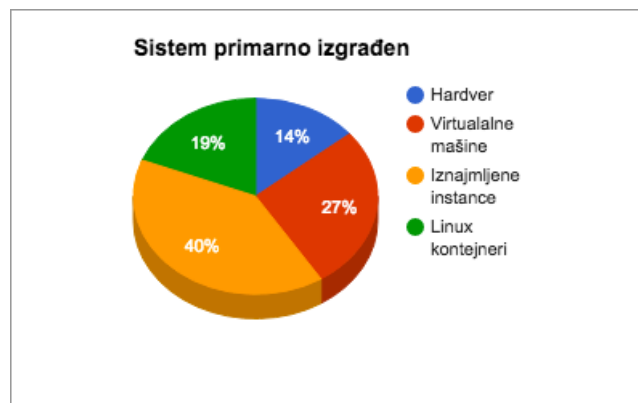
Profil firmi koje smo ispitivali su visokotehnološke firme iz kategorije kasne startup kompanije ka srednjim kompanijama. Nalaze se između osme i petnaeste godine poslovanja i imaju između 1.000 i 10.000 zaposlenih.

12.2. Analiza Rezultata Ankete

U nastavku sledi analiza rezultata ankete. Anketni uzorak je 100 visokotehnoloških kompanija koje su same izgradile ili grade svoje cloud sisteme. Anketna pitanja sa više detalja se mogu videti u Prilogu A, na kraju ovog rada.

Pitanje #1: Kojom od sledećih tehnologija je kompanijski cloud system primarno izgrađen?

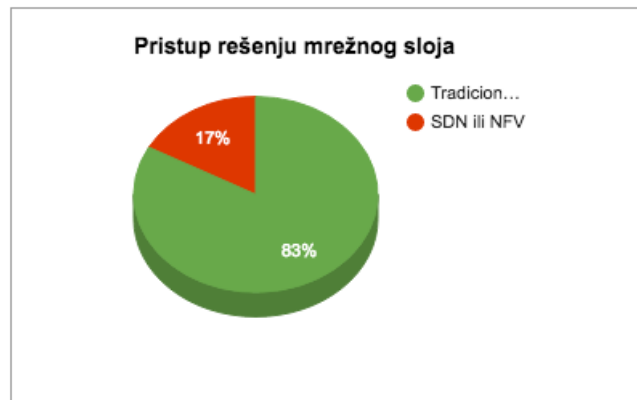
- Slike 32 se može zaključiti da većina kompanija i izgrađuje svoje sisteme uglavnom koristeći Amazon instance, a zatim, kao sledeću varijantu koriste virtualne mašine. Linux kontejneri polako postaju sve aktuelniji ali da bi ovde došlo do porasta, aplikacije je potrebno prebaciti. Smatramo da je od svih navedenih tehnologija ovo tehnologija za koju se veruje da će se sve više povećavati.



Slika 32 – Tehnologija kojom je cloud primarno izgrađen

Pitanje #2: Koja rešenja koristite na mrežnom sloju?

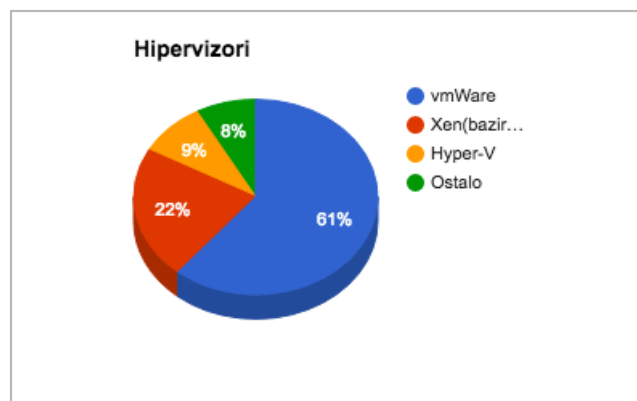
- Sa slike 33 se može zaključiti da većina ispitanih kompanija koristi tradicionalna mrežna rešenja. Pod tradicionalnim rešenjima podrazumevamo rešenja koja nisu iz domena softverski definisanih mreža ili virtualizacije mrežnih funkcija.



Slika 33 - Pristup rešenju mrežnog sloja

Pitanje #3: Da li koristite i koji od navedenih Tip 1 hipervizora?

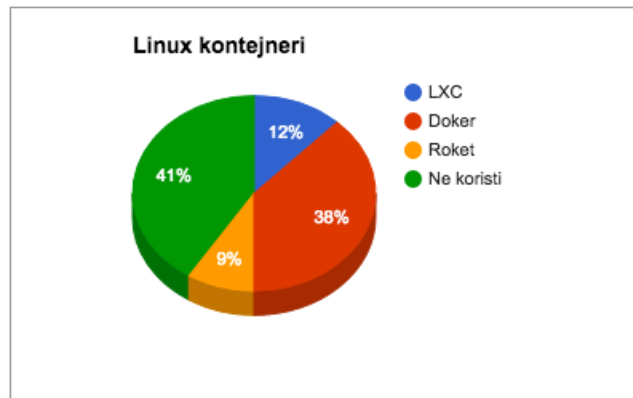
11. Sa slike 34 se može zaključiti da od onih koji koriste punu virtualizaciju, većina koristi VMware, zatim rešenja bazirana na Xen hipervizoru, uključujući (uglavnom) Citrix



Slika 34 - Tip 1 hipervizora koji se najviše koristi

Pitanje #4: Da li koristite i koji od navedenih Linux kontejnera?

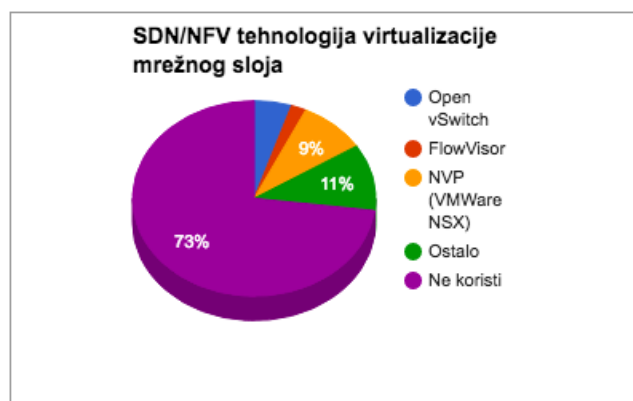
12. Sa slike 35 se može zaključiti da je Docker dominantna vrsta Linux kontejnera, mada većina još uvek nije prešla na kontejnere.



Slika 35 - Tip Linux kontejnera u upotrebi

Pitanje #5: Da li koristite neku od SDM/NFV tehnologija virtualizacije mrežnog sloja?

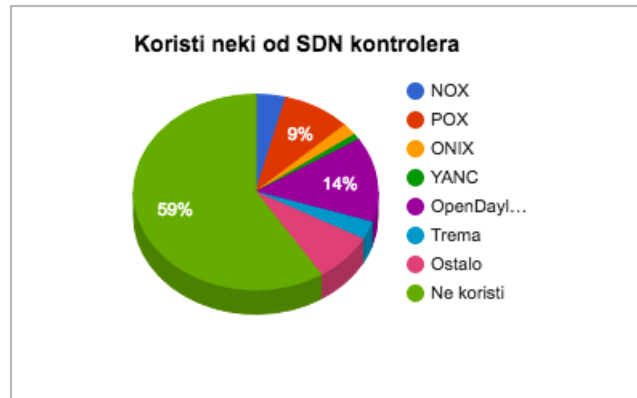
- Sa slike 36 se može zaključiti da većina ne koristi uopšte, a oni koji koriste, uglavnom se oslanjaju na VMWare NSX. Ovo je skup tehnologija u fazi sazrevanja i slika se menja svakodnevno. Firme koje koriste FlowVisor, uglavnom eksperimentišu sa rešenjima koja bi bila pogodna za velike *multitenant* sisteme.



Slika 36 - Tehnologija virtualizacije mrežnog sloja

Pitanje #6: Da li koristite neki od SDM kontrolera?

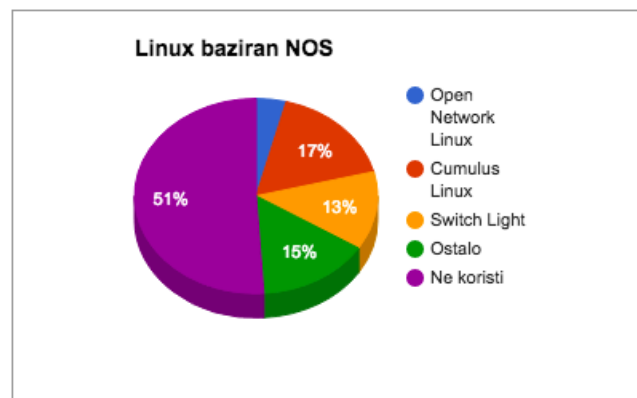
- Radi normalizacije podataka ispitanike smo pitali da uvek izaberu izvorni proizvod, ukoliko je proizvod koji koriste baziran na nekom drugom rešenju uglavnom otvorenog izvornog koda. Većina se izjasnilo da koristi OpenDaylight ili neku verziju, odnosno proizvod, derivat OpenDaylight-a (slika 37).



Slika 37 - Tip SDM kontrolera u upotrebi

Pitanje #7: Da li koristite neku od tehnologija mrežnog sloja zasnovanu na Linux operativnom sistemu i koje je portabilno na hardver više proizvođača?

- Sa slike 38 se može zaključiti da i dalje većina ne pribegava *whitebox* ili *bare-metal* rešenjima. Od onih koji koriste, većina se opredelila za Cumulus Linux.



Slika 38 - Linux kao NOS

Pitanje #8: Da li i koliko imate starih (eng. legacy) sistema koji zahtevaju specifičan operativni sistem ili specifičnu verziju ili postoje samo za jednu platformu?

13. Sa slike 39 se može zaključiti da većina kompanija ima potrebu da koristi neki softver koji ima zahtev da se izvršava u okviru specifičnog okruženja. Uglavnom se ovakvi slučajevi rešavaju punom virtualizacijom uz instalaciju kompletnog operativnog sistema, neretko samo za upotrebu od strane takvog Sistema.



Slika 39 - Upotreba starih i specifičnih sistema

Pitanje #9: Ako imate stare sisteme, da li je to bilo opredeljenje za pristup izgradnji cloud sistema baziranog na virtualizaciji mašina ili virtualne instance (naspram kontejnerima)?

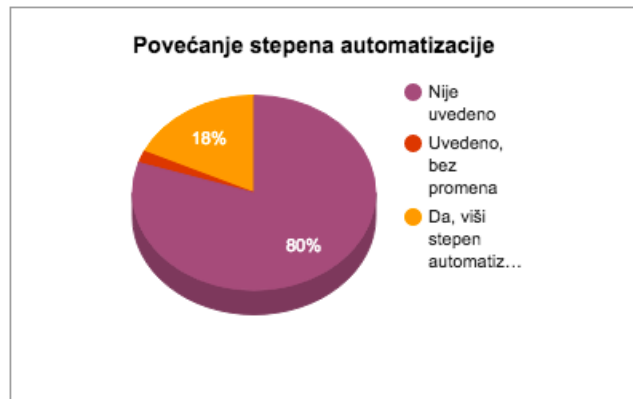
▪ Sa slike 40 se može zaključiti da je uticaj zastarelih sistema na arhitekturu i potreba da se oni podrže još uvek vrlo značajna u kontekstu opredeljenja za konačno rešenje arhitekture kompletnog cloud Sistema.



Slika 40 – Uticaj specifičnih ili zastarelih sistema na arhitekturu cloud sistema

Pitanje #10: Sa uvođenjem tehnologija na mrežnom sloju, da li ste uspeali da ostvarite viši stepen automatizacije (slika 41)? Primer: stvari koje su rađene ručnom konfiguracijom su automatizovane.

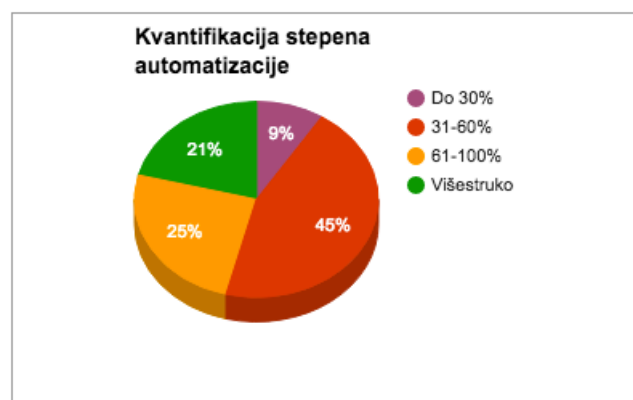
14. 90% kompanija koje su uvele tehnologije mrežnog sloja su ostvarila viši stepen automatizacije. Samo 10% kompanija je ostalo na približnom nivou i nijedna kompanija nije prijavila nepromenjeno stanje automatizacije (identično ili lošije).



Slika 41 - Ostvarena automatizacija

Pitanje #11: Sa uvođenjem tehnologija na mrežnom sloju, da li ste uspeali da ostvarite viši stepen automatizacije na mrežnom sloju i koliko (slika 42)? Primer: Automatizacija i propagacija konfiguracije.

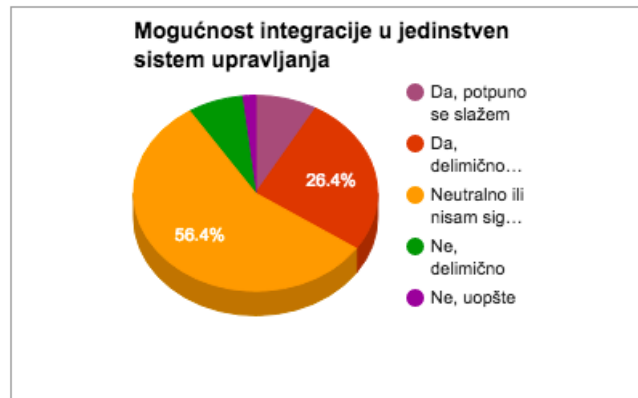
15. Većina kompanija koje su uvele tehnologije mrežnog sloja su ostvarila viši stepen automatizacije. Ovde vidimo da većina koja je uvela ovu vrstu tehnologija vidi povećanje automatizacija na nivou od 31-60%.



Slika 42 - Ostvaren stepen automatizacije

Pitanje #12: Da li upravljanje možete da integrišete u jedinstven sistem upravljanja?

16. Sa slike 43 se može zaključiti da većina nije uspela da mrežnom infrastrukturom upravlja na istovetan način ili kroz isti sistem kao što to čini sa serverskom infrastrukturom.



Slika 43 - Mogućnost integracije u jedinstven sistem upravljanja

Pitanje #13: Da li je mrežnim slojem moguće upravljati pomoću standardnih serverskih frejmvorka za automatizaciju i orkestraciju kao što su Chef, Puppet, CFEngine, Salt, Ansible?

17. Sa slike 44 se može zaključiti da većina ispitanika ne uspeva da ostvari to ali postoji velika grupa (37%) koja je uspela da to delimično ostvari i manja grupa koja je uspela da ostvari potpunu automatizaciju (11%).



Slika 44 - Mogućnost upravljanja mrežnim slojem korišćenjem alata za automatizaciju

Pitanje #14: Da li imate potrebu za podrškom više raznorodnih operativnih serverskih sistema u okviru cloud sistema i kojim?

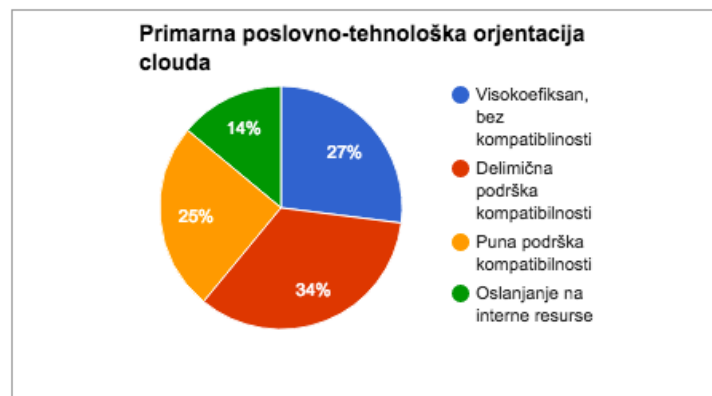
18. Sa slike 45 se može zaključiti da većina korisnika uglavnom ima potrebu za podršku Linux operativnim sistemima, odnosno nema potrebu za mnogo raznorodnih sistema.



Slika 45 - Potreba za multi OS podrškom

Pitanje #15: Koja je primarna poslovno-tehnološka namena sistema?

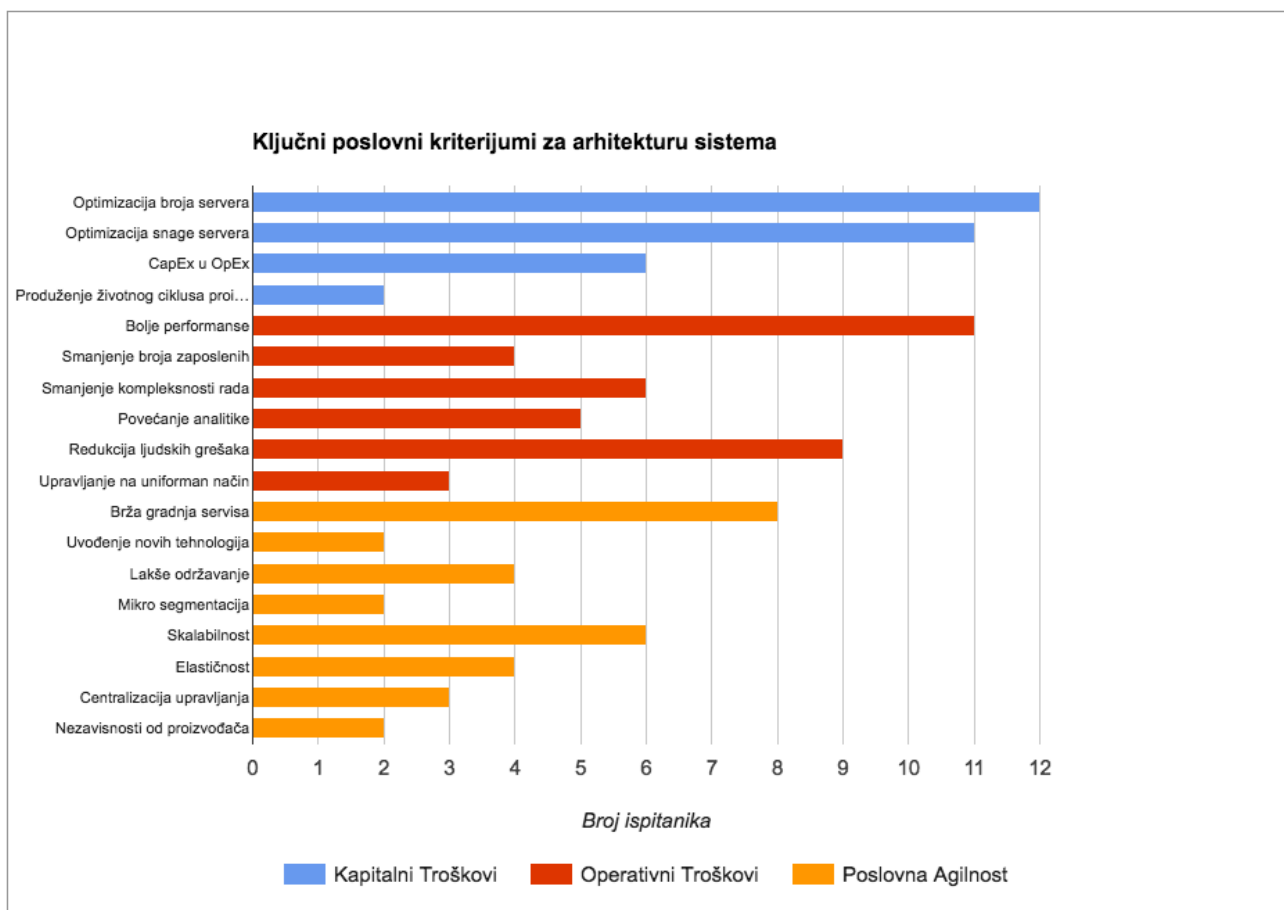
19. Kriterijum su rangirani na sledeći način: delimična podrška kompatibilnosti (34%), visoka efikasnost, bez kompatibilnosti (27%), puna podrška kompatibilnosti (25%) i oslanjanje na interne resurse (14%) (slika 46).



Slika 46 - Primarna poslovno-tehnološka orijentacija clouda

Pitanje #16: Koji od sledećih poslovnih kriterijuma su bili ključni u dizajnu arhitekture Sistema?

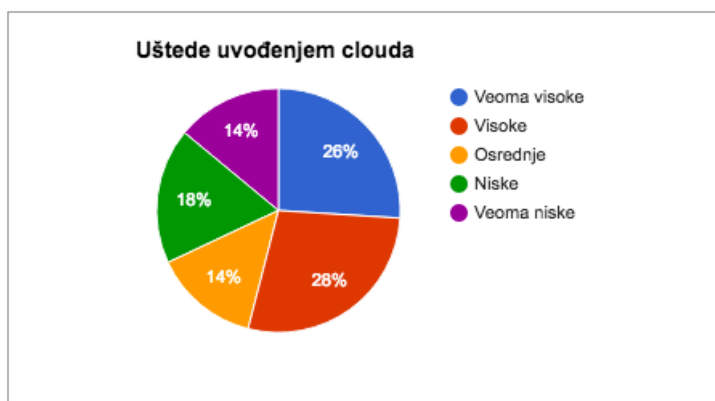
20. Slika 47 prikazuje klasifikaciju poslovnih kriterijuma na kapitalne troškove, operativne troškove i poslovnu agilnost. Većina firmi se trudi da bolje optimizuje performance i broj servera. Ove dve stvari su u čvrstoj sprezi. Ukupan napor je usmeren ka tome da je ili da će biti moguće izvršavati više različitih poslova na istoj mašini. Ukoliko se to odnosi na iznajmljene instance, većina se trudi da optimizacijom koristi ili pređe na manje instance. Treći kriterijum je skalabilnost. Čini se napor ka tome da se poveća skalabilnost, koja je u tesnoj sprezi sa elastičnošću. Skalabilnost i elastičnost u današnjem kontekstu, ako izuzmemo resurse, zavise na poslovnom planu od modela naplate a u tehnološkom od stepena automatizacije koji omogućava dobavljanje resursa i njihovo upravljanje.



