Primerjava učinkovitosti delovanja aplikacij v oblaku in v lokalnem okolju

Boris Ovčjak, Gregor Jošt, Gregor Polančič, Marjan Heričko
Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, Inštitut za informatiko, Smetanova 17, 2000 Maribor
{boris.ovcjak, gregor.jost, gregor.polancic, marjan.hericko}@uni-mb.si

Izvleček

Računalništvo v oblaku je vse bolj priljubljena metoda pri razvoju, izvajanju in uporabi informacijskih rešitev. Hitra vzpostavitev izvajalnega okolja, skalabilnost in poslovna agilnost so le nekatere prednosti, ki jih obljublja t. i. oblačna infrastruktura. Zato postaja računalništvo v oblaku vse bolj konkurenčno tradicionalnim metodam pri gostovanju in uporabi informacijskih rešitev.

Namen prispevka je primerjati lokalno strežniško okolje in okolje v oblaku z vidika učinkovitosti ter ugotoviti, pri kakšni konfiguraciji dosegata okolji podobno učinkovitost. Zato smo izvedli preizkuse obremenitve testne aplikacije v lokalnem okolju in v različnih konfiguracijah t. i. oblačnega okolja. Pri tem smo učinkovitost opredelili s sedmimi metrikami, ki smo jih pridobili iz standarda ISO/IEC 9126-2 in orodja za izvajanje obremenitev JMeter.

Rezultati testov obremenitev so pokazali, da je platforma kot storitev v primeru večjega števila primerkov učinkovitejša od lokalnega okolja. V primeru manjšega števila primerkov se je lokalno okolje izkazalo za učinkovitejše, zato smo v sklopu raziskave iskali tudi konfiguracijo, pri kateri se izenačita učinkovitosti delovanja obeh okolij. Pomembno vlogo pri zagotavljanju ustrezne učinkovitosti ima skalabilnost okolja, saj omogoča optimalno delovanje aplikacij ne glede na okoliščine; pri tem moramo upoštevati tudi stroške, ki naraščajo z zmogljivostjo platforme.

Ključne besede: računalništvo v oblaku, platforma kot storitev, učinkovitost, programska oprema kot storitev, Windows Azure.

Abstract

Comparsion of Local and Cloud Based Aplications Efficiency

Cloud computing represents an increasingly popular approach to the development, implementation and use of information technology (IT) solutions. Fast establishment of runtime environments, scalability and business agility are just some of the advantages that the cloud infrastructure promises to ensure. Therefore, cloud computing is becoming a competitive approach to the traditional hosting and usage of IT solutions.

The purpose of this paper was to compare the local server environment with the cloud environment in terms of effectiveness and to determine the configuration where both environments achieve similar efficiency. Therefore, we performed a load testing of the test application in the local environment and in different configurations of the cloud environment. We measured efficiency with seven performance metrics from the ISO/IEC 9126-2 standard and the load testing tool, JMeter.

The results of load testing indicate that in case of more instances, the platform as a service (PaaS) performed better than the local environment. However, the local environment has proven to be more efficient compared to low number of PaaS instances. While we compared the efficiency of both environments, we found that the local environment ranged between four and five instances of the cloud counterpart. We found that scalability has an important role in ensuring appropriate efficiency as it allows optimal application performance, regardless of the circumstances; however we need to take into consideration the costs of using the platform which grows with the capacity of the platform.

Key words: cloud computing, platform as a service, efficiency, software as a service, Windows Azure.

1 UVOD

Računalništvo v oblaku se omenja od leta 2007[1] in je še vedno najbolj aktualno področje informatike. Ključna prednost novega načina uporabe informacijske tehnologije je zmanjšanje investicij v strojno in programsko opremo, saj do storitev dostopamo kar prek omrežja (Gospodarska zbornica Slovenije, 2011).

Zaradi posrednega najema infrastrukture običajno nimamo stroškov z nadgradnjami in vzdrževanjem strojne opreme, kar zmanjša tudi obseg vzdrževalnega osebja. Sodoben tehnološki pristop omogoča razvoj novih in inovativnih storitev, kar posledično dviguje konkurenčnost.

[1] Razvidno iz grafa, pridobljenega s http://www.google.com/trends?q=cloud+computing.

O priljubljenosti računalništva v oblaku priča študija, ki jo je opravilo podjetje IBM maja 2011, ki je zajela več kot 3.000 globalnih direktorjev informatike (angl. CIO), med njimi jih je bilo deset tudi iz Slovenije. Rezultati so pokazali, da je 60 odstotkov organizacij pripravljenih sprejeti računalništvo v oblaku v naslednjih petih letih za potrebe poslovnega napredka in doseganja konkurenčnosti, kar je dvakrat več, kot je pokazala enaka študija iz leta 2009 (O'Malley & Tomasco, 2011).

Pomembnost paradigme je prepoznalo tudi raziskovalno podjetje Gartner Inc., saj je na seznam področij informacijske tehnologije, ki bodo imela v letu 2011 največjo strateško vlogo v poslovanju, na prvo mesto uvrstilo računalništvo v oblaku. V prihodnjih treh letih naj bi bili priča storitveno usmerjenim pristopom v oblaku. Po Gartnerjevih napovedih bodo ponudniki storitev ponujali zasebne oblake, ki bodo omogočali vključitev storitvenih tehnologij in metodologij v obliki implementacije znotraj podjetij (Pettey, b. d.).

Tudi v Sloveniji vse bolj spodbujajo razvoj aplikacij in storitev računalništva v oblaku. Pošta Slovenije je nedavno predstavila informacijsko storitev Digitalna pisarna, ki je namenjena poslovnim uporabnikom in je zasnovana na konceptu programske opreme kot storitve (Pošta Slovenije, 2011). Tudi Skupina Telekom Slovenije se strateško vključuje v računalništvo v oblaku z zagotavljanjem infrastrukture kot storitve (najem in upravljanje strežnikov) in programske opreme kot storitve (CRM, elektronska pošta itd.) (Siol.net, 2011). Gospodarska zbornica Slovenije (Združenje za informatiko in telekomunikacije) spodbuja razvoj rešitev za računalništvo v oblaku; sodeluje pri kompetenčnem centru Zavod e-Oblak, ki je tudi član evropskega združenja Euro Cloud Europe. Cilj delovanja Zavoda je razvoj mednarodno priznane blagovne znamke in uspešno trženje storitev na mednarodnih trgih (Gospodarska zbornica Slovenije, 2011).

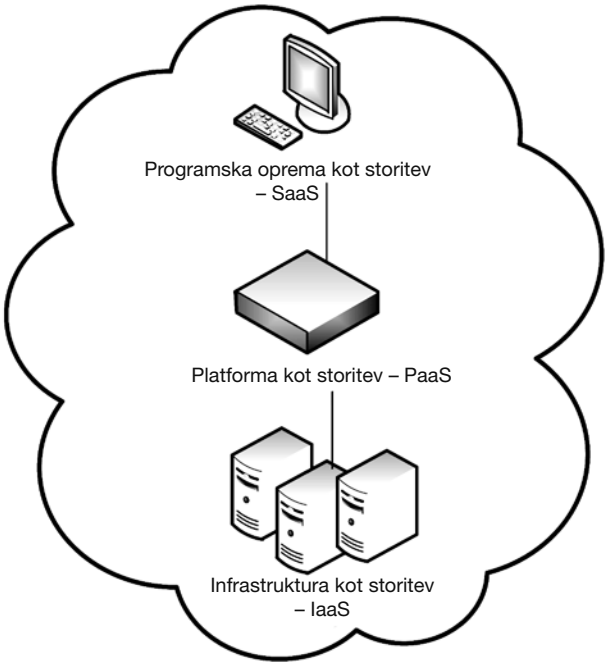
Kljub prednostim, ki jih prinaša računalništvo v oblaku, je vprašanje selitve v oblak še vedno aktualno. Glavni razlogi za selitev so hitra postavitev, manjši stroški in zmožnost skalabilnosti[2] (angl. scalability) aplikacij (Subashini & Kavitha, 2011), ki se prilagajajo potrebam. V prispevku želimo ugotoviti, kako prednosti računalništva v oblaku (predvsem skalabilnost) vplivajo na učinkovitost delovanja aplikacij v primerjavi z lokalnim okoljem.

Prispevek je urejen po razdelkih: v drugem smo definirali računalništvo v oblaku in platformo kot storitev ter podali pogled nekaterih večjih ponudnikov na omenjenem področju. Izpostavili smo ponudnika, ki smo ga vključili v raziskavo. V tretjem razdelku smo predstavili postopek primerjalne analize, vključno z aplikacijo, ki smo jo testirali v različnih konfiguracijah in okoljih. Dobljene rezultate smo interpretirali in podali sklepe. V zadnjem razdelku smo podali omejitve in uporabno vrednost raziskave ter načrte za naše nadaljnje delo.

