Peter Zagoranski

IZDELAVA PROGRAMA V OBLAČKU

Diplomsko delo

Maribor, november 2009

Diplomsko delo visokošolskega strokovnega študijskega programa

IZDELAVA PROGRAMA V OBLAČKU

Študent:	Peter Zagoranski
Študijski program:	visokošolski, Računalništvo in informatika
Smer:	Programska oprema
Mentor:	izr. prof. dr. Milan Ojsteršek

Maribor, november 2009

FERI
Fakulteta za elektrotehniko, računalništvo in informatiko
Smetanova ulica 17
2000 Maribor

Številka: RI.0198
Datum in kraj: 02. 10. 2009, Maribor

Na osnovi 330. člena Statuta Univerze v Mariboru (Ur. l. RS, št. 90/2008)

SKLEP O DIPLOMSKEM DELU

1. **Petru Zagoranskemu,** študentu visokošolskega strokovnega študijskega programa RAČUNALNIŠTVO IN INFORMATIKA, smer PROGRAMSKA OPREMA, se dovoljuje izdelati diplomsko delo pri predmetu Porazdeljeni računalniški sistemi.

2. **MENTOR: izred. prof. dr. Milan Ojsteršek**

3. **Naslov diplomskega dela:**
IZDELAVA PROGRAMA V OBLAČKU

4. **Naslov diplomskega dela v angleškem jeziku:**
BUILDING APPLICATION IN THE CLOUD

5. Diplomsko delo je potrebno izdelati skladno z "Navodili za izdelavo diplomskega dela" in ga oddati v treh izvodih ter en izvod elektronske verzije do 02. 10. 2010 v referatu za študentske zadeve.

Pravni pouk: Zoper ta sklep je možna pritožba na senat članice v roku 3 delovnih dni.

Dekan:

UNIVERZA V MARIBORU
FAKULTETA ZA ELEKTROTEHNIKO, RAČUNALNIŠTVO IN INFORMATIKO

Obvestiti:
- kandidata,
- mentorja,
- odložiti v arhiv.

ZAHVALA

Zahvaljujem se mentorju izr. prof. dr. Milanu Ojsteršku za pomoč in vodenje pri opravljanju diplomskega dela.

Posebna zahvala velja staršem, ki so mi omogočili študij.

IZDELAVA PROGRAMA V OBLAČKU

Ključne besede: računalništvo v oblaku, oblak, Windows Azure, spletni dnevnik, oxite, infrastruktura kot storitev, platforma kot storitev, programska oprema kot storitev

UDK: 004.4:004.738.5(043.2)

Povzetek

V predstavitvi so opisani koncepti računalništva v oblaku. Najprej smo predstavili klasifikacijo po nivojih, na katerih lahko uporabljamo storitve v oblaku, in si pri tem ogledali ponudbe trenutno najpomembnejših ponudnikov posameznih nivojev. V nadaljevanju smo se osredotočili na ponudbo plaftorme Windows Azure, saj smo z njeno pomočjo razvili aplikacijo, ki se izvaja v oblaku. Pri tem smo preučili storitve, katere nam trenutno ponuja platforma Windows Azure, in si zraven tega ogledali še, kakšne so vizije zaračunavanja za posamezne storitve. Končni produkt diplomske naloge je bila spletna aplikacija, ki se izvaja v oblaku in za svoje delovanje uporablja množico storitev omenjene platforme.

BUILDING APPLICATION IN THE CLOUD

Key words: cloud computing, cloud, Windows Azure, blog, oxite, Infrastructure as a Service, Platform as a Service, Software as a Service

UDK: 004.4:004.738.5(043.2)

Abstract

The present dissertation deals with the concepts of cloud computing. As there are several recognized layers in cloud computing, we first introduced classification by layers and studied the most important vendors offering services in one or more of the layers. In addition, we focused on the Windows Azure platform. We used Windows Azure to develop an application that runs in the cloud. We studied services currently offered by Windows Azure platform and pricing strategies for each service. The final result of the present dissertation is a web application that runs in a cloud, using a wide range of services provided by the above-mentioned platform.

VSEBINA

UPORABLJENE KRATICE, IZRAZI IN SIMBOLI

$	- Dolar
ADO.NET	- ActiveX Data Objects
AJAX	- Asynchronous JavaScript + XML
API	- Application Programming Interface
ASP.NET	- Application Service Provider
AWS	- Amazon Web Services
BlogML	- Blog Markup Language
BLOB	- Binary large object
CTP	- Community Technology Preview
CRM	- Customer relationship management
EaaS	- Everything as a Service
EC2	- Amazon Elastic Computer Cloud
IaaS	- Infrastructure as a Service
IIS	- Internet Information Services
IT	- Information Technology
JSON	- JavaScript Object Notation
GAE	- Google App Engine
GB	- Gigabyte
HTTP	- Hypertext Transfer Protocol
HTTPS	- Hypertext Transfer Protocol Secure
LINQ	- Language Integrated Query
MB	- Megabyte
Ms-PL	- Microsoft Public License
MVC	- Model View Controller

.NET	- Microsoft software framework
ODBC	- Open Database Connectivity
PaaS	- Platform as a Service
PHP	- Hypertext Preprocessor
REST	- Representational state transfer
RSS	- Really Simple Syndication
SaaS	- Software as a Service
SOAP	- Simple Object Access Protocol
SQL	- Structured Query Language
TB	- Terabyte
TDS	- Tabular Data Stream
ZIP	- Popularni format za stiskanje podatkov
WCF	- Windows Communication Fundation
XML	- Extensible Markup Language

1 UVOD

»Računalništvo v oblaku« je ena izmed smeri razvoja računalništva. Izraz »Oblak« se v tej besedni zvezi uporablja kot prispodoba za internet in izvira iz simbola oblaka, ki ga najpogosteje uporabljamo v diagramih, ko bi radi ponazorili globalno omrežje. Prav tako pa je simbol oblaka tudi zelo dobra abstrakcija za predstavitev zapletene infrastrukture, ki se skriva v ozadju.

Selitev računalništva v oblak bi lahko primerjali z revolucijo razvoja električne industrije kakšno stoletje nazaj, ko je še vsako gospodinjstvo z lastnimi generatorji ustvarjalo energijo za lastne potrebe. Z izgradnjo električnega omrežja so gospodinjstva izključila lastne generatorje in pričela kupovati električno energijo od obstoječih električnih distributerjev. Razlogi za to so bili, da ekonomija obsega pač zagotavlja, da lahko specializirani ponudniki zagotavljajo boljšo, cenejšo in zanesljivejšo storitev.

Podobna vizija se kaže tudi v računalništvu, katere namen je umik zahtevnih računalniških operacij in podatkovnih zbirk iz namiznih in prenosnih računalniških naprav na oddaljene podatkovne centre (strežnike), kjer sta dostopnost in zmogljivost na virtualiziranih platformah bistveno večja od posameznega računalnika. Primerjamo lahko tudi model zaračunavanja storitev, saj v obeh primerih plačamo le toliko kolikor porabimo.

Poleg razvoja virtualizacije je razvoju storitvenega modela v računalništvu zagotovo največ doprinesel razvoj hitrih, zanesljivih in razširljivih internetnih povezav, ki je bil dolgo časa eden izmed glavnih mankajočih členov v tej verigi ponube računalništva v oblaku.

1.1 Namen in cilj tega dela

Namen tega dela je predstaviti trenutne glavne platforme »računalništva v oblakih« (Amazon Web Services, Google App Engine, Windows Azure, VMWare in SalesForce) in si ogledati prednosti in slabosti posamezne platforme ter razlike med posameznimi ponudbami. Največjo pozornosti bomo posvetili Microsoftovi platformi Windows Azure.

Cilj tega dela je raziskovanje in izdelava aplikacije kot storitve, ki bo tekla v oblaku na Microsoftovi platformi Windows Azure. Kot primer za selitev aplikacije v oblak bomo uporabili obstoječo aplikacijo za pisanje osebnega spletnega dnevnika in program ponudili kot storitev v oblaku za končne uporabnike

1.2 Struktura dela

Delo je sestavljeno iz šestih poglavij. V prvem poglavju smo opisali od kod ideja za poimenovanje. Prav tako smo si pogledali vizijo razvoja in to vizijo primerjali z revolucijo razvoja elektične industrije.

V drugem poglavju si bomo pogledali splošne lastnosti računalništva v oblaku. Prav tako si bomo pogledali katere so njegove prednosti, ki so prispevale k hitremu razvoju in katere so njegove slabosti, ki bi lahko odvrnile uporabnike od uporabe.

V tretjem poglavju si bomo ogledali katere ponudbe nivojev storitev so se uveljavile pri modelu računalništva v oblaku. Omenili bomo tri glavne nivoje in našteli ponudnike posameznega nivoja.

V četrtem poglavju bomo predstavili trenutne glavne platforme računalništva v oblakih. Predstavili bomo: Amazon Web Services, Google App Engine, Windows Azure, VMWare in SalesForces.

V petem poglavju si bomo pogledali zgradbo platforme Windows Azure, opisali storitve, ki jih trenutno ponuja in si ob zaključku poglavja še ogledali kakšna je vizija zaračunavanja za posamezne storitve.

V šestem poglavju si bomo pogledali uporabo »računalništva v oblaku« v praksi. Opisali bomo namreč korake, ki so bili potrebni, da smo izvajanje obstoječe aplikacije za vodenje spletnega dnevnika preselili v oblak.

2 SPLOŠNO O RAČUNALNIŠTVU V OBLAKU

Kaj pravzaprav je »oblak«? Če bi želeli poenostaviti bi lahko rekli, da je »oblak« skupek računalnikov in strežnikov, ki ponujajo različne storitve, do katerih dostopamo s pomočjo interneta. Glavni namen tega modela je ponuditi uporabnikom visoko zmogljive, zanesljive, prilagodljive in cenovno ugodne računalniške storitve, do katerih dostopamo s pomočjo enostavnih spletnih vmesnikov. Kot lahko vidimo na sliki 1, se uporabniki do teh spletnih vmesnikov oz. do storitev v oblaku povezujejo s pomočjo raznovrstnih računalniških naprav in tako vidijo »oblak« kot aplikacijo, napravo ali dokument. Notranjost »oblaka« oz. strojna oprema in operaciski sistem, ki skrbijo za delovanje pa so za uporabnike nevidni oz. nepomembni.

Slika 1: Povezovanje z storitvami v oblaku

Arhitektura »oblaka« je navzven videti zelo enostavna, vendar se v notranjosti skriva kar zahtevna kompleksnost. Za uspešno delovanje potrebujejo namreč upravljalci oz.

ponudniki kar veliko znanja, da združijo moči vseh teh računalniških naprav in porazdelijo procesiranje opravil na množico uporabnikov. Infrastruktura, ki sestavlja »oblak« je tako last tretjih oseb oz. ponudnikov, ki to infrastrukturo tudi vzdržujejo. Trenutno imamo na trgu množico ponudnikov, ki ponujajo uporabo storitev v tako imenovanem »javnem oblaku« (public cloud), vendar to ni edina možnost za uporabo, kot alternativa internetni uporabi se je pojavil tudi pojem uporabe zasebnega oblaka (private cloud) in pojem uporabe hibridnega oblaka (hybrid cloud). Namen zasebnega oblaka je, da emulira računalništvo v oblaku znotraj lastnih zasebnih omrežij (s pomočjo virtualizacije). O hibridnem pristopu pa govorimo kadar za delovanje uporabljamo kombinacijo zasebnega in javnega oblaka. V diplomskem delu smo se osredotočili na ponudbo oz. uporabo »javnega oblaka«.

Velikokrat se nam zgodi da pojem »cloud computing« zamenjujemo z mrežnim računalništvom (grid computing), ki je oblika porazdeljene obdelave večjih opravil na »virtualnem superračunalniku«, ki je sestavljen iz rahlo povezanih računalnikov, kar pa ni isto kot model, ki nam ga prinaša računalništvo v oblaku.

2.1 Prednosti in slabosti

Kot vsaka stvar ima tudi »računalništvo v oblaku« določene prednosti in določene slabosti. Tako si bomo v tem poglavju pogledali katere so te prednosti, ki so pripomogle k tako hitremu razvoju storitvenega modela računalništva v oblaku in katere so njegove slabosti, ki bi lahko upočasnile oz. celo odvrnile uporabnike od uporabe.

2.1.1 Prednosti

- **Povečana zmogljivost**. Prav gotovo je to ena izmed glavnih prednosti uporabe računalništva v oblaku. Saj pri uporabi nismo več omejeni na zmogljivost enega samega računalnika ampak imamo na razpolago zmogljivost celotnega oblaka.
- **Finančna ugodnost**, pridobiva na svoji vrednosti, ker so virtualni viri v »oblaku« cenejši, kot če bi imeli namenski fizični strežnik. Uporabniki tako ne potrebujejo

več dragih visoko-zmogljivih sistemov za poganjanje aplikacij ampak jim že zadoščajo nezmogljivi sistemi, ki pa so mnogo cenejši.

- **Takojšnji popravki**. Pri uporabi aplikacije v oblaku uporabljamo vedno zadnjo verzijo aplikacije z vsemi popravki, kateri se nameščajo samodejno in nam tako ni potrebno skrbeti za nalaganje ali celo za plačevanje popravkov.
- **Neomejen prostor za shranjevanje podatkov**. Eden glavni razlogov za uporabo »računalništva v oblaku« je zagotovo shranjevanje podatkov v »oblak«. Pri shranjevaju podatkov v »oblak« so ti shranjeni na množici strežnikov širom sveta, namesto na namenskemu strežniku, ki se uporablja pri tradicionalnem shranjevanju podatkov na podatkovne strežnike.
- **Povečana varnost podatkov**. Kot smo že omenili pri računalništvu v oblaku shranjujemo naše podatke v oblak, kateri pa te podatke tudi replicira znotraj oblaka. Tako smo zašičiteni tudi, če pride do nepredvidljivega brisanja ali do poškodb na strojni opremi, saj se ti podatki nahajajo na več lokacijah.
- **Povečana združljivost med operacijskimi sitemi**. Zagotovo se nam je že velikokrat zgodilo, da smo želeli deliti podatke med računalniki z nameščenimi različnimi operacijskimi sistemi, kar zna biti včasih kar prava nočna mora. Pri uporabi računalništvu v oblaku se teh težav znebimo, saj se lakho s podatki v oblaku povežemo neglede na to kakšen operacijski sistem imamo nameščen.
- **Povečana združljivost med formati dokumentov**. Kot primer si lahko vzamemo dokument ki je ustvarjena na sistemu z nameščenim Wordom 2007 in dokument ki je ustvarjena na sistemu z nameščenim Wordom 2003. Prve datoteke namreč ne bomo mogli odpreti na slednjem sistemu. Dokumente ustvarjene z aplikacijo v oblaku pa bomo lahko odprli ne glede na to od kod dostopamo do dokumentov.
- **Poenostavljeno delo v skupinah**. Deljenje dokumentov nas velikokrat pripelje do skupne uporabe dokumentov med več uporabniki. Mnogim uporabnikom se ta prednost zdi ena izmed najpomembnejših, saj omogoča enostavno skupno uporabo istih dokumentov. Kot primer si lahko vzamemo podjetje, ki ima več poslovnih enot. Pred uvedbo računalništva v oblaku so si te poslovne enote med seboj pošiljale dokumente po elektronski pošti, sedaj to več ni potrebno, saj lahko ne glede na lokacijo množično dostopamo do istih dokumentov.