Slika 47 - Ključni poslovni kriterijumi za arhitekturu sistema

Pitanje #17: Kako ocenjujete postignute uštede uvođenjem cloud tehnologija?

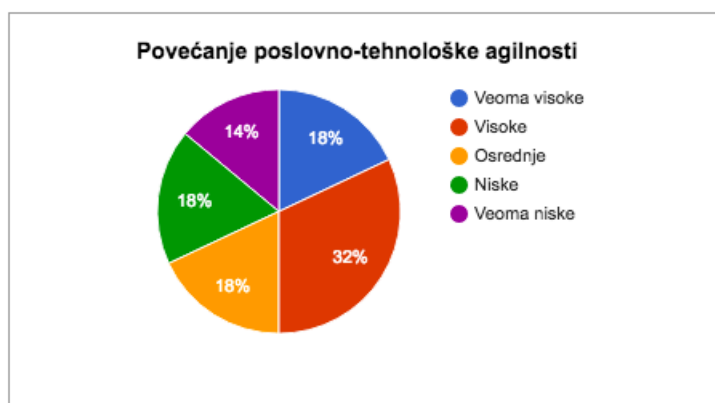
21. Većina kompanija (preko 50%) ocenjuje da su postigle neku vrstu uštede uvođenjem cloud tehnologija (slika 48).



Slika 48 - Uštede uvođenjem cloud tehnologija

Pitanje #18: Sa uvođenjem cloud tehnologija, u kojoj meri je firma u mogućnosti da brže reaguje na poslovno-tehnološku potrebu?

22. 68% kompanija vidi povećanje poslovno tehnološke agilnosti, od čega 18% veoma visoke, 32% visoke i 18% osrednje (slika 49).

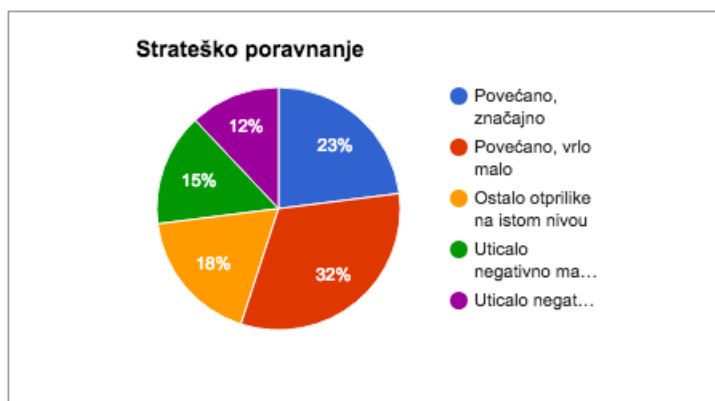


Slika 49 - Povećanje poslovno-tehnološke agilnosti

Napomena: U kontekstu povezanosti poslovne produktivnosti i konkurentnosti, vidimo [94] i [95] da postoji pozitivna povezanost između ove dve aktivnosti.

Pitanje #19: Da li je uvođenje novih, prethodno navedenih sistema, povećalo broj uspešno isporučenih projekata i ukupno poverenje u IT odeljenje?

23. 73% kompanija smatra da je strateško poravnanje ostalo isto ili bolje sa uvođenjem novih tehnologija dok 32% smatra da se povećalo vrlo malo, dok 23% smatra da se povećalo značajno (slika 50).



Slika 50 - Strateško poravnanje (isporučivanje projekata i poverenje u IT odeljenje)

12.3. *Predlog rešenja*

Ključni poslovno-tehnološki kriterijumi za odabir arhitekture cloud sistema su:

1. Optimizacija kapitalnih troškova,
2. Optimizacija operativnih troškova,
3. Poslovno-tehnološki faktori.

U nastavku diskutujemo o svakoj od navedenih oblasti.

12.3.1. Kapitalni troškovi

Smanjenje kapitalnih troškova je ključni kriterijuma gotovo svake firme. U opredeljenju za arhitekturu cloud sistema preovlađuju sledeći kriterijumi:

1. **Optimizacija broja servera:** Optimizacija broja servera se postiže na više načina. Pre svega, arhitekturom koja uključuje efikasnu upotrebu postojećih resursa i pravilnim upravljanjem servisima koji su angažovani kada su potrebni i otpušteni kada to više nisu [96]. Od značajnog uticaja je i izbor programskog jezika. Broj mogućnosti odziva servera na kome se izvršava aplikacija pisana u kompajliranom jeziku, naspram jezika koji se izvršavaju kroz interpretere (Python, Ruby) ili virtualne mašine (Java Virtual Machine) može biti osetno drugačiji. Optimizacija je istorijski išla od hardverskih servera, preko punih, rezervisanih instanci pa do kontejnera.
2. **Optimizacija snage servera:** Pod optimizacijom snage server [96] smatramo maksimalno iskorišćenje kapaciteta jedinice za obradu resursa. Pod jedinicom za obradu smatramo nekoliko kategorija: (1) fizičke mašine; (2) virtuelne mašine, (a) rezervisane i (b) deljene; (3) virtuelne instance, (a) rezervisane i (b) deljene; (4) kontejnere. Ukoliko normalizujemo gore navedena rešenja radi mogućnosti jednakog upoređivanja ovih, konceptualno raznorodnih resursa, odbijamo sledeće zajedničke karakteristike: (1) Optimizacija broja poslova po jedinici; (2) Optimizacija vremena dužine trajanja poslova po jedinici; (3) Klasifikacija prioriteta poslova; (4) Optimalan i visoko efikasan programski jezik; (5) Optimalan tehnološki stek aplikacija; (6) Potencijalna sprega sa komplementarnim tehnologijama kao što su load balanser i radi raspoređivanja poslova po mašini i drugo. Uz punu virtualizaciju, optimizacija dodeljenih resursa

je veoma laka kroz izbor parametara. Instance na javnim cloud sistemima imaju manju fleksibilnost u tom pogledu i mogu predstavljati suboptimalni izbor.

- 3. Konverzija kapitalnih u operativne troškove (CapEx u OpEx):** Umesto kapitalno intenzivnog ulaganja, moguće je iznajmljivati bilo opremu, instance ili određene usluge. Određene kompanije, umesto da kupe servere, uzmu ih na lizing [97]. Još jedan vid najma su iznajmljene bilo rezervisane, bilo deljene virtuelne instance na postojećim cloud sistemima. Dalje imamo mogućnost cloud *burstinga* u okviru hibridnih cloud rešenja. Ovoj strategiji se može pristupiti zbog više različitih razloga. Neki od njih mogu biti: radi jasnog i ujednačenog modela obračuna i naplate, radi smanjenja rizika pre ulaska u veće investicije, radi testiranja konceptualnih rešenja ili radi smanjenja visokih troškova vezanih za dokapitalizaciju postojećih resursa. Pri realizaciji ovog kriterijuma opredeljenje za iznajmljene instance u cloudu je preovlađujuće rešenje.

12.3.2. Operativni troškovi

- 1. Bolje performanse:** Performanse sistema se uglavnom mere kroz kriterijum brzine odziva i sposobnosti da se odgovori na broj upita po jedinici vremena [98]. Najbolje performanse pruža hardver [99], zatim puna rezervisana instanca a tek na kraju idu tehnologije iz javnih cloud sistema [98].
- 2. Smanjenje broja zaposlenih:** Uz pretpostavku o nepromenjenim poslovnim potrebama, potreba za brojem zaposlenih se može redukovati kroz automatizaciju poslova [100]. Uglavom se postiže korišćenjem raznih sistema i alata za automatizaciju, odabirom tehnologija čije upravljanje je moguće vršiti centralizovano, boljom organizacijom dužnosti u okviru administracije sistema.
- 3. Smanjenje kompleksnosti rada:** Pojednostavljenjem strukture rada moguće je uštedeti troškove jer su potrebne kvalifikacije za izvršenje ovakvih poslovnih zadataka praćene nižom cenom rada [100].
- 4. Povećanje analitike:** Povezivanjem različitih podsistema, a naročito sistema za nadzor, u jedinstvenu celinu dobijamo veliki broj ulaznih signala koje možemo koristiti u razne svrhe [101]. Primarna i očigledna svrha bila bi radi dalje optimizacije sistema, uočavanja uskih grla, razumevanja ključnih problema na sistemu ali i mogućnosti otkrivanja novih načina upotrebe sistema i izgradnje novih sistema i poslovnih modela [102]. Takođe, može značiti bolju

bezbednost, samoregulaciju i samopopravljanje sistema, aktiviranje mehanizama za redundantnost i visoku dostupnost.

5. **Redukcija ljudskih grešaka:** Ljudske greške se mogu desiti kroz netačnu interpretaciju zahteva, kroz grešku u sintaksi i semantici kroz proces konfiguracije sistema. Tehnologija u ovom domenu pokušava da reši problem na više nivoa [103]. Prva stvar, u ovom kontekstu, bila bi kroz centralizaciju sveukupnog upravljanja sistemom i jedinstven način izražavanja konfiguracione polise, naspram heterogenog za različite podsisteme. Ovo dalje omogućava izražavanje polise na višem nivou apstrakcije, te takođe izražavanje konfiguracione polise čini bližom i razumljivijom ljudskom načinu izražavanja. Radi realizacije prethodnog podrazumeva se automatizacija propagacije komandi kroz celokupan sistem, bez ljudske intervencije.
6. **Upravljanje na uniforman način:** Imati jedinstven interfejs ka raznorodnim tipovima resursa i njima upravljati na istovetan način [104]. Ovakav jedna primer je kada Linux koristimo i kao generalni operativni sistem i kao mrežni operativni sistem. U oba slučaja možemo koristiti isti set alata za upravljanje. Ovakav vid upravljanja utiče i na redukciju ljudskih grešaka ali i smanjenje kompleksnosti rada jer nam više nije neophodan mrežni inženjer da bismo upravljali mrežnim uređajima.

12.3.3. Poslovno-tehnološki faktori

1. **Brže građenje servisa:** U okviru kriterijuma bržeg građenja servisa, vidimo nekoliko ključnih stvari. Prva bi bila arhitektura mikroservisa [105]. Mikroarhitekture, naspram makroarhitekture (monolitne) arhitekture su znatno fleksibilnije za razvoj i održavanje i, ukoliko je ove funkcije potrebno značajnije ili u potpunosti preraditi, to je dosta lakše. U tom kontekstu prepoznamo tehnologiju kontejnera kao podesniju od monolitnih struktura. Druga bi bila viši nivo apstrakcije. Ovde dolaze do izražaja alati i programski jezici za izgradnju ovakvih sistema. Dok su jedni efikasniji i pružaju bolje performanse u izvršavanju, nisu obavezno optimalni sa aspekta izgradnje novih funkcionalnosti.
2. **Uvođenje novih tehnologija:** Ovde se postavlja ključno pitanje, koliko i kako je moguće uvesti novu tehnologiju. Puna virtualizacija i gotovo svaki Tip 1 hipervizor omogućava kompatibilnost sa veoma širokim spektrom mogućnosti eksperimentisanja sa novim tehnologijama, dok je to u slučaju kontejnera ograničeno. Pre svega, postojeće aplikacije bi morale biti prepisane i pripremljene da se izvršavaju u okviru kontejnera [106]. Ukoliko

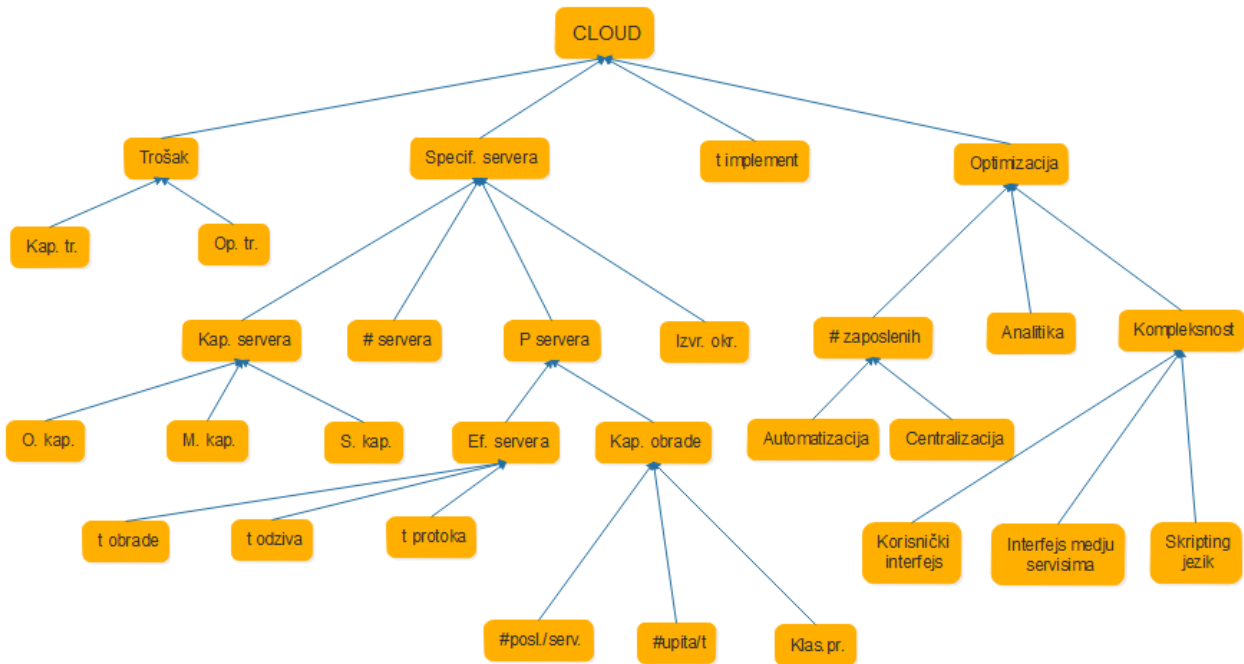
posmatramo situaciju na mrežnom nivou cloud Sistema, korišćenjem SDM pristupa omogućeno je eksperimentisanje i uvođenje novih protokola ili funkcija.

3. **Lakše održavanje:** Lakše održavanje se postiže kroz mikrosegmentaciju aplikacionih celina, definisanje jasnih interfejsa, korišćenjem standardnih protokola, alata i jezika [94]. U kontekstu opredeljenja za konkretno rešenje koje se lako održava, karakteristike takvog sistema bile bi da ima što manje zavisnih komponenti ali i da ne bude monolitna struktura. U zavisnosti od željenog cilja, odgovor može nekad biti u domenu kontejnera a nekad u domenu pune virtualizacije.
4. **Mikro segmentacija:** Mikro segmentacija sistemskih funkcionalnosti omogućava fleksibilniji razvoj i održavanje [107]. Ukoliko je pojedinačnu funkciju potrebno značajnije ili u potpunosti preraditi, to je dosta lakše. U pogledu mikrosegmentacije, virtualizacija i mašina ali i na mrežnom sloju omogućavaju širok spektar mogućnosti. Takođe, zavisno od toga šta se gradi, mikrosegmentacija može biti optimalna na nivou virtualnih mašina ili na nivou kontejnera.
5. **Skalabilnost:** Skalabilnost sistema je u direktnoj vezi za arhitekturom sistema. Sistem je poželjno projektovati tako da može da raste u skladu sa potrebama [108]. Sve prethodno opisane tehnologije to omogućavaju. Razliku čini pristup dizajnu i organizaciji resursa. Virtuelne mašine i virtuelne instance mogu da rastu (na nivou pojedinačne instance), da se konfigurišu i automatizuju i na kraju integrišu u jedinstven sistem. Isto tako to je moguće učiniti i sa kontejnerima. Kontejneri u novije vreme postaju agnostični u pogledu operativnih sistema te se mogu izvršavati, na primer, kako na Linux distribucijama, tako i na Windows familiji operativnih sistema.
6. **Elastičnost:** Porast na više i smanjenje na niže [108] u upotrebi resursa mora biti i jeste moguća takođe u okviru svih razmatranih tehnologija za izgradnju cloud sistema. U okviru potreba realizacije pojedinačnog sistema za date namene, bira se odgovarajuća tehnologija a zatim konfiguriše tako da podrži zahteve za elastičnošću servisa.
7. **Centralizacija upravljanja:** Mogućnost centralizacije se razlikuje kroz tehnologije. Teoretski sve je moguće uvezati kombinacijom sistemskih skriptova, korišćenjem postojećih alata za upravljanje i programiranjem dodatih funkcionalnosti. Sa druge strane, stepen centralizacije i integracije, u smislu prirodne podrške i jednoobraznosti, se razlikuje. Tako je jednoobrazne sisteme znatno lakše uvezati sa standardizovanim programskim interfejsima i podrškom za industrijske alate nego heterogene sisteme [109].

8. **Nezavisnost od proizvođača:** U pogledu povećanja nezavisnosti od proizvođača, tu je jasno opredeljenje za korišćenje industrijskih standarda i otvorenih tehnologija [110]. Tamo gde nedostaju, nadomešćuju se defakto proizvođačkim standardima. Međutim, kako sve više proizvođača bude podržavalo određene standarde i tehnologije, tako će mogućnost opredeljenja za datu tehnologiju biti manje limitirajuća u pogledu buduće arhitekture sistema. Na svakom nivou, tehnologija virtualizacije omogućava da, čak i kada ne postoji interoperabilnost, različita proizvođačka rešenja mogu da koegzistiraju.
9. **Mogućnost strateškog poravnanja (eng. alignment), poslovna skalabilnost i organizaciona fleksibilnost:** Pod mogućnošću strateškog poravnanja podrazumevamo mogućnost poravnanja informacionog sistema organizacije sa organizacionim procesima, ciljevima i strategijama [111]. Zbog ideje centralizovanog upravljanja uočavamo znatno manji broj komunikacionih kanala što je u direktnoj pozitivnoj korelaciji sa efikasnošću komunikacije. Visoki menadžment ima mogućnost da ostvari direktniji kontakt sa IT-IS odeljenjem te se broj mogućih tačaka greške u sistemu prenosa informacije redukuje. Oblast mogućnosti greške, kako ljudske, tako i systemske, u izvršnom delu je takođe smanjena značajno. Obzirom da se nadzor i upravljanje sistemom vrše sa jedne tačke, kompleksnost pregovora, planiranje i implementacija promena su značajno pojednostavljeni. Kembel, Kaj i Avison [111] dalje zaključuju direktnu korelaciju između uspešno isporučenih projekata od strane IT odeljenja strateškom menadžmentu i kredibiliteta odeljenja. Zatim, mogućnost pravilne implementacije poslovno-tehnoloških polisa postaje dostupnija i organizacijama sa manjim ljudsko-tehnološkim resursima zbog pojednostavljenog modela organizacione interakcije i smanjenja komunikacionih kanala. Poželjno je takođe implementirati sistem povratne sprege u smislu praćenja implementacije i mogućnosti korektivnih akcija radi postizanja željenog poslovnog rezultata (poravnanja). Takođe je moguće meriti ukupan kvalitet i zadovoljstvo novim sistemom u odnosu na prethodni. Jasno je da integrisano upravljanje otvara dosta mogućnosti u ovom smislu.

12.3.4. Ekspertni Sistem *Klaudifikator*

Kao objedinjujuć odgovor na izazove prilikom odabira tehnologija za izgradnju cloud sistema ali i na opšte i pojedinačne hipoteze, napravljen je ekspertni sistem *Klaudifikator*. Sistem je implementiran korišćenjem DEXi softvera za višekriterijumsko odlučivanje, gde svi atributi poseduju kvalitativne karakteristike. *Klaudifikator* uzima u obzir prethodno diskutovane agregirane ili ključne poslovne i tehnološke attribute (slika 51). Atributi su organizovani hijerarhijski i mogu se predstaviti putem stabla na sledeći način:



Slika 51 - Stablo atributa ekspertnog sistema

Osnovni atributi su: Kap. tr., Op. tr., O. kap., M. kap., S. kap., t obrade, t odziva, t protoka, # servera, Izvr. okr., # posl./serv., # upita/t, Klas. pr., t implement., Automatizacija, Centralizacija, Analitika, Korisnički interfejs, Interfejs među servisima i Skripting jezik. Osnovni atributi su oni atributi čije će karakteristike donosilac odluka morati da definiše da bi se dobila evaluacija modela, odnosno da bi se ocenila predložena opcija.

Agregirani atributi su: Cloud, Trošak, Specif. server, Optimizacija, Kap. server, P server, # zaposlenih, Kompleksnost, Ef. server i Kap. obrade. Agregirani atributi se procenjuju na osnovu osnovnih atributa, uz pomoć funkcije odlučivanja. Funkcija odlučivanja se definiše za svaki agregirani atribut.

Koreni atribut je CLOUD. Koreni atribut je onaj atribut čije se opcije ocenjuju. On je uvek agregiran.