2 RAČUNALNIŠTVO V OBLAKU

Čeprav uradne definicije računalništva v oblaku še ni (Mell & Grance, 2011), je inštitut The National Institute of Standards and Technology (v nadaljevanju NIST), ki skrbi za razvoj standardov, definiral računalništvo v oblaku kot *»model, ki omogoča vseprisoten, priročen in na zahtevo pridobljen mrežni dostop do bazena računalniških virov (npr. omrežja, strežnikov, shrambe, aplikacij in storitve)«* (Mell & Grance, 2011). Med prednosti računalništva v oblaku štejemo zmanjšanje stroškov, skalabilnost in neprekinjeno delovanje (Furht & Escalante, 2010).

Čeprav računalništvo v oblaku obravnava vse kot storitev (angl. Everything-as-a-Service), štejemo med glavne sloje arhitekture (Furht & Escalante, 2010): 1) infrastrukturo kot storitev (angl. Infrastructure-



Slika 1: **Sloji arhitekture računalništva v oblaku**

[2] Skalabilnost je sposobnost sistema, da prilagaja računalniške vire glede na trenutne potrebe aplikacije.

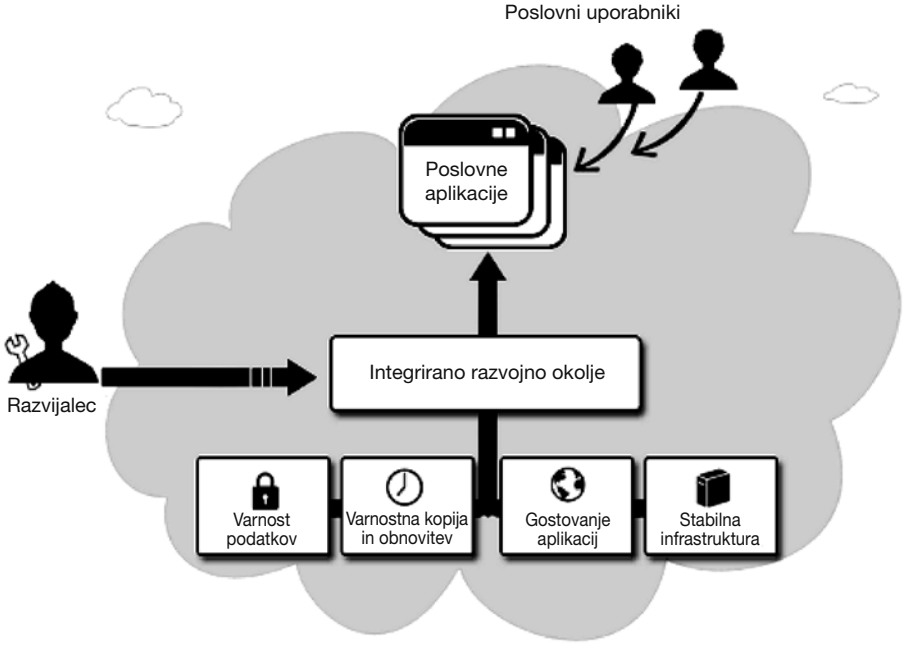
-as-a-Service, v nadaljevanju IaaS), 2) platformo kot storitev (angl. Platform-as-a-Service, v nadaljevanju PaaS) in 3) programsko opremo kot storitev (angl. Software-as-a-Service, v nadaljevanju SaaS) (slika 1).

IaaS je najnižji sloj, ki ponuja računalniške vire v obliki storitev. Sem štejemo virtualizacijo računalnikov z zajamčeno zmogljivostjo procesorja in pasovno širino za shrambo (angl. Storage bandwidth) ter dostop do omrežja. PaaS je nadgradnja IaaS, saj vključuje tudi operacijski sistem in storitve za specifične aplikacije. Na najvišjem sloju arhitekture računalništva v oblaku se nahajajo storitve informacijskih tehnologij, ki jim pravimo tudi SaaS. Te omogočajo, da uporabniki poganjajo aplikacije neposredno iz sloja oblaka PaaS (Furht & Escalante, 2010).

Ker bomo v sklopu prispevka primerjali učinkovitost delovanjeaplikacije v oblaku in v lokalnem okolju,se bomo osredotočili le na PaaS.

2.1 Platforma kot storitev

NIST definira platformo kot storitev (v nadaljevanju PaaS) kot *»možnost, da v infrastrukturo računalništva v oblaku postavimo uporabniško ustvarjene ali pridobljene aplikacije z uporabo programskih jezikov in orodij, ki jih podpira ponudnik. Uporabnik načeloma ne nadzoruje ali upravlja infrastrukture (mreža, strežniki, operacijski sistemi in shrambe), ima pa nadzor nad postavljenimi aplikacijami in potencialno tudi nad okolji za njihovo gostovanje«* (Mell & Grance, 2011). Slika 2 prikazuje koncept PaaS s pripadajočimi vlogami.



Slika 2: **Vloge v PaaS (ZOHO Creator, b. d.)**

PaaS omogoča razvijalcem popoln nadzor nad razvojem in postavitvijo aplikacij. Pri tem je proces izdaje (angl. release) programskih rešitev enostavnejši in hitrejši, saj je mnogo opravil, povezanih z izdajanjem programskih rešitev (npr. nastavitev gostovanja, strežnikov, podatkovnih baz in procesov uporabniške interakcije), že vnaprej pripravljenih (ZOHO Creator, b. d.). Glede na ponujene funkcionalnosti lahko definiramo dva tipa PaaS (Pavlinek, Košič, Jošt, Gradišnik & Ovčjak, 2011):

1) platforme za računalništvo v oblaku, ki prinašajo razvojna okolja in jih mora razvijalec še vedno lokalno namestiti, preden jih uporablja;
2) platforme, ki razvojna orodja selijo v oblak, dostopna pa so kot novodobna razvojna okolja.

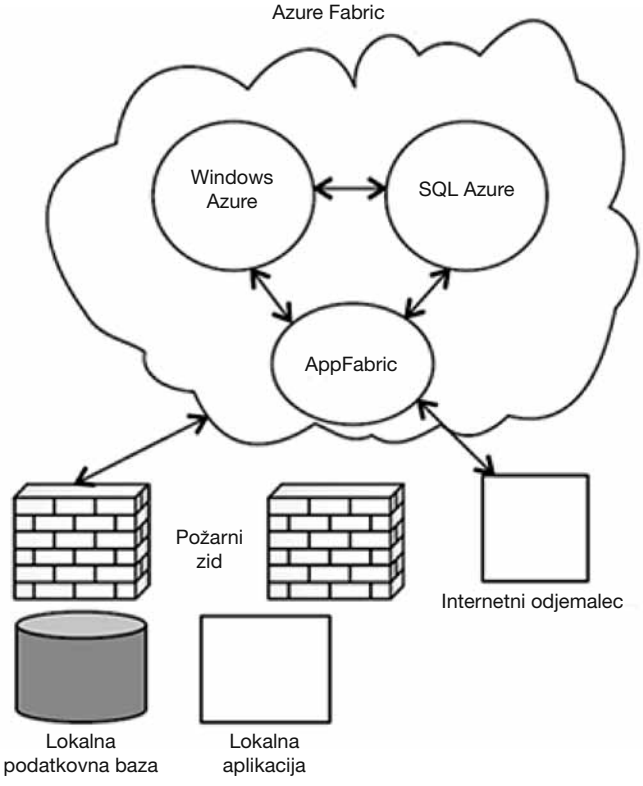
2.1 Ponudniki platforme kot storitve

Trenutno se na trgu pojavlja mnogo ponudnikov platform kot storitev, med katerimi med bolj prepoznavne štejemo (SearchCloudComputing.com, b. d.):

1) *Amazon Web Services* (AWS), ki je zbirka spletnih storitev, ki skupaj sestavljajo platformo kot storitev in vključujejo Amazon EC2, Amazon S3, Amazon Simple DB, Amazon SQS, Amazon FPS idr. AWS prinaša fleksibilno, skalabilno in cenovno ugodno računalniško platformo za podjetja vseh velikosti. Z AWS pridobimo dostop do zanesljive in varne tehnološke platforme, ki poganja tudi Amazonovo globalno spletno lastnino. Do AWS dostopamo prek protokolov HTTP ali SOAP, vse storitve pa zaračunavajo po porabi. Z uporabo platforme pridobimo dostop do infrastrukture podjetja Amazon in najamemo vire, ki jih potrebujemo za skaliranje svoje aplikacije. AWS ponuja primerke na zahtevo, pri čemer plačamo za računalniške vire po uri (AWS, b. d.);
2) *Google App Engine* (GAE) je platforma za razvoj in gostovanje spletnih aplikacij v Googlovih podatkovnih in aplikacijskih centrih. Omogoča izgradnjo in upravljanje aplikacij, zagotavlja podatkovne shrambe in skalabilnost v primeru rasti prometa. Platforma odpravi skrbi glede vzdrževanja infrastrukture in omogoča razvijalcem, da skrbijo izključno za svoje aplikacije. GAE trenutno podpira programske jezike, kot so Python, Java in Go ter posledično tudi druge jezike, ki temeljijo na JVM (angl. Java Virtual Machine). Mednje štejemo Groovy, JRuby, Scala, Clojur in Jython. GAE temelji na modelu plačila storitve glede na uporabo (angl. pay-as-you-go pricing), pri čemer uporabniki plačujejo le vire, ki jih porabljajo (Google, b. d.);
3) *Salesforce.com – Force.com* je produkt PaaS podjetja Salesforce. Začeli so kot ponudnik programske opreme kot storitve za upravljanje odnosov s strankami (angl. Customer relationship management, CRM), kasneje pa so razvili tudi platformo, imenovano Force.com. Lastnosti in filozofija Force.com se razlikujejo od ostalega tržišča ponudnikov PaaS, saj je Force.com ustvarjen specifično za poslovne aplikacije in temu primerno je prirejena tudi infrastruktura. Platforma vključuje tudi svoj programski jezik, imenovan Apex. Razvijalcem omogoča skriptiranje interakcij z drugimi funkcijami platforme, vključno z uporabniškim vmesnikom. Življenjski cikel razvoja aplikacij za Force.com je podoben razvoju spletnih aplikacij v lokalnem okolju. Za razliko od GAE Salesforce.com ponuja vnaprej zakupljene količine virov (Ouellette, b. d.);
4) Windows Azure (v nadaljevanju Azure) je Microsoftova rešitev na področju računalništva v oblaku. Ponuja dve možnosti obračunavanja: naročniško (angl. Subscription offers) in plačilo storitve glede na uporabo (Microsoft Corporation, 2011a). Ker smo se v eksperimentalnem delu članka omejili na Azure, bomo v nadaljevanju omenjeno platformo predstavili podrobneje.