- **Enoten dostop do dokumentov**. Zagotovo se nam je že velikokrat zgodilo, da smo se vrnili iz službe in smo ugotovili, da smo si pozabili shraniti dokument, ki bi ga trenutno potrebovali. Tem težavam se lahko izognemo z uporabo shranjevanja podatkov v oblak. Tako ne nosimo več podatkov s sabo ampak jih rajši shranimo v oblak, kjer nas ti čakajo ko jih želimo spet uporabiti ne glede na to od kod dostopamo do njih.
- **Zadnja verzija dokumenta**. V oblaku je vedno shranjena zadnja verzija dokumenta. Tako se nam ne more zgoditi, da bi kdaj uporabili verzijo dokumenta, ki je že zastarela.
- **Neodvisnost od specifične naprave**. Pri svojem delu, nismo omejeni na specifično napravo, saj lahko do aplikacije ali dokumentov dostopamo iz katerekoli naprave, ki ima nameščen spletni brskalnik.
- **Zmanjšana potreba po IT strokovnjakih**. Kot smo že omenili ponudnik storitev skrbi za upravljanje in vzdrževanje računalniške infrastrukture, tako nam ni potrebno imeti znotraj podjetja dodatnih IT strokovnjakov.

2.1.2 Slabosti

- **Potreba po konstantni internetni povezavi.** Za računalništvo v oblaku je internet bistvenega pomena, saj preko njega dostopamo do aplikacij in dokumentov, ki jih uporabljamo pri našem delu. Enostavno povedano, brez interneta ni računalništva v oblaku.
- **Finančna negotovost.** Pri prednostih uporabe smo omenili, da so začetni investicijski finančni stroški zagotovo manjši, vendar če pogledamo dolgoročno lahko operativni stroški morda celo presežejo dejanske stroške, ki bi jih potrebovali za nakup lastne infrastrukture.
- **Slabše delovanje pri počasnih internetnih povezavah**. Spletne aplikacije ponavadi potrebujejo široko podatkovno propustnost, saj se ponavadi prenašajo velike količine podatkov oz. dokumentov, kar pa privede do slabšega delovanja ali pa nam celo onemogoči normalno delo.

- **Možnost počasnega delovanja**. Čeprav imamo na rapolago hitro internetno povezavo, se nam lahko dostikrat zgodi, da bodo spletne aplikacije vseeno delovale počasneje, kot njihove namizne različice. Razlog za to je, da pri spletnih aplikacijah prenašamo zraven podatkov tudi uporabniški vmesnik, kar pa lahko bistveno upočasni samo delovanje.
- **Možnost omejene funkcionalnosti**. Čeprav se bo to čez čas zagotovo spremenilo, se nam trenutno velikokrat zgodi, da nam spletne verzije programov ne ponujajo polne funcionalnosti kot njihove namizne različice. Kot primer bi si lahko pogledali razlike med Googlovo rešitvijo Google Presentations in med Microsoftovim PowerPointom, saj imasta isto zasnovo pa vendar trenutno namizna rešitev prekosi spletno aplikacijo pri marsikateri napredni funkcionalnosti.
- **Varnostni pristopi**. Pri računalništvu v oblaku, shranjujemo vse naše podatke v oblak. Pri tem pa dostikrat pozabimo na varnost naših zaupnih podatkov. Lahko se nam namreč zgodi, da obstaja kakšna varnostna luknja in nam preko nje nepooblaščene osebe odtujijo naše dokumente. Prav tako se nam poraja tudi vprašanje ali lahko zaupamo ponudniku, da bo spoštoval pravila zasebnosti in v podatke ne bo posegal ali jih celo zlorabil?
- **Odvisnost od ponudnika**. Pri razvoju aplikacij, ki bodo tekle v oblaku, smo omejeni na gradnike, ki jih ponuja ponudnik. S tem pa naletimo na veliko oviro, če bi hoteli zamenjati ponudnika, saj so zaradi pomankanja standardov migracije med ponudniki večinoma nezdružljive.

3 NIVOJI RAČUNALNIŠTVA V OBLAKU

V prejšnjem poglavju smo si pogledali splošne lastnosti »računalništva v oblaku« in kakšne so njegove prednosti in slabosti pri uporabi v praksi. V tem poglavju pa se bomo lotili obrazložitvi nivojev »računalništva v oblaku«. Koncepti uporabe se namreč kažejo v vseh pomembnih tehnoloških trendih, kot so »infrastruktura kot storitev« (Infrastructure as a Service - IaaS), »platforma kot storitev« (Platform as a Service - PaaS) in »programska oprema kot storitev« (Software as a Service - SaaS). Vsem tem konceptom je skupno, da uporabniki za uporabo teh storitev potrebujejo stalno internetno povezavo in da računalniške zmogljivosti zagotavlja ponudnik storitev. Uporabniku tako za uporabo teh storitev ni potrebno imeti zmogljivih računalniških sistemov, saj se na njihovim računalniških sistemih nahajajo le nezmogljivi odjemalci, ki komunicirajo z storitvami v oblaku. Kot prikazuje slika 2, si lahko te sloje predstavljamo kot piramido, katera nam prikazuje medsebojne odvisnosti med posamenimi nivoji. Na dnu piramide lahko vidimo ponudbo infrastrukture kot storitve. Naslednji nivo je nadgradnja ponudbe infrastrukture kot storitve, ki predstavlja ponudbo platforme kot storitve. Na koncu nam ostane še ponudba programske opreme kot storitve, katera je produkt prej omenjenih nivojev. Na sliki lahko vidimo tudi trenutno glavne ponudnike po posameznem nivoju, ki smo jih opisali v naslednjem poglavju.

Slika 2: Piramida nivojev računalništva v oblaku

3.1 Infrastruktura kot storitev (IaaS)

Če se ozremo nekoliko nazaj, lahko vidimo, da korenine »infrastrukture kot storitve« segajo že v polovico devedesetih let, vendar uporaba teh storitve takrat še ni bila razširjena. Korak naprej je prispevala uveljavitev virtualizacije in razvoj hitrih, zanesljivih in razširjenih internetnih povezav.

Če pogledamo ponudbo programske opreme kot storitve vidimo, da nam ta ponuja celotno aplikacijo kot storitev, platforma kot storitev pa nam omogoča razvijanje lastne programske opreme. V obeh primerih vidimo da je končni produkt programska oprema. Če to primerjamo s ponudbo infastrukture kot storitve pa vidimo, da na tem nivoju programska oprema ni končni produkt, saj nam ta nivo ponuja uporabo strojne opreme in sistemskih virov katere pa lahko uporabimo za karkoli želimo. Tako bi lahko ponudbo »infrastrukture kot storitve« poimenovali tudi »vse kot storitev« (Everything as a Service – EaaS).

Glavna prednost uporabe infrastrukture kot storitve je zagotovo zmanjševanje začetnih stroškov, saj ne potrebujemo začetnih investicij za lastno strojno opremo ampak rajši plačujemo za najem te opreme in pri tem plačujemo toliko kolikor jo uporabljamo. Kot lahko vidimo so ciljni uporabniki infrastrukture kot storitve predvsem načrtovalci sistemov in omrežij ter sistemski administratorji. V uvodu tega poglavja smo videli, da sta tipična ponudnika infrastrukture kot storitve zagotovo Amazon s svojo rešitvijo Amazon Web Services in podjetje VMWare s svojo rešitvijo vCloud.

3.2 Platforma kot storitev (PaaS)

V uvodu tega poglavja smo že omenili, da je ponudba platforme kot storitve nadgradnja ponudbe infrastrukture kot storitve. Ponudba predstavlja model, ki nam s pomočjo interneta ponuja uporabo že omenjene infrastrukture, razvijalnega okolja in množico gradnikov s katerimi lahko razvijamo lastne storitve. Kot vidimo gre pri tem nivoju za ponudbo, ki nam omogoča razvijanje novih storitev, tako vidimo da so ključni uporabniki tega nivoja predvsem razvijalci, katerim ni potrebno skrbeti za infrastrukturo ampak lahko

takoj začnejo z razvijanjem lastnih rešitev. Ponudniki platforme kot storitve pa omogočijo razvijalcu enostaven način za objavo svojih rešitev na internetu. Omeniti moremo, da je večina ponudnikov omejenih na razvijanje le v določenem programskem okolju. V večini primerov razvijalci in storitve nimajo dostopa do operacijskega sistema na katerem se nahajajo, prav tako pa obstajajo nekatere omejitve pri razvoju, katere varujejo ponudnike pred zlorabami (nezaželena koda in nepooblaščena uporaba dodatnih virov). Kot smo lahko videli v uvodu tega poglavja sta tipična ponudnika platforme kot storitve zagotovo Google s svojo rešitvijo Google App Engine in Microsoft s svojo rešitvijo Windows Azure, katero smo uporabili za razvoj praktičnega dela diplomske naloge.

3.3 Programska oprema kot storitev (SaaS)

Bistvo tega nivoja je ponudba programske opreme kot storitve (Software as a service – SaaS). Ta nivo je zagotovo eden izmed največkrat uporabljenih nivojev uporabe računalništva v oblaku, saj je namenjen za uporabo množici končnih uporabnikov. Uporabniki do programske opreme dostopamo s pomočjo interneta in uporabniškega vmesnika, ki nam pomaga pri komunikaciji z aplikacijo. Razlogi za ponudbo takšnih storitev se skrivajo v tem, da bi ponudili enotno aplikacijo množici različnih uporabnikov brez začetnih investicij v strojno opremo ali v licenciranje programske opreme. Na drugi strani pa za razvijalce to prinaša enostavnost pri vzdrževanju aplikacije, saj vzdržujejo le eno verzijo aplikacije za množico uporabnikov. Kot smo že omenili uporabniku za uporabo programske opreme te ni potrebno kupiti, ampak rajši plačuje najemnino za uporabo (plačaj kolikor uporabljaš). Kot smo videli v uvodu tega poglavja lahko za dober primer ponudbe programske opreme kot storitve vzamemo ponudbo podjetja Google, ki nam ponuja množico aplikacij v oblaku (GMail, Google Apps,…).

4 TRENUTNO NAJPOMEMBNEJŠE PLATFORME

V prejšnjem poglavju smo si pogledali kateri nivoji storitev računalništva v oblakih so se razvili skozi čas. V piramidi storitvenih nivojev smo že omenili trenutne glavne ponudnike storitev posameznega nivoja. V tem poglavju pa si bomo ogledali lastnosti in ponudbo omenjenih platform. Osredotočili smo se na platforme: Amazon Web Services, Google App Engine, Windows Azure, VMWare in SalesForces.

4.1 Amazon Web Services

Amazon je eden izmed največjih trgovcev na spletu. Prav tako je tudi eden izmed glavnih ponudnikov »računalništva v oblaku«. Vse se je začelo z vzpostavitvijo množice strežnikov za poganjanje njihove priljubljene spletne strani, v katero so vložili veliko časa in denarja. Zaradi zelo dobrih izkušenj, so se odločili da bodo del svojih zmogljivosti začeli ponujati tudi drugim, storitev so poimenovali Elastic Compute Cloud znano tudi pod kratico EC2. Po nekaterih besedah bi naj bila to največja skupina združenih računalnikov na svetu. Storitev je plačljiva in omogoča razvijalcem in podjetjem, da najamejo zmogljivost strežnikov v oblaku.

Amazon ponuja uporabnikom tri različne tipe zmogljivosti navideznih strežnikov:

- Majhni: zmogljivost je enaka, kot če bi imeli sistem z 1,7 GB notranjega pomnilnika, 160 GB podatkovnega prostora in enim 32-bit-nim navideznim procesorjem.
- Veliki: zmogljivost je enaka, kot če bi imeli sistem z 7,5 GB notranjega pomnilnika, 850 GB podatkovnega prostora in dvema 64-bit-nima navideznima procesorjema.

- Zelo veliki: zmogljivost je eneka, kot če bi imeli sistem z 15 GB notranjega pomnilnika, 1,7 TB podatkovnega prostora in štirimi 64-bit-nimi navideznimi procesorji.

Z drugimi besedami bi lahko rekli, da nam omogočajo izbiro velikosti in zmogljivosti naših navideznih strežnikov, za vse ostalo pa bodo poskrbeli oni. Amazon je trenutno najverjetneje eden izmed najbolj popularnih ponudnikov storitev »računalništva v oblaku«. Sami trdijo, da pokrivajo trg z več kot 330.000 uporabniki, ki so mešanica razvijalcev, začetnikov in že vpeljanih podjetij.

4.2 Google App Engine

Google je vodilni ponudnik spletno usmerjenih aplikacij, zato ni prav nič presenetljivo, da je ponudil tudi platformo za razvijanje storitev v oblaku, ki so jo poimenovali Google App Engine. Platforma omogoča podporo praktično celotnemu življenskemu ciklu programa: od razvoja, ki ga lakho opravimo na lastnem namizju, do objave, nadzora in vzdrževanja, ki poteka na Googlovi visoko zmogljivi infrastrukturi.

Google nam s svojo platformo Google App Engine ponuja popolnoma integrirano razvojno okolje. Z uporabo Googlovih razvijalskih orodij in računalniškega oblaka lahko našo aplikacijo na zelo enostaven način zgradimo, vzdržujemo in prilagajamo. Vse kar moremo narediti je, da razvijemo našo aplikacijo, jo naložimo v Googlov oblak in od tukaj naprej je aplikacija na voljo našim uporabnikom.

Kot smo lahko od Googla pričakovali, ta ponuja robustno razvijalsko okolje katero vsebuje kar nekaj zanimivih rešitev:

- Omogočena nam je dinamična spletna strežba.
- Podpira vse najbolj razširjene spletne tehnologije
- Ponudba zmogljive podatkovne shrame, ki podpira povpraševanja, razvrščanja in transakcije.
- Avtomatično prilagodljivost oz. razširljivost in izenačevanje obremenitev.

- Gradnike za identifikacijo uporabnikov in gradnike za pošiljanje elektronske pošte.

Kot dodatek še Google ponuja polno funkcionalno lokalno razvijalno okolje, s katerim lahko popolnoma simuliramo Google App Engine na bilokaterem namiznem računalniku.

Ena od bistvenih prednosti Googlove ponudbe je ta, da ponujajo osnovno verzijo računalniškega oblaka povsem brezplačno. Ponudba vsebuje 500MB diskovnega prostora in dovolj delovne zmogljivosti in pasovne širine, ki bo zadoščala postrežbi 5 miljonov strani na mesec.