Izveštaj iz DEXi softvera o strukturi atributa je predstavljen na sledeći način, kao na slici 52:

Attribute	Description
Cloud	tip Clouda kao poslovno resenje
Trošak	ukupan trošak
Kap.tr.	kapitalni trošak
Op.tr.	operativni trošak
Specif.servera	serverska specifikacija
#Servera	broj servera
Kap.servera	kapacitet servera
O.kap.	obradni kapacitet
M.kap.	mrežni kapacitet
S.kap.	skladišni kapacitet
P servera	serverska snaga
Ef.servera	efikasnost servera
t obrade	vreme obrade
t odziva	vreme odziva
t protoka	vreme protoka
Kap.obrade	kapacitet obrade
#posl.serv.	broj poslova po serveru
#upita/t	broj upita po jedinici vremena
Klas.pr.	klasifikacija prioriteta
Izvr.okr.	izvršno okruženje
Optimizacija	optimizacija poslovnih faktora
#zaposlenih	broj zaposlenih
Automatizacija	automatizacija poslova
Centralizacija	centralizovano upravljanje
Kompleksnost	kompleksnost rada
Korisnički interfejs	kompleksnost korisničkog interfejsa
Interfejs medju servisima	postojanost nterfejsa među servisima
Skripting jezik	kompleksnost skripting jezika
Analitika	kvalitet analitike
t implmnt.	vreme implementacije

Slika 52 - Struktura atributa

Objašnjenje modela: Da bi se donela odluka o izboru clouda, razmatra se ukupan trošak, specifikacija servera, optimizacija i vreme implementacije. Ukupan trošak zavisi od kapitalnog i operativnog troška. Specifikacija servera se definiše preko izvršnog okruženja, broja servera, snage servera i kapaciteta servera. Kapacitet se dalje dekomponuje na obradni, mrežni i skladišni kapacitet. Snaga servera se dekomponuje na efikasnost servera i kapacitet obrade. Efikasnost servera zavisi od vremena obrade, vremena odziva i vremena protoka, dok je kapacitet obrade razložen na broj poslova po serveru, broj upita po jedinici vremena i klasifikaciju prioriteta. Optimizacija je dekomponovana preko broja zaposlenih, kompleksnosti i analitike. Broj zaposlenih se dalje razlaže na automatizaciju i centralizaciju, dok se kompleksnost ogleda preko kompleksnosti korisničkog interfejsa, postojanosti interfejsa među servisima i kompleksnosti skripting jezika.

Na sledećoj, slici 53 je prikazana skala prema kojoj su vrednovani atributi. Kao što je ranije navedeno, atributi poseduju samo kvalitativne karakteristike i najčešće su interpretirani preko ordinalne rastuće skale. Vrednosti rastuće skale se definišu od „loših“ ka „dobrim“ vrednostima. „Loše“ vrednosti su one vrednosti koje nisu poželjne kao karakteristike opcije o kojoj se odlučuje (u našem slučaju cloud rešenja). Nasuprot njima, „dobre“ vrednosti predstavljaju prednosti opcije o kojoj se odlučuje i poželjne su od strane donosilaca odluka. U našem sistemu samo je atribut vreme implementacije definisan na opadajućoj skali. Svi atributi su definisani na skalama od po tri vrednosti (negativno, neutralno i pozitivno), osim atributa koji se agregiraju u kompleksnost, a koji su definisani na skalama od po dve vrednosti (negativno, pozitivno). Smatrano je da su te veličine skale dovoljne da se napravi razlika među karakteristikama ponuđenih opcija za cloud rešenje.

Attribute	Scale
Cloud	neprihvatljivo ; zadovoljavajuće; odlicno
Trošak	visok ; srednji; nizak
Kap.tr.	visok ; srednji; nizak
Op.tr.	visok ; srednji; nizak
Specif.servera	slab ; osrednji; odlican
#Servera	nizak ; srednji; visok
Kap.servera	nizak ; srednji; visok
O.kap.	nizak ; srednji; visok
M.kap.	nizak ; srednji; visok
S.kap.	nizak ; srednji; visok
P servera	niska ; srednja; visoka
Ef.servera	nizak ; srednji; visok
t obrade	nisko ; srednje; visoko
t odziva	nisko ; srednje; visoko
t protoka	nisko ; srednje; visoko
Kap.obrade	nizak ; srednji; visok
#posl.serv.	nizak ; srednji; visok
#upita/t	nizak ; srednji; visok
Klas.pr.	niska ; osrednja; visoka
Izvr.okr.	interpreter ; vm; kompajlirano
Optimizacija	niska ; osrednja; visoka
#zaposlenih	visok ; srednji; nizak
Automatizacija	niska ; osrednja; visoka
Centralizacija	niska ; srednja; visoka
Kompleksnost	visoka ; srednja; niska
Korisnički interfejs	komplikovan ; jednostavan
Interfejs medju servisima	ne postoji ; postoji
Skripting jezik	komplikovan ; jednostavan
Analitika	slaba ; osrednja; dobra
t implmnt.	kratko ; osrednje; dugo

Slika 53 - Model ekspertnog sistema

Da bi se ocenila određena opcija, potrebno je definisati funkcije odlučivanja (slika 53). Kao što je ranije rečeno funkcije odlučivanja se definišu za svaki agregirani atribut. U predstavljenom modelu ih ima deset, pa će tako postojati i deset funkcija odlučivanja. Za svaku funkciju su definisana osnovna pravila odlučivanja na osnovu svih mogućih vrednosti koje određeni kriterijum može da uzme. Pa tako, na primer, atribut specifikacija servera je definisan 81. pravilom, jer se agregira preko 4 atributa, od kojih svaki može da uzme po 3 vrednosti – $3^4=81$. Ukupan broj definisanih pravila u modelu je:

$$81 \text{ (Cloud)} + 9 \text{ (Trošak)} + 81 \text{ (Spec. servera)} + 27 \text{ (Kap. servera)} + 9 \text{ (P servera)} + 27 \text{ (Ef. servera)} + 27 \text{ (Kap. obrade)} + 27 \text{ (Optimizacija)} + 9 \text{ (# zaposlenih)} + 8 \text{ (Kompleksnost)} = \mathbf{305 \text{ pravila}}$$

DEXi ima mogućnost da spajanjem osnovnih pravila koja imaju istu vrednost funkcije kreira tzv. kompleksna pravila u cilju predstavljanja funkcije odlučivanja. Na sledećim slikama su prikazane funkcije odlučivanja uz pomoć kompleksnih pravila.

	Trošak	Specif.servera	Optimizacija	t implmnt.	Cloud
	28%	39%	15%	18%	
1	<=srednji	slab	*	kratko	neprihvatljivo
2	<=srednji	<=osrednji	niska	kratko	neprihvatljivo
3	*	slab	<=osrednja	kratko	neprihvatljivo
4	visok	*	*	dugo	neprihvatljivo
5	<=srednji	<=osrednji	*	dugo	neprihvatljivo
6	<=srednji	*	niska	dugo	neprihvatljivo
7	*	slab	*	dugo	neprihvatljivo
8	*	<=osrednji	niska	dugo	neprihvatljivo
9	<=srednji	slab	>=osrednja	*	neprihvatljivo
10	*	slab	osrednja	*	neprihvatljivo
11	visok	*	osrednja	>=osrednje	neprihvatljivo
12	<=srednji	<=osrednji	osrednja	>=osrednje	neprihvatljivo
13	*	slab	>=osrednja	>=osrednje	neprihvatljivo
14	<=srednji	osrednji	niska	*	neprihvatljivo
15	visok	>=osrednji	<=osrednja	>=osrednje	neprihvatljivo
16	<=srednji	osrednji	<=osrednja	>=osrednje	neprihvatljivo
17	<=srednji	>=osrednji	niska	>=osrednje	neprihvatljivo
18	srednji	slab	*	*	neprihvatljivo
19	srednji	<=osrednji	niska	*	neprihvatljivo
20	>=srednji	slab	<=osrednja	*	neprihvatljivo
21	srednji	<=osrednji	<=osrednja	>=osrednje	neprihvatljivo
22	srednji	*	niska	>=osrednje	neprihvatljivo
23	>=srednji	slab	*	>=osrednje	neprihvatljivo
24	visok	slab	niska	osrednje	zadovoljavajuce
25	<=srednji	osrednji	>=osrednja	kratko	zadovoljavajuce
26	<=srednji	>=osrednji	osrednja	kratko	zadovoljavajuce
27	<=srednji	osrednji	visoka	<=osrednje	zadovoljavajuce
28	visok	>=osrednji	visoka	osrednje	zadovoljavajuce
29	*	osrednji	visoka	osrednje	zadovoljavajuce
30	<=srednji	odlican	<=osrednja	kratko	zadovoljavajuce
31	srednji	odlican	osrednja	*	zadovoljavajuce
32	nizak	slab	visoka	kratko	zadovoljavajuce
33	nizak	osrednji	niska	osrednje	zadovoljavajuce
34	nizak	osrednji	>=osrednja	dugo	zadovoljavajuce
35	*	odlican	visoka	kratko	odlicno
36	>=srednji	odlican	visoka	*	odlicno
37	nizak	>=osrednji	*	kratko	odlicno
38	nizak	>=osrednji	osrednja	<=osrednje	odlicno
39	nizak	odlican	*	*	odlicno

Slika 54 – Funkcije odlučivanja za Cloud sa pravilima

	Kap.tr.	Op.tr.	Trošak
	83%	17%	
1	visok	*	visok
2	srednji	*	srednji
3	>=srednji	visok	srednji
4	nizak	>=srednji	nizak

Slika 55 - Funkcija odlučivanja za trošak

	#Servera 29%	Kap.servera 24%	P servera 27%	Izvr.okr. 20%	Specif.servera
1	nizak	nizak	<=srednja	*	slab
2	nizak	nizak	*	<=vm	slab
3	nizak	<=srednji	niska	*	slab
4	nizak	*	niska	<=vm	slab
5	<=srednji	nizak	niska	<=vm	slab
6	<=srednji	nizak	<=srednja	interpreter	slab
7	<=srednji	<=srednji	niska	interpreter	slab
8	*	nizak	niska	interpreter	slab
9	nizak	nizak	visoka	kompajlirano	osrednji
10	nizak	>=srednji	srednja	*	osrednji
11	nizak	>=srednji	>=srednja	<=vm	osrednji
12	<=srednji	srednji	srednja	*	osrednji
13	<=srednji	srednji	>=srednja	<=vm	osrednji
14	<=srednji	>=srednji	srednja	<=vm	osrednji
15	<=srednji	>=srednji	>=srednja	interpreter	osrednji
16	*	srednji	srednja	<=vm	osrednji
17	*	srednji	>=srednja	interpreter	osrednji
18	nizak	visok	<=srednja	kompajlirano	osrednji
19	<=srednji	visok	niska	kompajlirano	osrednji
20	srednji	<=srednji	<=srednja	kompajlirano	osrednji
21	srednji	*	niska	kompajlirano	osrednji
22	>=srednji	nizak	<=srednja	kompajlirano	osrednji
23	>=srednji	<=srednji	niska	kompajlirano	osrednji
24	srednji	<=srednji	srednja	>=vm	osrednji
25	srednji	<=srednji	>=srednja	vm	osrednji
26	srednji	*	srednja	vm	osrednji
27	>=srednji	nizak	srednja	>=vm	osrednji
28	>=srednji	<=srednji	srednja	vm	osrednji
29	srednji	<=srednji	visoka	<=vm	osrednji
30	srednji	*	visoka	interpreter	osrednji
31	>=srednji	<=srednji	visoka	interpreter	osrednji
32	srednji	srednji	<=srednja	>=vm	osrednji
33	srednji	srednji	*	vm	osrednji
34	srednji	>=srednji	niska	>=vm	osrednji
35	srednji	>=srednji	<=srednja	vm	osrednji
36	>=srednji	srednji	niska	>=vm	osrednji
37	>=srednji	srednji	<=srednja	vm	osrednji
38	srednji	visok	niska	*	osrednji
39	srednji	visok	<=srednja	<=vm	osrednji
40	srednji	visok	*	interpreter	osrednji
41	>=srednji	visok	niska	interpreter	osrednji
42	visok	nizak	<=srednja	>=vm	osrednji
43	visok	<=srednji	niska	>=vm	osrednji
44	visok	<=srednji	<=srednja	vm	osrednji
45	visok	nizak	srednja	*	osrednji
46	visok	<=srednji	srednja	<=vm	osrednji
47	visok	<=srednji	>=srednja	interpreter	osrednji
48	visok	srednji	niska	*	osrednji
49	visok	srednji	<=srednja	<=vm	osrednji
50	visok	srednji	*	interpreter	osrednji
51	visok	>=srednji	niska	interpreter	osrednji
52	*	>=srednji	visoka	kompajlirano	odlican
53	>=srednji	*	visoka	kompajlirano	odlican
54	>=srednji	visok	>=srednja	kompajlirano	odlican
55	>=srednji	visok	visoka	>=vm	odlican
56	visok	*	visoka	>=vm	odlican
57	visok	>=srednji	>=srednja	kompajlirano	odlican
58	visok	visok	*	>=vm	odlican
59	visok	visok	>=srednja	*	odlican

Slika 56 - Funkcija odlučivanja za specifikaciju servera

	#Servera	Kap.servera	P servera	Izvr.okr.	Specif.servera
	29%	24%	27%	20%	
1	nizak	nizak	<=srednja	*	slab
2	nizak	nizak	*	<=vm	slab
3	nizak	<=srednji	niska	*	slab
4	nizak	*	niska	<=vm	slab
5	<=srednji	nizak	niska	<=vm	slab
6	<=srednji	nizak	<=srednja	interpreter	slab
7	<=srednji	<=srednji	niska	interpreter	slab
8	*	nizak	niska	interpreter	slab
9	nizak	nizak	visoka	kompajlirano	osrednji
10	nizak	>=srednji	srednja	*	osrednji
11	nizak	>=srednji	>=srednja	<=vm	osrednji
12	<=srednji	srednji	srednja	*	osrednji
13	<=srednji	srednji	>=srednja	<=vm	osrednji
14	<=srednji	>=srednji	srednja	<=vm	osrednji
15	<=srednji	>=srednji	>=srednja	interpreter	osrednji
16	*	srednji	srednja	<=vm	osrednji
17	*	srednji	>=srednja	interpreter	osrednji
18	nizak	visok	<=srednja	kompajlirano	osrednji
19	<=srednji	visok	niska	kompajlirano	osrednji
20	srednji	<=srednji	<=srednja	kompajlirano	osrednji
21	srednji	*	niska	kompajlirano	osrednji
22	>=srednji	nizak	<=srednja	kompajlirano	osrednji
23	>=srednji	<=srednji	niska	kompajlirano	osrednji
24	srednji	<=srednji	srednja	>=vm	osrednji
25	srednji	<=srednji	>=srednja	vm	osrednji
26	srednji	*	srednja	vm	osrednji
27	>=srednji	nizak	srednja	>=vm	osrednji
28	>=srednji	<=srednji	srednja	vm	osrednji
29	srednji	<=srednji	visoka	<=vm	osrednji
30	srednji	*	visoka	interpreter	osrednji
31	>=srednji	<=srednji	visoka	interpreter	osrednji
32	srednji	srednji	<=srednja	>=vm	osrednji
33	srednji	srednji	*	vm	osrednji
34	srednji	>=srednji	niska	>=vm	osrednji
35	srednji	>=srednji	<=srednja	vm	osrednji
36	>=srednji	srednji	niska	>=vm	osrednji
37	>=srednji	srednji	<=srednja	vm	osrednji
38	srednji	visok	niska	*	osrednji
39	srednji	visok	<=srednja	<=vm	osrednji
40	srednji	visok	*	interpreter	osrednji
41	>=srednji	visok	niska	interpreter	osrednji
42	visok	nizak	<=srednja	>=vm	osrednji
43	visok	<=srednji	niska	>=vm	osrednji
44	visok	<=srednji	<=srednja	vm	osrednji
45	visok	nizak	srednja	*	osrednji
46	visok	<=srednji	srednja	<=vm	osrednji
47	visok	<=srednji	>=srednja	interpreter	osrednji
48	visok	srednji	niska	*	osrednji
49	visok	srednji	<=srednja	<=vm	osrednji
50	visok	srednji	*	interpreter	osrednji
51	visok	>=srednji	niska	interpreter	osrednji
52	*	>=srednji	visoka	kompajlirano	odlican
53	>=srednji	*	visoka	kompajlirano	odlican
54	>=srednji	visok	>=srednja	kompajlirano	odlican
55	>=srednji	visok	visoka	>=vm	odlican
56	visok	*	visoka	>=vm	odlican
57	visok	>=srednji	>=srednja	kompajlirano	odlican
58	visok	visok	*	>=vm	odlican
59	visok	visok	>=srednja	*	odlican

Slika 57 - Funkcija odlučivanja za kapacitet, snagu i efikasnost servera

	#posl.serv.	#upita/t	Klas.pr.	Kap.obrade
	39%	43%	17%	
1	nizak	nizak	*	nizak
2	nizak	<=srednji	niska	nizak
3	<=srednji	nizak	<=osrednja	nizak
4	nizak	>=srednji	>=osrednja	srednji
5	<=srednji	srednji	>=osrednja	srednji
6	nizak	visok	*	srednji
7	<=srednji	visok	niska	srednji
8	srednji	<=srednji	visoka	srednji
9	>=srednji	nizak	visoka	srednji
10	srednji	srednji	*	srednji
11	srednji	>=srednji	niska	srednji
12	>=srednji	srednji	niska	srednji
13	visok	nizak	*	srednji
14	visok	<=srednji	niska	srednji
15	>=srednji	visok	>=osrednja	visok
16	visok	>=srednji	>=osrednja	visok
17	visok	visok	*	visok

	#zaposlenih	Kompleksnost	Analitika	Optimizacija
	33%	33%	33%	
1	visok	visoka	*	niska
2	visok	*	slaba	niska
3	*	visoka	slaba	niska
4	<=srednji	srednja	>=osrednja	osrednja
5	<=srednji	>=srednja	osrednja	osrednja
6	*	srednja	osrednja	osrednja
7	srednji	<=srednja	>=osrednja	osrednja
8	srednji	*	osrednja	osrednja
9	>=srednji	<=srednja	osrednja	osrednja
10	srednji	srednja	*	osrednja
11	srednji	>=srednja	<=osrednja	osrednja
12	>=srednji	srednja	<=osrednja	osrednja
13	*	niska	dobra	visoka
14	nizak	*	dobra	visoka
15	nizak	niska	*	visoka

	Automatizacija	Centralizacija	#zaposlenih
	57%	43%	
1	niska	<=srednja	visok
2	niska	visoka	srednji
3	osrednja	<=srednja	srednji
4	>=osrednja	niska	srednji
5	>=osrednja	visoka	nizak
6	visoka	>=srednja	nizak

	Korisnički interfejs	Interfejs medju servisima	Skripting jezik	Kompleksnost
	60%	20%	20%	
1	komplikovan	ne postoji	*	visoka
2	komplikovan	*	komplikovan	visoka
3	komplikovan	postoji	jednostavan	srednja
4	jednostavan	ne postoji	komplikovan	srednja
5	jednostavan	*	jednostavan	niska
6	jednostavan	postoji	*	niska

Slika 58 – Pravila za osnovne atribute

Legenda: Znak “*” na slikama 54 do 58 predstavlja bilo koju vrednost, pa tako ako je Kapitalni trošak visok, bez obzira na vrednost operativnog troška, ukupan trošak biće visok. Procenti u prvom redu svake od tabela predstavljaju težinske vrednosti atributa, odnosno npr. korisnički interfejs je tri puta važniji (ponder 0.6 na prema 0.2) za definisanje kompleksnosti u odnosu na skripting jezik. Kod optimizacije je svaki atribut podjednako značajan (0.33).

Uočava se da je broj kompleksnih pravila ukupno jednak 180, što je značajno manje od broja 305.

12.3.5. Evaluacija – studija slučaja

Opisani ekspertni sistem je definisan da bi se procenilo koji tip clouda treba da se uvede u poslovni sistem. Odnosno da li je bolje iznajmiti cloud (opcija 1), izgraditi cloud (opcija 2) ili da se pronade neko hibridno rešenje (opcija 3). Sledeća slika predstavlja evaluaciju datih opcija na osnovu unetih karakteristika svake od opcija. Unesene su samo karakteristike za osnovne atribute, dok su agregirani atributi procenjeni automatski na osnovu funkcije odlučivanja.

Attribute	Iznajmiti CLOUD	Izgraditi CLOUD	Hibridno resenje
Cloud	<i>odlicno</i>	zadovoljavajuće	neprihvatljivo
Trošak	visok	srednji	srednji
Kap.tr.	visok	<i>nizak</i>	srednji
Op.tr.	<i>nizak</i>	visok	srednji
Specif.servera	<i>odlican</i>	<i>odlican</i>	osrednji
#Servera	visok	visok	srednji
Kap.servera	visok	visok	srednji
O.kap.	visok	srednji	nizak
M.kap.	srednji	visok	srednji
S.kap.	srednji	visok	srednji
P servera	niska	srednja	srednja
Ef.servera	nizak	srednji	srednji
t obrade	nisko	srednje	nisko
t odziva	<i>visoko</i>	<i>visoko</i>	srednje
t protoka	nisko	srednje	<i>visoko</i>
Kap.obrade	srednji	srednji	nizak
#posl.serv.	nizak	nizak	nizak
#upita/t	srednji	visok	srednji
Klas.pr.	osrednja	<i>visoka</i>	niska
Izvr.okr.	<i>kompajlirano</i>	vm	vm
Optimizacija	<i>visoka</i>	osrednja	niska
#zaposlenih	<i>nizak</i>	srednji	srednji
Automatizacija	osrednja	osrednja	osrednja
Centralizacija	<i>visoka</i>	srednja	niska
Kompleksnost	niska	niska	visoka
Korisnički interfejs	<i>jednostavan</i>	<i>jednostavan</i>	komplikovan
Interfejs medju servisima	ne postoji	<i>postoji</i>	<i>postoji</i>
Skripting jezik	<i>jednostavan</i>	<i>jednostavan</i>	komplikovan
Analitika	osrednja	osrednja	slaba
t implmnt.	<i>kratko</i>	dugo	osrednje

Slika 59 - Rezultat evaluacije

Sa slike 59 vidimo da je koreni atribut (Cloud) kod opcije 1 procenjen „odlično“, kod opcije 2 „zadovoljavajuće“, dok je kod opcije 3 „neprihvatljiv“. To znači da je prema kriterijumima koje je definisao donosilac odluka i njihovoj važnosti, procenjeno da je za taj konkretan poslovni sistem (iz koga dolazi donosilac odluke) najbolje da iznajme cloud. Prihvatljiva je i opcija izgradnje cloud sistema, dok je hibridno rešenje neprihvatljivo za dati poslovni sistem. Sledeće slike prikazuju jake i slabe aspekte svake od procenjenih opcija.