2.2 Platforma Windows Azure

Platforma Azure se nahaja v zmogljivih Microsoftovih podatkovnih centrih. Platformo sestavljajo tri komponente(Dudley, 2010): operacijski sistem Windows Azure, SQL Azure in Windows Azure AppFabric. Slika 3 prikazuje medsebojne relacije komponent.



Slika 3: **Komponente platforme Windows Azure (Dudley, 2010)**

2.2.1 Operacijski sistem Windows Azure

Windows Azure je operacijski sistem platforme, ki omogoča poganjanje skalabilnih aplikacij na Microsoftovih strežnikih in podatkovnih centrih. Operacijski sistem v danem kontekstu pomeni, da Windows Azure opravlja dela, ki so značilna za konvencionalni operacijski sistem (Hay & Prince, 2010), med katera lahko štejemo gostovanje in izvajanje aplikacij, abstrakcijo strojne opreme in zagotavljanje vmesnika

med uporabniki ter aplikacijami. Glavna razlika med Windows Azure in konvencionalnim operacijskim sistemom je pri dodeljevanju procesorske moči in pomnilnika. Aplikacije, gostujoče na Windows Azure, se ne izvajajo vedno na enem strežniku, zaradi česar za dodeljevanje virov ne more biti zadolžen operacijski sistem platforme. To nalogo opravlja virtualni stroj, ki skrbi za porazdelitev storitev na različne strežnike.

Windows Azure v bistvu omogoča gostovanje spletnih aplikacij ali aplikacij, ki obdelujejo podatke v ozadju. Prav tako pa zagotavlja storitve, namenjene obdelavi (angl. Windows Azure Compute Services) in shrambi (angl. Windows AzureStorageServices) (Microsoft Corporation, 2010).

V primeru obdelave podatkov imamo opravka z dejansko programsko kodo [4]. Aplikacije so strukturirane v obliki vlog, ki jih poganjamo v izvajalnem okolju. V vsakem trenutku imamo lahko eno ali več primerkov določene vloge. Windows Azure podpira (Hay & Prince, 2010; Microsoft Corporation, 2010):

1) *spletno vlogo* (angl. Web Role), ki je namenjena razvoju spletnih aplikacij, skladno z IIS (Internet Information Services) 7 in ASP.NET;
2) *delovno vlogo* (angl. Worker Role), ki je uporabna za splošni razvoj in lahko skrbi za procesiranje (v ozadju) za potrebe spletne vloge. Delovna vloga je primerljiva s storitvami na lokalnih operacijskih sistemih;
3) *vlogo virtualne naprave* (angl. Virtual Machine Role), ki zagotavlja sliko (angl. image) sistema Windows Server 2008 R2, ki olajša selitev obstoječe aplikacije na Windows okolje Azure.

Storitve, namenjene shrambi, zagotavljajo trajno shrambo v oblaku. Med temeljne storitve štejemo (Hay & Prince, 2010; Microsoft Corporation, 2010):

1) zbirke binarnih podatkov (angl. Blob services), namenjene hranjenju besedilnih ali binarnih podatkov;
2) vrste (angl. Queue service), ki skrbijo za zanesljivo pošiljanje in izmenjavo sporočil med storitvami;
3) tabele (angl. Table services), ki so namenjene strukturirani shrambi podatkov. Teh storitev ne smemo enačiti z relacijsko podatkovno bazo, saj standardni jezik za povpraševanje (SQL) ni podprt [4].

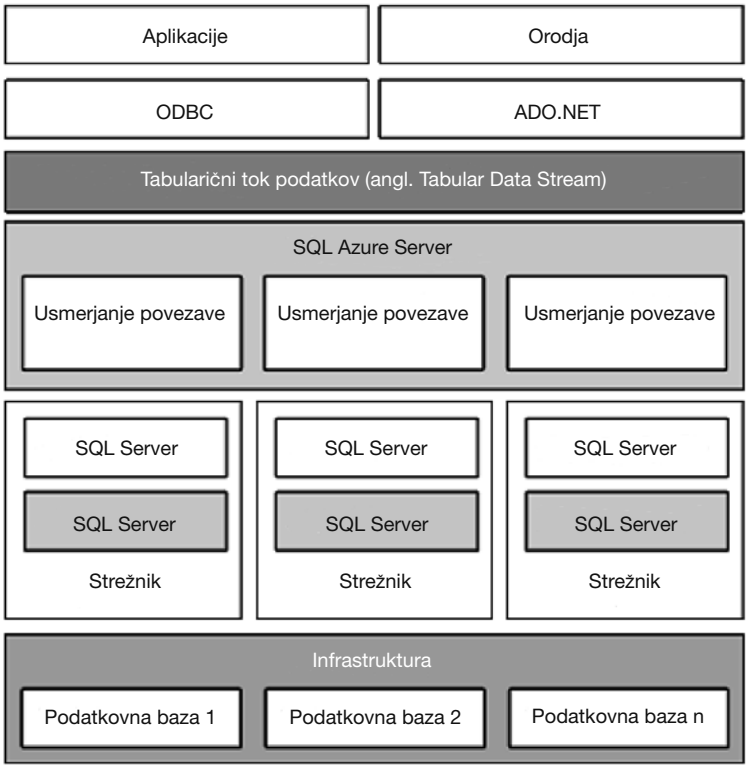
2.2.2 Podatkovne storitve SQL Azure

SQL Azure je skoraj popolna implementacija strežnika SQL Server 2008 [4] in ponuja storitve relacijske podatkovne baze. Nahaja se na platformi Azure in podpira relacijski model podatkovne baze, ki temelji na jeziku T-SQL (Microsoft Corporation, 2011b).

Podatkovno bazo SQL Azure lahko upravljamo prek 1) portala SQL Azure Portal, 2) ukazne vrstice sqlcmd ali 3) v okolju SQL Server Management Studia 2008 R2. Za uporabo SQL Azure ni treba lokalno ali v oblaku nameščati, nastavljati ali vzdrževati dodatne programske opreme (Microsoft Corporation, 2011b).

Skladno s koncepti platforme kot storitve omogoča SQL Azure tudi večnajemništvo (angl. multitenancy), plačilo storitve glede na uporabo in minimizacijo administracije infrastrukture (Microsoft Corporation, 2011b).

Izvedba SQL Azure se razlikuje od klasičnega sistema SQL Server. Pri povezavi na SQL Azure se ne povezujemo na fizični SQL Server, ampak na simulacijo strežnika. Gre torej za oblačne storitve, ki zagotavljajo večino značilnosti podatkovne baze SQL Server. SQL Azure deluje tako, da ovija (angl. wrapper) fizično infrastrukturo (slika 4) (Hay& Prince, 2010).



Slika 4: **Fizični pogled na strukturo SQL Azure (Hay & Prince, 2010)**

Do SQL Azure lahko dostopamo z ADO.NET, ODBC in programskim jezikom PHP. Kljub temu da SQL Azure temelji na SQL Server, se soočamo z nekaterimi omejitvami. Zaradi nivoja abstrakcije fizične implementacije npr. ne moremo uporabiti ukaza USE, prav tako niso podprti nekateri ukazi T-SQL, ki se sklicujejo na fizični nivo (npr. BACKUP).

Z omejitvami se soočamo tudi pri povezavi na SQL Azure. Ta namreč omogoča dostop le prek vrat (angl. port) 1433 (Hay & Prince, 2010). SQL Azure v splošnem podpira transakcije, vendar mu manjka podpora porazdeljenim transakcijam, saj vrata za njihovo izvedbo niso podprta.