4.3 Windows Azure

Če pogledamo piramido nivojev storitev v oblaku spada Windows Azure med storitve, katere ponujajo uporabo »platforme kot storitve« (PaaS). Windows Azure je razširljiva platforma, ki teče na Microsoftovih podatkovnih strežnikih. Ti strežniki ponujajo operacijski sistem za poganjanje naših aplikacij in množico storitev, katere lahko uporabimo pri razvijanju naših aplikacij. Storitve so neodvisne in jih lahko uporabljamo vsako posebej ali pa v skupini. Prav tako je Azure prilagodljiva platforma, na kateri lahko gradimo nove aplikacije, ki bodo tekle v oblaku, ali pa prilagodimo že obstoječe aplikacije za delovanje v oblaku.

Razvijalcem je omogočeno, da lahko enostavno ustvarijo aplikacijo, ki teče v oblaku z uporabo obstoječega znanja in spretnosti pri uporabi razvijalskega orodja Microsoft Visual Studio v povezavi z ogrodjem Microsoft .NET. Microsoft si je zadal nalogo, da bo Azure v prihodnosti podpiral razvoj v več programskih jezikih in razvijalskih okoljih oz. orodjih.

S pomočjo platforme Windows Azure je poskrbljenjeno za avtomatizirano vzdrževanje infrakstrukture, ki skrbi za visoko dosegljivost posamezne storitve. Prav tako je poskrbljeno za dinamično prilagajanje trenutnim potrebam po zagotavljanju storitev in pri tem upošteva model »plačaj kolikor uporabljaš«. Platforma tudi omogoča podporo in uporabo množice internetnih protokolov, ki so danes zelo razširjeni, npr. HTTP, REST, SOAP in XML.

Microsoft trenutno že ponuja kar nekaj aplikacij, ki tečejo v oblaku in so na razpolago uporabnikom za integriranje v njihove sisteme. Omenimo le nekaj glavnih, kot so: Windows Live, Microsoft Dynamics in druge Microsoftove spletne storitve kot je npr.: Microsoft Exchange Online in SharePoint Online. S temi gradniki pa nam platforma Windows Azure omogoča, da lahko uporabniki ustvarijo lastno unikatno ponudbo za končne uporabnike.

4.4 VMWare

Virtualizacija je ključnega pomena za ponudnike računalništva v oblaku. Glede na dosedanje izkušnje z virtualizacijo, je bilo samo vprašanje časa kdaj bo tudi VMWare ponudil svojo platformo za podporo računalništvu v oblaku. Poimenovali so jo vCloud in je trenutno še v fazi razvoja. Vizija podjetja VMWare je predvsem v ponudbi infrastrukture kot storitve, pri tem pa obljubljajo zanimiv pristop k virtualizaciji, ki zna odpreti povsem nov trg z množico številnih ponudnikov, ki bodo s konkurenčnimi pristopi poskrbeli za inovativne rešitve in nizke cene.

Kot smo omenili ima VMWare številne izkušnje z virtualizacijo in to mu prinaša veliko prednost pred ostalimi tekmeci pri ponudbi infrastrukture kot storitve. Prednost je v tem da gradijo oblak na preizkušeni podlagi za virtualizacijo in imajo na voljo že številne izdelane in preizkušene virtualne strojne podobe (virtual machine images). Zanmivo je tudi to, da so v izgradnjo oblaka povabili številne partnerje, ki bodo naknadno poskrbeli za zanesljivo gostovanje že izdelanih virtualnih strojev. Ta pristop jim je omogočil, da so uporabnike zavarovali pred odvisnostjo od enega samega ponudnika. Standardizirane podobe virtualnih strojev bo namreč mogoče brez večjih težav prenašati med številnimi ponudniki, ki si bodo med seboj konkurirali s ceno in kakovostjo storitve.

Za razvijalce vCloud ne prinaša kakšnih posebnih novosti, saj gre predvsem za ponudbo, ki bo prinesla novost med ponudniki infrastrukture kot storitve. Najrazličnejše programske podlage, ki se bodo rojevale na njej pa najverjetneje ne bodo del ponudbe samega vClouda.

4.5 SalesForces

SalesForces je pravgotovo eden izmed najbol znanih ponudnikov storitev, ki zajemajo ponudbo aplikacij kot storitev. Svoj sloves so si pridobili predvsem na podlagi poslovnih rešitev, ki jih ponujajo. Med temi rešitvami je pravzagotovo najbol znana aplikacija, ki služi urejanju odnosov med strankami (Customer relationship management - CRM). Rešitev računalništva v oblakih, ki so jo oblikovali pa so poimenovali Force.com.

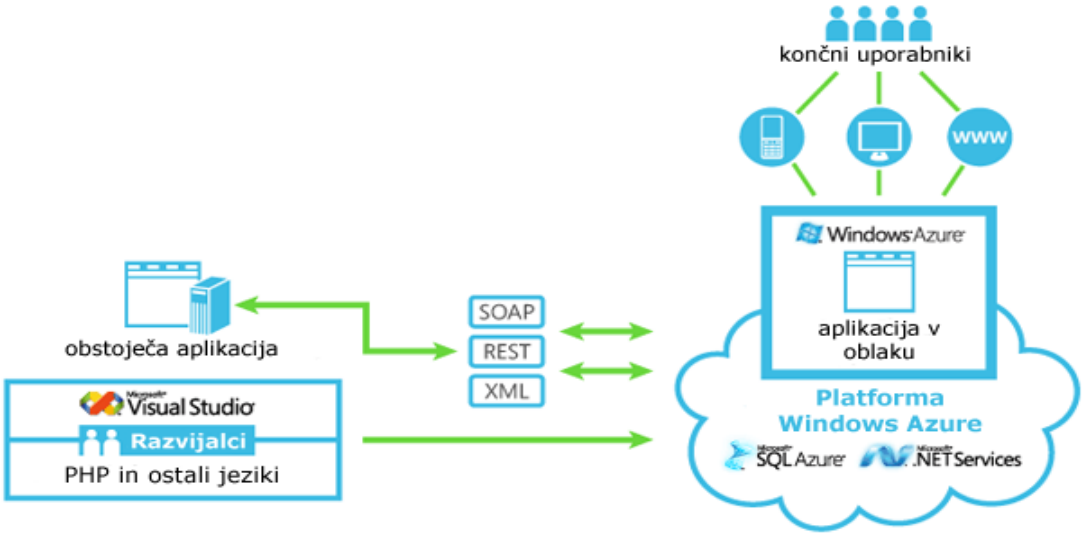
SalesForces je prav tako poskrbel za razvijanje novih storitev, saj je za razvijalce pripravil razvijalsko okolje za razvijanje na platformi Foce.com, prav tako pa so poskrbeli tudi za pripravo razvijalnih orodij, katere razvijalcem olajšajo delo pri razvoju aplikacij. Svojo ponudbo so dopolnili z storitvijo AppExcahnge, katere namen je deljenje aplikacij med uporabniki. Razvijalci lahko namreč uporabljajo obstoječe aplikacije, ki jih ponuja AppExcange ali pa razvijejo svojo aplikacijo in omogočijo njeno uporabo tudi drugim uporabnikom. Mnoge aplikacije, ki jih trenutno ponuja AppExcahnge so na voljo za uporabo povsem brezplačno, prav tako pa je omogočeno, da lahko razvijalec ponudi svojo aplikacijo v uporabo in za to prejema denarna nadomestila. Omeniti moremo, da je večina aplikacij, katere so trenutno zbrane v AppExcahnge komercialno usmerjenih – prodajno analitična orodja, elektronski marketinški sistemi, finančno analitične aplikacije in še bi lahko naštevali, vendar to ni omejitev platforme Force.com, saj nam platforma omogoča da razvijamo aplikacije poljubnega tipa.

5 PLATFORMA WINDOWS AZURE

V prejšnjem poglavju smo si pogledali trenutne glavne ponudnike »računalništva v oblakih«. Eden izmed teh ponudnikov je tudi Microsoft s svojo platformo Windows Azure, katero smo v nekaj odstavkih že opisali v prejšnjem poglavju. V tem poglavju pa si bomo nekoliko podrobneje pogledali kakšna je njena zgradba, opisali storitve ki jih trenutno ponuja in si ob zaključku poglavja pogledali še kakšna je vizija zaračunavanja za posamezne storitve. Razlog za podrobnejši opis pa se skriva v tem, da smo platformo Windows Azure uporabili za naš praktični del diplomske naloge.

5.1 Zgradba

Za začetek si poglejmo zgradbo platforme, katere shematski prikaz je podan na sliki 3. Kot lahko vidimo je platforma sestavljena iz množice gradnikov, ki so namenjeni aplikacijam, ki delujejo v oblaku in tudi aplikacijam, ki tečejo izven oblaka (lokalni sistemi). Namen vsakega od teh gradnikov pa je, da pokrije določeno področje in poenostavi razvijanje in uporabo storitev v oblaku. Na sliki tudi vidimo, da lahko končni uporabniki do teh storitev dostopajo preko različnih kanalov in so pri tem neodvisni od sistema, ki ga poganja posamezna naprava. Na žalost mnogokaterega razvijalca nam trenutno Windows Azure ponuja razvijanje aplikacij le s pomočjo ogrodja .NET v povezavi z razvijalskim orodjem Visual Studio, vendar bi se naj to v kratkem spremenilo, saj je vizija Microsofta, da bi podprli množico različnih razvijaslkih jezikov.



Slika 3: Zgradba platforme Windows Azure

5.1.1 Opis platformeWindows Azure

Windows Azure temelji na operacijskem sistemu Windows in nam tako ponuja okolje za shranjevanje podatkov in okolje za poganjanje naših aplikacij. Kot lahko vidimo na sliki 4, teče okolje Windows Azure na množici strežnikov, ki so nameščeni v Microsoftovih podatkovnih centrih širom sveta. Vsak od teh strežnikov pa z uporabo virtualizacije in tehnologije Hypervisior poganja množico virtualnih podob (ang. virtual images), na katerih so nameščene naše aplikacije.

Kot smo že omenili Windows Azure temelji na okolju Windows in s tem razvijalcem ponuja možnost razvijanja aplikacij, katere za svoje delovanje uporabljajo ogrodje .NET. Pri razvoju pa se lahko odločimo med dvema različnima načinoma delovanja aplikacije, kot nam prikazuje slika 4. Ta dva načina sta:

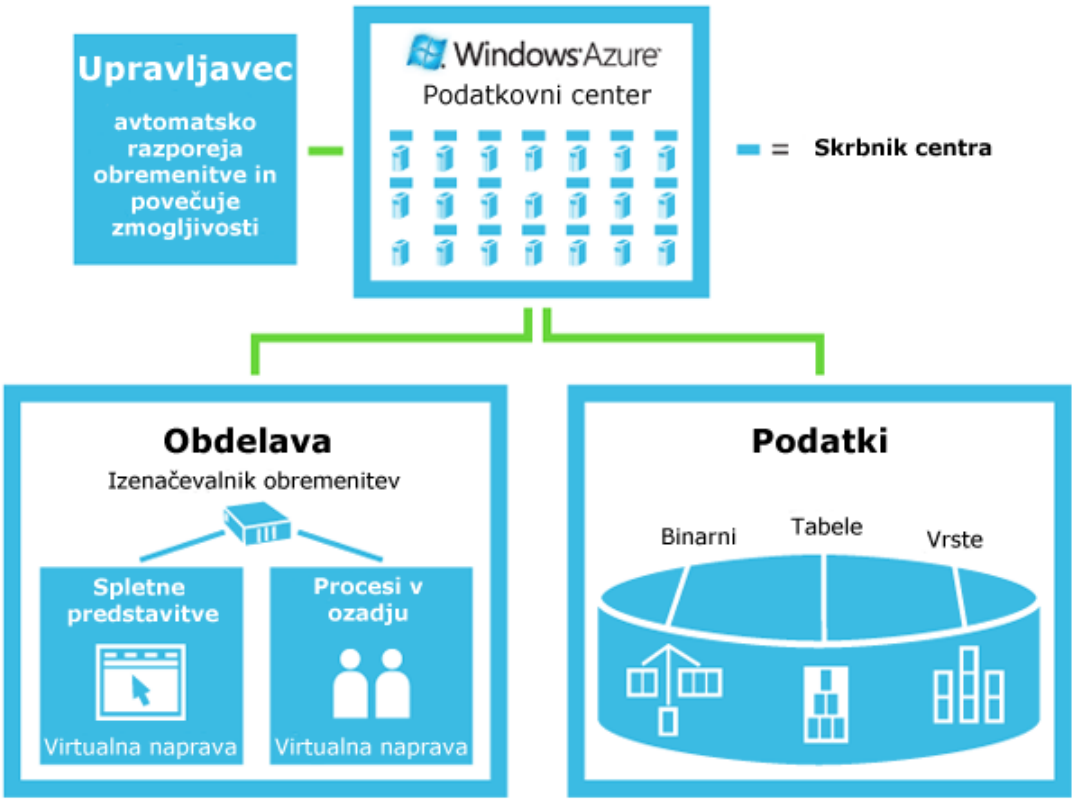
- **Web Role**: Ta način delovanja aplikacije je namenjen aplikacijam, ki za svojo delovanje potrebujejo predstavitev. Vsaka od teh aplikacij je navzven dostopna preko HTTP (ali HTTPS) poizvedbe na spletni strežnik (IIS). Uporabimo lahko različne vrste implementacij: ASP.NET, WCF ali katero drugo .NET tehnologijo, ki deluje v povezavi z spletnim strežnikom (IIS).

- **Worker Role**: Ta način delovanja je namenjen različnim obdelavam, ki se izvajajo nekje v odzadju. Njihovo delovanje jim narekujejo »Web Roles«, ponavadi s pomočjo »čakalnih vrst« (ang. queue), saj te vloge navzven niso direktno dostopne (virtualne podobe nimajo nameščenega spletnega strežnika - IIS). Rezultati obdelave pa se lahko shranijo v podatkovne zbirke ali pa na kakšen drugi način posredujejo podatke zunanjemu svetu (povezave navzven so namreč omogočene).

Aplikacije upravljajo s podatki na mnogo različnih načinov. Včasih nam zadošča že enostavno shranjevanje binarnih podatkov, v nekaterih drugih situacijah pa je potreba po bolj strukturiranem shranjevanju podatkov ali pa nam že zadošča pot, po kateri bi lahko izmenjevali podatke med dvema različnima deloma aplikacije. Podatkovna zbirka v Windows Azure nam omogoča vse tri omenjene zahteve, kot nam prikazuje slika 4:

- **Blobs**: So zbirke namenjene shranjevanju velikih objektov, kot so npr.: videi in slike.
- **Tables**: Predstavlja visoko razširljivo entitetno orientirano zbirko za shranjevanje podatkov.
- **Queues**: So čakalne vrste, ki nam omogoča pošiljanje in sprejemanje sporočil, kot npr. sporočila, ki so namenjena komunikaciji med dvema aplikacijama (WebRole – WorkerRole).

Na koncu še omenimo, da so vsi ti trije tipi možnosti shranjevanja podatkov dostopni tudi aplikacijam, ki ne tečejo na Windows Azure, saj nam Windows Azure ponuja možnost dostopa do teh podakov s pomočjo standardnih HTTP GET, PUT in DELETE zahtev.