Weak points

Attribute	Iznajmiti CLOUD
—Trošak	visok
—Kap.tr.	visok
—P servera	niska
—Ef.servera	nizak
—t obrade	nisko
—t protoka	nisko
—#posl.serv.	nizak
—Interfejs medju servisima	ne postoji

Strong points

Attribute	Iznajmiti CLOUD
Cloud	odlicno
—Specif.servera	odlican
—#Servera	visok
—Kap.servera	visok
—O.kap.	visok
—Izvr.okr.	kompajlirano
—Optimizacija	visoka
—#zaposlenih	nizak
—Centralizacija	visoka
—Kompleksnost	niska
—Korisnički interfejs	jednostavan
—Skripting jezik	jednostavan
—t implmnt.	kratko
—Op.tr.	nizak
—t odziva	visoko

Slika 60 - Aspekti opcije 1 (jaki i slabi)

Kao što se uočava sa slike 61 opcija 1 ima visok obradni kapacitet, koji, ako se pogleda slika 58, ima važnost od 0.5 pri određivanju kapaciteta servera. Izvršno okruženje je kompajlirano i uz visok broj servera koji poseduje opcija 1 čini da specifikacija servera bude procenjena kao odlična. Pozitivni aspekti opcije 1 su jednostavan korisnički interfejs i skripting jezik, što čini nisku kompleksnost sistema, a uz nizak broj zaposlenih optimizacija opcije 1 je visoko procenjena. Vreme implementacije kod opcije 1 je kratko pa i ono ide na ruku pozitivnoj stranu te opcije. Ukupan trošak spada u slab aspekt opcije 1, i to prvenstveno zbog toga što je kapitalni trošak visok (ranije je pokazano da on nosi 0.83 važnosti). Međutim iako je trošak visok (što je negativno), uz ostale karakteristike koje su sve pozitivne (vreme implementacije, specifikacija servera i optimizacija), čini da se opcija 1 proceni kao „odlična“.

Weak points

Attribute	Izgraditi CLOUD
└ Op.tr.	visok
└ └ #posl. serv.	nizak
└ implmnt.	dugo

Attribute	Izgraditi CLOUD
└ Kap.tr.	nizak
└ Specif.servera	odlican
└ └ #Servera	visok
└ └ Kap.servera	visok
└ └ └ M.kap.	visok
└ └ └ S.kap.	visok
└ └ └ └ t odziva	visoko
└ └ └ └ #upita/t	visok
└ └ └ └ Klas.pr.	visoka
└ Kompleksnost	niska
└ └ Korisnički interfejs	jednostavan
└ └ Interfejs medju servisima	postoji
└ └ Skripting jezik	jednostavan

Slika 61 - Aspekti opcije 2 (jaki i slabi)

Opcija 2 takođe ima odličnu specifikaciju servera i nisku kompleksnost kao i opcija 1, ali je zbog dugog vremena implementacije (i uz osrednje troškove) ipak procenjena samo kao zadovoljavajuća.

Weak points

Attribute	Hibridno resenje
Cloud	neprihvatljivo
└ Optimizacija	niska
└ └ Kompleksnost	visoka
└ └ └ Korisnički interfejs	komplikovan
└ └ └ Skripting jezik	komplikovan
└ Analitika	slaba
└ └ O.kap.	nizak
└ └ └ obrade	nisko
└ └ Kap.obrade	nizak
└ └ └ #posl. serv.	nizak
└ └ └ Klas.pr.	niska
└ Centralizacija	niska

Attribute	Hibridno resenje
└ └ └ protoka	visoko
└ └ Interfejs medju servisima	postoji

Slika 62 - Aspekti opcije 3 (jaki i slabi)

Opcija 3 nema nijedan atribut većeg nivoa koji bi predstavljao jaču stranu te opcije. Sa slike 63 se vidi da je i od atributa većeg nivoa koji predstavljaju slabe strane, samo optimizacija niska. To sve skupa, uz ostale osrednje karakteristike daje neprihvatljivo rešenje.

13. Zaključak

13.1. Cilj rada

Dakle, cilj rada je da: 1) prikaže kritički uporedni pregled aktuelnih pristupa uvođenju cloud rešenja u svakodnevno poslovanje preduzeća, 2) analizira aktuelne tehnologije po rešenjima i sloju kom pripadaju i da 3) predloži ekspertski sistem koji bi poslužio u odlučivanju za gradivne komponente i izbor cloud rešenja, za date kriterijume.

Deo rada koji se odnosi na pregled ima za cilj da olakša donošenje kvalitetne poslovne odluke vezane za uvođenje cloud tehnologija. U okviru pregleda, rad primarno razmatra SPI model (eng. *Software, Platform, Infrastructure*).

Računarski centri današnjice poznaju različite vrste gradivnih blokova čiji je često zajednički imenitelj tehnologija virtualizacije. Dva domena na koje se rad fokusira u ovom smislu jesu server (operativni sistem) i računarska mreža. Primenjeno na sloj servera, tehnologija omogućava apstrakciju fizičkih obradnih resursa i nalazi se u zreloj tehnološkoj fazi, dok donedavno, uglavnom tradicionalni mrežni sloj doživljava transformaciju sa integrisanog pristupa na razdvojene ravni upravljanja i prosleđivanja podataka uz sve viši stepen programabilnosti i sa mogućnošću integrisanja sa ostalim slojevima. Ove paradigme promena na mrežnom sloju prepoznajemo kroz termine koji još uvek konvergiraju kroz industriju: (1) virtualizacija mrežnih funkcija i (2) softverski definisane mreže.

Tehnologije virtualizacije omogućavaju znatno viši stepen iskorišćenja fizičkih resursa, kao i elastičnost resursa koji se dobavljaju na zahtev korisnika cloud sistema. Pored toga što omogućavaju dobavljanje na zahtev, cloud sistemi menjaju iskustvo interakcije korisnika koji popunjava web forme ili koristi API i vrši dobavljanje svih vrsta traženih resursa bez ljudske interakcije.

Ovakav način upotrebe resursa omogućava vrstu optimizacije kakva ranije nije bila moguća. Osnova ovih mogućnosti je u transformaciji serverske infrastrukture u virtualne mašine zahvaljujući čemu se fizički

resursi mnogo bolje optimizuju, migriraju i transformišu da odgovore na poslovne potrebe. Takođe, na mrežnom sloju, imamo nove mogućnosti koje se odnose na migraciju, automatizaciju i upravljanje mrežnim resursima, kao i mogućnost prevazilaženja određenih tehnoloških prepreka poznatih tradicionalnom okruženju.

Na ovaj način, počevši od najnižeg tehnološkog sloja, imamo konsolidaciju infrastrukture na nivou mrežnog sloja a zatim i virtualnih mašina na serverskom, to jest infrastrukturnom sloju, koje vrši obradu podataka. Za ovaj sloj vezujemo pojam infrastrukture na zahtev gde je ključna mogućnost dobavljanja i upravljanja slojem operativnog sistema na zahtev, IaaS. Sloj iznad, nalazi se sloj platforme na zahtev PaaS. Na ovom sloju korisnik gubi kontrolu nad operativnim sistemom ali dobija kontrolu nad platformom kao celinom. Konačno, najviši sloj apstrakcije cloud sistema podrazumeva dobavljanje softvera kao usluge, SaaS.

Bilo da se radi o infrastrukturi, platformi ili softveru koji se dobavlja kao usluga, ovakve sisteme odlikuju ranije nepostojeća i/ili bitno poboljšani kvaliteti poput elastičnosti i skalabilnosti. U novije vreme, sve više dolaze do izražaja mogućnosti horizontalne integracije sa javnim cloud sistemima.

Cloud sistemi pružaju mogućnost postojanja dobavljača usluge, *SP*, koji kao sloj ispod, ima uslugu infrastrukture ili platforme na zahtev. Pored ovog koncepta, formira se čitava svita novih koncepata, kako organizacije računarskih resursa, tako i poslovnih modela. S tim u vezi, postaje izazov u poslovnom odlučivanju da li i u kojoj meri graditi i oslanjati se na sopstvenu infrastrukturu naspram najma računarskih resursa kao servisa.

Rad razmatra pravce razvoja, šanse i niše za poslovne poduhvate i budući razvoj ovakvih sistema.

13.2. *Predmet istraživanja*

Predmet istraživanja predstavlja izrada kritičkog uporednog prikaza aktuelnih pristupa uvođenju cloud rešenja u preduzeće. Na osnovu pregleda tehnologija za virtuelizaciju i raspoloživih rešenja, u radu su istraženi i obrađeni relevantni kriterijumi za upoređivanje postojećih rešenja i donošenje odluke o izboru. Na osnovu identifikovanih kriterijuma, vrši se komparacija tri reprezentativna rešenja za primenu cloud tehnologija u svakodnevno poslovanje preduzeća.

U radu se daje i pregled izbora nekoliko vodećih softverski definisanih rešenja (SDM) u odnosu na tradicionalna, koja u ovom slučaju smatramo normalizovanim i ne ulazimo u konkretne proizvođačke specifičnosti.

13.3. *Cilj istraživanja*

Osnovni cilj primene informacionih tehnologija jeste odgovor na poslovnu potrebu, praćenu povećanjem produktivnosti, efikasnosti i efektivnosti odvijanja poslovnih procesa. Važna stvar u dostizanju ovog cilja jeste izbor odgovarajućeg rešenja informacionog sistema koji pruža podršku poslovnim funkcijama na svim nivoima, od operativnog do strateškog.

Imajući u vidu da su poslednjih godina aktuelna rešenja zasnovana na korišćenju cloud tehnologija, cilj ovog istraživanja jeste kritički uporedni pregled aktuelnih pristupa uvođenju cloud rešenja u svakodnevno poslovanje preduzeća. Ovakav pregled ima za cilj da olakša donošenje kvalitetne poslovne odluke o uvođenju cloud tehnologija. Glavni fokus rada se odnosi na SPI modelu ali i ostalim cloud tehnologijama. Rad se još bavi i tehnologijama serverske i strane virtualizacije, kao osnovom za cloud sisteme i pregledom aktuelnih industrijskih rešenja od značaja. U radu su prikazani i mogući pravci razvoja, te i šanse i niše za poslovne poduhvate kroz budući razvoj cloud sistema.

13.4. Metode i hipoteze

U radu koristimo opšte i posebne naučne metode kao ispod.

- Od opštih metoda:
 - Sistemski pristup,
 - Analitičko-sintetičku metodu, kao postupak kojim se prilazi suštini problema, na način da se obrade međusobno uslovljeni odnosi, a potom i problem u celini,
 - Deduktivno-logička metoda pomoću koje će se, na temelju spoznaje utvrđene praksom, utemeljiti vlastiti teorijski stavovi,
 - Komparativna metoda.
- Od posebnih metoda:
 - Anketni upitnik za utvrđivanje uspešnosti primene cloud rešenja i ostvarenja konkurentskih prednosti u poslovanju u kompanijama,
 - Statističke metode radi obrade podataka i prezentacije rezultata

Kroz rad ispitujemo sledeće hipoteze:

24. Opštu: Polazna hipoteza je da pri sadašnjem stanju tržišnih ponuda cloud tehnologija, ne postoji jedno optimalno rešenje za sve poslovne sisteme, već da se za konkretni poslovni sistem može iz datog skupa cloud sistema naći bar jedno rešenje koje je bolje od ostalih.

25. Pojedinačne:

H1 - Postoji pozitivna povezanost između produktivnosti - konkurentnosti preduzeća i nivoa primene cloud tehnologija.

H2 - Postoji mogućnost federacije različitih resursa – privatnih sa javnim ili drugim privatnim resursima.

H3 - Postoji mogućnost izgradnje rešenja u slojevima – aplikativni servisi na platformi.

H4 - Informacionim sistemima zasnovanim na cloud rešenjima postiže se ušteda u resursima u odnosu na tradicionalni pristup realizaciji infrastrukture.

H5 - Informacionim sistemima zasnovanim na cloud rešenjima postiže se veća poslovna agilnost u izgradnji sistema u odnosu na tradicionalni pristup realizaciji infrastrukture.

H6 - Opređenjem za cloud rešenja može se postići bolja skalabilnost sistema, odnosno jednostavnija alokacija ili dealokacija potrebnih resursa u odnosu na klasični pristup realizaciji informacione infrastrukture.

H7 - Opredeljenjem za virtualizaciju na mrežnom sloju postiže se viši stepen automatizacije i bolje upravljanje celokupnim sistemom u odnosu na klasični pristup realizaciji informacione infrastrukture.

13.5. Razmatranje hipoteza

Na opštu hipotezu smo odgovorili kroz anketu i analizu tehnologija. Takođe, kroz definiciju funkcija odlučivanja u okviru ekspertnog sistema *Klaudifikator* smo takođe potvrdili opštu hipotezu. Vidimo da često postoji više rešenja, zatim više dobrih rešenja i, u zavisnosti od kriterijuma, jedno je je uvek optimalno u odnosu na ostala. Nemoguće je naći jedno univerzalno rešenje jer je proces odlučivanja složen i sadrži velik broj faktora čija parametrizacija ima direktan uticaj na odlučivanje. Kompletan izgrađen ekspertni sistem je u funkciji dokaza opšte hipoteze, kao i anketa koju prati statistička obrada podataka u SPSS-u, kao i dalja obrada i tumačenje dobijenih rezultata u okviru ovog rada. U nastavku vršimo razmatranje pojedinačnih hipoteza.

Cloud tehnologije u okviru trougla: produktivnost-konkurentnost-nivo_primene_cloud_tehnologija imaju zajedničku konotaciju i pružaju fleksibilnost poslovnim entitetu da skalira resurse na vrlo efikasan način bez kapitalnih ulaganja te, što kroz anketu, što kroz ekspertni sistem, potvrđujemo ovu korelaciju u pozitivnom smislu.

Kroz pregled cloud ponuđača i pregled njihovih ponuda vidimo da postoje, kako mogućnosti, tako i slučajevi federacije resursa, gde se kroz aplikativne programske interfejske inicijalizuju ili raspuštaju potrebni resursi u traženim količinama. Ovo je često korisno u, na primer, slučajevima već postojećim private infrastrukture gde je potrebno obaviti, bilo kratkoročan posao, bilo za kratko vreme obezbediti dodatne obradne resurse. Ekspertni sistem i analiza tržišta cloud ponuđača adresiraju ovu hipotezu.

U prvom preglednom delu rada, gde dajemo pregled rešenja, osvrćemo se na SPI model. Ovakav model podrazumeva slojevitost arhitekture sa fokusima na infrastrukturu, platformu ili aplikaciju.

Ekspertni sistem nam pomaže da odgovorimo na ovu hipotezu ali i obrada ankete. Kludifikator uzima u obzir faktor kapitalnih, odnosno operativnih ulaganja u okviru optimizacije odlučivanja vezanog za izbor cloud rešenja. Anketa kaže da preko 50% kompanija ocenjuje da su postigle neku vrstu uštede uvođenjem cloud tehnologija.

Kroz anketu vidimo da 68% kompanija vidi povećanje poslovno tehnološke agilnosti, od čega 18% veoma visoke, 32% visoke i 18% osrednje. Kludifikator takođe uzima faktor agilnosti u razmatranje.

Opređenjem za cloud rešenja može se postići bolja skalabilnost Sistema, odnosno jednostavnija alokacija ili dealokacija potrebnih resursa u odnosu na klasični pristup realizaciji informacione infrastrukture.

Kroz anketu vidimo da se većina firmi trudi da bolje optimizuje performanse i broj servera. Ove dve stvari su u čvrstoj sprezi. Ukupan napor je usmeren ka tome da jeste ili bude moguće izvršavati više različitih poslova na istoj mašini. Ukoliko se to odnosi na iznajmljene instance, većina se trudi da ih optimizacijom koristi ili pređe na manje instance. Treći kriterijum je skalabilnost. Čini se napor ka tome da se poveća skalabilnost, koja je u tesnoj sprezi sa elastičnošću. Skalabilnost i elastičnost u današnjem kontekstu, ako izuzmemo resurse, zavise na poslovnom planu od modela naplate a u tehnološkom od stepena automatizacije koji omogućava dobavljanje resursa i njihovo upravljanje.

Opređenjem za virtualizaciju na mrežnom sloju postiže se viši stepen automatizacije i bolje upravljanje celokupnim sistemom u odnosu na klasični pristup realizaciji informacione infrastrukture.

Kroz anketu vidimo da je većina kompanija koje su uvele tehnologije mrežnog sloja ostvarilo viši stepen automatizacije na nivou od 31-60%.

U nastavku rada sumiraćemo rezime teze i zaključke.

13.6. Rezime disertacije

U ovom odeljku nalazi se sažeta teza sa zaključcima.

13.6.1. Cloud

U radu prvo razmatramo definiciju, koncept i početke cloud računarstva iz više uglova.

Počeci cloud računarstva se vezuju za Likliderovu viziju intergalaktičke računarske mreže dok je radio za ARPANET. Vizija je podrazumevala pristup aplikacijama sa bilo kog mesta, bilo kom mestu. Ovu ideju dalje nastavlja i unapređuje Džon Makarti, uvodeći koncept servisnog (eng. *utility*) računarstva.

Konstatujemo da su ključne tehnologije za razvoj cloud sistema bile web 2.0 tehnologije za isporuku do korisnika, kao i razne vrste tehnologija virtualizacije ali i sve brži razvoj mrežnih propusnih opsega i, na kraju, razvoj univerzalnih softverskih standarda interoperabilnosti.

ENISA definiše cloud računarstvo [35]: „Cloud računarstvo je „*on-demand*“ model usluge za IT snabdevanje koji se najčešće zasniva na virtualizaciji i distribuiranim računarskim tehnologijama. Cloud računarska arhitektura poseduje: (1) visoko abstraktne resurse, (2) skoro trenutnu skalabilnost i fleksibilnost, (3) skoro trenutno rezervisanje resursa, (4) zajedničke resurse (hardver, bazu podataka, memoriju...), (5) uslugu na zahtev koju najčešće prati “plati prema korišćenju” sistem plaćanja, (6) programsko upravljanje putem aplikativnog programskog interfejsa (API)“.

S obzirom na prepoznavanje tehnologija virtualizacije kao jednog od osnovnog gradivnog bloka cloud sistema, rad se u nastavku fokusira na tehnologije virtualizacije i komparativnu analizu tri rešenja.

Neke od ključnih prednosti cloud računarstva, kao servisnog računarstva su: (1) veća skalabilnost, (2) brži pristup infrastrukturi, (3) veća dostupnost servisa, (4) brža mogućnost izlaska na tržišta, (5) povećana efikasnost IT osoblja, (6) mogućnost bolje i šire geografske pokrivenosti servisa, (7) ušteda u ukupnim troškovima (uz bolji balans između kapitalnih i operativnih troškova), (8) mogućnost boljeg obezbeđenja poslovnog kontinuiteta, (9) ukupno više performanse celokupnog sistema.

Cloud računarstvo je isporuka računarskih servisa, nasuprot proizvodu, gde se deljeni resursi, softver i informacije isporučuju računarima i drugim uređajima u vidu merljivih servisa putem računarskih mreža, obično Interneta.

Cloud je apstrakcija računarskih resursa u toj meri da korisnik brine isključivo o usluzi koju dobavlja i o interfejsima koji mu služe u tu svrhu. Cloud računarstvo uključuje obradu podataka, softver, pristup podacima i skladišne resurse, bez potrebe da korisnik sistema zna i vodi računa o fizičkoj lokaciji podataka ili drugim detaljima računarskih resursa korišćenih u procesu. Krajnji korisnici pristupaju cloud aplikacijama kroz veb-čitač ili kroz mobilnu aplikaciju napisanu za neku od mobilnih platformi, kao što su telefon i tablet, dok su poslovni softver i podaci skladišteni na serverima koji se nalaze na udaljenoj lokaciji.

Konstatujemo da se cloud računarstvo bazira na konceptima konvergirane infrastructure i deljenih resursa, gde pod konvergiranom infrastrukturom podrazumevamo grupisanje resursa po tipu (na primer: server, skladišni prostor, mrežni resursi, i drugo). Ovaj tip obradnog okruženja računarskog centra omogućava poslovnim entitetima da svoje aplikacije pokreću brzo i jednostavno uz jednostavnije upravljanje i manje održavanja, i omogućava ICT službama da svoje resurse prilagođavaju rapidnom brzinom, uz jednostavniju manipulaciju i upravljanje istim. Postaje veoma lako dobiti potrebne skladišne, mrežne i računarske resurse kao i skoro jednako lako da se odgovori na fluktuacije i nepredvidivosti u tražnji za ovim resursima.

Cloud koncept servisi su pravljani sa idejom da zadovolje princip višestrukog zakupa aplikacije/servisa (eng. *Multitenancy*). Ovaj princip se zasniva na konceptu višekorisničkog zakupa jedne instance aplikacije nasuprot konceptu *multi-instance* arhitekture gde se kreira posebna instanca za svakog korisnika aplikacije/usluge.

Cloud računarstvo ne menja ove stvari, međutim daje mogućnost dobavljačima SaaS usluga da bez formiranja sopstvenih datacentara nude ovakve servise, baš kao što je to slučaj sa ugovornom proizvodnjom u oblasti proizvodnje poluprovodničkih komponenti. Cloud računarstvo omogućava postavljanje SaaS i skaliranje na zahtev, bez da se gradi ili dobavlja datacenter. Analogno, kako korisnik može izvesnu kategoriju problema da prebaci na dobavljača SaaS usluga, tako SaaS dobavljač može da prebaci neke od svojih problema na cloud dobavljača.