Omejen je tudi časovni interval izvedbe ukazov in poizvedb. Ti ne smejo trajati dlje kot 30 minut, kolikor je privzeta časovna omejitev. Vse transakcije, ki trajajo dlje, se prekličejo (Hay & Prince, 2010).

2.2.3 Komponenta AppFabric

AppFabric je najstarejša komponenta Azure. Prvotno se je AppFabric imenoval BizTalk Services in pomeni dopolnilno oblačno ponudbo strežniku BizTalk.[3] Kasneje so BizTalk Services preimenovali v .NET Services in nato v Azure AppFabric z namenom, da bi komponento bolje povezali z Azure platformo (Hay & Prince, 2010).

AppFabric je knjižnica storitev in pomaga pri poganjanju oblačnih storitev ter povezovanju z zunanjimi programskimi okolji. Je varen most med obstoječimi (angl. legacy) aplikacijami in oblačnimi storitvami. AppFabric je sestavljen iz dveh ključnih delov (Dudley, 2010): 1) storitvenega vodila (angl. Service Bus) in 2) nadzora dostopa (angl. Access Control, v nadaljevanju ACS).

Storitveno vodilo zagotavlja vodilo v oblaku, ki služi za povezovanje storitev z odjemalci. Skladno s karakteristikami storitvenih vodil omogoča poimenovanje, odkrivanje, izpostavljanje in orkestracijo storitev neodvisno od njihovih fizičnih lokacij (Dudley, 2010). Storitveno vodilo pomeni način, kako povezati storitve in usmerjati sporočila. Prednost takšnega pristopa je, da lahko storitve povezujemo brez poznavanja povezovalne tehnologije (Hay & Prince, 2010). Storitveno vodilo abstrahira fizično lokacijo storitve in ponuja URI, prek katerega lahko prikličemo storitev.

V storitvenem vodilu potrebujemo še mehanizem, kako omejiti dostop do aplikacij, kar zagotavlja komponenta AppFabric ACS (angl. Access Control Service). ACS skrbi za varnost naših "oblačnih storitev" in aplikacij, saj abstrahira overjanje (Hay & Prince, 2010). S pomočjo ACS definiramo pravila (z uporabo storitev, ki temeljijo na Windows Communication Foundation), preko katerih lahko do oblačnih storitev dostopajo tudi drugi. ACS je podoben storitvam, kot so Open ID ali Live ID (Dudley, 2010).

3 PRIMERJALNA ANALIZA DELOVANJA APLIKACIJE V OBLAKU IN V LOKALNEM OKOLJU

V okviru primerjalne analize smo izvedli test obremenitve (angl. Stress testing) testne aplikacije v lokalnem okolju in v okolju Azure. Test obremenitve je namenjen ugotavljanju robustnosti, dosegljivosti in zanesljivosti aplikacije v skrajnih okoliščinah. Med te okoliščine štejemo velike obremenitve, sočasno delovanje in omejene vire. Namen testa obremenitve je identificiranje težav, povezanih z viri, s prioritetami itd. (Microsoft Corporation, 2007).

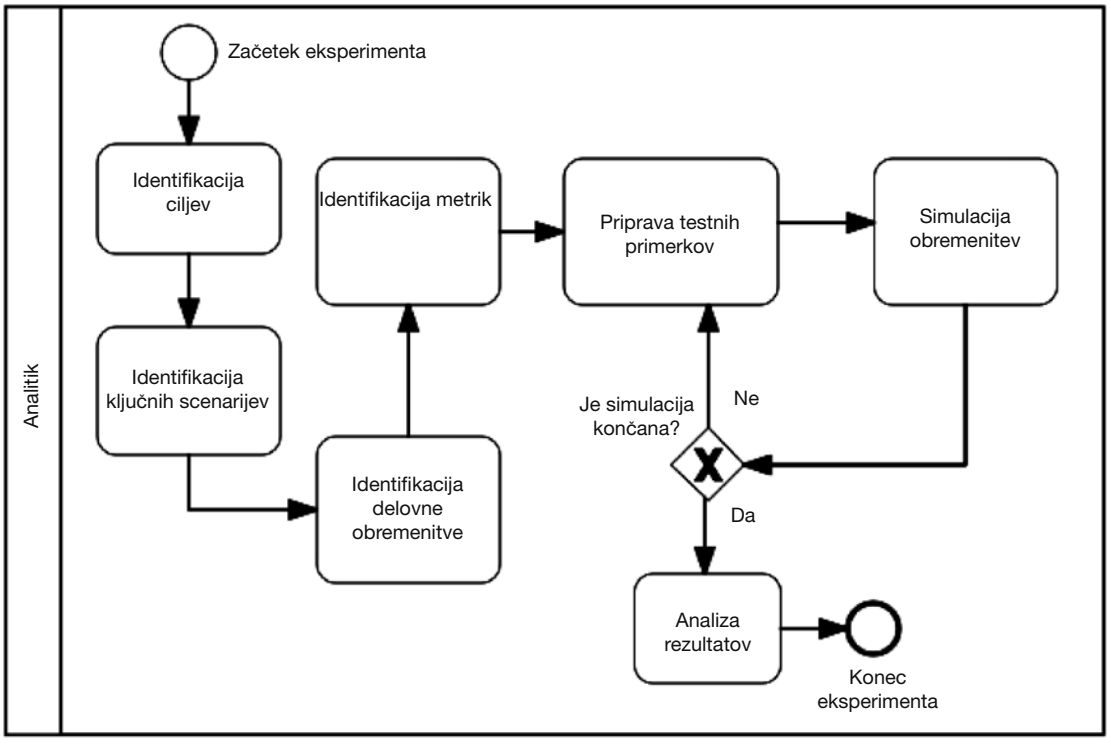
Pri testu obremenitve običajno izvajamo simulacije več scenarijev v različnih okoliščinah. V našem primeru smo se osredinili na test obremenitve spletnih aplikacij z namenom, da opazujemo učinkovitost delovanja aplikacij v skrajnih okoliščinah. Slika 5 prikazuje proces poteka primerjalne analize v notaciji BPMN.[4]

3.1 Opredelitev testne aplikacije

Za testiranje zmogljivosti različnih platform in konfiguracij smo pripravili testno aplikacijo, ki ima sicer enostavne, a procesorsko potratne funkcionalnosti. Glavni delež procesiranja smo tako delegirali na razčlenjevalnik XML, saj je obdelovanje dokumentov XML v splošnem procesorsko zahtevno opravilo (Nicola & John, 2003). Za potrebe testne aplikacije smo izbrali podatkovno zbirko iz portala SI-STAT, nad katero smo z uporabo omenjenega razčlenjevalnika izvajali operacije in s tem simulirali različne obremenitve.

Testna aplikacija ASP.NET je izdelana s programskim jezikom C#. V primeru lokalne infrastrukture smo lahko datoteko XML shranili na fizični lokaciji

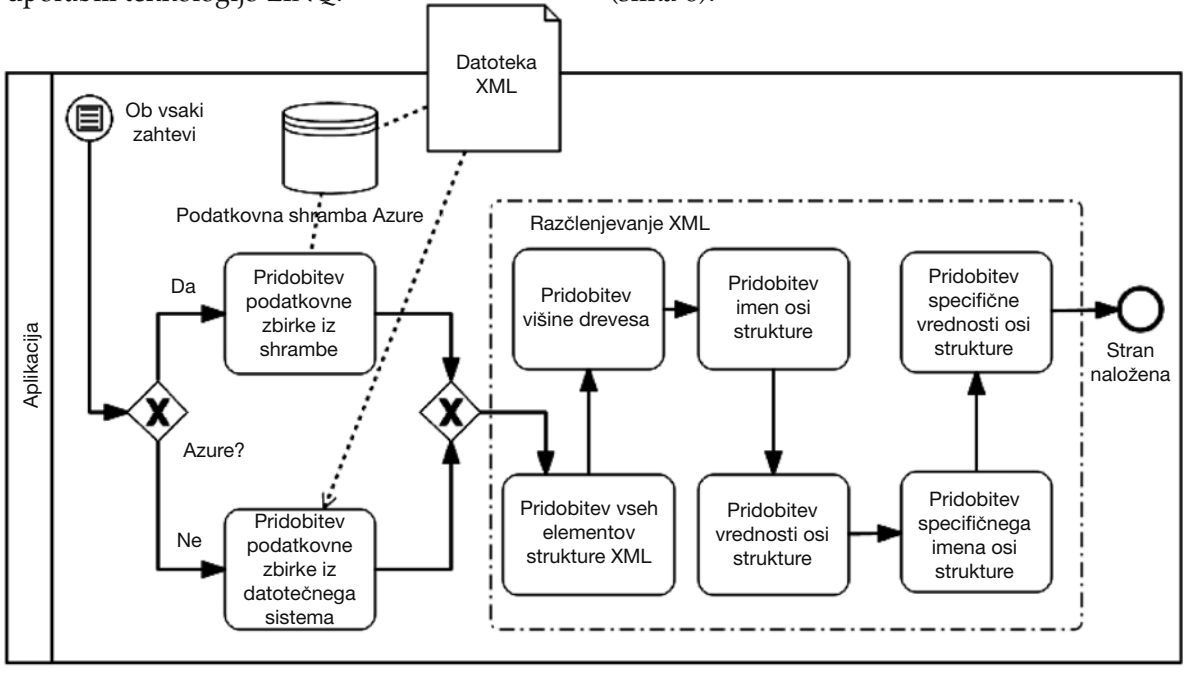
[3] BizTalk je strežnik za integracijo in izvajanje poslovnih procesov.
[4] BPMN (BusinessProcessModellingNotation) je standard za grafično modeliranje poslovnih procesov. Vir: http://www.bpmn.org/



Slika 5: **Model primerjalne analize BPMN**

(podatkovnem disku), za razliko od lokalne infrastrukture Azure, ki ne omogoča fizičnega shranjevanja. Datoteko XML smo zato shranili v nestrukturirano podatkovno shrambo (Azure Storage). V obeh primerih smo za prebiranje podatkov iz datoteke XML uporabili tehnologijo LINQ.