Slika 4: Shema Windows Azure

5.1.2 SQL Serivces

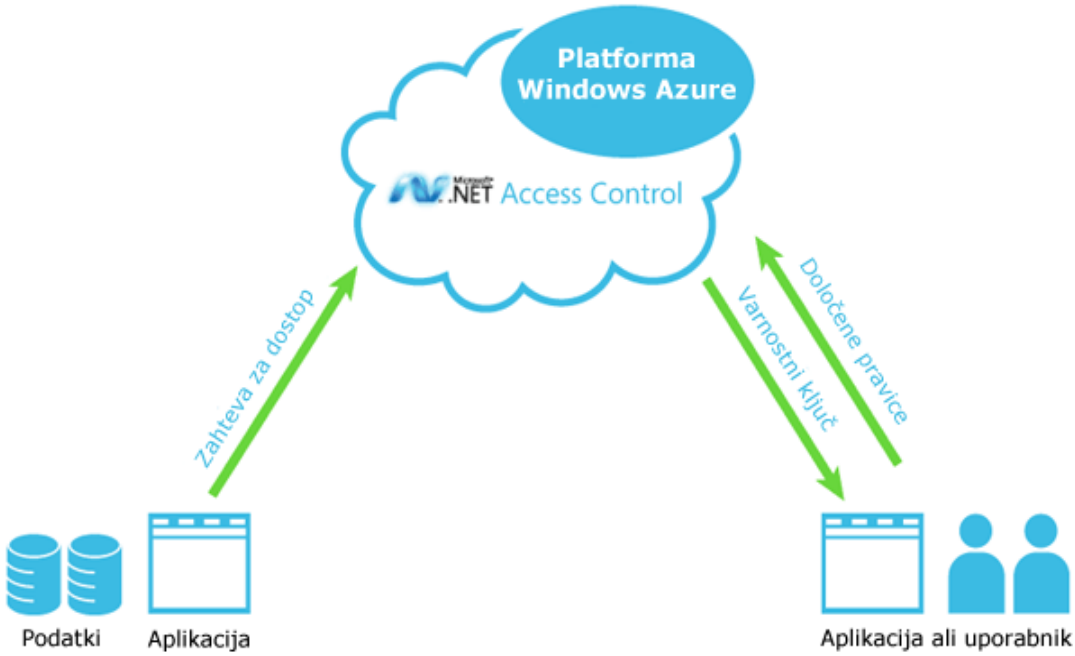
Ena od najbolj atraktivnih poti uporabe računalništva v oblaku je prav gotovo shranjevanje podatkov v oblak. Kot smo omenili v prejšnji točki nam Windows Azure že ponuja osnovno shranjevanje podatkov, vendar ta ne podpira shranjevanja relacij med podatki. Z namenom da pokrijejo tudi to področje, so se pri Microsoftu odločili, da ponudijo rešitev tudi za to področje. Rešitev so poimenovali SQL Services, ki temelji na znanem Microsoftovem SQL strežniku.

Kot smo že omenili v uvodu tega poglavja lahko do SQL Services dostopamo ne glede na to kateri operacijski sistem uporabljamo oz. s katero napravo dostopamo do te storitve. SQL strežnik v oblaku namreč za delovanje uporablja standardni protokol Tabular Data Stream (TDS), katerega uporabljajo tudi lokalni SQL strežniki in nam je s tem omogočena komunikacija z strežnikom z uporabo obstoječih SQL knjižnic, kot so ADO.NET, ODBC in še številne druge.

5.1.3 .NET Services

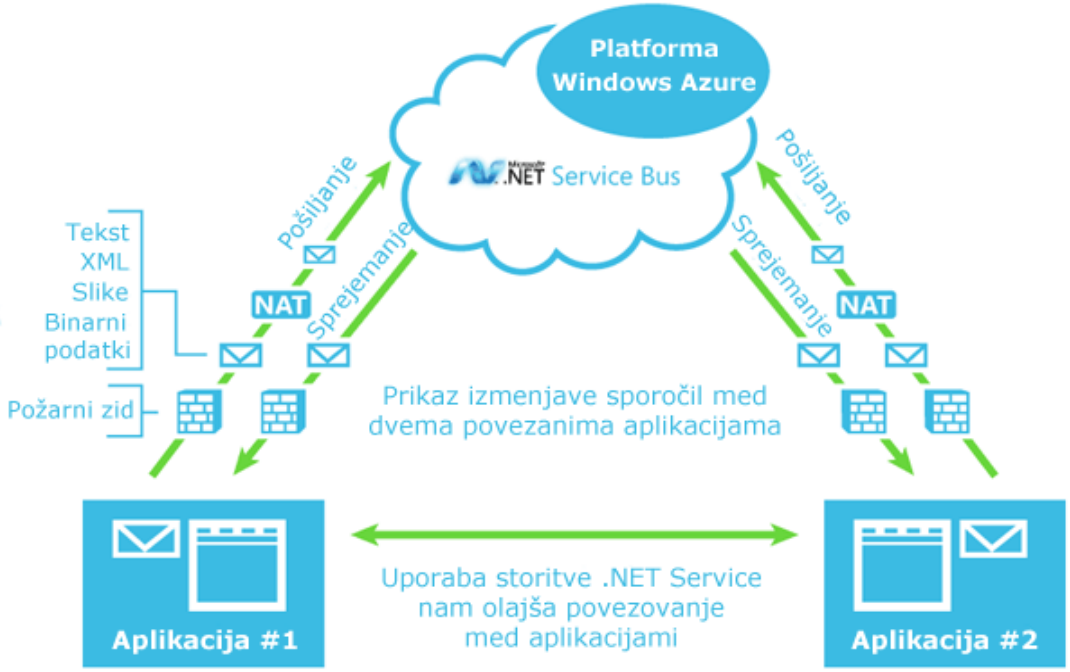
Microsoft .NET Services je del storitev, ki jih Microsoft ponuja v svoji platformi Windows Azure in nam omogočajo integracijo z obstoječimi indetifikacijskimi sistemi (slika 6) in povezovanje različnih aplikacij preko interneta (slika 7). Na voljo imamo gradnika Access Control in Service Bus:

Access Control: Kot smo omenili nam storitev .NET Services ponuja možnost integracije obstoječega identifikacijskega sistema in se nam tako ni potrebno obremenjevati z lastnimi implementacijami, ki skrbijo za varnost aplikacije. Z uporabo modela vlog in pravic, nam omogoča enostavno in fleksibilno nastavljanje različnih vlog s katerimi lahko pokrijemo varnostne potrebe za različne identifikacijsko urejene sisteme.



Slika 5: Shema delovanja Access Control-e

Service bus: Njegova naloga je, da nam omogoča enostavno in varno povezovanje različnih aplikacij preko interneta.



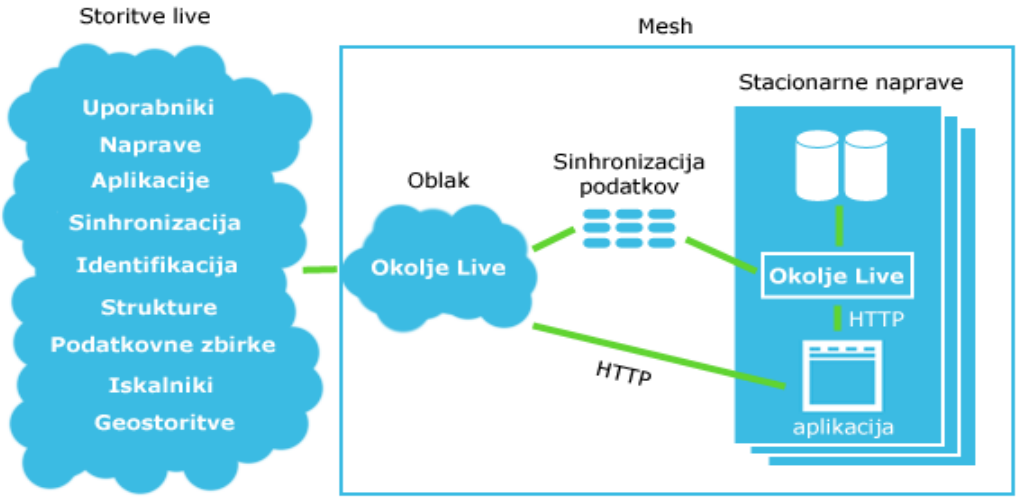
Slika 6: Shema delovanja Serivce Bus-a

5.1.4 Live Services

Kot lahko vidimo na sliki 7 je ključna kompontenta storitev Live (Live Services) njegovo delovno okolje (Live Operating Environment). To okolje se nahaja v oblaku in ponuja aplikacijam dostop do ponujenih storitev v oblaku. Povezava med aplikacijo in delovnim okoljem pa poteka preko standardnih povezav HTTP oz. poizvedb, ki nam omogočajo dostop do storitev ne glede na to s pomočjo katere tehnologije je bila naša aplikacija implementirana (.NET, Java, JavaScript,…).

Slika 7: Prikaz sheme delovanja okolja Live Services

Kot smo omenili se delovno okolje nahaja v oblaku, vendar to ni edina možnost, saj lahko to okolje namestimo na posamezne naprave in te naprave združimo v tako imenovani »Live Mesh«. Kot lahko vidimo na sliki 8 je storitev Live Mesh namenjena združevanju digitalnih vsebin in omogoča uporabnikom dostop do dokumentov, slik, videoposnetkov in drugih vsebin iz kateregakoli računalniškega sistema ali prenosne naprave, ki je priključena na svetovni splet. Okolje bo namreč poskrbelo za sinhronizacijo datotek med posameznimi napravami.



Slika 8: Sinhronizacija podatkov med posameznimi napravami

5.2 Cenik in zaračunavanje

V času skupinskega tehnološkega pregleda (Community Technology Preview - CTP) bodo storitve, ki jih platforma ponuja na voljo brezplačno vendar z nekaterimi omejitvami. Te omejitve so:

- Skupno število porabljenih procesorskih ur je omejeno na 2000 ur.
- Skupna kapaciteta za shranjevanje podatkov je omejena na 50GB.
- Dnevna omejitev prenosa podatkov je omejena na 20GB.

V naslednjih točkah tega poglavja pa si bomo pogledali, kakšna je Microsoftova vizija za zaračunavanja posamezne storitve, ko bo platforma Windows Azure zrela za komercialno uporabo.

5.2.1 Zaračunavanje platforme Windows Azure

Shema je tukaj dokaj enostavna saj uporablja za izračun cene štiri spremenljivke. Te spremenljivke so procesorski čas, velikost shranjenih podatkov, število operacij nad shranjenimi podatki in porabljena podatkovna širina. Za procesorski čas se plačuje mesečna najemnina za vsako uro, ki jo porabi virtualna naprava. Pri shranjevanju podatkov se nam mesečno zaračuna vsak GB, ki ga hranimo v oblaku. Pri operacijah nad shranjenimi podatki se nam mesečno zaračuna vsaka 10.000 operacija. Vsak mesec se nam zaračuna tudi najemnina za vsak poslani in prejeti GB, ki je obremenil podatkovno propustnost.

- Za vsako uro procesorskega časa bomo morali odšteti $0,12 na mesec.
- Za vsak shranjen GB podatkov bomo morali odšteti $0,15 na mesec.
- Za vsako 10.000 operacijo nad shranjenimi podatki bomo morali odšteti $0,01 mesečno.
- Za vsak GB podatkov, ki ga bomo prejeli bomo morali mesečno odšteti $0,10. Za vsak GB podatkov, ki ga bomo poslali ven pa bomo odšteli mesečno $0,15.

5.2.2 Zaračunavanje storitev SQL Services

Microsoft nam pri uporabi SQL strežnika v oblaku ponuja dve verziji, kateri pa se razlikujeta po skupni velikosti podatkovne zbirke.

- Spletna verzija: Uporaba podatkovnega strežnika v velikosti do 1 GB nas bo mesečno stala $9,99.
- Poslovna verzija: Uporaba podatkovnega strežnika v velikosti do 1 GB nas bo mesečno stala $99,99 .
- Za vsak GB podatkov, ki ga bomo prejeli, bomo morali mesečno odšteti $0,10. Za vsak GB podatkov, ki ga bomo poslali pa bomo odšteli mesečno $0,15.

5.2.3 Zaračunavanje storitve .NET Services

Shema za izračun opravljenih storitev je enostavna, saj uporablja za izračun dve spremenljivki. Ti dve spremenljivki sta število operacij in porabljena podatkovna širina. Za vsak mesec posebej si platforma beleži število opravljenih operacij, na podlagi števila opravljenih operacij se nam izračuna mesečna poraba. Vsak mesec se nam zaračuna tudi najemnina za vsak poslani in sprejeti GB, ki je obremenil podatkovno propustnost

- Za vsako 100.000 sporočilo oz. operacijo nad Access Control in Service Bus-om bomo morali mesečno odšteti $0,15.
- Za vsak GB podatkov, ki ga bomo prejeli, bomo morali mesečno odšteti $0,10. Za vsak GB podatkov, ki ga bomo poslali pa bomo odšteli mesečno $0,15.

6 APLIKACIJA ZA VODENJE SPLETNEGA DNEVNIKA, KI SE IZVAJA V OBLAČKU

V prejšnjem poglavju smo si nekoliko pobližje ogledali Microsoftovo platformo Windows Azure, ki smo jo uporabili za naš praktični del naloge. Namen praktičnega dela naloge je bil, da bi preselili izvajanje obstoječe aplikacije za vodenje spletnega dnevnika v oblak. Izbrali smo Microsoftovo rešitev z imenom Oxite, ki je odprto-kodna rešitev na voljo pod »Microsoft Public License« (Ms-PL). Za razvojem Oxite-a se skriva uveljavljena ekipa, katero poznamo iz projektov kot so Channel 9, Channel 10, Channel 8, TechNet Edge in MixOnline. Glavni razlog za razvoj tega projekta je bil, da so želeli uporabiti Oxite kot osnovo za njihov spletni portal MixOnline.

Za uporabo rešitve Oxite smo se odločili zato, ker nam ta ponuja močne temelje na katerih lahko nadaljujemo razvoj in tako nadgrajujejmo našo aplikacijo. V osnovi nam Oxite ponuja množico osnovnih funkcionalnosti za vodenje spletnega dnevnika, kot so: pingbacks, trackbacks, komentiranje, podporo avatarjem, RSS vire, podprti je protokol MetaWebLog API (npr. integracija z Windows Live Writter-jem), spletno administratorsko nadzorno ploščo, podpora OpenSearch formatu (uporabniku omogočimo, da lahko išče s spletnim brskalnikom v iskalnem oknu po vsebini naše spletne aplikacije) in še bi lahko naštevali. To pa nam zagotavlja, da se lahko bolje osredotočimo na načrtovanje novih storitev saj se nam ni potrebno ukvarjati z osnovami, kar je v današnjih časih bistvenega pomena, če želimo hitro in uspešno prodreti na trg.