Sa hardverske tačke gledišta, postoji tri aspekta cloud računarstva [38]: (1) iluzija o beskonačnim računarskim resursima raspoloživim na zahtev, te na taj način eliminiše se potreba za korisnika cloud usluga da planira unapred resurse; (2) eliminacija potrebe za prethodnim obavezivanjem cloud korisnika, što omogućava korisnicima da počnu kao veoma mali te da kasnije povećavaju računarske resurse u skladu sa povećanjem obima potreba; (3) sposobnost da se plate upotrebljeni računarski resursi za vrlo male vremenske jedinice (obrada podataka na sat i skladište podataka na dan), čuvajući resurse oslobađanjem istih kada nisu potrebni.

U okviru tradicionalnih datacentara, takođe je moguće alocirati i osloboditi resurse, međutim ne na automatizovan način tako da se korisnik samooposluži.

Uvođenje clouda ne znači automatski da će aplikacije, servisi i procesi automatski biti migrirani na cloud rešenja. Mnoge kompanije su vrlo obazrive i analiziraju detaljno svoje strateške poslovne procese i intelektualnu svojinu kako bi utvrdile koja tehnološka infrastruktura treba da ostane pod kompanijskim direktnim nadzorom a koja će biti preseljena u cloud rešenje.

Cloud servis je opšti termin za bilo šta što uključuje isporuku hostovanih servisa putem Interneta. Ovi servisi su, šire gledano, podeljeni u tri osnovne kategorije, u okviru SPI modela: (1) Softver kao servis (Software as a service, **SaaS**) - Klijentu se pruža mogućnost korišćenja dobavljačevih aplikacija koje se izvršavaju na cloud infrastrukturi, (2) Platforma kao servis (Platform as a service, **PaaS**) - skup softvera i razvojnih alata hostovanih na infrastrukturi dobavljača usluge. Razvojni timovi razvijaju aplikacije na platformi koja se fizički nalazi u posedu dobavljača usluge. Za pristup ovakvoj infrastrukturi koristi se Internet, (3) Infrastruktura kao servis (Infrastructure as a service, **IaaS**) - pruža mogućnost da se pokrenu, zaustave, pristupi, konfiguriraju virtualni serveri i sistemi skladištenja podataka

Cloud može biti javni ili privatni. Javni cloud poseduje resurse koji su javno dostupni svima. Privatni cloud označava mrežne servise ili datacentar hostovanih servisa u privatnom vlasništvu koji pruža usluge ograničenom broju ljudi. Kada isporučilac usluga koristi resurse javnog clouda da napravi svoj privatni cloud, rezultat se zove virtualni privatni cloud. Privatni ili javni, cilj cloud računarstva je da pruži jednostavan, skalabilan pristup računarskim resursima i ICT uslugama.

Osnovne karakteristike cloud sistema: (1) Samousluživanje na zahtev – Krajnji korisnik može jednostrano da obezbedi računarske usluge, poput serverskog obradnog vremena ili mrežnog sistema skladištenja podataka, i to po potrebi, a bez ljudske interakcije sa dobavljačem usluga, (2) Širok mrežni pristup – Mogućnosti su na raspolaganju kroz mrežu i pristupa im se kroz standardne mehanizme koji promovišu upotrebu od strane heterogenih punih ili tankih klijentskih platformi, (3) Udruživanje resursa – Računarski resursi dobavljača usluga se udružuju u službi upotrebe u višeklijentskom okruženju, sa različitim fizičkim i virtualnim resursima dinamički dodeljenim i raspoređivanim u skladu sa korisničkim zahtevima, (4) Rapidna elastičnost – Mogućnosti mogu biti elastično obezbeđene i oslobođene, u mnogim slučajevima automatski, sa skaliranjem ka spolja ili unutra, proporcionalno tražnji. Iz korisničke perspektive, resursi često deluju neograničeno i mogu biti prisvojeni u bilo kom obimu i vremenu, (5) merena usluga – cloud sistemi vrše merenje upotrebe resursa a zatim i automatsku kontrolu i optimizaciju korišćenja istih resursa.

13.6.2. Ekonomija cloud računarstva

Rad razmatra nekoliko scenarija odlučivanja da li se držanje servisa u cloudu isplati dugoročno gledano. Čini se da fina granularnost ekonomskog modela u okviru cloud ponude čini ovo odlučivanje fluidnijim. Tako, na primer, osobina elastičnosti cloud sistema omogućava da se prelazak izvede uz prenos rizika.

Takođe, iako cena hardverskih resursa ima tendenciju stalnog pada, to se takođe dešava u nepravilnom rasporedu. Na primer, cena obrade i cena skladišta podataka brže padaju od cene WAN konekcija. Cloud računarstvo može da prati ove promene i potencijalno ih prosledi do klijenta efikasnije u odnosu na samostalni centar za obradu podataka koji bi taj isti klijent napravio. Ovo rezultuje bližoj ceni stvarnih troškova upotrebe tih resursa.

Prilikom odlučivanja za pomeranje servisa u cloud, korisno je utvrditi očekivan prosečni skok korišćenja resursa, posebno ako aplikacija može imati veoma varijabilne trendove na strani tražnje. Limiti kapaciteta su određeni karakteristikama instalirane opreme, kao i različitim operacionim troškovima koji variraju i zavise od tipa razmatranog cloud okruženja.

Iako se ekonomska privlačnost cloud računarstva često opisuje kao pretvaranje kapitalnih u operativne troškove, čini se da je “plati na osnovu korišćenja” dosta precizniji način da se opiše dobrobit za korisnika cloud sistema.

Tako, iako na primer, ukupna cena najma cloud sistema može biti viša od kupovine i upotrebe uporedivog servera za isti period, izbor prevaže prevaže u korist cloud koncepta koji nudi ekonomske prednosti iz kategorije elastičnosti sistema i transfera rizika, naročito rizika od prevelike ili nedovoljne alokacije resursa.

Rad dalje obrađuje primer odlučivanja za obradu podataka jedne laboratoriju lokalno naspram obrade u cloud sistemu, uz transfer podataka do cloud sistema. S tim u vezi zaključujemo iz datog primera:

- Moguće je u okviru cloud sistema angažovati kratkoročno znatno više resursa za dati posao i time ga brže završiti. U datom primeru je vreme obrade smanjeno 50 puta.
- Obrada i prenos podataka bili bi skuplji u poređenju sa već postojećim rešenjem koje razmatrana laboratorija ima na raspolaganju.
- Vreme transfera podataka do laboratorije je (marginalno) duže (50 naspram 55 sati).
- Zaključak laboratorije je da je cena pojedinačnog eksperimenta vršena na cloud infrastrukturi viša u odnosu na korišćenje sopstvene infrastrukture, gde je vreme izvršenja nešto duže na cloudu ali uz napomenu da, uz povećano angažovanje resursa može biti znatno kraće, umesto 50 sati, može se obraditi za 1 sat. Ukoliko je kriterijum odlučivanja finansijske prirode, izbor bi ostao na korišćenju sopstvene infrastrukture, dok bi kriterijum brzine izvršavanja u prvi plan stavio cloud kao izbor.

Dalje u radu diskutujemo o problemima i šansama za cloud računarstvo poput: (1) dostupnosti usluga, (2) zaključavanja podataka, (3) privatnosti podataka i mogućnosti revizije, (4) uskih grla prenosa podataka, (5) nepredvidivosti performansi, (6) skalabilnosti skladištenja podataka, (7) grešaka u velikim distribuiranim sistemima, (8) brzog skaliranja, (9) deljenja reputacije i (10) licenciranja.

13.6.3. Virtualizacija kao osnova cloud sistema

Uobičajen cilj virtualizacije je centralizacija administrativnih zadataka uz poboljšanje skalabilnosti i opšte iskorišćenosti hardverskih resursa.

U konkretnom smislu virtualizacija se pojavljuje kao odgovor na neadekvatnu iskorišćenost hardverskih resursa u oba smera. To se odnosi pre svega na klasične scenarije u kojima se planira hardver za određenu namenu i, s tim u vezi jednom kapacitira i dobavi pa ostaje nepromenjen do kraja svog životnog veka, uz minimalne ili nikakve mogućnosti proširivanja. Mogućnosti promene elastičnosti ovakvog resursa se, gotovo isključivo, odnose jednosmerno i to u smeru proširivanja. Ukoliko se ispostavi da je dobavljeni resurs veći u odnosu na realne potrebe ili je prosto vremenom to postao, sav neupotrebljen potencijal hardvera ostaje neiskorišćen. Takođe, proširivanje hardverskog resursa je moguće samo u okvirima proizvođačkih specifikacija i poseduje faktor vremenske oročenosti u smislu podrške do kraja života proizvoda. Često je nakon izvesnog vremena teško, skupo ili nemoguće izvršiti nadogradnja hardverske konfiguracije.

Pored gore navedenog, postoji i problem kompanijskog *contingency* i *business continuity* plana. *Contingency* plan podrazumeva izvršavanje procedure u slučaju nepredviđenih situacija u poslovanju, te raznih otkaza resursa i servisa. *Business continuity* plan podrazumeva dostupnost, zatim visoku dostupnost kao i skalabilnost u kontekstu obezbeđenja kontinuiteta poslovanja i mogućnosti njegovog proširivanja.

Razmatran odgovor na navedene potrebe je virtualizacija infrastrukture. Rad u nastavku diskutuje vrste virtualizacija, a u potom razmatra primere virtualizacionih scenarija poput: (1) izvršavanje jedne ili više aplikacija koje nisu podržane od strane host operativnog sistema, (2) evaluacija alternativnog operativnog sistema, (3) višestruki serveri bi mogli da se izvršavaju na jednoj fizičkoj platformi, kako bi efikasnije koristili raspoložive hardverske resurse. (4) duplikacija specifičnih okruženja: Virtualna mašina bi mogla, zavisno od toga koji se virtualizacioni softver koristi, da bude duplicirana na više hostova, ili da bude ponovo uspostavljena (eng. *restored*) pod pretpostavkom da je prethodno izvršen *backup* iste virtualne mašine, (5) formiranje zaštićenog okruženja: ukoliko se guest operativni sistem koji se izvršava na virtualnoj mašini ošteti na način koji je teško popraviti, primera radi takav slučaj može nastati prilikom izučavanja ponašanja zlonamernog koda (eng. *Malware*) ili instalacijom softvera koji kompromituje stabilnost sistema iz bilo kog razloga, virtualna mašina se može odbaciti bez negativnih posledica po host sistem, te se može naredni put koristiti nova, istovetna kopija (*image* ili *template*) ove virtualne mašine.

Rad potom u nastavku nudi pregled sledećih vrste virtualizacije: 1) hardverska virtualizaciju, (2) puna virtualizacija, (3) delimična virtualizacija, (4) paravirtualizacija, (5) desktop virtualizacija, (6) softverska virtualizacija, (7) virtualizacija memorije, (8) virtualizacija sistema za skladištenje podataka, (9) virtualizacija u oblasti računarskih mreža, (10) virtualizacija podataka.

Kao zaključak na temu serverske virtualizacije, rad razmatra tri rešenja serverskog hipervizora tipa 1: vSphere, xen i hyper-v. U prvom delu, koncentrišemo se na dve kategorije poslovnih kriterijuma evaluacije:

1. Kriterijum sposobnosti izvršavanja - nije vezan isključivo za funkcionalnosti proizvoda, već u velikoj meri za održavanje konstantno promenljivog poslovnog modela sa veoma dinamičnim trendom:
 - a. Proizvod i usluga - Ključni faktori koji se evaluiraju uključuju: opseg operativnih sistema, skalabilnost i efikasnost, elastičnost, zrelost, upravljanje *embedded* resursima, upravljanje opcijama za efikasniju administraciju, mogućnost administriranja sistema kao celine, virtualizovanih ekosistema, administrativna skalabilnost i integracija se drugim dobavljačima usluge upravljanja i održavanja ovakvih sistema.
 - b. Opšta vitalnost - Procena celokupne finansijske vitalnosti organizacije, finansijski i praktičan uspeh poslovnih jedinica i verovatnoća individualnih poslovnih jedinica kako bi se nastavilo sa investicijom u proizvod.
 - c. Realizovane prodaje - Ovo uključuje upravljanje sklapanjem poslova, cenovnom politika i pregovaranjem, pretprodajnom podrškom i celokupnom efektivnošću prodajnih kanala.
 - d. Tržišna responzivnost i praćenje - Sposobnost odgovora, promene pravca, fleksibilnosti i konkurentnosti sa razvojem mogućnosti, konkurentnog delovanja, evolucije korisničkih zahteva i tržišne dinamičke promene.
 - e. Marketing - Jasnost, kvalitet, kreativnost i efikasnost programa dizajniranih da isporuče poruku koju organizacija želi da iskomunicira i na taj način utiče na tržište, promocija brenda i posla, povećanje svesti vezane za proizvod, i uspostavljanje pozitivne identifikacije sa proizvodom i/ili brendom od strane kupaca.
 - f. Korisničko iskustvo - Ovo uključuje načine na koji korisnici primaju tehničku i ostalu podršku. Ovo takođe može uključivati pomoćne alate, programe korisničke podrške (te i kvalitet), dostupnost korisničkih grupa, ugovore servisne podrške, itd.

- g. Operacije - Sposobnost organizacije da ispuni svoje ciljeve i obaveze. Faktori uključuju kvalitet organizacione strukture poput veština, iskustava, programa, sistema i drugih elemenata potrebnih organizaciji kako bi efektivno i efikasno nastavila da funkcioniše.
2. Kriterijum kompletnosti vizije - Evaluirani su tehnološki dobavljači po osnovu svoje sposobnosti da ubedljivo artikulišu logičke izjave povodom trenutnih i budućih tržišnih pravaca, inovacija, korisničkih potreba i kompetitivnih snaga.
- a. Razumevanje tržišta - Sposobnost tehnološkog dobavljača da razume potrebu svojih klijenata i da ih prevedu proizvode i usluge.
 - b. Marketing strategija - jasan i diferenciran skup poruka konzistentno komuniciran kroz organizaciju i slat kroz veb-sajt, reklame, korisničke programe i izjave za pozicioniranje.
 - c. Prodajna strategija - Strategija prodaje proizvoda koji koriste odgovarajuću mrežu direktne i indirektno prodaje, marketinga, usluga i komunikacionih posrednika koji šire okvire doseganja tržišta, veština, ekspertize, tehnologije, usluga i baze korisnika.
 - d. Ponuda (proizvodne) strategije - Pristup tehnološkog dobavljača razvoju proizvoda i isporuci koja uključuje strategiju diferencijacije, funkcionalnosti, metodologije i skupa funkcionalnosti, kako se mapiraju na trenutne i buduće zahteve.
 - e. Biznis model - Čvrstoća i logika fundamentalne logike poslovne ponude tehnološkog dobavljača.
 - f. Vertikalna ili industrijska strategija - Strategija tehnološkog dobavljača da upravlja resursima, veštinama i ponudama kako bi se odgovorilo na specifične potrebe ili pojedinačne tržišne segmente, uključujući vertikalne.
 - g. Inovacija - Direktni, povezani, komplementarni i sinergijski raspored resursa, ekspertiza ili kapitalna investicija, konsolidacija, odbrambena ili preventivna svrha.
 - h. Geografska strategija - Strategija tehnološkog dobavljača da upravlja resursima, veštinama i ponudama izvan "kuće" ili svoje matične geografije, bilo direktno ili preko partnera, kanala i predstavništava na odgovarajući način u kontekstu geografije i tržišta.

Rad dalje razmatra konkurentne prednosti svakog od proizvođača i vrši razmatranje taksonomije snaga i slabosti ova tri proizvođačka rešenja.

Na kraju odeljka o virtualizaciji, rad vrši uporedni pregled tehnoloških kriterijuma tri hipervizora: (1) zauzeće prostora na disku, (2) nezavisnost od operativnog sistema, (3) drajveri, (4) napredno upravljanje memorijom, (5) napredno upravljanje skladištem, (6) visoka I/O skalabilnost, (7) upravljanje resursima hostova, (8) unapređenje performansi, (9) tehnologija bezbednosti u okviru virtualne infrastrukture, (10) fleksibilna alokacija resursa, (11) kreiranje i upravljanje prilagođenom slikom (eng. *image file*), (12) automatsko puštanje u produkciju, auto *deployment*, (13) upravljanje fajervol interfejsom, (14) unapređeni virtualni hardver.

Na osnovu razmatranja poslovnih i tehnoloških kriterijuma, u radu se zaključuje da VMware još uvek ima najkompletniju ponudu a da ostala dva proizvođača tek treba da razviju alate, pre svega iz oblasti menadžmenta, skalabilnosti i poslovnog kontinuiteta.

13.6.4. Kontejneri

Bez obzira na implementaciono rešenje, rešenja u oblasti Linux kontejnera se vrlo malo međusobno razlikuju i uglavnom su to unapređenja osnovne verzije kontejnera (eng. *Linux Containers*, LXC).

Zajedničko za sve tehnologije kontejnera su: (1) izolaciju procesa i (2) kontrola resursa. Pod izolacijom podrazumevamo mogućnost da izvršavanje poslova na jednom podsistemu ne može da utiče na izvršavanje nekog drugog podsistema u okviru istog sistema, na primer u okviru istog operativnog sistema.

Kontrola resursa se odnosi na sposobnost zadržavanja resursa podsistema u okviru željenog i unapred definisanog skupa resursa.

Linux kontejneri omogućavaju virtualizaciju na nivou operativnog sistema. Ovakav sistem omogućava izvršavanje višestrukih izolovanih korisničkih domena na jednom sistemskom domenu (na nivou jednog kernela). Virtualizacija sistema na ovom nivou značajno je manje resursno zahtevna od pune virtualizacije i nudi dosta veliku fleksibilnost. Takođe, kontejneri se mogu izvršavati na svakom operativnom sistemu koji ih podržava.

Kompletno upravljanje resursima radi kernel, kao uslugu kontejnerima. Primeri upravljanja resursima su upravljanje memorijom, procesorskim vremenom, sistemskim prekidima i drugim resursima.

U okviru kontejnera rad razmatra LXC, Docker i Rokit uz napomenu da je Docker danas defakto standard u industriji.

13.6.5. Softverski definisane mreže kao osnova cloud sistema

Računarske mreže se sastoje od mrežnih elemenata koji obavljaju različite funkcije. Primeri nekih od tih funkcija su funkcija sviča, rutera, fajervola, itd. Po funkcionalnosti, računarske mreže delimo na dve ravni: ravan upravljanja i ravan (prosleđivanja) podataka. Kontrolna ravan ima ulogu odlučivanja u pogledu saobraćaja i smatra se *mozgom* mreže. Ravan podataka ima izvršnu funkciju prosleđivanja saobraćaja na osnovu instrukcija dobijenih od kontrolne ravni [76].

U tradicionalnim računarskim mrežama, ravan kontrole i ravan podataka čine nerazdvojivu celinu i obično se nalaze u mrežnom elementu nekog proizvođača. Mrežni elementi su nezavisni, distribuirani i konvergiraju ka određenom stanju, u zavisnosti od informacija koje razmenjuju. U SDM arhitekturi, ravan kontrole i ravan podataka su razdvojene i postoji centralni uvid u mrežne resurse. U tradicionalnim mrežama, mrežni elementi imaju u sebe ugrađene mnoge mrežne protokole, koji su u često, zatvorena i patentirana kompanijska rešenja.

Osnovna definicija SDM-a bila bi separacija kontrolne ravni i ravni prosleđivanja podataka uz programabilnost funkcija. U tradicionalnim mrežama, ove dve ravni su vertikalno integrisane u uređaj poput rutera ili sviča i obično su proizvođačka rešenja zaštićena patentom.

SDM je pristup računarskim mrežama koji podrazumeva sledeće: (1) razdvajanje ravni prosleđivanja podataka od ravni kontrole, (2) definiciju interakcija između prethodno navedenih ravni kroz jasno definisane apstrakcije (aplikativni programski interfejs, API), (3) logički centralizovan menadžment, (4) programabilnost, (5) otvoreni standard.

Ravan podataka je kolekcija jednostavnih hardverskih elemenata za prosleđivanje podataka koja obavlja funkcije prosleđivanja na osnovu odluka ravni kontrole. Neke od ovih odluka su prosledi, ispusti, prosledi kontroleru i prepisi zaglavlje. Kontrolna ravan donosi odluke vezane za saobraćaj i daje nalog ravni prosleđivanja podataka (hardveru) kroz južni API interfejs, najčešće OpenFlow [16]. Kontrolna ravan ima pregled kompletne mrežne topologije u okviru svog domena, kao i kompletnu kontrolu nad višestrukim elementima ravni prosleđivanja podataka i njihovim stanjima. Ova kontrolna logika se nalazi u SDM kontroleru, koji se naziva još i mrežnim operativnim sistemom.

Zavisno od konkretnog dizajna arhitekture, može biti jedan ili više kontrolera. U slučaju više kontrolera, oni mogu da uspostave *peer-to-peer* komunikaciju kroz istočno-zapadni API. Kontroler omogućava usluge SDM aplikacijama preko severnog API-ja.