Postopek izvajanja primerjave je potekal tako, da je vsak od simuliranih uporabnikov izvedel poizvedbe HTTP na domačo stran spletne aplikacije in s tem sprožil pridobitev podatkovne zbirke ter šest različnih poizvedovalnih operacij po zbirki (slika 6).



Slika 6: **Model izvajanja testne aplikacije BPMN**

3.2 Predstavitev testnih okolij

Za potrebe izvedbe testov obremenitev potrebujemo dve izvajalni okolji za omenjeno aplikacijo. Ker je namen raziskave primerjati učinkovitost delovanja aplikacij v lokalni in oblačni infrastrukturi, smo želeli primerjati rešitve istega ponudnika, saj tako dobimo najbolj primerljive rezultate. Kot predstavnika tehnologije računalništva v oblaku smo izbrali platformo Azure, medtem ko je lokalno okolje predstavljala infrastruktura z operacijskim sistemom Windows Server 2008 R2 64-bit.

Definicija platforme kot storitve pove, da razvijalci oz. uporabniki nimajo neposrednega vpogleda v infrastrukturo, v kateri tečejo aplikacije oz. storitve. Te se fizično nahajajo na virtualnih napravah, pri čemer vsaka virtualna naprava gosti en primerek aplikacije. Azure ima na voljo pet različnih velikosti virtualnih naprav: ExtraSmall, Small, Medium, Large in Extra Large. Vsaka velikost ima vnaprej določene vire skupaj z jasno definiranimi specifikacijami (število jeder, količina spomina, diskovna kapaciteta in pasovna širina). Da bi zmanjšali vpliv nastavitev na rezultate primerjalne analize, smo tako v lokalnem okolju kot v primeru Azure uporabili privzeto nastavitev (velikost primerka Small). Ta zagotavlja za vsak primerek aplikacije oz. storitve procesorsko jedro s frekvenco 1,6 GHz, 1,75 GB spomina, 225 GB diskovne kapacitete in 100 Mbit pasovne širine (Microsoft Corporation, b. d.).

Za razliko od PaaS, pri kateri smo izbrali evropsko regijo, imamo v primeru lokalnega strežnika popoln nadzor nad infrastrukturo, operacijskim sistemom in njegovo optimizacijo. Za potrebe simulacije smo izbrali osebni računalnik z operacijskim sistemom Windows Server 2008 R2 64-bit. Tabela 1 prikazuje primerjavo konfiguracij obeh testnih okolij.

Tabela 1: **Primerjava konfiguracij infrastrukture okolja v oblaku in lokalnega okolja**

Atributi	Okolje v oblaku	Lokalno okolje
Število procesorskih jeder	2, 10 in 20	4
Zmogljivost procesorskih jeder	1,6 GHz	2,66 GHz
Pomnilnik (RAM)	1,75 GB	4 GB
Diskovna kapaciteta (GB)	225 GB	500 GB

3.3 Predstavitev metrik učinkovitosti

Pri definiranju metrik učinkovitosti smo se zgledovali po standardu ISO/IEC 9126-2. Standard se osredinja na ocenjevanje kakovosti produkta, in sicer na zunanje metrike, ki naslavljajo učinkovitost programske opreme. Za potrebe analize smo izbrali metrike (International Organization for Standardization, b. d.):

- **metrike časovnega obnašanja** (angl. Time behaviour metrics):
  - ***odzivni čas (ms)*** (angl. Response time) meri povprečni čakalni čas, ki preteče od podane zahteve do pridobitve rezultata pod specifičnimi obremenitvami sistema v smislu sočasnih nalog in sistemske uporabe (angl. utlilisation);
  - ***prepustnost (št./s)*** (angl. Throughput) meri povprečno število sočasnih nalog, ki jih lahko sistem obdela v določenem časovnem intervalu;
- **metrike uporabe virov** (angl. Resource utilisation metrics):
  - ***delež napak pri prenosu v določenem času (%)*** (angl. Mean of transmission error per time) meri delež napak med prenosom v določenem časovnem intervalu in pri specifični sistemski uporabi.

Poleg navedenih metrik smo uporabili še metrike izbranega simulacijskega orodja (Apache Jakarta Project, b. d.):

- **mediana (ms) –** je srednji čas množice rezultatov. Polovica primerov je pod to vrednostjo, druga polovica pa nad njo;
- **črta 90 % (ms) –** pove, da je 90 odstotkov primerkov vzelo vsaj toliko časa, kot je definirana vrednost, preostalih 10 odstotkov pa je trajalo dlje;
- **minimum (ms) –** je najkrajši čas zahteve množice primerkov;
- **maksimum (ms) –** je najdaljši čas zahteve množice primerkov.

3.4 Izvedba testov obremenitev

V fazi izvedbe testa obremenitve smo uporabili različno število primerkov aplikacije in primerjali učinkovitost delovanja pod različnimi obremenitvami. Naš uporabniški račun Azure je omogočal izkoriščanje dvajsetih procesorskih jeder. Na podlagi analize platforme smo za testne namene definirali: 1) dva primerka – Azure priporoča vsaj toliko za posamezno aplikacijo, saj s tem ob morebitnih težavah zagotovimo nemoteno delovanje aplikacij, 2) deset primerkov in 3) dvajset primerkov.

Za simulacijo obremenitev aplikacije smo uporabili programsko orodje *Apache JMeter* (v nadaljevanju JMeter). JMeter je odprtokodno orodje za merjenje obremenitev in učinkovitosti delovanja. Uporablja se za testi-

ranje delovanja statičnih in dinamičnih virov (datotek, servletov, Java objektov, podatkovnih baz itd.). Prav tako lahko izvajamo simuliranje obremenitev strežnika, omrežja ali objektov z namenom opazovanja stabilnosti in analize učinkovitosti delovanja pod različnimi obremenitvami (Apache Jakarta Project, b. d.).

V okviru analize smo pripravili več različnih scenarijev. Za vsako od okolij, v katerem je delovala aplikacija, smo simulirali obremenitev pri različnem številu odjemalcev (tabela 2).

Tabela 2: **Konfiguracije obremenitve aplikacije**

Atributi	Vrednosti
Število uporabnikov	200, 500, 800, 1000
Število zahtev	20.000
Zamik	5 sekund

Kot je razvidno iz tabele, smo simulirali 200, 500, 800 in 1000 odjemalcev. Za potrebe testiranja aplikacije smo najprej ugotovili maksimalno vrednost hkratno prisotnih uporabnikov, ki znaša 1000 (točka, na kateri so se začele pojavljati napake). Vrednost 500 smo določili kot polovično vrednost maksimalne (za lažjo primerjavo), medtem ko smo se z vrednostma 200 in 800 želeli bolj približati skrajnima vrednostma (0 in 1000) in obenem tudi zadostiti, da je med številoma faktor potence števila 2.

Za vsakega smo določili petsekundno zakasnitev med pošiljanjem zahtev, kar se ujema z običajnim časom zadrževanja uporabnika na spletni strani.[5]

Ker smo morali predhodno definirati število zahtev, smo najprej na lokalnem okolju izvedli simulacijo, ki je trajala 10 minut, kar smo postavili kot izhodiščno vrednost pri nadaljnjih testih.

3.5 Rezultati raziskave

Tabela 3 prikazuje rezultate testov obremenitev in je razdeljena na štiri sklope: lokalno okolje in okolje Azure za dve, deset in dvajset primerkov.