V tem poglavju si bomo torej ogledali korake, ki so bili potrebni za selitev aplikacije v oblak, ogledali si bomo tudi arhitekturno postavitev in na koncu še opisali ponujene funkcionalnosti obstoječe aplikacije.

6.1 Namestitev okolja

Kot smo si lahko v uvodu tega poglavja pogledali, smo si za našo nalogo izbrali obstoječo aplikacijo Oxite. Preden smo lahko začeli z uporabo in razvojem obstoječe aplikacije smo si morali namestiti naslednja orodja:

- **Visual Studio 2008 (z nameščenim .NET 3.5 SP1)**, je odlično razvojno okolje, ki se najpogosteje uporablja za razvoj aplikacij, ki temeljijo na operacijskem sistemu Microsoft Windows.
- **SQL Server (Express edition)**, je podatkovna platforma, katere glavni namen je shranjevanje podatkov v podatkovne zbirke.
- **ASP.NET MVC 1.0**, je ogrodje, ki nam olajša razvoj spletnih aplikacij, ki temeljijo na uporabi patenta »Model-View-Controler« (MVC).
- **Unity Application Block 1.2**, je ogrodje ki nam omogoča enostavno dodajanje novih funkcionalnosti, katere so implementirane po vnaprej definiranih vmesnikih.
- **BlogML 2.0**, knjižnica ki omogoča enostaven uvoz starejših blog zapisov v naš novi sistem.
- **AntiXSS 3.1**, knjižnica varuje aplikacijo pred uporabniki, ki bi želeli vključiti v zapise zlobnonamerno kodo.

Po namestitvi teh orodij smo bili pripravljeni za lokalno uporabo naše aplikacije. Ker pa je bil naš namen preseliti izvajanje aplikacije v oblak, smo morali namestiti še orodja, ki so nam pomagala pri razvoju in prenosu aplikacije v oblak. Omeniti moramo, da imajo ta orodja eno omejitev oz. slabost, saj potrebujemo za uporabo prvih dveh orodij nameščen operacijski sistem Windows Vista ali Windows 2008. Ta orodja so:

- **Windows Azure Software Development Kit**, ki nam ponuja navidezni oblak na našem lokalnem razvijalnem okolju in nam omogoča da gradimo in lokalno poganjamo Windows Azure aplikacije, kot da bi te tekle v pravem oblaku.
- **Windows Azure Tools za Visual Studio**, nam ponuja gradnike integrirane v Visual Studio, ki nam olajšajo in poenostavijo razvijanje aplikacij za Windows Azure.

- **SQL Azure Migration Wizard v0.2.7**, program, ki nam pomaga pri selitvi lokalne podatkovne zbirke SQL strežnika na strežnik SQL Azure, ki teče v oblaku.

6.2 Arhitekturna postavitev

V prvi točki tega poglavja smo že omenili razlog za uporabo aplikacije Oxite vendar to ni bil edini razlog. Če pogledamo kaj vse nam ponujajo današnje odprto-kodne rešitve za vodenje spletnega dnevnika ugotovimo, da je ponujena funkcionalnost dokaj identična, opazimo pa lahko bistvene razlike v arhitekturnih postavitvah. Pri ocenjevanju arhitekturne postavitve ima Oxite veliko prednost pred ostalimi tekmeci, saj uporablja moderne arhitekturne prijeme oz. patente, kot so npr.: »Model View Controler«, »Test-driven design« in »Dependency injection«.

Oxite ima tro-nivojsko arhitekturno razdelitev in za to uporablja patent »Model View Controler« (MVC). Na nivoju modela so shranjena stanja aplikacije, ki se najpogosteje preslikajo v modele podatkovnih tabel. Krmilni program skrbi za upravljanje akcij, ki spreminjajo stanje modelov. Na koncu nam ostane še nivo View-a, ki poskrbi za izluščenje informacij iz modela, ki jih nato prikaže uporabniku s pomočjo uporabniškega vmesnika.

Pod pojmom »Test-driven design« se skriva ideja, da moremo pri razvoju poskrbeti za čim večjo neodvisnost programskih komponent. To nam omogoča, da lahko neodvisno testiramo manjše kose programske kode. Pri tem pa si pomagamo s patentom »Dependency injection«, ki nam omogoča, da lahko enostavno zamenjamo programske komponente brez da bi morali spremeniti ostalo funkcionalnost.

Rešitev Oxite je tako sestavljena iz petih sklopov oz. projektov:

- **Oxite.Database**, v tem projektu so zbrane sheme, ki so namenjene shranjevanju podatkov v podatkovni SQL strežnik.
- **Oxite**, po patentu »Model-View-Controler« je ta projekt nivo »Model«. V njem so zbrane entitete in definicije storitev brez implementacije.

- **Oxite.LinqToSqlDataProvider**, v tem projektu najdemo dejanske implemetacije storitevv, katere smo definirali v »Modelu«. Storitve so implementirane s pomočjo tehnologije LinqToSql.
- **Oxite.MVC**, v tem projektu najdemo kodo za krmilne programe, prav tako se v tem projektu nahaja koda, ki skrbi za komunikacijo oz. povezavo med »Controler«-jem in »View«-om.
- **Oxite.Site**, v tem projektu najdemo uporabniške vmesnike, ki so namenjeni za komunikacijo z uporabnikom.

6.3 Prilagoditev aplikacije za izvajanje na platformi Windows Azure

V prejšnji točki tega poglavja smo si ogledali kakšna je arhitekturna postavitev naše aplikacije. Kot smo videli, gre za tipično spletno aplikacijo ki za svoje delovanje potrebuje še podatkovni strežnik za shranjevanje podatkov in spletni brskalnik za komunikacijo z uporabnikom. Kot smo že omenili je bila naša naloga prilagoditev aplikacije za izvajanje na platformi Windows Azure. Na sliki 9 lahko vidimo, da smo aplikacijo izdelali v dveh korakih.

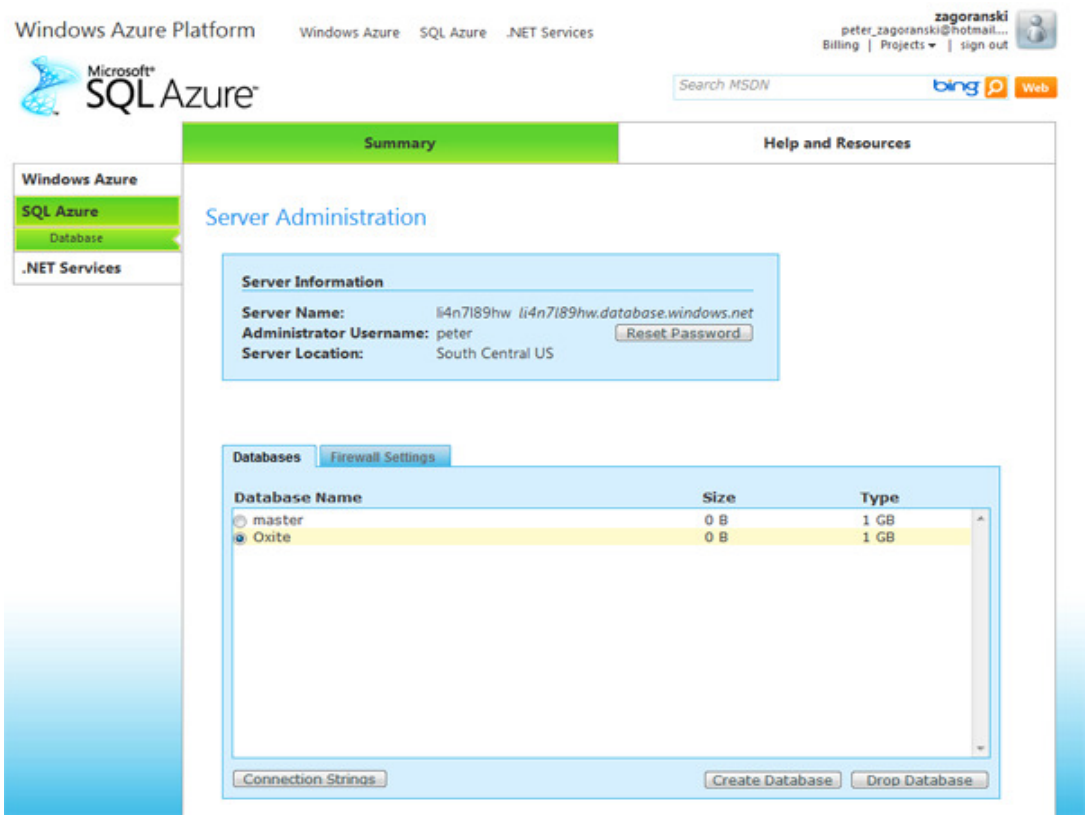
V arhitekturni postavitvi smo videli, da aplikacija za shranjevanje podatkov uporablja SQL strežnik. Če pogledamo v poglavje, kjer smo predstavljali gradnike, ki jih trenutno ponuja Platforma Windows Azure, ugotovimo da je eden izmed gradnikov tudi SQL strežnik, ki se nahaja v oblaku. Naš prvi korak je tako bil preseliti podatkovno zbirko iz lokalnega strežnika na strežnik v oblaku. S tem korakom smo želeli tudi dokazati, da lahko do strežnika v oblaku dostopamo tudi iz lokalnih aplikacij in ne samo iz aplikacij katere tečejo v oblaku.

V drugem koraku, kot lahko vidimo na sliki 9, pa smo zraven podatkovnega strežnika v oblak preselili še izvajalno kodo, ki skrbi za delovanje naše aplikacije.

Slika 9: Prikaz poteka selitve v oblak

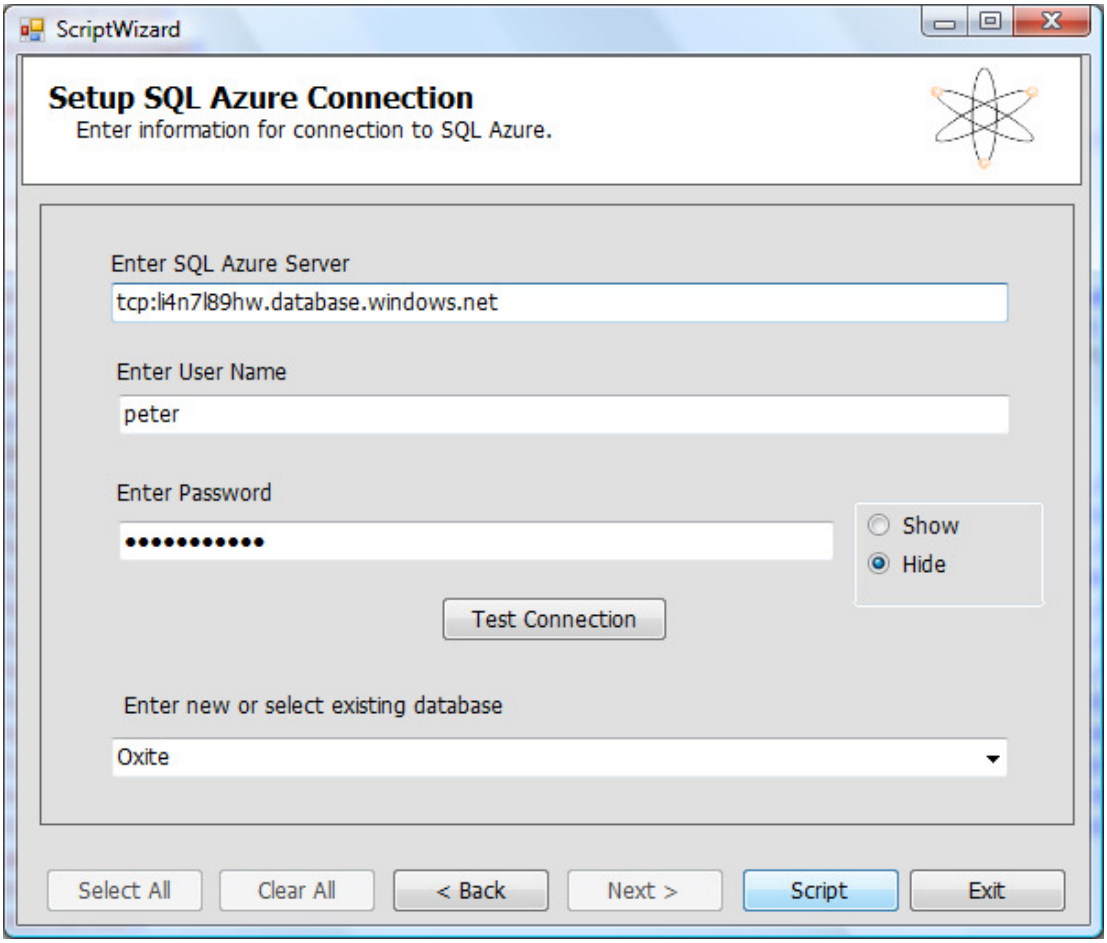
6.3.1 Selitev podatkov na SQL strežnik v oblaku

Preden smo lahko začeli z uporabo SQL strežnika v oblaku, smo morali opraviti registracijo na portalu, ki je namenjen upravljanju strežnika v oblaku. Po opravljenih formalnostih smo bili pripravljeni na začetek selitve. Kot je vidno iz slike 10, smo za začetek kreirali podatkovno zbirko, ki smo jo poimenovali Oxite.



Slika 10: Portal za upravljanje strežnika SQL Azure

Po uspešnem kreiranju podatkovne zbirke smo bili pripravljeni na kopiranje strukture podatkovne zbirke v oblak. Kot je prikazano na sliki 11, nam je bil pri tem v veliko pomoč program »SQL Azure Migration Wizard«, s katerim smo kreirali sheme iz obstoječe podatkovne zbirke prilagojene za SQL strežnik v oblaku in jih nato tudi s pomočjo istega programa prenesli v oblak.



Slika 11: Program za selitev podatkov na strežnik SQL Azure

Kot smo že omenili SQL Azure za delovanje uporablja standardni protokol Tabular Data Stream (TDS), ki omogoča komunikacijo s strežnikom z uporabo obstoječih SQL knjižnic, kot so ADO.NET, ODBC in številne druge. V našem primeru je aplikacija za komunikacijo s podatkovnim strežnikom že uporabljala knjižnico ADO.NET in smo tako za prilagoditev aplikacije morali spremeniti samo povezavo do podatkovnega strežnika, kot je razvidno iz slike 12. S tem korakom smo tudi zaključili našo prilagoditev za uporabo strežnika SQL v oblaku.