Neki od ključnih izazova u tradicionalnim računarskim mrežama su: (1) značajan broj korisnika ima potrebnu za manjim brojem mogućnosti koje su dostupne na uređajima, (2) ovako vertikalno integrisane opcije i funkcije čine da mrežni uređaji imaju višu cenu nego što bi mogli imati, (3) put za bilo kakvu promenu protokola je dugotrajan, često meren u godinama, čineći na ovaj način inovaciju veoma sporom i neefikasnom, (4) eksperimentisanje sa novim protokolima je teško ili čak nemoguće, (5) svaka promena postojećih funkcija u sistemu mora da prođe kroz proceduru prihvatanja specifičnog proizvođača i može da bude neprihvaćena, da implementacija traje (znatno) duže od očekivane ili željene, ne odgovori na potrebe i zahteve u potpunosti, ne odgovori kvalitetom ili nekim drugim željenim kriterijumom, (6) konfiguracioni interfejsi mogu da variraju, a obično i variraju u zavisnosti od proizvođača a često i (linije) proizvoda, (7) upravljanje ne-programabilnim mrežama je decentralizovano, stvarajući značajne izazove u automatizaciji, čineći operativne troškove visokim i sam proces podložnim greškama, (8) budući da je Internet kritična infrastruktura sveta, u čije je razvoj uloženo mnogo sredstava i opreme u velikim razmerama, otežan je razvoj i njegova evolucija. Ključni razlog za sporu evoluciju je potreba za usaglašavanjem između raznih organizacija, poput standardizacionih tela. Ova pojava naziva se okoštavanjem Interneta [2].

Kao odgovor na gore pomenute izazove, ideja programabilnih mreža se pojavila. Koncept softverski definisanih mreža (*Software Defined Networking*, SDN) pojavio se sa željom da se inovacija u oblasti računarskih mreža odvija većom brzinom i uz isto iskustvo razvoja kao i u ostatku softverske industrije.

U prvom delu smo govorili o ranim programabilnim mrežama i objasnili neke ključne primere koji su obeležili razvoj ove oblasti podeljene u podoblasti kao što su: programabilna ravan podataka, razdvajanje ravni kontrole od ravni podataka, odnosno hardvera, i na kraju mrežnu virtualizaciju.

U radu su razmatrana samo prva dva jer su od ključnog značaja za razvoj oblasti. Iako je bilo pokušaja za otvaranjem, generalizacijom i korišćenjem *node* operativnih sistema, ovaj pristup, nije doživeo značajniji razvoj. Klasifikaciju istorije programabilnih mreža pre SDM-a vršimo u sledećim kategorijama i sa sledećim primerima: (1) programabilnost ravni podataka: dodatak programabilnosti u tradicionalnim računarskim mrežama i aktivne mreže: (i) Xbind, (ii) ANTS, (iii) SwitchWare, (iv) NetScript; (2) razdvajanje ravni kontrole od ravni podataka: razdvajanje dve ravni kontrole od ravni podataka i pristup otvorenih signala: (i) GSMP, (ii) Tempest, (iii) ForCES, (iv) RCP, (v) SoftRouter; (3) mrežna virtualizacija: abstrakcija mrežnih resursa kroz dodatak sloja mrežne virtualizacije i mrežnog overleja (eng. *network overlay*): (i) Tempest, (ii) MBone, (iii) RON, (iv) GENI, (v) VINI.

U narednom delu diskutujemo terminologiju i definiciju softverski definisanih mreža. Razmatramo šta je to SDM i kako se razlikuje ili dopunjuje u odnosu na koncepte poput virtualizacije mrežnih funkcija (*Network Function Virtualization*, NFV) i mrežna virtualizacija (*Network Virtualization*, NV).

Iza toga razmatramo pregled rešenja u kategorijama: programabilnost ravni podataka, razdvajanje kontrolne ravni od ravni podataka, mrežna virtualizacija (mrežni hipervizori) i završavamo sekciju sa SDM kontrolerima i mrežnim operativnim sistemima: (1) programabilnost ravni podataka: (i) OpenFlow, (2) razdvajanje ravni kontrole od ravni prosleđivanja podataka, (i) SANE, (ii) ETHANE; Mrežna virtualizacija (mrežni hipervizori): (i) Open vSwitch, (ii) FlowVisor, (ii) NVP; (3) SDM kontroleri: (i) NOX, (ii) ONIX, (iii) Yanc; (4) Mrežni operativni sistemi: (i) Cumulus Linux, (ii) Open Network Linux, (iii) Switch Light.

U poslednjem odeljku ove sekcije diskutujemo potencijalne poslovne i tehnološke dobrobiti, uticaj i promene koje mogu ili se već dešavaju sa uplivom raznih SDM rešenja.

13.6.6. Mrežni svič operativni sistemi

Postoji rastući broj operativnih sistema za svičeve baziranih na Linux-u koji se mogu instalirati na poseban tip mrežnih uređaja koje nazivamo *bare metal* svičevima ili *whitebox* svičevima. Ovakvi uređaji se razlikuju od klasičnih, proizvođačkih, po tome što je na njih moguće instalirati više različitih operativnih sistema.

Iako u povoj, raste broj ovih sistema pa anticipiramo nastavak pozitivnog trenda i sve veći uticaj ovakvih rešenja.

Od rešenja razmatramo sledeća: (1) Cumulus Linux, (2) Open Network Linux, (3) Switch Light. Zaključujemo: (1) da se većina rešenja u oblastima bazira na *debian* distribucijama, (2) da je arhitektura troslojna: (i) hardver, (ii) operativni sistem baziran na Linux-u i (iii) aplikacije, (3) da je upravljanje mrežnim resursima potencijalno moguće na sličan ili isti sistem kao sa serverima, (4) da zbog generalizacije hardvera kroz operativni sistem i dobre podrške drajverima, postoji dobra podrška različitim arhitekturama i uređajima, (5) da postoji dobra podrška za hardver otvorenog dizajna, (6) da većina rešenja omogućava centralizovano upravljanje firmverom.

13.6.7. Mogućnosti i pravci na mrežnom sloju

Rani rad na programabilnim mrežama se fokusirao na pristup aktivnih mreža, koji je imao dosta praktičnih izazova. Enkapsulacija mrežnih programa i njihovo procesiranje na aktivnim čvorovima nije doživela razvoj. Iako su mreže postale programabilne, način kako se promene puštaju u produkciju, alati za razvoj i ceo pristup nije adresirao fundamentalne potrebe istraživača u oblasti mreža i industrije. Drugi, pristup otvorenih mrežnih signala razmatrao je razdvajanje ravni kontrole i ravni prosleđivanja podataka. Ovaj pristup je naišao na bolji odziv i dokazao se kao više primenjiv za potrebe pojednostavljenog upravljanja i izazova povećanja kompleksnosti. Dalje, programabilne apstrakcije su bile ključni izvor za omogućavanje daljeg napretka koji je pratila ekspanzija mrežnih kontrolera i platformi. Ovo je omogućilo dalje apstrakcije, kako za one koji upravljaju mrežama, tako i za one koji projektuju nove arhitekture.

Softverski definisane mreže nisu jedno rešenje. To je pristup arhitekturi izgradnje računarskih mreža na modularan način i uz punu programabilnost. Još jedna vrlo važan aspekt evaluacije računarskih mreža je inicijativa da se koristi hardver opšteg tipa koji je i jeftiniji. Veruje se da će ovakva promena na hardverskoj strani otvoriti put ka većem rastu i širem usvajanju novog pristupa ravni podataka u računarskim centrima budućnosti, ali i u manjim firmama. SDM nije ekskluzivno arhitektura za velike.

Dalje diskutujemo pravce razvoja i šanse za razvoj savremenih SDM rešenja od kojih biznis može potencijalno da ima koristi. Rad detaljno razrađuje svaku od sledećih poslovno-tehnoloških mogućnosti koje SDM potencijalno nosi.

Tabela 13 - Potencijal i uticaj SDM-a na poslovne operativne modele

Poslovni	Tehnološki	
	Razvoj i Istraživanje	Strateški i Operativni
<ul style="list-style-type: none"> ▪ Porast konkurencije ▪ Ugovoreno održavanje mreža ▪ Povećana agilnost ▪ Redukovana cena uređaja ▪ Produžen životni ciklus opreme ▪ Pojednostavljenje poslova mrežnih operatera 	<ul style="list-style-type: none"> ▪ Povećana brzina inovacije u mrežama ▪ Povećana brzina inovacije uređaja ▪ Omogućeno eksperimentisanje ▪ Porast analitike 	<ul style="list-style-type: none"> ▪ Otvoren pristup tehnologiji ▪ Povećan izbor dizajna mrežne arhitekture ▪ Dobavljanje mrežnih resursa kao usluge ▪ Centralizacija upravljanja mrežama ▪ Centralizacija upravljana kompletnom infrastrukturom ▪ Smanjen faktor ljudske greške ▪ Pojednostavljenje poslova mrežnih operatera ▪ Upravljanje isključivo serverima ▪ Razdvojene mrežne funkcionalnosti ▪ Povećane performanse prosleđivanja paketa

13.6.8. Pristup izbora rešenja pri izgradnji cloud sistema

U okviru rada, kao jedna od naučnih metoda u okviru istraživačkog rada korišćena je anketa. U prilogu A se nalaze detalji i upitnik. Rad u nastavku sadrži analizu rezultata ankete nad ispitivanih 100 visokotehnoloških firmi u SPSS-u u okviru priloga B.

Na kraju rada, sumiramo sva znanja kroz predlog rešenja. Ključni poslovno-tehnološki kriterijumi za odabir arhitekture cloud sistema su:

1. Optimizacija kapitalnih troškova: (i) optimizacija broja servera, (ii) optimizacija snage servera, (iii) konverzija kapitalnih u operativne troškove,
2. Optimizacija operativnih troškova: (i) bolje performanse, (ii) smanjenje broja zaposlenih, (iii) smanjenje kompleksnosti rada, (iv) povećanje analitike, (v) redukcija ljudskih grešaka, (vi) upravljanje na uniforman način,
3. Poslovno-tehnološki faktori: (i) brže građenje servisa, (ii) uvođenje novih tehnologija, (iii) lakše održavanje, (iv) mikro segmentacija, (v) skalabilnost, (vi) elastičnost, (vii) centralizacija upravljanja, (viii) nezavisnost od proizvođača, (ix) mogućnost strateškog poravnanja, poslovna skalabilnost i organizaciona skalabilnost.

Rad u nastavku diskutuje svaku od navedenih oblasti koje služe kao deo modela ekspertnog sistema *Klaudifikator*. Rad objašnjava logiku rada sistema, funkcije odlučivanja, pravila sistema i, na kraju, evaluaciju slučaja odabira tehnologije na osnovu zadatih kriterijuma. Sistem se može videti u okviru priloga C, datog u elektronskom obliku (fajl).

13.7. *Doprinos i budući pravci istraživanja*

Doprinosi rada su sledeći:

- Pregled istorije, vrste, koncepta, razvoja i tehnoloških trendova cloud sistema, SPI modela i servisnog računarstva
- Kreiranje modela za odlučivanje u evaluaciji prelaska na cloud rešenje (razumevanje ekonomije, principa i vršenje cost-benefit analize)
- Klasifikacija izazova i poslovnih potencijala u okviru cloud Sistema
- Pregled tehnoloških oblasti: virtualizacije, Linux kontejnera, tip 1 hipervizora i softverski definisanih mreža
- Kreiranje taksonomije i definisanje kriterijuma odlučivanja u okviru upoređivanja virtualizacionih tehnologija sa naglaskom na tip 1 hipervizore
- Kreiranje taksonomije i definisanje poslovnih i tehnoloških kriterijuma za odlučivanje u vezi sa izgradnjom cloud sistema
- Izgradnja ekspertskog sistema za podršku u odlučivanju prelaska na cloud po osnovu prethodno kreirane taksonomije kriterijuma

Mogućnosti za unapređenja i budući pravci istraživanja:

- Unapređenja sistema za podršku u odlučivanja *Klaudifikator* su zaista velike i možemo ih podeliti u dve kategorije: kvalitativne i kvantitativne. U okviru kvantitativnih mogućnosti za poboljšanje treba navesti činjenicu da *Klaudifikator* ima 305 pravila te je moguće proširiti sistem dodatnim atributima i funkcijama odlučivanja.
- Što se kvalitativnih mogućnosti poboljšavanja sistema *Klaudifikator* tiče, svi atributi su definisani na skalama od po tri vrednosti (negativno, neutralno i pozitivno), osim atributa koji se agregiraju u kompleksnost, koji su definisani na skalama od po dve vrednosti (negativno, pozitivno). Smatrano je da su te veličine skale dovoljne da se napravi razlika među karakteristikama ponuđenih opcija za cloud rešenje. Ovako nešto ne mora biti dovoljno za sistem koji bi morao da reprezentuje dosta složenije vidove odlučivanja. Stoga smatramo da su mogućnosti u okviru definisanja skale u domenu prilagođavanja sistema određenim realnim potrebama. Skale za attribute bi možda bilo dobro definisati u okviru Likertove skale radi bolje obrade i/ili reprezentacije izlaznih podataka sistema a kao deo bolje interakcije računar-čovek.
- Povodom mogućnosti razvoja tehnologija virtualizacije na serverskom i mrežnom sloju, pojedinačna poglavlja daju detaljan uvid u izazove ali i šanse u daljem razvoju svake od oblasti.

14.Citirani Radovi

- [1] C. H. Museum, “John McCarthy Biography.” 2016.
- [2] D. Taylor and J. Turner, “Towards a diversified internet,” *White Pap. Novemb.*, 2004.
- [3] A. Lazar, “Project xbind,” in *Gigabit Networking Workshop*, 1995.
- [4] D. J. Wetherall, J. V Guttag, and D. L. Tennenhouse, “ANTS: A toolkit for building and dynamically deploying network protocols,” in *Open Architectures and Network Programming, 1998 IEEE*, 1998, pp. 117–129.
- [5] D. S. Alexander *et al.*, “The SwitchWare active network architecture,” *IEEE Netw.*, vol. 12, no. 3, pp. 29–36, 1998.
- [6] S. Da Silva, Y. Yemini, and D. Florissi, “The NetScript active network system,” *Sel. Areas Commun. IEEE J.*, vol. 19, no. 3, pp. 538–551, 2001.
- [7] A. Doria, F. Hellstrand, K. Sundell, and T. Worster, “General switch management protocol (GSMP) V3,” 2002.
- [8] J. E. der Merwe, S. Rooney, I. Leslie, and S. Crosby, “The Tempest-a practical framework for network programmability,” *Network, IEEE*, vol. 12, no. 3, pp. 20–28, 1998.
- [9] L. Yang, R. Dantu, T. Anderson, and R. Gopal, “Forwarding and control element separation (ForCES) framework,” 2004.
- [10] M. Caesar, D. Caldwell, N. Feamster, J. Rexford, A. Shaikh, and J. van der Merwe, “Design and implementation of a routing control platform,” in *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2*, 2005, pp. 15–28.
- [11] T. V Lakshman, T. Nandagopal, R. Ramjee, K. Sabnani, and T. Woo, “The softrouter architecture,” in *Proc. ACM SIGCOMM Workshop on Hot Topics in Networking*, 2004, vol. 2004.
- [12] M. R. Macedonia and D. P. Brutzman, “MBone provides audio and video across the Internet,” *Computer (Long. Beach. Calif.)*, vol. 27, no. 4, pp. 30–36, 1994.
- [13] R. Hauben, “The Information Processing Techniques Office and the Birth of the Internet: A Study in Governance.” 2012.
- [14] GENI, “Global environment for networking innovations (GENI).” 2006.
- [15] A. Bavier, N. Feamster, M. Huang, L. Peterson, and J. Rexford, “In VINI veritas: realistic and controlled network experimentation,” in *ACM SIGCOMM Computer Communication Review*, 2006, vol. 36, no. 4, pp. 3–14.
- [16] N. McKeown *et al.*, “OpenFlow: enabling innovation in campus networks,” *ACM SIGCOMM*

- Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, 2008.
- [17] M. Casado *et al.*, “SANE: A Protection Architecture for Enterprise Networks,” in *Usenix Security*, 2006.
- [18] M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker, “Ethane: Taking control of the enterprise,” in *ACM SIGCOMM Computer Communication Review*, 2007, vol. 37, no. 4, pp. 1–12.
- [19] B. Pfaff *et al.*, “The Design and Implementation of Open vSwitch,” *12th USENIX Symp. Networked Syst. Des. Implement.*, pp. 117–130, 2015.
- [20] R. Sherwood *et al.*, “Flowvisor: A network virtualization layer,” *OpenFlow Switch Consortium, Tech. Rep*, 2009.
- [21] “VMware and Nicira,” *vmWare*, 2012. [Online]. Available: <https://www.vmware.com/company/acquisitions/nicira.html>.
- [22] Z. Chen, W. Dong, H. Li, P. Zhang, X. Chen, and J. Cao, “Collaborative Network Security in Multi-Tenant Data Center for Cloud Computing,” *Tsinghua Sci. Technol.*, vol. 19, no. 1, pp. 82–94, 2014.
- [23] T. Koponen *et al.*, “Onix: A Distributed Control Platform for Large-scale Production Networks,” in *OSDI*, 2010, vol. 10, pp. 1–6.
- [24] N. Gude *et al.*, “NOX: towards an operating system for networks,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 3, pp. 105–110, 2008.
- [25] M. Monaco, O. Michel, and E. Keller, “Applying operating system principles to SDN controller design,” in *Proceedings of the Twelfth ACM Workshop on Hot Topics in Networks*, 2013, p. 2.
- [26] C. Networks, “Cumulus Linux,” 2014. [Online]. Available: <https://cumulusnetworks.com>.
- [27] O. N. Linux, “Open Network Linux.” [Online]. Available: <https://opennetlinux.org/>.
- [28] B. S. Networks, “Switch Light.” [Online]. Available: <http://www.bigswitch.com/products/switch-light>.
- [29] J. C. R. Licklider and V. W. Clapp, *Libraries of the Future*, vol. 2. mit Press Cambridge, MA, 1965.
- [30] T. (Thompson C. Gale, “World of Computer Science on John McCarthy,” *BookRags*, 2005. [Online]. Available: <http://www.bookrags.com/biography/john-mccarthy-wcs/>.
- [31] D. Shasha and C. Lazere, *Out of their minds: the lives and discoveries of 15 great computer scientists*. Springer Science & Business Media, 1998.
- [32] M. Benioff and C. Adler, *Behind the cloud: the untold story of how Salesforce.com went from idea to billion-dollar company-and revolutionized an industry*. John Wiley & Sons, 2009.
- [33] “Public Cloud Computing - Gartner IT Glossary,” 2016. [Online]. Available:

- <http://www.gartner.com/it-glossary/public-cloud-computing/>. [Accessed: 18-Dec-2016].
- [34] T. J. Bittman, “Clarifying private cloud computing,” *GARTNER. Gartner Blog Network [online]*, 2010. .
- [35] ENISA, “FAQs to the report ‘Cloud Computing: Benefits, risks and recommendations for information security.’” 2009.
- [36] P. Mell and T. Grance, “The NIST definition of cloud computing,” 2011.
- [37] J. Hurwitz, R. Bloor, M. Kaufman, and F. Halper, *Cloud computing for dummies*. John Wiley & Sons, 2010.
- [38] W. Vogels, “Beyond server consolidation,” *Queue*, vol. 6, no. 1, pp. 20–26, 2008.
- [39] J. Dean and S. Ghemawat, “MapReduce: simplified data processing on large clusters,” *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008.
- [40] F. Chang *et al.*, “Bigtable: A distributed storage system for structured data,” *ACM Trans. Comput. Syst.*, vol. 26, no. 2, p. 4, 2008.
- [41] G. DeCandia *et al.*, “Dynamo: amazon’s highly available key-value store,” in *ACM SIGOPS Operating Systems Review*, 2007, vol. 41, no. 6, pp. 205–220.
- [42] S. Ghemawat, H. Gobioff, and S.-T. Leung, “The Google file system,” in *ACM SIGOPS operating systems review*, 2003, vol. 37, no. 5, pp. 29–43.
- [43] J. Hamilton, “Cost of power in large-scale data centers,” *Blog entry dated*, vol. 11, p. 28, 2008.
- [44] A. Bechtolsheim, “Cloud Computing and Cloud Networking.” Berkeley, 2008.
- [45] K. Rangan, A. Cooke, J. Post, and N. Schindler, “The Cloud Wars: 100+ billion at stake,” 2008.
- [46] I. S. & A. U.S. Energy Information Administration, “Factors Affecting Electricity Prices,” 2015. [Online]. Available: http://www.eia.gov/energyexplained/index.cfm?page=electricity_factors_affecting_prices.
- [47] B. Despotović, “Cena Struje u Srbiji,” 2010. [Online]. Available: <http://www.ekobalans.net/vesti/3-energija/64-cena-struje-u-srbiji.html>.
- [48] E. E. Portal, “Electricity Prices - Industry,” 2012. [Online]. Available: <http://www.energy.eu/#Industrial-Elec>.
- [49] M. Armbrust *et al.*, “Above the Clouds : A Berkeley View of Cloud Computing Cloud Computing : An Old Idea Whose Time Has (Finally) Come,” *Computing*, vol. 53, no. UCB/EECS-2009-28, pp. 07–013, 2009.
- [50] J. Gray, “Distributed computing economics,” *Queue*, vol. 6, no. 3, pp. 63–68, 2008.
- [51] D. A. Patterson, “A conversation with Jim Gray,” *ACM Queue*, vol. 1, no. 4, pp. 53–56, 2003.
- [52] L. Siegele, *Let it rise: A special report on corporate IT*. Economist Newspaper, 2008.