Tabela 3: **Rezultati meritev**

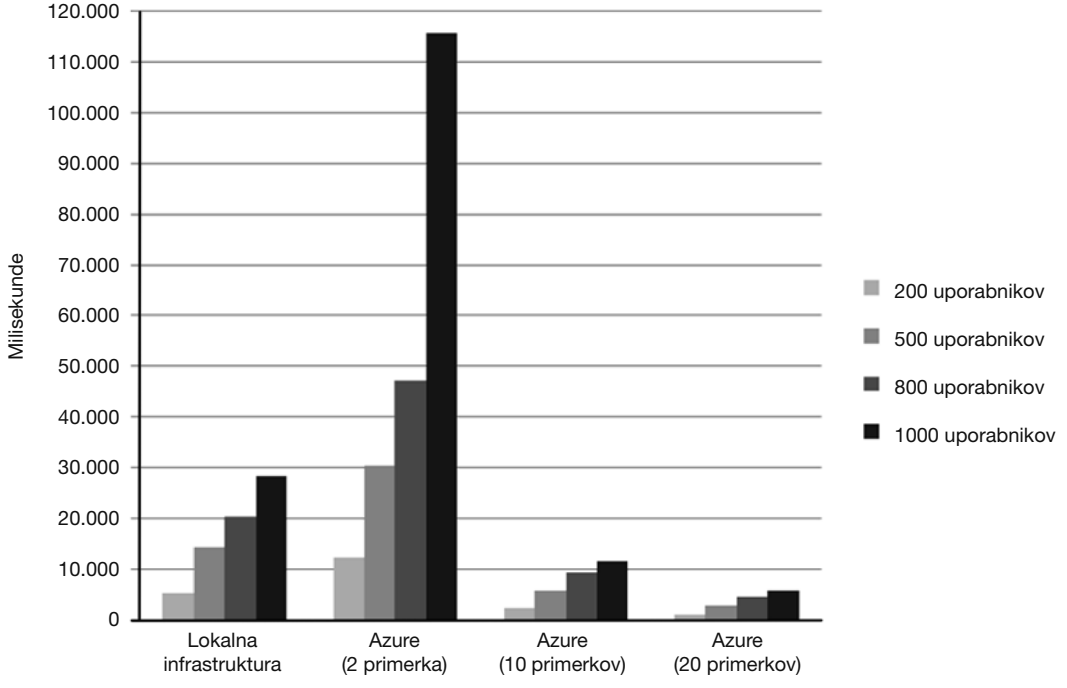
Atributi	**Vrednosti**			
Izvajalno okolje	**Lokalno okolje – IIS 7**			
Število primerkov/procesorskih jeder	**Štiri procesorska jedra**			
Število uporabnikov	**200**	**500**	**800**	**1000**
Povprečni odzivni čas (ms)	5.217,67	14.415,00	20.374,33	28.301,33
Napaka pri prenosu (%)	0,00	0,00	0,00	0,00
Prepustnost (št. zahtev/s)	38,65	34,43	39,00	35,80
Mediana (ms)	5.211,00	14.291,00	20.657,67	28.765,00
Črta 90 % (ms)	5.570,67	16.018,33	21.120,00	30.449,33
Minimalni odzivni čas (ms)	131,33	58,67	240,50	87,67
Maksimalni odzivni čas (ms)	8.829,67	22.540,00	25.509,67	36.326,33
Izvajalno okolje	**Azure**			
Število primerkov/procesorskih jeder	**Dva primerka**			
Število uporabnikov	**200**	**500**	**800**	**1000**
Povprečni odzivni čas (ms)	12.320,50	30.374,50	47.156,50	115.438,00
Napaka pri prenosu (%)	0,00	0,00	0,00	84,00
Prepustnost (št. zahtev/s)	14,75	16,10	16,90	8,50
Mediana (ms)	13.032,00	30.907,50	47.634,00	123.595,00
Črta 90 % (ms)	16.255,00	32.643,00	51.446,50	127.991,00
Minimalni odzivni čas (ms)	176,50	545,50	482,50	213,00
Maksimalni odzivni čas (ms)	75.466,50	25.892,50	117.318,50	169.093,00

[5] http://ezinearticles.com/?Reasons-Why-Your-Site-Has-Only-5-Seconds-to-Get-Someones-Attention&id=5674345.

Atributi	**Vrednosti**			
Izvajalno okolje	**Azure**			
Število primerkov/procesorskih jeder	**Deset primerkov**			
Število uporabnikov	**200**	**500**	**800**	**1000**
Povprečni odzivni čas (ms)	2.309,00	5.843,00	9.315,00	11.688,67
Napaka pri prenosu (%)	0,00	0,00	0,00	0,00
Prepustnost (št. zahtev/s)	81,80	81,10	75,80	76,97
Mediana (ms)	2.265,00	5.716,00	8.968,33	11.325,33
Črta 90 % (ms)	2.871,00	7.147,67	12.404,33	14.782,00
Minimalni odzivni čas (ms)	154,67	147,33	163,33	154,00
Maksimalni odzivni čas (ms)	15.819,67	57.620,67	90.997,00	107.191,67
Izvajalno okolje				**Azure**
Število primerkov/procesorskih jeder				**Dvajset primerkov**
Število uporabnikov	**200**	**500**	**800**	**1000**
Povprečni odzivni čas (ms)	1.175,00	2.881,33	4.587,67	5.838,67
Napaka pri prenosu (%)	0,00	0,00	0,00	0,00
Prepustnost (št. zahtev/s)	108,33	160,40	156,00	156,30
Mediana (ms)	1.159,67	2.804,00	4.418,00	5.567,67
Črta 90 % (ms)	1.501,00	3.811,00	5.993,67	7.776,00
Minimalni odzivni čas (ms)	145,67	144,67	147,67	150,33
Maksimalni odzivni čas (ms)	5.713,33	24.632,33	44.304,50	59.166,00

Kot je razvidno iz tabele, so med izvajanjem testa obremenitve nastopile napake le pri Azure v primeru dveh primerkov v skrajnih okoliščinah (1000 zahtevkov v razmiku petih sekund). Temu botruje privzeto nastavljena časovna omejitev na zahtevo platforme. Preostali podatki, ki smo jih pridobili s testi obremenitev, so podrobneje razčlenjeni v nadaljevanju.

Pri simulaciji obremenitev aplikacije smo najprej spremljali povprečni odzivni čas glede na število uporabnikov. Slika 7 prikazuje, kako se je povečeval odzivni čas z naraščanjem števila uporabnikov.



Slika 7: **Povprečni odzivni čas v odvisnosti od števila uporabnikov**

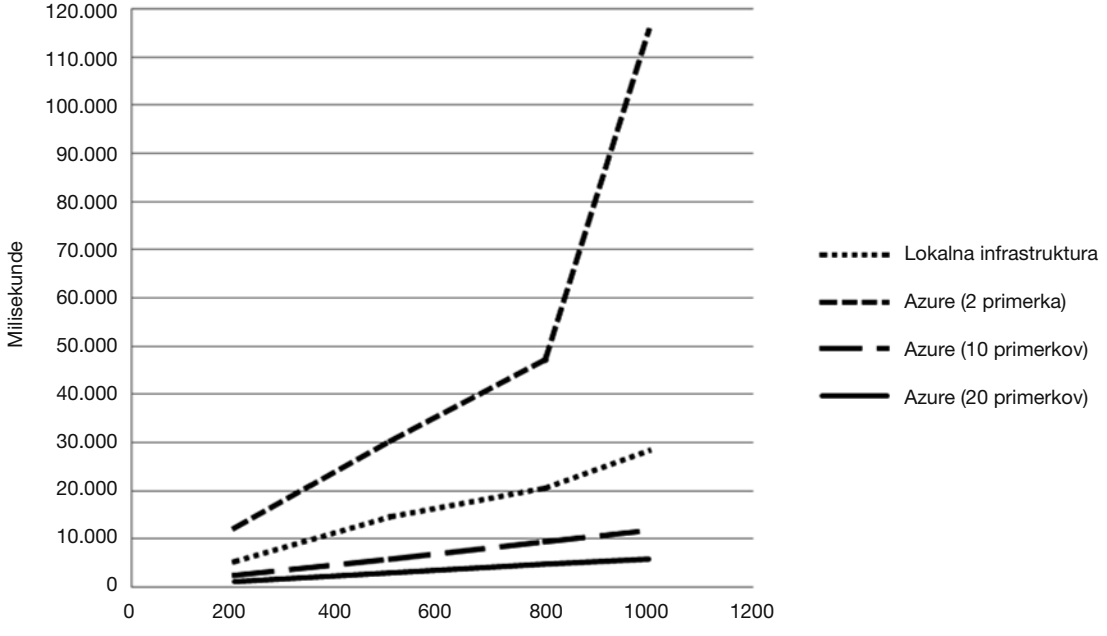
Iz slike 7 je razvidno, da je Azure v primeru dveh primerkov dosegel najvišji odzivni čas in se v skrajnem primeru približal meji dveh minut. Pri tem je treba upoštevati dejstvo, da dva primerka v našem primeru pomenita dvoje procesorskih jeder, kar je polovico manj kot v lokalnem okolju. Opazimo lahko, da se s številom primerkov (procesorskih jeder) odzivni čas opazno zmanjšuje in pri dvajsetih primerkih doseže signifikantno boljše rezultate tudi v primeru skrajnih obremenitev. Pri 1000 uporabnikih je povprečni odzivni čas v primeru Azure 5,83 sekunde, za razliko od infrastrukture lokalnega okolja, ki je pri enakih pogojih potrebovala 28,3 sekunde. Tudi v primeru desetih primerkov je bil povprečni odzivni čas 142 % krajši od lokalnega okolja (11,68 proti 28,3 sekunde).Tabela 4 prikazuje spremembe koeficientov povprečnega odzivnega časa.