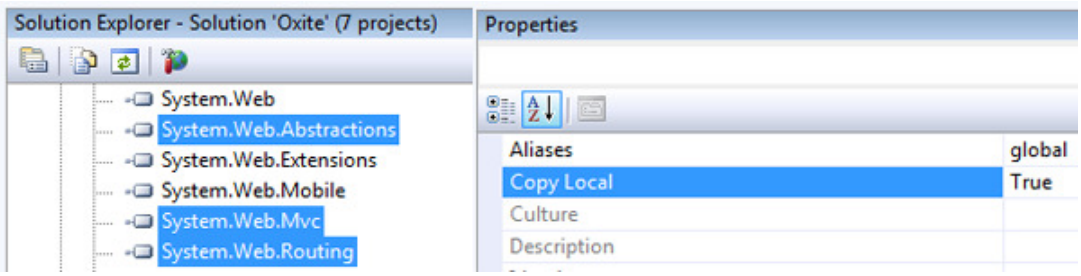
```
<connectionStrings>
  <add
    name="ApplicationServices"
    connectionString="
    Server=tcp:li4n7189hw.database.windows.net;
    Database=Oxite;
    User ID=peter;
    Password=**********;
    Trusted_Connection=False;"
  />
</connectionStrings>
```

Slika 12: Definranje nove povezave do strežnika SQL v oblaku

6.3.2 Selitev izvajalne kode v oblak

Po uspešni selitvi podatkov iz lokalnega strežnika na strežnik v oblaku smo bili pripravljeni na naslednji korak. Ta korak je bil namenjen selitvi izvajalne kode, ki je potrebna za delovanje naše aplikacije.

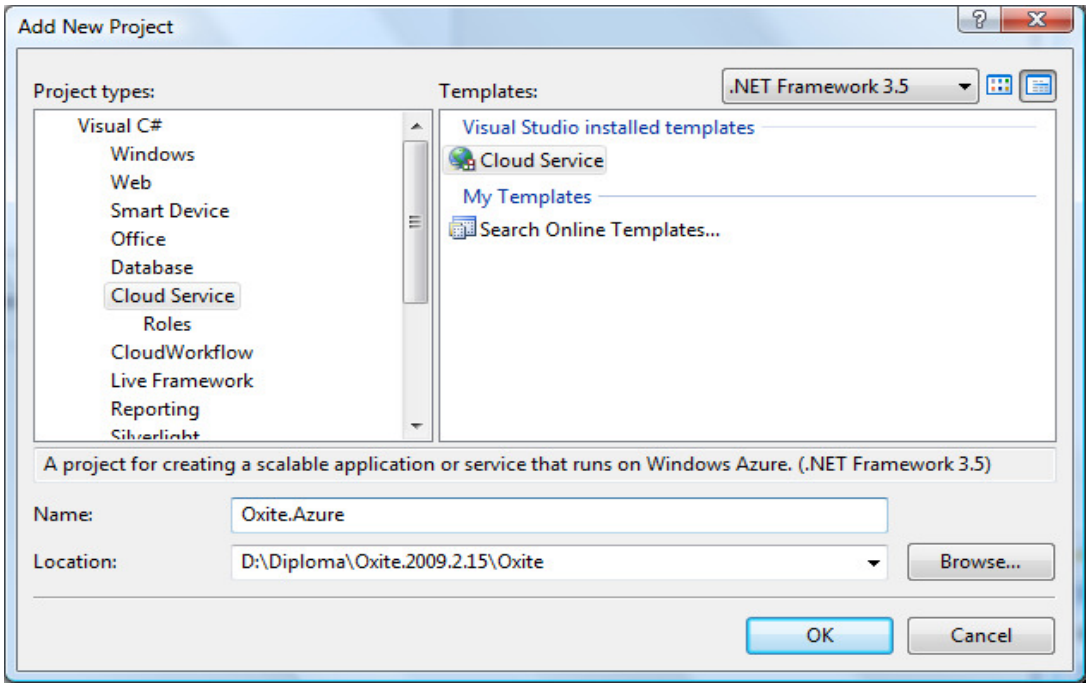
Kot smo videli v arhitekturni postavitvi, smo omenili da Oxite uporablja patent MVC (Model View Controler), za katerega skrbi ogrodje ASP.NET MVC. Žal pa trenutna verzija platforme Windows Azure (CTP) ne podpira ASP.NET MVC projektov. Tako je bil naš prvi korak, da smo knjižnicam, ki so del ASP.NET MVC ogrodja nastavili, da se skopirajo oz. vljučijo v končni produkt, kot je razvidno iz slike 13.



Slika 13: Vključitev ogrodja ASP.NET MVC v končni produkt

Naslednji korak, ki smo ga morali narediti je bil, da smo v obstoječo rešitev dodali novi projekt, ki bo poskrbel za pripravo in zapakiranje končnega produkta, prav tako bo

poskrbel za poganjanje in razhroščevanje na lokalnem razvijalskem okolju. Iz slike 14 je razvidno da smo kreirali projekt tipa »Cloud Service« in ga poimenovali »Oxite.Azure«.



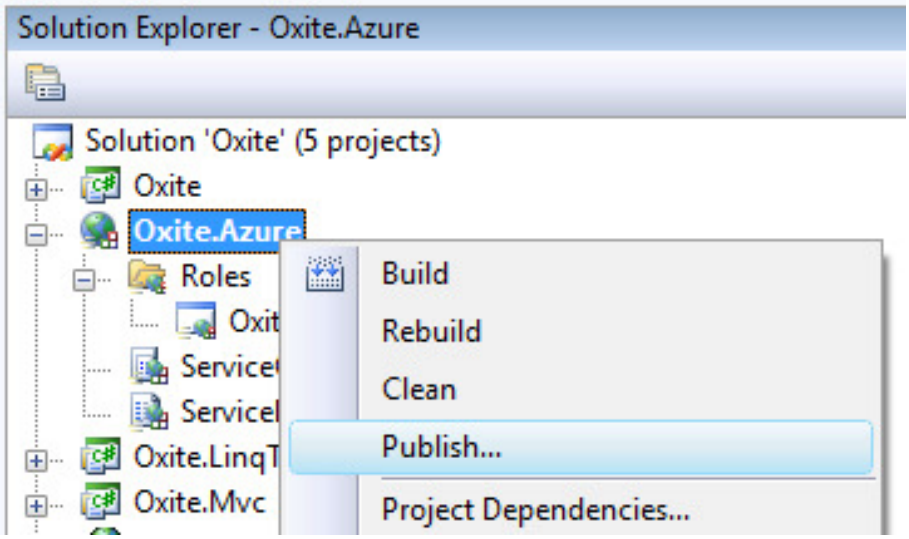
Slika 14: Kreiranje projekta ki skrbi za upravljanje platforme Windows Azure

Naslednji korak je bil, da smo znotraj našega novega projekta »Oxite.Azure« definirali, kateri obstoječi projekt je namenjen predstavitvi. Kot smo že omenili imamo lahko znotraj aplikacije dva različna načina delovanja aplikacije. To sta »Web Role« in »Worker Role«. Prvi način je namenjen delovanju kot predstavitvi, drugi način pa je namenjen obdelavi podatkov nekje v ozadju. Kot vemo naša aplikacija že vsebuje predstavitev, vendar ta ni definirana kot predstavitev katero bi znal naš projekt, ki skrbi za pripravo in zapakiranje produkta vključiti v svoj končni paket. Kot je razvidno iz slike 15 smo morali zato v projekt »OxiteSite« ročno dodati novo lastnost oz. opis projekta in sicer RoleType, s katero smo določili da bo aplikacija delovala kot predstavitev oz. Web Role.

```
<Project ToolsVersion="3.5" DefaultTargets="Build" xmlns="http://
  <PropertyGroup>
    <RoleType>Web</RoleType>
    <MvcBuildViews>true</MvcBuildViews>
    <Configuration Condition=" '$(Configuration)' == '' ">Debug</
    <Platform Condition=" '$(Platform)' == '' ">AnyCPU</Platform>
    <ProductVersion>9.0.30729</ProductVersion>
```

Slika 15: Dodajanje definicije načina delovanja

Kot lahko vidimo iz slike 16, smo tako bili pripravljeni na zadnji korak, ki nam je pripravil namestitveni paket za dodajanje aplikacije v oblak. Rezultat publikacije sta bile dve datoteki in sicer »Oxite.Azure.cspkg« in »ServiceConfiguration.cscfg«. Prva datoteka je navadna ZIP datoteka s spremenjeno končnico, ki v notranjosti skriva izvajalno kodo naše rešitve. V drugi daoteki so shranjene nastavitve za oblak, kot je npr.: število instanc, ki jih potrebuje naša rešitev za delovanje in nastavitve za podatkovne shrambe. Z navedenima datotekama smo bili pripravljeni za nalaganje aplikacije v oblak.



Slika 16: Postopek za pripravo paketa, ki ga bomo naložili v oblak

6.4 Dodajanje novih funkcionalnosti v obstoječo aplikacijo

V prejšnjih točki tega poglavja smo opisali postopek prilagoditve aplikacije za delovanje na platformi Windows Azure in tako uporabili nekaj gradnikov, ki jih trenutno ponuja

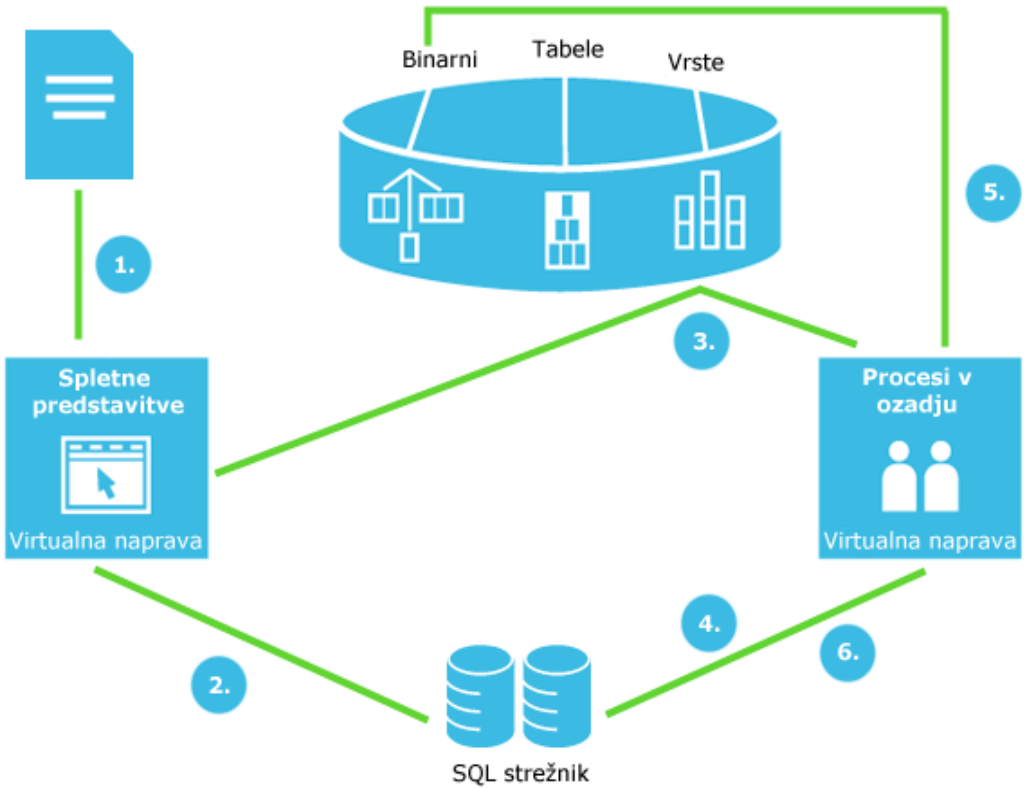
platforma. Z željo po vključitvi čimveč obstoječih gradnikov, smo se odločili za nadgradnjo funkcionalnosti aplikacije z novimi gradniki.

Zagotovo smo že velikokrat naleteli na spletne strani, katere so v opisu vsebovale slike, a žal te slike niso bile več dostopne. Do tega poblema ponavadi pride, ker uporabniki velikokrat v svoje strani vključujejo slike iz drugih spletnih strani nad katerimi pa nimajo kontrole. Da bi preprečili ta problem, smo se odločili dodati funkcionalnost, ki bo na nek način preprečevala takšne situacije. Vedeli smo, da bomo ta problem rešili tako, če bomo vsako sliko, ki je vključena v novi zapis shranili na neko varno lokacijo in potem zapise prikazovali z slikami iz varne lokacije. Ker pri tem nismo želeli obremeniti obstoječe aplikacije, ki služi predstavitvi, smo se odločili, da bomo namesto tega izdelali storitev, ki bo delovala v ozadju in bo pregledovala zapise, če vsebujejo kakšne slike in pri tem poskrbela za prenašanje slik in spreminjanje povezav do slik.

Kot lahko vidimo iz slike 17 ima trenutna funkcionalnost implementirana le prva dva koraka. Pri implementaciji nove funkcionalnosti pa je potrebno narediti še naslednje korake.

1. Uporabnik preko uporabniškega vmesnika vnese zapis v svoj spletni dnevnik in potrdi novi zapis.
2. S potrditvijo se zapis shrani v podatkovno zbirko v oblaku (SQL Azure). S tem korakom se tudi konča osnovna funkcionalnost obstoječe aplikacije.
3. Po shranjevanju zapisa se identifikacija novega zapisa doda v čakalno vrsto (queue) in tako signalizira procesu, ki teče v ozadju (Worker Role), da je na voljo novi zapis, ki ga je potrebno pregledati.
4. Proces, ki deluje v ozadju (Worker Role), zazna novo nalogo v čakalni vrsti (queue). Iz podatkovne zbirke vzame obstoječ zapis in pregleda, če vsebuje slike.
5. Vsako sliko, ki jo zasledimo v zapisu shranimo v zbirko, ki je namenjena shranjevanju binarnih podatkov (blob) in pri tem zamenjamo povezave do slik.

6. Obstoječemu zapisu zamenjamo vsebino za prikaz in to vsebino posodobimo tudi v podatkovni zbirki. Nova vsebina tako vsebuje povezave do slik iz lokacij, ki so pod našim nadzorom.



Slika 17: Diagram obstoječe in nove funkcionalnosti

V arhitekturni postavitvi smo že omenili, da aplikacija uporablja patent »dependency injection«, ki nam je omogočil, da nam ni bilo potrebno posegati v obstoječo funkcionalnost. Implementirali smo namreč vmesne člene, ki so poskrbeli za izvajanje nove funkcionalnosti in katere smo preko konfiguracijskih datotek vključili v izvajanje naše aplikacije. Kot lahko vidimo smo pri implementaciji obstoječega problema uporabili kar nekaj gradnikov, ki nam jih ponuja platforma Windows Azure in s tem zagotovo izboljšali samo delovanje naše aplikacije.