- [53] Cloudera, “Apache Hadoop training from Cloudera University.” [Online]. Available: <http://www.cloudera.com/training.html>. [Accessed: 14-Dec-2016].
- [54] Amazon, “Public Data Sets on AWS.” 2010.
- [55] P. Mell and T. Grance, “Effectively and securely using the cloud computing paradigm,” *NIST, Inf. Technol. Lab.*, pp. 304–311, 2009.
- [56] J. Hsiao, “Amazon.com CEO Jeff Bezos on Animoto.” Apr-2008.
- [57] D. Abramson, R. Buyya, and J. Giddy, “A computational economy for grid computing and its implementation in the Nimrod-G resource broker,” *Futur. Gener. Comput. Syst.*, vol. 18, no. 8, pp. 1061–1074, 2002.
- [58] K. Hosanagar, R. Krishnan, M. Smith, and J. Chuang, “Optimal pricing of content delivery network (CDN) services,” in *System Sciences, 2004. Proceedings of the 37th Annual Hawaii International Conference on*, 2004, p. 10--pp.
- [59] G. Stuer, K. Vanmechelen, and J. Broeckhove, “A commodity market algorithm for pricing substitutable Grid resources,” *Futur. Gener. Comput. Syst.*, vol. 23, no. 5, pp. 688–701, 2007.
- [60] J. Hamilton, “Cooperative expendable micro-slice servers (CEMS): low cost, low power servers for internet-scale services,” in *Conference on Innovative Data Systems Research (CIDR'09)(January 2009)*, 2009.
- [61] J. Brodtkin, “Loss of customer data spurs closure of online storage service - The Linkup,” *Netw. World (August 2008)*, 2008.
- [62] U. Holzle, “Private Communication,” 2009.
- [63] J. D. McCalpin, “Memory bandwidth and machine balance in current high performance computers,” 1995.
- [64] B. Krebs, “Amazon: Hey spammers, get off my cloud,” *Washingt. Post (July 2008)*, 2008.
- [65] C. Horne, “Understanding full virtualization, paravirtualization, and hardware assist.” 2014.
- [66] A. Kovari and P. Dukan, “KVM & OpenVZ virtualization based IaaS Open Source Cloud Virtualization Platforms: OpenNode, Proxmox VE.”
- [67] Wikipedia, “Partial Virtualization,” *Wikipedia*, Jan-2017. [Online]. Available: https://en.wikipedia.org/wiki/Hardware_virtualization#Partial_virtualization.
- [68] “Desktop Virtualization,” *Wikipedia*, 2017. [Online]. Available: https://en.wikipedia.org/wiki/Desktop_virtualization.
- [69] K. Adams and O. Agesen, “A Comparison of Software and Hardware Techniques for x86 Virtualization.”
- [70] Wikipedia, “Paravirtualization,” Jan-2016. [Online]. Available:

<https://en.wikipedia.org/wiki/Paravirtualization>.

- [71] Gartner, “Gartner Highlights Five Attributes of Cloud Computing,” 2012. [Online]. Available: <https://www.gartner.com/newsroom/id/1035013>.
- [72] R. Arpaci-Dusseau and A. Arpaci-Dusseau, “Operating Systems: Three Easy Pieces.” [Online]. Available: <http://pages.cs.wisc.edu/~remzi/OSTEP/#book-chapters>. [Accessed: 14-Dec-2016].
- [73] S. Julian, M. Shuey, and S. Cook, “Containers in Research,” in *Proceedings of the XSEDE16 on Diversity, Big Data, and Science at Scale - XSEDE16*, 2016, pp. 1–6.
- [74] “What is Docker?,” 2016. [Online]. Available: <https://www.docker.com/what-docker>.
- [75] J. Purrier, “What is Rocket and How It’s Different Than Docker,” 2015. [Online]. Available: <https://www.ctl.io/developers/blog/post/what-is-rocket-and-how-its-different-than-docker/>.
- [76] V. Djurica and M. Minovic, “Linux based virtual networking laboratories for software defined networking,” *IJEE*, vol. vol 33-2B, 2017.
- [77] B. Carpenter and S. Brim, “Middleboxes: Taxonomy and issues,” 2002.
- [78] A. T. Campbell, H. G. De Meer, M. E. Kounavis, K. Miki, J. B. Vicente, and D. Villela, “A survey of programmable networks,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 29, no. 2, pp. 7–23, 1999.
- [79] K. Calvert, “Reflections on network architecture: an active networking perspective,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 2, pp. 27–30, 2006.
- [80] K. Psounis, “Active networks: Applications, security, safety, and architectures,” *Commun. Surv. IEEE*, vol. 2, no. 1, pp. 2–16, 1999.
- [81] A. Lazar, K.-S. Lim, F. Marconcini, and others, “Realizing a foundation for programmability of atm networks with the binding architecture,” *Sel. Areas Commun. IEEE J.*, vol. 14, no. 7, pp. 1214–1227, 1996.
- [82] A. T. Campbell, I. Katzela, K. Miki, and J. Vicente, “Open signaling for ATM, internet and mobile networks (OPENSIG’98),” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 29, no. 1, pp. 97–108, 1999.
- [83] OpenNetworkFoundation, “Open Network Foundation.” .
- [84] D. Kreutz, F. M. V Ramos, P. Esteves Verissimo, C. Esteve Rothenberg, S. Azodolmolky, and S. Uhlig, “Software-defined networking: A comprehensive survey,” *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, 2015.
- [85] P. Newman *et al.*, “Ipsilon’s General Switch Management Protocol Specification Version 2.0,” 1998.

- [86] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, *Resilient overlay networks*, vol. 35, no. 5. ACM, 2001.
- [87] Pica8, “XORPlus.” [Online]. Available: <http://sourceforge.net/p/xorplus/home/Pica8Xorplus/>.
- [88] Pica8, “Pica8 PicOS.” [Online]. Available: <http://www.pica8.com/products/picos>.
- [89] Fa, “Introducing ‘Wedge’ and ‘FBOSS,’ the next steps toward a disaggregated network.” [Online]. Available: <https://code.facebook.com/posts/681382905244727/introducing-wedge-and-fboss-the-next-steps-toward-a-disaggregated-network/>.
- [90] T. Koponen *et al.*, “Network virtualization in multi-tenant datacenters,” in *USENIX NSDI*, 2014.
- [91] Facebook, “Facebook FBOSS.” [Online]. Available: <https://code.facebook.com/posts/843620439027582/facebook-open-switching-system-fboss-and-wedge-in-the-open/>.
- [92] R. Sherwood, “TUTORIAL: WHITE BOX/BARE METAL SWITCHES,” *BigSwitch Networks*, 2014. [Online]. Available: <https://www.bigswitch.com/sites/default/files/presentations/onug-baremetal-2014-final.pdf>. [Accessed: 14-Dec-2016].
- [93] V. Djurica and M. Minovic, “Poslovne i tehnološke prednosti centralizovanog upravljanja klad tkanjem,” in *InfoTeb*, 2017.
- [94] F. Bergeron, L. Raymond, and S. Rivard, “Ideal patterns of strategic alignment and business performance,” *Inf. Manag.*, vol. 41, no. 8, pp. 1003–1020, 2004.
- [95] T. M. Smith and J. S. Reece, “The relationship of strategy, fit, productivity, and business performance in a services setting,” *J. Oper. Manag.*, vol. 17, no. 2, pp. 145–161, 1999.
- [96] M. S. Peer Mohamed and S. R. Swarnammal, “An efficient framework to handle integrated VM workloads in heterogeneous cloud infrastructure,” *Soft Comput.*, pp. 1–10, Jan. 2016.
- [97] M. Armbrust *et al.*, “Above the Clouds: A Berkeley View of Cloud Computing,” 2009.
- [98] W. Felter, A. Ferreira, R. Rajamony, and J. Rubio, “An Updated Performance Comparison of Virtual Machines and Linux Containers,” *Technology*, vol. 25482, pp. 171–172, 2014.
- [99] A. Gordon *et al.*, “ELI: bare-metal performance for I/O virtualization,” in *ASPLOS ’12 Proceedings of the seventeenth international conference on Architectural Support for Programming Languages and Operating Systems4*, 2012, pp. 411–422.
- [100] M. Attaran, “Exploring the relationship between information technology and business process reengineering,” *Inf. Manag.*, vol. 41, no. 5, pp. 585–596, 2004.
- [101] K. Kambatla, G. Kollias, V. Kumar, and A. Grama, “Trends in big data analytics,” *J. Parallel*

Distrib. Comput., vol. 74, no. 7, pp. 2561–2573, 2014.

- [102] H. Demirkan and D. Delen, “Leveraging the capabilities of service-oriented decision support systems: Putting analytics and big data in cloud,” *Decis. Support Syst.*, vol. 55, no. 1, pp. 412–421, 2013.
- [103] A. Dix, J. Finlay, G. D. Abowd, and R. Beale, “Human-Computer Interaction,” *Human-Computer Interact.*, vol. Third, no. January, p. 834, 2004.
- [104] C. Ebert, G. Gallardo, J. Hernantes, and N. Serrano, “DevOps,” *IEEE Softw.*, vol. 33, no. 3, pp. 94–100, 2016.
- [105] M. Fowler, “Microservices,” *martinfowler.com*, 2014. [Online]. Available: <http://martinfowler.com/articles/microservices.html>.
- [106] D. S. Lathicum, “Practical Use of Microservices in Moving Workloads to the Cloud,” *IEEE Cloud Comput.*, vol. 3, no. 5, pp. 6–9, Sep. 2016.
- [107] A. S. Patwardhan, “Analysis of Software Delivery Process Shortcomings and Architectural Pitfalls,” 2000.
- [108] I. Gorton, A. B. Bener, and A. Mockus, “Software Engineering for Big Data Systems,” *IEEE Softw.*, vol. 33, no. 2, pp. 32–35, Mar. 2016.
- [109] G. Buchgeher, R. Weinreich, and T. Kriechbaum, “Making the Case for Centralized Software Architecture Management,” Springer, Cham, 2016, pp. 109–121.
- [110] J. Opara-Martins, R. Sahandi, and F. Tian, “Critical review of vendor lock-in and its impact on adoption of cloud computing,” in *International Conference on Information Society, i-Society 2014*, 2015, pp. 92–97.
- [111] B. Campbell, R. Kay, and D. Avison, “Strategic alignment: a practitioner’s perspective,” *J. Enterp. Inf. Manag.*, vol. 18, no. 6, pp. 653–664, Dec. 2005.

15. Prilog A – Anketna pitanja

Normalizacija podataka: Radi normalizacije podataka ispitanike smo pitali da uvek izaberu izvorni proizvod ukoliko je proizvod koji koriste baziran na nekom drugom rešenju, uglavnom otvorenog izvornog koda.

1. Kojom od sledećih tehnologija je kompanijski cloud system primarno izgrađen? (jedan odgovor)
 - a. Hardver (operativni sistemim direktno instaliran na hardver)
 - b. Tehnologije virtualizacije (Tip 1 hipervizora)
 - c. Iznajmljene instance na cloud sistemima (Azure, GCE, EC2)
 - d. Linux Kontejneri
2. Koja rešenja koristite na mrežnom sloju? (jedan odgovor)
 - a. Tradicionalna rešenja iz oblasti računarskih mreža (integrisana kontrolna ravan sa ravni prosleđivanja podataka)
 - b. Virtualizaciju mrežnih funkcija i/ili softverski definisane mreže (NFV, SDN)
3. Da li koristite i koji od navedenih Tip 1 hipervizora? (jedan odgovor)
 - a. VMware
 - b. Xen ili Xen bazirano rešenje (npr. Citrix)
 - c. Microsoft Hyper-V
 - d. Neko drugo rešenje (navedite koje)
 - e. Nijedno rešenje zasnovano na Tip 1 hipervizorima
4. Da li koristite i koji od navedenih Linux kontejnera? (jedan odgovor)
 - a. LXC
 - b. Docker
 - c. Rocket
 - d. Neko drugo rešenje (navedite koje)
 - e. Nijedno rešenje bazirano na kontejnerima
5. Da li koristie neku od SDN/NFV tehnologija **virtualizacije** mrežnog sloja? (jedan odgovor)
 - a. Open vSwitch
 - b. FlowVisor
 - c. NVP (VMWare NSX)

- d. Neko drugo rešenje (navedite koje)
 - e. Nijedno rešenje virtualizacije mrežnog sloja
6. Da li koristite neki od SDN kontrolera? (jedan odgovor)
- a. NOX
 - b. POX
 - c. ONIX
 - d. YANC
 - e. OpenDaylight (ili fork istog kao što je Brocade SDN controller)
 - f. Trema
 - g. Neki drugi, navedite koji
 - h. Ne koristimo nijedno rešenje u oblasti SDN kontrolera
7. Da li koristite neku od tehnologija mrežnog sloja zasnovanu na Linux operativnom sistemu i koje je portabilno na hardver više proizvođača? (jedan odgovor)
- a. Open Network Linux
 - b. Cumulus Linux
 - c. Switch Light
 - d. Neki drugi Linux baziran mrežni operativni sistem (navedite koji)
 - e. Nijedan Linux baziran operativni system
8. Da li i koliko starih (eng. *legacy*) sistema imate koji zahtevaju specifičan operativni sistem ili specifičnu verziju ili postoje samo za jednu platformu? (jedan odgovor)
- a. Imamo jedan ili više takvih sistema
 - b. Nemamo takve sisteme
9. Ako imate stare sisteme, da li je to bilo opredeljenje za pristup izgradnji cloud sistema baziranog na virtualizaciji mašina ili virtualne instance (naspram kontejnerima)? (jedan odgovor)
- a. Da, u potpunosti
 - b. Delimično da, to je bio jedan od faktora odlučivanja
 - c. Ne, uopšte
10. Sa uvođenjem tehnologija na mrežnom sloju, da li ste uspeli da ostvarite viši stepen automatizacije? Primer: stvari koje su rađene ručnom konfiguracijom su automatizovane. (jedan odgovor)
- a. Kompanija nema uvedene tehnologije mrežnog sloja
 - b. Uvedeno, ali je automatizacija ostala na približno istom nivou
 - c. Da, viši stepen automatizacije je ostvaren

11. Da li ste uspjeli da ostvarite viši stepen automatizacije na mrežnom sloju i koliko? Primer: Automatizacija i propagacija konfiguracione promene (jedan odgovor)
- 0 - 30%
 - 31 – 60 %
 - 61 – 100%
 - Preko 100% (višestruko ubrzanje propagacije konfiguracione promene)
12. Da li upravljanje možete da integrišete u jedinstven sistem upravljanja? (jedan odgovor)
- Da, potpuno
 - Da, delimično
 - Ne, uopšte
13. Da li je mrežnim slojem moguće upravljati pomoću standardnih serverskih frejmvorka za automatizaciju i orkestraciju? (npr. *Chef, Puppet, CFEngine, Salt, Ansible*)? (jedan odgovor)
- Da, potpuno se slažem
 - Da, delimično se slažem
 - Neutralno ili nisam siguran
 - Ne, delimično (moguće je automatizovati sistem i povezati ga sa ostatkom sistema putem neke vrste standardnog interfejsa: API, formata podataka ili tome slično)
 - Ne, uopšte nije moguće
14. Da li imate potrebu za podrškom više raznorodnih operativnih serverskih sistema u okviru cloud sistema i kojim? (jedan odgovor)
- Windows
 - Linux (bilo koja distribucija)
 - Ostalo (navesti)
15. Koja je primarna poslovno-tehnološka namena sistema? (jedan odgovor)
- Interno izgrađeni sistemi visoke efikasnosti bez potrebe za kompatibilnošću
 - Interno izgrađeni sistemi sa delimičnom podrškom za zastarele ili specifične sisteme
 - Puna podrška raznorodnim sistemima, uključujući i zastarele i specifične sisteme
 - Mogućnost samostalne kontrole, razvoja i inovacije internih sistema
 - Ostalo (navesti)
16. Koji od sledećih poslovnih kriterijuma su bili ključni u dizajnu arhitekture Sistema? (jedan odgovor)
- Optimizacija broja servera
 - Optimizacija veličine odnosno snage pojedinačnog servera

- c. Prebacivanje dela kapitalnih troškova u operativne (na primer, kroz eksterne ugovore ili iznajmljivanje)
- d. Produženje životnog ciklusa uređaja
- e. Poboljšanje performansi sistema
- f. Ušteda na broju zaposlenih koji održavaju infrastrukturu
- g. Smanjenje kompleksnosti poslova koje zaposleni obavljaju (na primer, ono što je radio inženjer je posle optimizacije mogao da radi tehničar)
- h. Povećana mogućnost analize delova sistema radi optimizacija i boljeg razumevanja sistema uopšte
- i. Smanjen faktor ljudskih grešaka
- j. Upravljanje na unforman (unificiran način) svim sistemima. Primer su mrežni operativni sistemi bazirani na Linux-u kojima se može upravljati sa standardnim alatima za orkestraciju serverske infrastructure
- k. Povećanje brzine izgradnje novih servisa
- l. Povećanje mogućnosti uvođenja novih tehnologija i internih inovacija
- m. Lakše održavanje postojećih sistema (manje zavisnih komponenti podsistema)
- n. Mikro segmentacija servisa
- o. Skalabilnost sistema
- p. Elastičnost sistema (u oba pravca)
- q. Centralizacija upravljanja sistemom
- r. Nezavisnost od dobavljača i robne marke proizvođača

17. Kako ocenjujete postignute uštede uvođenjem cloud tehnologija?

- a. Veoma visoke
- b. Visoke
- c. Osrednje
- d. Niske
- e. Veoma niske

18. Sa uvođenjem cloud tehnologija, u kojoj meri je firma u mogućnosti da brže reaguje na poslovno-tehnološku potrebu?

- a. Veoma visoke
- b. Visoke
- c. Osrednje
- d. Niske

e. Veoma niske

19. Da li je uvođenje novih, prethodno navedenih sistema povećao broj uspešno isporučenih projekata i ukupno poverenje u IT odeljenje?

- a. Povećano, značajno
- b. Povećano, vrlo malo
- c. Ostalo je otprilike na istom nivou
- d. Uticalo je negativno u manjoj meri
- e. Uticalo je negativno u većoj meri

16.Prilog B – Šifrarnik za SPSS

Tabela 14 - Šifrarnik za SPSS

#	Pitanje	Naziv promenljive	Uputstvo za šifrovanje
0	Identifikacioni broj	Id	Identifikacioni broj subjekta
1	Kojom od sledećih tehnologija je kompanijski cloud system primarno izgrađen? (jedan ili više odgovora)	primarno_izgradjen	1=Hardver (operativni sistemim direktno instaliran na hardver), 2=Tehnologije virtualizacije (Tip 1 hipervizora) 3=Iznajmljene instance na cloud sistemima (Azure, GCE, EC2), 4=Linux Kontejneri
2	Koja rešenja koristite na mrežnom sloju?	mrezni_sloj	1=Tradicionalna rešenja iz oblasti računarskih mreža (integrisana kontrolna ravan sa ravni prosleđivanja podataka), 2=Virtualizaciju mrežnih funkcija i/ili softverski definisane mreže (NFV, SDN)
3	Da li koristite i koji od navedenih Tip 1 hipervizora?	tip_hipervizora	1=VMware, 2=Xen ili Xen bazirano rešenje (npr. Citrix), 3=Microsoft Hyper-V, 4=Neko drugo rešenje (navedite koje), 5=Nijedno rešenje zasnovano na Tip 1 hipervizorima
4	Da li koristite i koji od navedenih Linux kontejnera?	tip_kontejnera	1=LXC, 2=Doker, 3=Roket, 4=Neko drugo rešenje (navedite koje), 5=Nijedno rešenje bazirano na kontejnerima

5	Da li koristite neku od SDN/NFV tehnologija virtualizacije mrežnog sloja?	sdn_virtualizacija	1=Open vSwitch, 2=FlowVisor, 3=NVP (VMWare NSX), 4=Neko drugo rešenje (navedite koje), 5=Nijedno rešenje virtualizacije mrežnog sloja
6	Da li koristite neki od SDN kontrolera?	sdn_kontroler	1=NOX, 2=POX, 3=ONIX, 4=YANC, 5=OpenDaylight (ili fork istog kao što je Brocade SDN controller), 6=Trema, 7=Neki drugi, navedite koji, 8=Ne koristimo nijedno rešenje u oblasti SDN kontrolera
7	Da li koristite neku od tehnologija mrežnog sloja zasnovanu na Linux operativnom sistemu i koje je portabilno na hardver više proizvođača?	linux_mrežno_rešenje	1=Open Network Linux, 2=Cumulus Linux, 3=Switch Light, 4=Neki drugi Linux baziran mrežni operativni sistem (navedite koji), 5=Nijedan Linux baziran operativni system
8	Da li i koliko starih (eng. <i>legacy</i>) sistema imate koji zahtevaju specifičan operativni sistem ili specifičnu verziju ili postoje samo za jednu platformu?	legacy_sistemi	1=Imamo jedan ili više takvih Sistema, 2=Nemamo takve sisteme
9	Ako imate stare sisteme, da li je to bilo opredeljenje za pristup izgradnji cloud sistema baziranog na virtualizaciji mašina ili virtualne instance (naspram kontejnerima)?	legacy_faktori	1=Da, u potpunosti, 2=Delimično da, to je bio jedan od faktora odlučivanja, 3=Ne, uopšte
10	Sa uvođenjem tehnologija na mrežnom sloju, da li ste uspeli da ostvarite viši stepen automatizacije? Primer: stvari	automatizacija_ostvarena	1=Kompanija nema uvedene tehnologije mrežnog sloja, 2=Uvedeno, ali je automatizacija ostala na

	koje su rađene ručnom konfiguracijom su automatizovane.		prbiližno istom nivou, 3=Da, viši stepen automatizacije je ostvaren
11	Da li ste uspjeli da ostvarite viši stepen automatizacije na mrežnom sloju i koliko? Primer: Automatizacija i propagacija konfiguracione promene	automatizacija_metrika	1=0 - 30%, 2=31 – 60 %, 3=61 – 100%, 4=Preko 100% (višestruko ubrzanje propagacije konfiguracione promene)
12	Da li upravljanje možete da integrišete u jedinstven sistem upravljanja?	integracija_jedinstveni	1=Da, potpuno, 2=Da, delimično, 3=Ne, uopšte
13	Da li je mrežnim slojem moguće upravljati pomoću standardnih serverskih frejmvorka za automatizaciju i orkestraciju? (npr. <i>Chef, Puppet, CFEngine, Salt, Ansible</i>)?	upravljanje_srv_frmwrk	1=Da, potpuno se slažem, 2=Da, delimično se slažem, 3=Neutralno ili nisam siguran, 4=Ne, delimično (moguće je automatizovati sistem i povezati ga sa ostatkom sistema putem neke vrste standardnog interfejsa: API, formata podataka ili tome slično)
14	Da li imate potrebu za podrškom više raznorodnih operativnih serverskih sistema u okviru cloud sistema i kojim?	multi_os_podrska	1=Windows, 2=Linux (bilo koja distribucija), 3=Ostalo (navesti)
15	Koja je primarna poslovno-tehnološka namena sistema?	namena_sistema	1=Interno izgrađeni sistemi visoke efikasnosti bez potrebe za kompatibilnošću 2=Interno izgrađeni sistemi sa delimičnom podrškom za zastarele ili specifične sisteme