Tabela 4: **Tabela sprememb koeficientov**

Koeficient spremembe povprečnega odzivnega časa				
Primerjava števila primerkov	200	500	800	1000
2 : 10	5,34	5,20	5,06	9,88
10 : 20	1,97	2,03	2,03	2,00

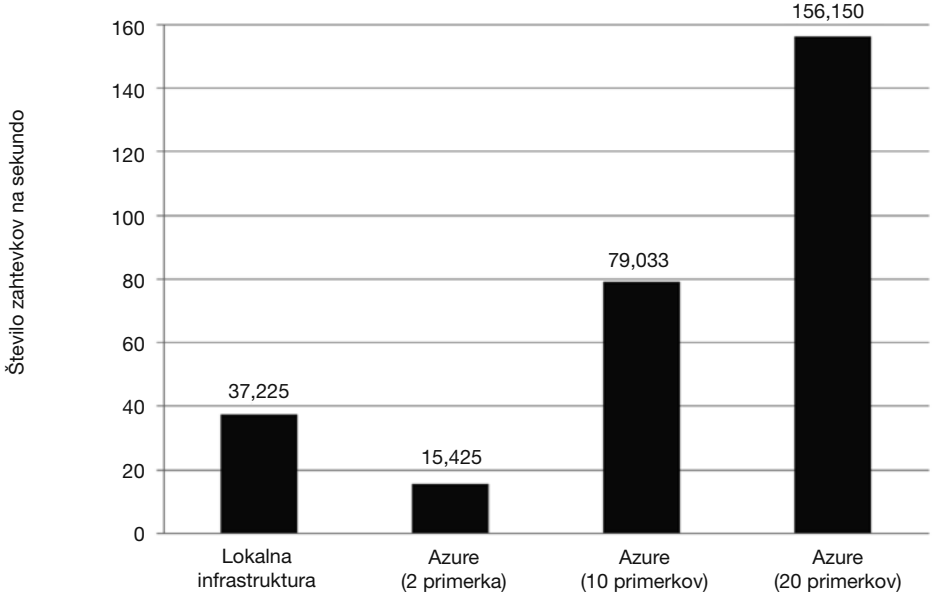
Kot je razvidno iz tabele, se pri petkratnem povečanju števila primerkov tudi koeficient spremembe povprečnega odzivnega časa zmanjša za faktor pet. Pri prehodu z 2 na 10 primerkov lahko opazimo, da je pri dveh primerkih prišlo do zasičenja, kar se odraža kot anomalija v rezultatih.

Slika 8 prikazuje linearno rast odzivnega časa v odvisnosti od število uporabnikov. Izjema je Azure pri dveh primerkih, pri katerih je odzivni čas pri 1000 uporabnikih zaradi preobremenitve presegel prednastavljeno časovno omejitev zahteve. Podatki so posledično izgubili zanesljivost, kar se odraža v obliki presežka odzivnega časa (število napak pri simulaciji).



Slika 8: **Povprečni odzivni čas za posamezna okolja**

Večanje števila primerkov aplikacije veča tudi prepustnost, ki raste sorazmerno s številom primerkov (slika 9).

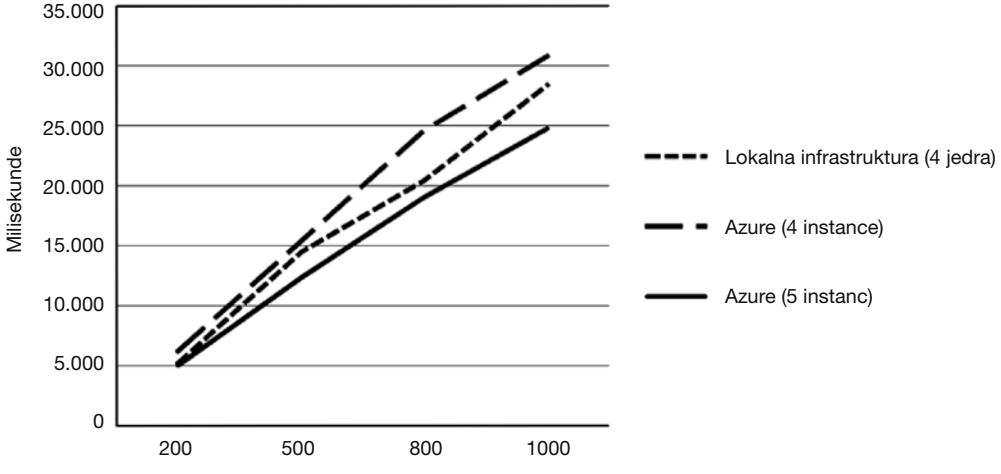


Slika 9: **Mediana prepustnosti**

Slika 9 prikazuje mediano (srednja vrednost) prepustnosti rezultatov simuliranih uporabnikov (200, 500, 800 in 1000). Največjo prepustnost je mogoče opaziti pri Azure s konfiguracijo 20 primerkov, pri čemer je platforma sposobna obdelati 156 zahtevkov na sekundo. Lokalno okolje se je izkazalo za boljše (37 zahtevkov na sekundo) od Azure pri konfiguraciji dveh primerkov (15 zahtevkov na sekundo), vendar pa je bil Azure pri konfiguraciji desetih primerkov za 114 % boljši (79 zahtevkov na sekundo).

V sklopu primerjave smo izvedli tudi simulacijo, s katero smo se želeli s konfiguracijo platforme kot storitve čim bolj približati zmogljivosti infrastrukture lokalnega okolja. Želeli smo ugotoviti konfiguracijo, pri kateri se obe okolji izenačita z vidika učinkovitosti delovanja. Upoštevati moramo, da je ena izmed omejitev primerjave, da nimamo neposrednega vpogleda v infrastrukturo platforme kot storitve. Zanašali smo se torej le na število procesorskih jeder, saj njihove zmogljivosti ne moremo preveriti (Microsoft Corporation, b. d.).

Infrastruktura lokalnega okolja uporablja štiri jedra, zato smo na Azure izbrali konfiguraciji s štirimi in petimi primerki ter primerjali učinkovitost delovanja (slika 10).



Slika 10: **Primerjava odzivnega časa konfiguracij okolij**

Iz slike 10 je razvidno, da je pri tisoč uporabnikih odzivni čas lokalnega okolja 28,3 sekunde, kar je za približno dve sekundi bolje od Azure pri konfiguraciji štirih primerkov (30,7 sekunde), čeprav bi naj imeli obe okolji v tem primeru enako število procesorskih jeder. Pri konfiguraciji petih primerkov je razlika v prid Azure, pri čemer je za približno 4 sekunde krajši povprečni odzivni čas (24,1 sekunde) od infrastrukture lokalnega okolja (28,3 sekunde). Tabela 5 prikazuje rezultate primerjav simulacije števila uporabnikov, iz katerih je razvidno, da je trend enak kot v primeru tisočih uporabnikov.

Tabela 5: **Primerjava odzivnih časov za posamezna okolja**

	Povprečni odzivni čas (ms)			
	Število uporabnikov			
Izvajalno okolje	**200**	**500**	**800**	**1000**
Lokalna okolje – IIS	5.217,67	14.415,00	20.374,33	28.301,33
Azure (4 primerki)	6.234,00	15.419,00	24.601,00	30.773,00
Azure (5 primerkov)	4.986,00	12.316,00	18.995,00	24.136,00

Iz grafa (slika 10) in tabele 5 je razvidno, da se pri opazovanju odzivnega časa aplikacija na infrastrukturi lokalnega okolja uvršča med aplikaciji v konfiguraciji oblaka s štirimi in petimi primerki. Rezultati kažejo, da smo lahko le približno ocenili neposredno primerjavo med okoljema.

3.3 Omejitve primerjalne analize

V prispevku smo se omejili na Windows Azure, ker smo imeli na voljo dostop do platforme, prav tako pa smo imeli možnost gostovati aplikacijo na infrastrukturi Windows Server 2008 R2. Tehnologiji istega ponudnika sta omogočali bolj enakovredno primerjavo med lokalnim in oblačnim okoljem.

Pri interpretaciji rezultatov moramo upoštevati, da bi lahko drugačne konfiguracije okolja Azure in lokalnega okolja vplivale na končni rezultat primerjalne analize. Kot smo že omenili, smo uporabili privzeto nastavitev Azure (majhna virtualna naprava), ki gosti posamezen primerek aplikacije.

Pri primerjavi smo se omejili le na razčlenjevanje datotek XML, zato ne moremo podajati ugotovitev za primere kakršnih koli drugih opravil. Prav tako nismo vključili podatkovne baze Azure SQL. Do datotek smo v primeru Azure dostopali prek shrambe Azure (Azure ne omogoča običajnega dostopa do datotek), medtem ko smo v primeru lokalnega okolja dostopali do datotek na disku. Upoštevati moramo, da lahko različna pristopa zajemanja podatkov vplivata na rezultate testa obremenitve.

4 SKLEP

Paradigma računalništva v oblaku bo tudi v prihodnje imela pomembno vlogo na področju informacijskih tehnologij. V njen prid govorijo predvsem te lastnosti: zniževanje stroškov, večja svoboda uporabnikov, centraliziranost, agilnost, varnost in preprostost upravljanja.