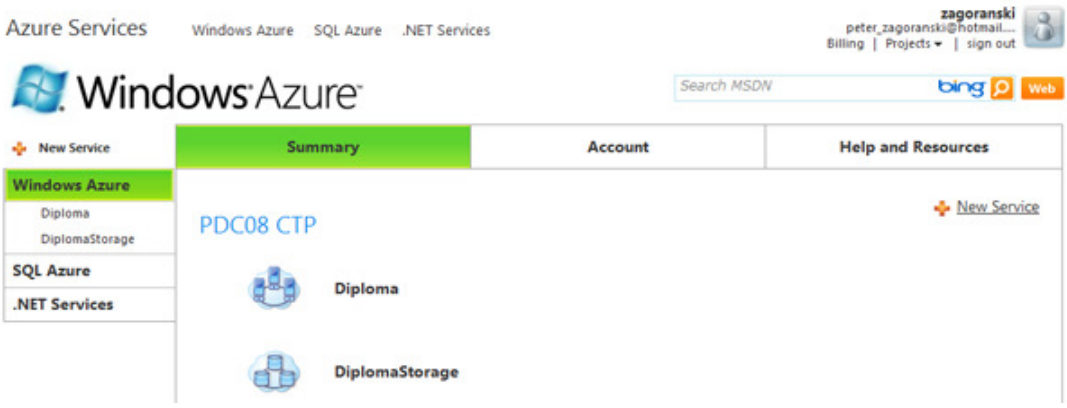
Naslednja funkcionalnost, ki smo jo dodali je bila ta, da smo omogočili uporabnikom spletnega dnevnika komunikacijo z avtorjem spletnega dnevnika s pomočjo spletnega

orodja , ki nam omogoča pogovor med udeležencema. Za to funkcionalnost smo uporabili gradnik Windows Live Messenger, ki je del ponudbe storitev Live Services. S tem korakom smo pokrili uporabo gradnikov iz vsake množice od storitev, ki nam jih ponuja Platforma Windows Azure.

Z namenom prikaza možnosti vključevanja storitev drugih ponudnikov smo si v ta namen izbrali storitev, ki nam jo ponuja podjetje Google. Gre za storitev, ki nam omogoča povezovanje uporabnikov (Friend Connect) z istimi interesi. V našem primeru gre za povezovanje uporabnikov, ki jim je všeč naš spletni dnevnik. Uporabniki se lahko namreč z uporabo enega izmed mnogih obstoječih avtentikacijskih sistemov včlanijo v skupino in bi lahko tako v realnih aplikacijah postali del velike socialne mreže.

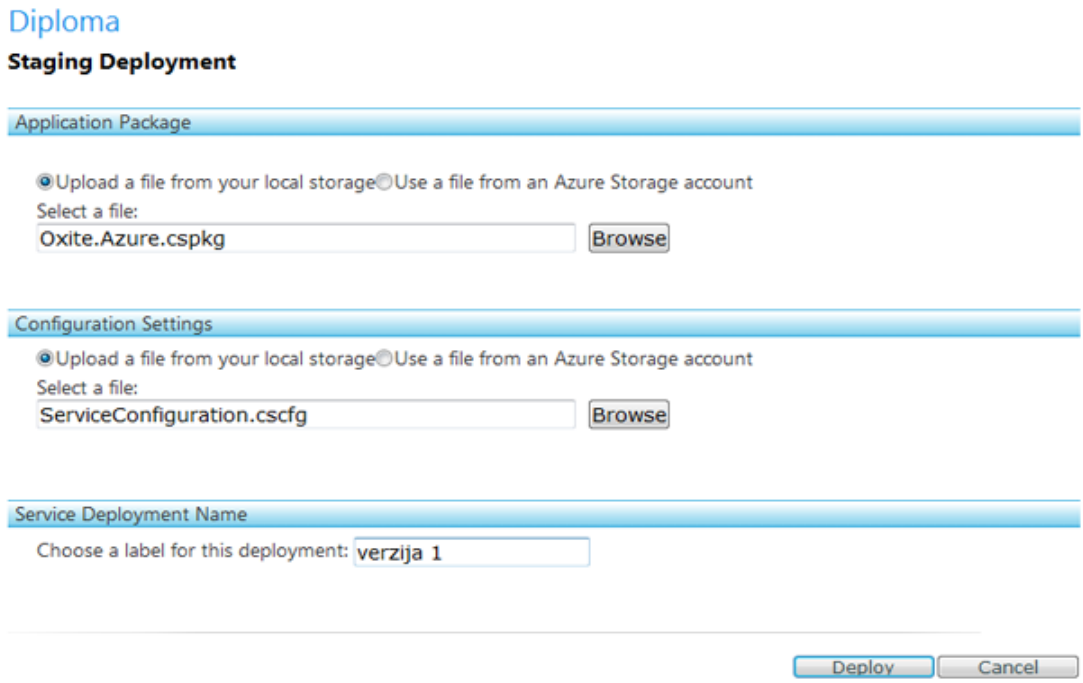
6.5 Upravljanje platforme Windows Azure

V prejšnjih dveh točkah tega poglavja smo si pogledali, kaj vse smo morali narediti, da smo prilagodili obstoječo aplikacijo na delovanje na platformi Windows Azure. Rezultat sta bile dve datoteki, ki ju potrebujemo za zagon aplikacije v oblaku. Preden smo lahko dejansko začeli uporabljat platformo Windows Azure smo morali opraviti še registriracijo na Microsoftovem portalu, ki je namenjen razvijalcem aplikacij Windows Azure in nam omogoča nalaganje aplikacij v oblak, spremljanje stanja aplikacij in vzdrževanje aplikacij. Ko smo opravili vse potrebne formalnosti z registracijo, kot je razvidno iz slike 18, smo lahko začeli z nalaganjem naše aplikacije v oblak.



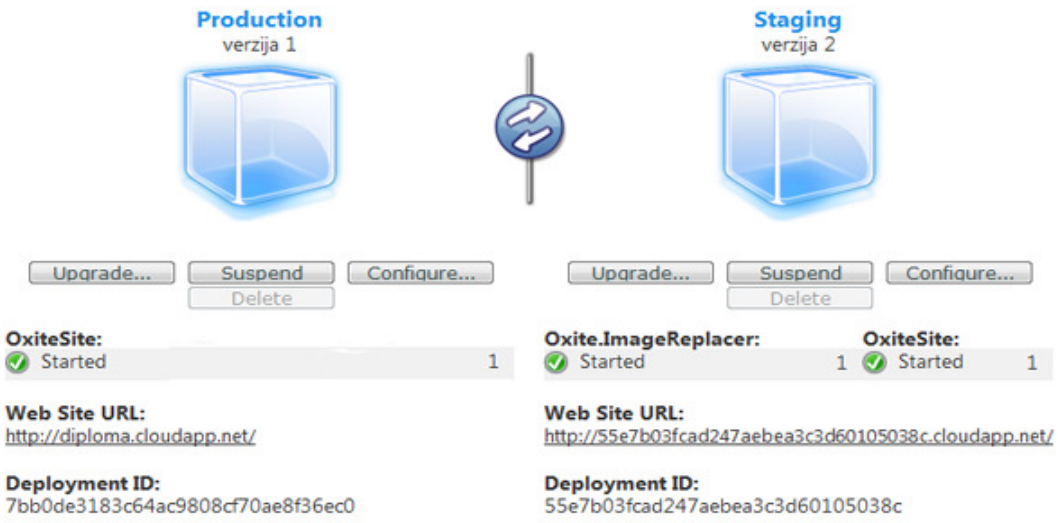
Slika 18: Portal za upravljanje platforme Windows Azure

Za začetek smo kreirali gostiteljsko storitev, v kateri bo tekla naša aplikacija. S pomočjo vmesnika, ki je prikazan na sliki 19 smo bili pripravljeni za nalaganje. Označili smo datoteko z izvajalno kodo, datoteko z nastavitvami, dodali kratek opis verzije in s potrditvijo naložili našo aplikacijo v oblak.



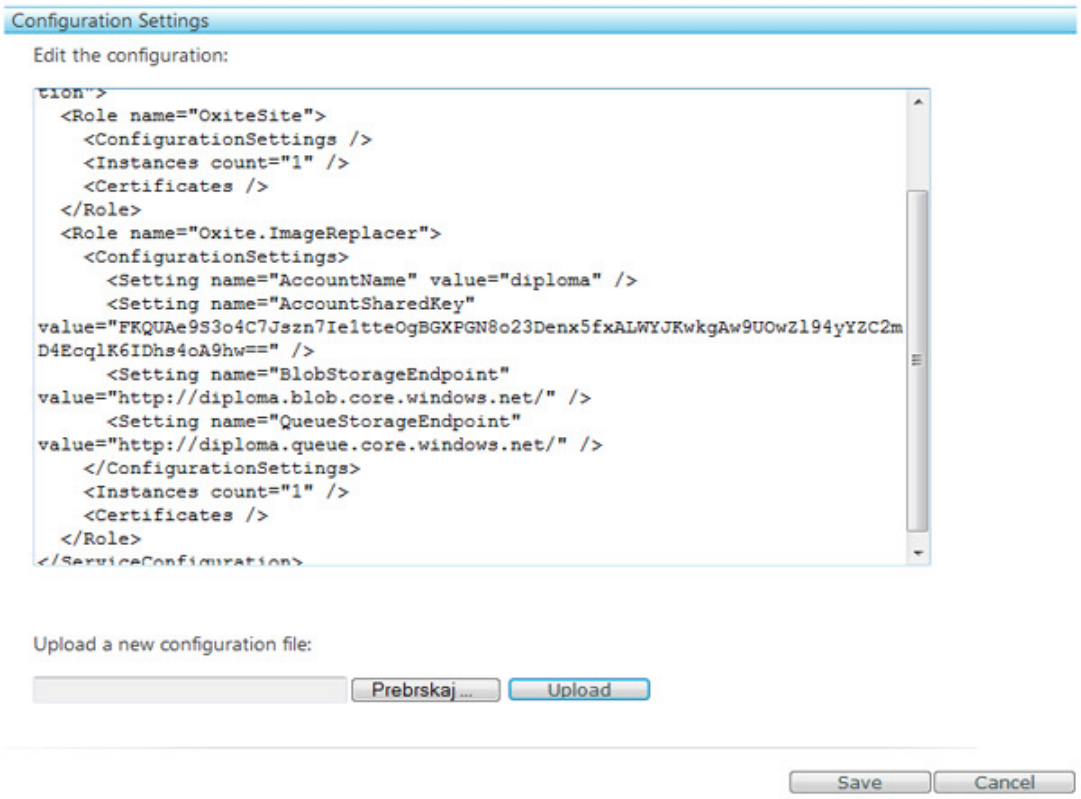
Slika 19: Vmesnik za nalaganje aplikacij v oblak

Po uspešni namestitvi aplikacije v oblak, smo bili pripravljeni na zagon aplikacije. Kot vidimo iz slike 20 nam platforma omogoča delovanje aplikacije na dve stopnjah. Prva je namenjena dejanski produkciji aplikacije, druga pa službi kot predprodukcijska stopnja, ki je namenjena zagonu aplikacije ter hitremu izmenjavanju med verzijama. Slednjo stopnjo lahko uporabljamo tudi za testiranje aplikacije. Kot vidimo na sliki 20 smo imeli v času nastanka slike v produkciji osnovno verzijo aplikacije, na drugi strani pa smo že preizkušali aplikacijo z novimi funkcionalnostmi (shranjevanje slik). Po zaključenem testiranju smo bili pripravljeni na zamenjavo verzij. S klikom na gumb v sredini sta se verziji zamenjali med seboj. Omeniti moramo, da je prehod med njima zelo hiter, enostaven in za uporabnika transparenten.



Slika 20: Prikaz vmesnika za poganjanje aplikacije v oblaku

S tem korakom je bila naša aplikacija pripravljena za uporabo.. Kot smo lahko videli, smo obe storitvi znotraj aplikacije poganjali na ločenih virtualnih napravah, kar je za začetek zadoščalo našim potrebam po uspešnem zagotavljanju storitve. Platforma Windows Azure pa nam omogoča, v primerih, ko pričakujemo povečano obiskanost naših strani, enostavno prilagajanje razširljivosti trenutnim potrebam in pri tem upošteva model »plačuj kolikor uporabljaš«. Kot lahko vidimo na sliki 21, nam platforma ponuja spletni vmesnik, s katerim lahko enostavno dodajamo oz. odvzemamo potrebo po dodatnih virih, ki bodo poskrbeli za uspešno zagotavljanje storitev.



Slika 21: Vmesnik za spreminjanje lastnosti posamezne storitve

6.6 Opis aplikacije

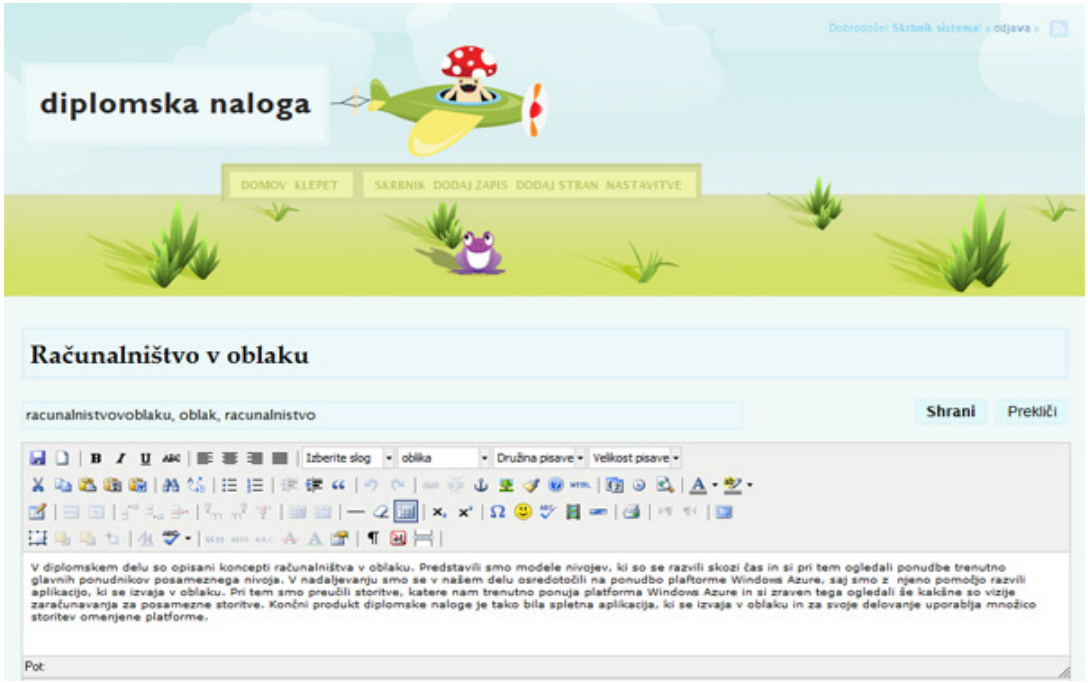
V uvodu tega poglavja smo videli, da smo za praktični del naše naloge uporabili obstoječo aplikacijo za vodenje spletnega dnevnika in njegovo izvajanje preselili v oblak. Namen spletnega dnevnika je, da omogoča avtorju vsakodnevno delitev svojih misli z ostalimi uporabniki na spletu in jim pri tem omogoča, da s pomočjo komentarjev izrazijo svoje mnenje o določeni temi. Iz tega lahko vidimo, da spletni dnevniki niso namenjeni samo izražanju misli, ampak služijo tudi kot dobro orodje za povezovanje med uporabniki. Pri dodajanju novega zapisa so ti urejeni po kronološkem vrstnem redu tako, da se novi zapisi vedno prikazujejo na vrhu strani in s tem obiskovalcem omogočajo spremljanje sprememb na strani. Kot vidimo bi lahko s temi zahtevami pokrili osnovno funkcionalnost spletnih dnevnikov, vendar nam naša aplikacija ponuja mnogo več in si bomo tako v naslednjih odstavkih pogledali katere so te dodatne funkcionalnosti.