			<p>3=Puna podrška raznorodnim sistemima, uključujući i zastarele i specifične sisteme</p> <p>4=Mogućnost samostalne kontrole, razvoja i inovacije internih sistema</p> <p>5=Ostalo (navesti)</p>
16	Koji od sledećih poslovnih kriterijuma su bili ključni u dizajnu arhitekture Sistema?	<p>op_capex_1 do op_capex_4</p> <p>op_opex_1 do op_opex_6</p> <p>op_agilnost_1 do op_agilnost_8</p>	1=da, 2=ne
17	Kako ocenjujete postignute uštede uvođenjem cloud tehnologija?	ustede	1=Veoma visoke, 2=Visoke 3=Osrednje, 4=Niske, 5=Veoma niske
18	Sa uvođenjem cloud tehnologija, u kojoj meri je firma u mogućnosti da brže reaguje na poslovno-tehnološku potrebu?	agilnost	1=Veoma visoke, 2=Visoke, 3=Osrednje, 4=Niske, 5=Veoma niske
19	Da li je uvođenje novih, prethodno navedenih sistema povećao broj uspešno isporučenih projekata i ukupno poverenje u IT odeljenje?	stratesko_poravnanje	1=Povećano, značajno, 2=Povećano, vrlo malo 3=Ostalo je otprilike na istom nivou 4=Uticalo je negativno u manjoj meri 5=Uticalo je negativno u većoj meri

17. Prilog C – Implementacija ekspertnog sistema *Klaudifikator*

U nastavku se nalazi izvorni kod fajla ***Klaudifikator.dxi*** koji predstavlja *Klaudifikator*, ekspertni sistem za podršku u odlučivanju izbora i uvođenja cloud tehnologija u preduzeća:

```
<?xml version="1.0" encoding="UTF-8"?>
<DEXi>
  <VERSION>5.02</VERSION>
  <CREATED>2017-08-01T15:22:19</CREATED>
  <OPTION>Iznajmiti CLOUD</OPTION>
  <OPTION>Izgraditi CLOUD</OPTION>
  <OPTION>Hibridno resenje</OPTION>
  <SETTINGS/>
  <ATTRIBUTE>
    <NAME>CLOUD</NAME>
    <SCALE>
      <SCALEVALUE>
        <NAME>neprihvatljivo</NAME>
        <GROUP>BAD</GROUP>
      </SCALEVALUE>
      <SCALEVALUE>
        <NAME>zadovoljavajuće</NAME>
      </SCALEVALUE>
      <SCALEVALUE>
        <NAME>odlično</NAME>
        <GROUP>GOOD</GROUP>
      </SCALEVALUE>
    </SCALE>
  </ATTRIBUTE>
  <FUNCTION>
    <LOW>01000000000010011010010021000000000000010011010011122200000010021022121122222222</
    LOW>
    <CONSIST>False</CONSIST>
  </FUNCTION>
  <OPTION>2</OPTION>
  <OPTION>1</OPTION>
  <OPTION>0</OPTION>
  <ATTRIBUTE>
    <NAME>Trosak</NAME>
    <SCALE>
      <SCALEVALUE>
        <NAME>visok</NAME>
        <GROUP>BAD</GROUP>
      </SCALEVALUE>
      <SCALEVALUE>
        <NAME>srednji</NAME>
      </SCALEVALUE>
      <SCALEVALUE>
        <NAME>nizak</NAME>
        <GROUP>GOOD</GROUP>
      </SCALEVALUE>
    </SCALE>
  </ATTRIBUTE>
</DEXi>
```

```

</SCALE>
<FUNCTION>
  <LOW>000111122</LOW>
  <ENTERED>+++++++</ENTERED>
</FUNCTION>
<OPTION>0</OPTION>
<OPTION>1</OPTION>
<OPTION>1</OPTION>
<ATTRIBUTE>
  <NAME>Kapitalni trosak</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>visok</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>srednji</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>nizak</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
  </SCALE>
  <OPTION>0</OPTION>
  <OPTION>2</OPTION>
  <OPTION>1</OPTION>
</ATTRIBUTE>
<ATTRIBUTE>
  <NAME>Operativni trosak</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>visok</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>srednji</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>nizak</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
  </SCALE>
  <OPTION>2</OPTION>
  <OPTION>0</OPTION>
  <OPTION>1</OPTION>
</ATTRIBUTE>
</ATTRIBUTE>
<ATTRIBUTE>
  <NAME>Specifikacija servera</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>slab</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>osrednji</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>odlican</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
  </SCALE>
</FUNCTION>

```



```

<OPTION>2</OPTION>
<OPTION>1</OPTION>
<OPTION>0</OPTION>
</ATTRIBUTE>
<ATTRIBUTE>
  <NAME>Mrezni kapacitet</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>nizak</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>srednji</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>visok</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
  </SCALE>
  <OPTION>1</OPTION>
  <OPTION>2</OPTION>
  <OPTION>1</OPTION>
</ATTRIBUTE>
<ATTRIBUTE>
  <NAME>Skladistni kapacitet</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>nizak</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>srednji</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>visok</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
  </SCALE>
  <OPTION>1</OPTION>
  <OPTION>2</OPTION>
  <OPTION>1</OPTION>
</ATTRIBUTE>
</ATTRIBUTE>
<ATTRIBUTE>
  <NAME>Snaga servera</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>niska</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>srednja</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>visoka</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
  </SCALE>
  <FUNCTION>
    <LOW>001111122</LOW>
  </FUNCTION>
  <OPTION>0</OPTION>
  <OPTION>1</OPTION>
  <OPTION>1</OPTION>

```

```

<ATTRIBUTE>
  <NAME>Efikasnost servera</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>nizak</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>srednji</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>visok</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
  </SCALE>
  <FUNCTION>
    <LOW>00001101201111112012112222</LOW>
    <ENTERED>+++++++-----+-----</ENTERED>
  </FUNCTION>
  <OPTION>0</OPTION>
  <OPTION>1</OPTION>
  <OPTION>1</OPTION>
</ATTRIBUTE>
  <NAME>Vreme obrade</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>nisko</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>srednje</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>visoko</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
  </SCALE>
  <OPTION>0</OPTION>
  <OPTION>1</OPTION>
  <OPTION>0</OPTION>
</ATTRIBUTE>
<ATTRIBUTE>
  <NAME>Vreme odziva</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>nisko</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>srednje</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>visoko</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
  </SCALE>
  <OPTION>2</OPTION>
  <OPTION>2</OPTION>
  <OPTION>1</OPTION>
</ATTRIBUTE>
<ATTRIBUTE>
  <NAME>Vreme protoka</NAME>
  <SCALE>
    <SCALEVALUE>

```

```

        <NAME>nisko</NAME>
        <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
        <NAME>srednje</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
        <NAME>visoko</NAME>
        <GROUP>GOOD</GROUP>
    </SCALEVALUE>
</SCALE>
<OPTION>0</OPTION>
<OPTION>1</OPTION>
<OPTION>2</OPTION>
</ATTRIBUTE>
</ATTRIBUTE>
<ATTRIBUTE>
    <NAME>Kapacitet obrade</NAME>
    <SCALE>
        <SCALEVALUE>
            <NAME>nizak</NAME>
            <GROUP>BAD</GROUP>
        </SCALEVALUE>
        <SCALEVALUE>
            <NAME>srednji</NAME>
        </SCALEVALUE>
        <SCALEVALUE>
            <NAME>visok</NAME>
            <GROUP>GOOD</GROUP>
        </SCALEVALUE>
    </SCALE>
    <FUNCTION>
        <LOW>000011111001111122111122222</LOW>
        <ENTERED>+++++++-----+++++</ENTERED>
    </FUNCTION>
    <OPTION>1</OPTION>
    <OPTION>1</OPTION>
    <OPTION>0</OPTION>
</ATTRIBUTE>
<ATTRIBUTE>
    <NAME>Broj poslova po serveru</NAME>
    <SCALE>
        <SCALEVALUE>
            <NAME>nizak</NAME>
            <GROUP>BAD</GROUP>
        </SCALEVALUE>
        <SCALEVALUE>
            <NAME>srednji</NAME>
        </SCALEVALUE>
        <SCALEVALUE>
            <NAME>visok</NAME>
            <GROUP>GOOD</GROUP>
        </SCALEVALUE>
    </SCALE>
    <OPTION>0</OPTION>
    <OPTION>0</OPTION>
    <OPTION>0</OPTION>
</ATTRIBUTE>
</ATTRIBUTE>
    <NAME>Broj upita po jedinici vremena</NAME>
    <SCALE>
        <SCALEVALUE>
            <NAME>nizak</NAME>
            <GROUP>BAD</GROUP>
        </SCALEVALUE>

```

```

    <SCALEVALUE>
      <NAME>srednji</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>visok</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
  </SCALE>
  <OPTION>1</OPTION>
  <OPTION>2</OPTION>
  <OPTION>1</OPTION>
</ATTRIBUTE>
<ATTRIBUTE>
  <NAME>Klasifikacija prioriteta</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>niska</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>osrednja</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>visoka</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
  </SCALE>
  <OPTION>1</OPTION>
  <OPTION>2</OPTION>
  <OPTION>0</OPTION>
</ATTRIBUTE>
</ATTRIBUTE>
</ATTRIBUTE>
<ATTRIBUTE>
  <NAME>Izvršno okruženje</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>interpreter</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>vm</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>kompajlirano</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
  </SCALE>
  <OPTION>2</OPTION>
  <OPTION>1</OPTION>
  <OPTION>1</OPTION>
</ATTRIBUTE>
</ATTRIBUTE>
<ATTRIBUTE>
  <NAME>Optimizacija poslovnih faktora</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>niska</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>osrednja</NAME>
    </SCALEVALUE>
    <SCALEVALUE>

```

```

    <NAME>visoka</NAME>
    <GROUP>GOOD</GROUP>
  </SCALEVALUE>
</SCALE>
<FUNCTION>
  <LOW>00001101201111112012112222</LOW>
  <ENTERED>+++++++-----</ENTERED>
</FUNCTION>
<OPTION>2</OPTION>
<OPTION>1</OPTION>
<OPTION>0</OPTION>
<ATTRIBUTE>
  <NAME>Broj zaposlenih</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>visok</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>srednji</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>nizak</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
  </SCALE>
  <FUNCTION>
    <LOW>001112122</LOW>
    <ENTERED>+++++--</ENTERED>
  </FUNCTION>
  <OPTION>2</OPTION>
  <OPTION>1</OPTION>
  <OPTION>1</OPTION>
</ATTRIBUTE>
  <NAME>Automatizacija poslova</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>niska</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>osrednja</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>visoka</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
  </SCALE>
  <OPTION>1</OPTION>
  <OPTION>1</OPTION>
  <OPTION>1</OPTION>
</ATTRIBUTE>
<ATTRIBUTE>
  <NAME>Centralizovano upravljanje</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>nisko</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>osrednje</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>visoko</NAME>
    </SCALEVALUE>
  </SCALE>

```



```

        <GROUP>GOOD</GROUP>
    </SCALEVALUE>
</SCALE>
<OPTION>2</OPTION>
<OPTION>1</OPTION>
<OPTION>0</OPTION>
</ATTRIBUTE>
</ATTRIBUTE>
<ATTRIBUTE>
    <NAME>Kompleksnost rada</NAME>
    <SCALE>
        <SCALEVALUE>
            <NAME>visoka</NAME>
            <GROUP>BAD</GROUP>
        </SCALEVALUE>
        <SCALEVALUE>
            <NAME>srednja</NAME>
        </SCALEVALUE>
        <SCALEVALUE>
            <NAME>niska</NAME>
            <GROUP>GOOD</GROUP>
        </SCALEVALUE>
    </SCALE>
    <FUNCTION>
        <LOW>00011222</LOW>
        <ENTERED>+++++</ENTERED>
    </FUNCTION>
    <OPTION>2</OPTION>
    <OPTION>2</OPTION>
    <OPTION>0</OPTION>
</ATTRIBUTE>
    <NAME>Korisnicki interfejs</NAME>
    <SCALE>
        <SCALEVALUE>
            <NAME>komplicovan</NAME>
            <GROUP>BAD</GROUP>
        </SCALEVALUE>
        <SCALEVALUE>
            <NAME>jednostavan</NAME>
            <GROUP>GOOD</GROUP>
        </SCALEVALUE>
    </SCALE>
    <OPTION>1</OPTION>
    <OPTION>1</OPTION>
    <OPTION>0</OPTION>
</ATTRIBUTE>
<ATTRIBUTE>
    <NAME>Interfejs medju servisima</NAME>
    <SCALE>
        <SCALEVALUE>
            <NAME>ne postoji</NAME>
            <GROUP>BAD</GROUP>
        </SCALEVALUE>
        <SCALEVALUE>
            <NAME>postoji</NAME>
            <GROUP>GOOD</GROUP>
        </SCALEVALUE>
    </SCALE>
    <OPTION>0</OPTION>
    <OPTION>1</OPTION>
    <OPTION>1</OPTION>
</ATTRIBUTE>
<ATTRIBUTE>
    <NAME>Skripting jezik</NAME>

```

```

    <SCALE>
      <SCALEVALUE>
        <NAME>komplikovan</NAME>
        <GROUP>BAD</GROUP>
      </SCALEVALUE>
      <SCALEVALUE>
        <NAME>jednostavan</NAME>
        <GROUP>GOOD</GROUP>
      </SCALEVALUE>
    </SCALE>
    <OPTION>1</OPTION>
    <OPTION>1</OPTION>
    <OPTION>0</OPTION>
  </ATTRIBUTE>
</ATTRIBUTE>
<ATTRIBUTE>
  <NAME>Analitika</NAME>
  <SCALE>
    <SCALEVALUE>
      <NAME>slaba</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>osrednja</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>dobra</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
  </SCALE>
  <OPTION>1</OPTION>
  <OPTION>1</OPTION>
  <OPTION>0</OPTION>
</ATTRIBUTE>
</ATTRIBUTE>
<ATTRIBUTE>
  <NAME>Vreme implementacije</NAME>
  <SCALE>
    <ORDER>DESC</ORDER>
    <SCALEVALUE>
      <NAME>kratko</NAME>
      <GROUP>GOOD</GROUP>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>osrednje</NAME>
    </SCALEVALUE>
    <SCALEVALUE>
      <NAME>dugo</NAME>
      <GROUP>BAD</GROUP>
    </SCALEVALUE>
  </SCALE>
  <OPTION>0</OPTION>
  <OPTION>2</OPTION>
  <OPTION>1</OPTION>
</ATTRIBUTE>
</ATTRIBUTE>
</DEXi>

```

18. Biografski Podaci

Obrazovanje:

- 2012-2018: Univerzitet u Beogradu, Fakultet Organizacionih Nauka
Doktorske studije na smeru za Informacione Tehnologije
Naziv teze: *Model izbora virtualizacionih tehnologija za izgradnju cloud sistema*
- 2005-2012: Univerzitet u Beogradu, Fakultet Organizacionih Nauka
Magistarske studije na smeru Elektronsko Poslovanje
Naziv teze: *Kritički uporedni prikaz aktuelnih pristupa uvođenju cloud rešenja u preduzeće*
- 1996-2005: Univerzitet u Beogradu, Fakultet Organizacionih Nauka
Dodiplomske studije, smer Menadžment

Spisak objavljenih radova:

1. Đurica Vladimir, Minović Miroslav, *Poslovne i tehnološke prednosti centralizovanog upravljanja kladom i kanjem*, InfoTeh konferencija, 2017.
2. Đurica Vladimir, Minović Miroslav, *Linux Based Virtual Networking Laboratories for Software Defined Networking*, International Journal of Engineering Education (IJEE), vol 33-2B, 2017 (ISSN 0949-149X).
3. Đurica Vladimir, *Biznis analiza modela uvođenja cloud sistema u preduzeće*, InfoM, vol. 12, str. 38-44, 2013.
4. Đurica Vladimir, Minović Miroslav, *Elektronsko Poslovanje*, Ekonomski Anali, 43. vol., br. 142, str. 3

Profesionalno iskustvo:

- 2016-Trenutno: Program Menadžer, *Brocade, San Jose, SAD*
- 2015-2016: Program Menadžer, *Ooyala, Santa Clara, SAD*
- 2014-2015: Program Menadžer, *Google Inc., Mountain View, SAD*
- 2010-2015: Projekt Menadžer, *EON+, Beograd, Srbija*
- 2008-2010: Regionalni Menadžer za IT, *Wolf Theiss, Beograd*
- 2002-2008: Menadžer za IT, *Ambasada Australije, Beograd*
- 2001-2002: Menadžer za Informacione Tehnologije, *CARE International, Beograd*
- 1999-2001: Novinar i organizacioni urednik, *PC World Jugoslavija, Beograd*

Profesionalni sertifikati:

- Virtualization Certified Professional – Data Centar Virtualization, VCP5-DCV
- Cisco Certified Network Associate, CCNA
- Microsoft Certified Systems Engineer, MCSE
- IT Service Management, ITIL v3
- Project Management, UC Berkeley Extension
- Project Management Professional, PMP
- Project Management Professional – Agile Certified Professional, PMI-ACP
- Certified Scrum Master, CSM

Изјава о ауторству

Име и презиме аутора: Владимир Ђурица

Број индекса: 104/95

Изјављујем

да је докторска дисертација под насловом:

MODEL IZBORA VIRTUALIZACIONIH TEHNOLOGIJA ZA IZGRADNJU CLOUD SISTEMA

- резултат сопственог истраживачког рада;
- да дисертација у целини ни у деловима није била предложена за стицање друге дипломе према студијским програмима других високошколских установа;
- да су резултати коректно наведени и
- да нисам кршио/ла ауторска права и користио/ла интелектуалну својину других лица.

Потпис аутора

У Београду, 2. Октобра 2017.



Изјава о истоветности штампане и електронске верзије докторског рада

Име и презиме аутора: Владимир Ђурица

Број индекса: 104/95

Студијски програм: Информационе Технологије

Наслов рада: MODEL IZBORA VIRTUALIZACIONIH TEHNOLOGIJA ZA IZGRADNJU
CLOUD SISTEMA

Ментор: др Мирослав Миновић, Универзитет у Београду, Факултет
Организационих Наука

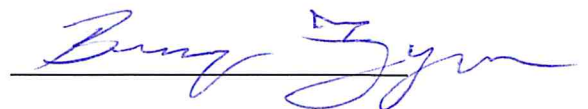
Изјављујем да је штампана верзија мог докторског рада истоветна електронској верзији коју сам предао/ла ради похрањена у **Дигиталном репозиторијуму Универзитета у Београду**.

Дозвољавам да се објаве моји лични подаци везани за добијање академског назива доктора наука, као што су име и презиме, година и место рођења и датум одбране рада.

Ови лични подаци могу се објавити на мрежним страницама дигиталне библиотеке, у електронском каталогу и у публикацијама Универзитета у Београду.

Потпис аутора

У Београду, 2. Октобра 2017.



Изјава о коришћењу

Овлашћујем Универзитетску библиотеку „Светозар Марковић“ да у Дигитални репозиторијум Универзитета у Београду унесе моју докторску дисертацију под насловом:

MODEL IZBORA VIRTUALIZACIONIH TEHNOLOGIJA ZA IZGRADNJU CLOUD SISTEMA

која је моје ауторско дело.

Дисертацију са свим прилозима предао/ла сам у електронском формату погодном за трајно архивирање.

Моју докторску дисертацију похрањену у Дигиталном репозиторијуму Универзитета у Београду и доступну у отвореном приступу могу да користе сви који поштују одредбе садржане у одабраном типу лиценце Креативне заједнице (Creative Commons) за коју сам се одлучио/ла.

1. Ауторство (CC BY)
2. Ауторство – некомерцијално (CC BY-NC)
3. Ауторство – некомерцијално – без прерада (CC BY-NC-ND)
4. Ауторство – некомерцијално – делити под истим условима (CC BY-NC-SA)
5. Ауторство – без прерада (CC BY-ND)
6. Ауторство – делити под истим условима (CC BY-SA)

(Молимо да заокружите само једну од шест понуђених лиценци.
Кратак опис лиценци је саставни део ове изјаве).

Потпис аутора

У Београду, 2. Октобра 2017.



1. **Ауторство.** Дозвољаваате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце, чак и у комерцијалне сврхе. Ово је најслободнија од свих лиценци.

2. **Ауторство – некомерцијално.** Дозвољаваате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца не дозвољава комерцијалну употребу дела.

3. **Ауторство – некомерцијално – без прерада.** Дозвољаваате умножавање, дистрибуцију и јавно саопштавање дела, без промена, преобликовања или употребе дела у свом делу, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца не дозвољава комерцијалну употребу дела. У односу на све остале лиценце, овом лиценцом се ограничава највећи обим права коришћења дела.

4. **Ауторство – некомерцијално – делити под истим условима.** Дозвољаваате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце и ако се прерада дистрибуира под истом или сличном лиценцом. Ова лиценца не дозвољава комерцијалну употребу дела и прерада.

5. **Ауторство – без прерада.** Дозвољаваате умножавање, дистрибуцију и јавно саопштавање дела, без промена, преобликовања или употребе дела у свом делу, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца дозвољава комерцијалну употребу дела.

6. **Ауторство – делити под истим условима.** Дозвољаваате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце и ако се прерада дистрибуира под истом или сличном лиценцом. Ова лиценца дозвољава комерцијалну употребу дела и прерада. Слична је софтверским лиценцама, односно лиценцама отвореног кода.