Namen članka je bil primerjati učinkovitost lokalnega okolja z okoljem v oblaku in preveriti, ali je skaliranje linearno. Porazdeljevanje prometa med primerki bi namreč lahko vplivalo na linearnost (Amdahlov zakon), zaradi česar ne moremo posplošiti, da je skaliranje linearno. Kljub temu smo pri dvajsetih primerkih ugotovili, da linearnost še vedno drži.

V raziskavi smo si prizadevali, da bi bili specifikaciji obeh okolij primerljivi, vendar zaradi različnih konfiguracij tega nismo uspeli doseči. To je narekovalo izbor privzete velikosti primerka, ki vključuje eno procesorsko jedro. Na podlagi tega smo želeli izvesti neposredno primerjavo glede na število procesorskih jeder. Prišli smo do spoznanja, da Azure v primeru štirih primerkov privzete konfiguracije ni ekvivalenten lokalnemu okolju, ki vsebuje štiri procesorska jedra, v primeru petih primerkov pa je njegova zmogljivost višja. Pri tem moramo upoštevati, da pri Azure vsak primerek aplikacije teče na svojem ločenem, oddaljenem strežniku, za razliko od lokalne infrastrukture, pri kateri imamo opravka z eno napravo.

Iz rezultatov je prav tako razvidno, da aplikacija, ki se izvaja na Azure, prinaša boljše rezultate v primeru večjega števila primerkov. Pri majhnem številu primerkov (dva primerka) se je učinkovitost delovanja aplikacije ustrezno zmanjšala. Za zagotavljanje optimalnega delovanja aplikacije moramo najti ravnovesje med številom primerkov in stroški uporabe platforme. V sklopu analize smo iskali tudi primerljivo konfiguracijo platforme kot storitve v primerjavi z infrastrukturo lokalnega okolja. Ugotovili smo, da sta okolji primerljivi pri petih primerkih privzete konfiguracije platforme Azure, vendar je podrobnejša primerjava otežena zaradi omejenih informacij o specifikaciji virov infrastrukture v oblaku.

V prihodnje nameravamo raziskave na področju učinkovitosti PaaS razširiti z vključevanjem podatkovne baze SQL Azure. Prav tako bo testna aplikacija

predstavljala realno poslovno okolje, učinkovitost delovanja pa bomo merili z uporabo kvalitativnih metrik. Na ravni platforme načrtujemo preizkusiti različne velikosti virtualnih naprav posameznih primerkov. Prav tako nameravamo izvesti primerjalno analizo učinkovitosti in lastnosti preostalih ponudnikov PaaS.

5 VIRI

[1] Apache Jakarta Project (b. d.)a. JMeter – User's Manual: Component Reference. Pridobljeno 25. 5. 2011 s http://jakarta.apache.org/jmeter/usermanual/component_reference.html#Aggregate_Report.

[2] Apache Jakarta Project (b. d.)b. JMeter – Apache JMeter. Pridobljeno 25. 5. 2011 s http://jakarta.apache.org/jmeter/.

[3] AWS (b. d.). AWS Free Usage Tier. Pridobljeno 25. 5. 2011 s http://aws.amazon.com/free/.

[4] Dudley, R. (2010). *Microsoft Azure enterprise application development: straight talking advice on how to design and build enterprise applications for the cloud*. Birmingham: Packt Publishing.

[5] Furht, B. & Escalante, A. (ur.) (2010). *Handbook of Cloud Computing*. Boston, MA: Springer US. Pridobljeno s http://www.springerlink.com/index/10.1007/978-1-4419-6524-0.

[6] Google (b. d.). What Is Google App Engine? Pridobljeno 25. 5. 2011 s http://code.google.com/intl/sl-SI/appengine/docs/whatisgoogleappengine.html.

[7] Gospodarska zbornica Slovenije (2011). Računalništvo v oblaku. Pridobljeno 20. 6. 2011 s http://www.gzs.si/slo/o_gzs/gzs_jeva_zgodba_v_casu/jutri/racunalnistvo_v_oblaku.

[8] Hay, C. & Prince, B. H. (2010). *Azure in action*. Greenwich Conn.; London: Manning; Pearson Education [distributer].

[9] International Organization for Standardization (b. d.). ISO – International Organization for Standardization. Pridobljeno 25. 5. 2011 s http://www.iso.org/iso/home.html.

[10] Mell, P. & Grance, T. (2011). The NIST Definition of Cloud Computing (Draft). Pridobljeno s http://csrc.nist.gov/publications/drafts/800-145/Draft-SP-800-145_cloud-definition.pdf.

[11] Microsoft Corporation (2007). Stress Testing Web Applications. Pridobljeno 25. 5. 2011 s http://msdn.microsoft.com/en-us/library/bb924374.aspx.

[12] Microsoft Corporation (2010). About Windows Azure. Pridobljeno 20. 6. 2011 s http://msdn.microsoft.com/en-us/library/dd179442.aspx.

[13] Microsoft Corporation (2011a). Windows Azure Pricing. Pridobljeno 5. 5. 2011 s http://www.microsoft.com/windowsazure/pricing/.

[14] Microsoft Corporation (2011b). The Next Generation Cloud Database. Pridobljeno 20. 6. 2011 s http://www.microsoft.com/windowsazure/sqlazure/database/.

[15] Microsoft Corporation (b. d.). Windows Azure Compute. Pridobljeno 31. 8. 2011 s http://www.microsoft.com/windowsazure/features/compute/.

[16] Nicola, M. & John, J. (2003). XML parsing. *Proceedings of the twelfth international conference on Information and knowledge management – CIKM '03* (str 175). Predstavljeno na The Twelfth International Conference, New Orleans, LA, USA. doi:10.1145/956863.956898.

[17] O'Malley, T. & Tomasco, S. (2011). New Global IBM Study Confirms Cloud Computing Poised to Take Off at Companies. Pridobljeno 3. 6. 2011 s http://www.03.ibm.com/press/us/en/pressrelease/34530.wss.

[18] Ouellette, J. (b. d.). Salesforce force.com platform. Pridobljeno 25. 5. 2011 s http://thecloudtutorial.com/salesforces-forcecom-platform.html.

[19] Pavlinek, M., Košič, K., Jošt, G., Gradišnik, M. & Ovčjak, B. (2011). Platforma kot storitev – med infrastrukturo in aplikacijo (str. 37–46). Predstavljeno na OTS 2011, Maribor.

[20] Pettey, C. (b. d.). Gartner Identifies the Top 10 Strategic Technologies for 2011. Pridobljeno 25. 5. 2011 s http://www.gartner.com/it/page.jsp?id=1454221.

[21] Pošta Slovenije (2011). Informacijske storitve Pošte Slovenije – Računalništvo v oblaku. Pridobljeno 20. 6. 2011 s http://www.posta.si/novica/26996/Informacijske-storitve-Poste-Slovenije-Racunalnistvo-v-oblaku?nodeid=493&page=1&year=0.

[22] Search Cloud Computing.com (b. d.). Verizon: 2010 top cloud computing provider. Pridobljeno 25. 5. 2011 s http://searchcloudcomputing.techtarget.com/feature/Top-cloud-computing-providers-Verizon#slideshow.

[23] Siol.net (2011). Računalništvo v oblaku – nebo je meja. Pridobljeno 20. 6. 2011 s http://www.siol.net/tehnologija/racunalnistvo/2011/05/telekom_racunalnistvo_v_oblaku.aspx.

[24] Subashini, S. & Kavitha, V. (2011). A survey on security issues in service delivery models of cloud computing. *Journal of Network and Computer Applications*, *34*(1), 1–11. doi:10.1016/j.jnca.2010.07.006.

[25] ZOHO Creator (b. d.). Platform as a Service, Cloud Computing Platform. Pridobljeno 25. 5. 2011 s http://www.zoho.com/creator/paas.html.

Boris Ovčjak je zaposlen na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Med njegove interesne dejavnosti spadajo področja odprtih podatkov javne uprave, računalništva v oblaku in mobilne tehnologije.

Gregor Jošt je raziskovalec in doktorski študent na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Med njegove interesne dejavnosti spadajo področja upravljanja poslovnih procesov, računalništva v oblaku in storitveno usmerjene arhitekture.

Gregor Polančič je docent na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Med njegova interesna področja spadajo tehnološki in netehnološki vidiki sistemov za komuniciranje, sodelovanje in upravljanje informacijskih procesov vključno z implikacijami sodobnih storitveno usmerjenih informacijskih rešitev z omenjena področja.

Marjan Heričko je redni profesor za informatiko na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Je namestnik predstojnika Inštituta za informatiko ter vodja Laboratorija za informacijske sisteme. Njegovo raziskovalno delo zajema vsa področja razvoja sodobnih informacijskih rešitev in storitev s poudarkom na naprednih pristopih k modeliranju in načrtovanju informacijskih sistemov, načrtovalskih vzorcih in metrikah. V zadnjem obdobju se posveča tudi področjem storitvene znanosti in storitvenega inženirstva ter uporabniško usmerjenega razvoja.