Prisotnost in uporabo spletnih dnevnikov lahko opazujemo na spletu že kar nekaj časa. V tem času se je razvilo mnogo sistemov, ki nam omogočajo vodenje spletnega dnevnika, vendar so bili prehodi med njimi prava nočna mora. S tem namenom je prišlo do razvoja označevalnega jezika BlogML (weblog markup language), ki nam omogoča označevanje zapisov po določenih pravilih in prenos zapisov med različnimi sistemi, ki podpirajo takšno označevanje. Naša aplikacija podpira takšen jezik in nam s tem trenutno omogoča uvoz starejših zapisov iz drugih sistemov za vodenje spletnega dnevnika. Izvoz pa nažalost trenutno še ni podprt.

Ena izmed pomembnih funkcionalnosti je tudi ta, da nam Oxite omogoča enostavno vizualno spreminjanje obstoječe aplikacije. Kot je prikazno na sliki 22 smo v našem delu uporabili vizualno podobo z imenom TheLab, katere avtor je Nathan Heskew.

Slika 22: Izgled spletnega devnika z vizualno podobo TheLab.

Glavni namen spletnih dnevnikov je, da omogočajo avtorjem enostaven vnos novih zapisov in prav tako enostavno urejanje obstoječih. Tako smo v našo aplikacijo za urejanje zapisov vključili zmogljiv spletni urejevalnik TinyMCE (slika 23), katerega funkcionalnost je do neke mere podobna namiznim urejevalnikom besedila. Urejevalnik nam namreč omogoča oblikovanje besedila na zelo enostaven način. Kot dodatno funkcionalnost smo v urejevalnik integrirali tudi pregledovanje slik iz oblaka in pravtako tudi nalaganje slik v oblak. S tem ko smo oblikovali zapis pa še naše delo za dodajanje zapisa ni bilo zaključeno. Vsakemu zapisu lahko namreč tudi podamo značilke (tags), s katerimi lahko enostavno združimo podobne zapise in nam tudi olajša kasnejše iskanje podobnih zapisov. Kot smo že omenili, je ena izmed funkcionalnosti spletnega dnevnika tudi ta, da nam in uporabnikom omogoča izražanje svojih misli oz. komentiranje zapisa, vendar lahko to tudi preprečimo, saj lahko pri dodanaju novega zapisa definiramo ali za ta zapis omogočamo komentiranje. Za vsak zapis lahko tudi definiramo »Prijazno spletno povezavo«, ki uporabnikom omogoča, da si lažje zapomnijo povezavo do zapisa. Na koncu nam preostane še samo, da definiramo ali želimo zapis objaviti ali pa ga želimo trenutno samo shraniti kot osnutek, katerega bomo kasneje nadgradili. V primeru objave moramo podati datum in čas objave.



Slika 23: Stran za dodajanje novega zapisa

Kot smo lahko videli nam aplikacija omogoča enostavno urejanje zapisov s pomočjo spletnega vmesnika, vendar to ni edina možnost za urejanje zapisov. Ena od funkcionalnosti aplikacije ja namreč tudi ta, da nam omogoča urejanje zapisov s pomočjo protokola MetaWeblog. Ta protokol nam omogoča dodajanje, urejanje in brisanje zapisov s pomočjo spletnih storitev. S tem so nam omogočene operacije nad zapisi s pomočjo namenskih programov, ki podpirajo takšen protokol. Ena izmed takšnih aplikacij je aplikacija Windows Live Writer.

Omenili smo, da lahko določene zapise tudi komentiramo. Avtor komentarja lahko pri dodajanju vnese tudi elektronsko pošto in označi sledenje spremembam nad zapisom. S tem se avtorju komentarja ob spremembi zapisa s pomočjo elektronske pošte posreduje sporočilo o spremembi zapisa. Ena od funkcionalnosti aplikacije je tudi, da omogoča skrbniku sistema filtriranje komentarjev pred prikazom širši množici. V aplikaciji lahko namreč nastavimo, da vsak novi komentar pred prikazom potrebuje tudi potrditev skrbnika. S tem se lahko zaščitimo pred prikazovanjem nezaželenih sporočil oz. komentarjev.

Naslednja funkcionalnost aplikacije je ta, da nam omogoča enostavno iskanje po vsebini zapisov. Iskanje nam je namreč omogočeno v samem spletnem dnevniku ali pa s pomočjo protokola OpeanSearch, ki omogoča uporabniku iskanje po vsebini spletnega dnevnika s pomočjo iskalnega polja v spletnem brskalniku, ki podpira omenjen protokol.

Omenili smo, da so novi zapisi pri pregledu urejeni po kronološkem vrstnem redu in tako omogočajo obiskovalcem spletnega dnevnika spremljanje sprememb na strani. Z namenom da bi poenostavili spremljanje raznovrstnih spletnih strani je nastal protokol »Resnično preprostih objav« (Really Simple Syndication - RSS). Za spremljanje takšnih objav potrebujemo namensko aplikacijo, ki spremlja spremembe objav iz različnih (nam zanimivih) virov. Prednosti teh virov so, da nam za ogled sprememb ni potrebno naložiti celotne spletne strani ampak samo ključno vsebino. Naša aplikacija nam tako ponuja možnost spremljanja sprememb tudi s pomočjo teh virov.

Aplikacija nam omogoča tudi povezovanje s pomočjo protokola »povratne povezave« (Trackback) z drugimi spletnimi dnevniki, ki pravtako podpirajo takšen protokol. Tako lahko s svojih zapisih dodamo povezave na druge spletne dnevnike s podobno vsebino in aplikacija bo tako poskrbela za obvestitev izvornga spletnega dnevnika na našo objavo. Pri tem pa bo izvorni spletni dnevnik objavil pod vsebino zapisa (na katerega se navezujemo) povezavo na naš spletni dnevnik (Pingback). S tem pa je uporabnikom omogočeno enostavno sledenje obravnavani tematiki zapisa po različnih spletnih dnevnikih.

Na koncu nam ostane še opis skrbniškega modula, ki ga prikazuje slika 24 in nam omogoča naslednje funkcionalnosti:

- Pregled in upravljanje komentarjev: kot smo že omenili nam stran omogoča filtriranje komentarjev. To možnost nam ponuja skrbniški modul, ki nam prikazuje zadnje komentarje in nam omogoča, da te komentarje potrdimo ali celo zavrnemo iz zbišemo.
- Pregled in upravljanje zapisov: skrbniški modul nam omogoča poleg osnovne funkcionalnosti, ki služi prikazu zapisov tudi povezave, ki služijo za urejanje posameznega zapisa ali celo brisanje zapisa.
- Spreminjanje nastavitev strani. Znotraj te funkcionalnosti lahko spreminjamo množico parametrov, ki narekujejo delovanje spletnega dnevnika. Ti parametri so: ime strani, opis strani, definicija izgleda strani, lokalizacija strani in različna obnašanja strani, kot so omogočanje iskanja s pomočjo protokola OpenSearch, onemogočanje komentiranja za celotno stran, privzeta odobritev komentarjev in avtomatsko sledenje zapisov za komentatorje.
- Upravljanje vtičnikov: aplikacija nam ponuja nadgrajevanje funkcionalnosti s pomočjo vtičnikov. Skrbniški modul nam omogoča vključevanje posameznega vtičnika v spletni dnevnik. Prav tako nam omogoča spreminjanje nastavitev oz. parametrov, ki se navezujejo na delovanje posameznega vtičnika.
- Dodajanje in upravljanje podstrani oz. področij. Aplikacija nam poleg dodajanja zapisov v spletni dnevnik omogoča tudi, dodajanje podstrani. Razlika med zapisi in podstranmi je, da se podstrani ne prikazujejo med zadnjimi vnosi v dnevnik, ampak služijo kot dodatna funkcionalnost osnovne strani. V našem primeru smo dodali

podstran s katero smo omogočili uporabnikom komunikacijo z skrbnikom sistema s pomočjo spletne aplikacije za kleept (Windows Live Messenger).

- Spreminjanje gesla za dostop do skrbniškega modula.

Slika 24: Izgled skrbniške nadzorne plošče

7 SKLEP

Računalništvo kot ga poznamo danes se bo v prihodnosti zagotovo spremenilo, eden izmed ključnih dejavnikov, ki bodo poskrbeli za te spremembe je zagotovo uveljavitev pristopa računalništva v oblaku. Videli smo, da ima model računalništva v oblaku mnogo pozitivnih lastnosti, na drugi strani pa ima tudi nekaj pomanjkljivosti oz. nedefiniranosti. Če pogledamo mnenja drugih uporabnikov lahko vidimo, da se ta razlikujejo. Nekateri govorijo o velikih uspehih, na drugi strani pa imamo tudi mnogo razočaranj. Glavni razlogi za kritike gredo predvsem na nedokončanost storitev, pojavljajo se tudi občasni izpadi ali slabša odzivnost ob večjih obremenitvah in v nekaterih primerih tudi pomanjkljivost definiranja standardov.

Zaradi naraščajoče pribljubljenosti računalništva v oblaku, imamo trenutno na voljo množico najrazličnejših storitev, nekatere izmed njih nažalost tudi nezadovoljive. K temu je največkrat botrovala želja po izkoriščanju popularnosti in želja po hitremu zaslužku. Tako bo zagotovo eden izmed nadaljnih korakov za razvoj izločitev teh neprimernih storitev. Omeniti moremo tudi, da so kljub pozitivni reklami vse storitve v oblaku še nekoliko nezrele in potrebujejo čas, da jih bodo ponudniki razvili do te mere kot bi pričakovali in ki jih navajajo trditve v oglasih. Razveseljivo pa je dejstvo, da večina izmed storitev že temelji na modernih tehnologijah in standardih, kot so AJAX, XML, JSON in REST. Za nadaljni uspeh bi bila zaželjena tudi standardizacija, ki bi nam zagotavljala združljivost in prenosljivost med posameznimi ponudniki oz. neodvisnost od ponudnika.

V diplomskem delu smo si pogledali, kakšne so možnosti uporabe računalništva v oblaku in si ogledali njegove prednosti in slabosti. Pri svojem raziskovanju smo se osredotočili na trenutne glavne ponudnike in si pri tem izbrali enega izmed teh za implementacijo naše diplomske naloge. Odločili smo se za uporabo platforme Windows Azure, saj nam je ta iz programerskega vidika in s poznavanjem razvijalskega okolja .NET omogočala hitro

vključitev in implementacijo novih storitev v oblaku. Z aplikacijo, ki smo jo izbrali, smo želeli dokazati možnost hitre in enostavne prilagoditve obstoječe aplikacije za izvajanje v oblaku. Izbirali smo med aplikacijami, ki za svoje izvajanje potrebujejo podatkovno zbirko, saj smo lahko s tem prikazali enega izmed glavnih razlogov za uporabo računalništva v oblaku, ki nam omogoča shranjevanje podatkov v oblak. Odločili smo se za aplikacijo, ki je namenjena vodenju osebnega spletnega dnevnika in nam omogoča enostavno nadgrajevanje osnovnih funkcionalnosti s pomočjo storitev v oblaku.

Pri uporabi Platforme Windows Azure, ki je trenutno v skupinskem tehnološkem pregledu (CTP), smo vedeli, da bomo lahko z uporabo naleteli na kopico težav, saj se platforma trenutno še ocenjuje in nadgrajuje. V zadnjem mesecu smo tako tudi opazili, da se pri Microsoftu temeljito pripravljajo na predstavitev platforme na njihovi zelo znani konferenci PDC 2009 (Professional Developers Conference), ki se bo odvijala novembra, saj so v zadnjem času dodobra spremenili uporabniške vmesnike za upravljanje same platforme. V našem primeru gre sicer za zelo enostavno aplikacijo, vendar nas je vseeno presenetilo dejstvo, da pri selitvi oz. implementaciji aplikacije nismo naleteli na bistvene težave, ki bi jih lahko pričakovali. Nekaj napak smo sicer zasledili v razvijaslkem okolju, ki poskrbi za lokalni zagon aplikacije v oblaku, vendar jih nismo ocenili kot kritične, saj menimo, da bodo odpravljene preden bo platforma namenjena dejanski uporabi. Z našim pregledom smo si tako lahko ustvarili dobro mnenje o sami platformi. Mislimo, da je Microsoft naredil tukaj korak v pravo smer, saj bo sama platforma oz. deli storitev, ki jih ponujajo tako zanimivi za marsikaterega uporabnika.

Na podlagi naših raziskav v diplomskem delu lahko z zagotovostjo trdimo, da je model ponudbe računalništva v oblaku gotovo eden izmed glavnih trendov pri razvoju računalništva današnjega časa. Ponudbe najrazličnejših storitev v oblaku se namreč povečujejo iz dneva v dan in nam tako nakazujejo, da bodo posamezne storitve v oblaku v prihodnosti zagotovo predstavljale najrazličnejše dele operacijskih sistemov prihodnosti, ki bodo izredno zmogljivi, globalno razpršeni in dostopni v vsakem trenutku, ne glede na to kje se nahajamo in s pomočjo katere naprave dostopamo do posamezne storitve. Mislimo, da si bodo lahko trenutni glavni ponudniki pridobili bistveno prednost pred zamudniki, saj bodo v primeru razširjene uporabe ti že imeli delujočo rešitev. S tega vidika vidimo, da so

zagotovo postavljeni zanimivi temelji za prihodnost računalništva, vendar bomo morali na dejansko, razširjeno in vsakodnevno uporabo še nekoliko počakati.

8 LITERATURA IN VIRI

[1] David Chappell, *Introducing the Azure services platform, An early look at Windows Azure, .NET services, SQL services nad Live services*, Microsoft corporation, oktober 2008.

[2] Michael Miller, *Cloud computing, Web-Based Applications That Change the Way You Work and Collaborate Online*, Que Publishing, avgust 2008.

[3] Windows Azure, http://www.microsoft.com/windowsazure/windowsazure, obiskano 16. oktober 2009.

[4] SQL Azure, http://www.microsoft.com/windowsazure/sqlazure, obiskano 16. oktober 2009.

[5] .NET Services, http://www.microsoft.com/windowsazure/dotnetservices, obiskano 16. oktober 2009.

[6] Cenik Microsoftovih storitev, http://www.microsoft.com/windowsazure/pricing, obiskano 16. oktober 2009.

[7] Amazon Web Services, http://aws.amazon.com/ec2/, obiskano 16. oktober 2009.

[8] Google App Engine, http://code.google.com/intl/sl/appengine/docs/whatisgoogleappengine.html, obiskano 16. oktober 2009.

[9] VMWare, http://www.vmware.com/solutions/cloud-computing/vcloud.html, obiskano 16. oktober 2009.

[10] SalesForces, http://www.salesforce.com/platform/cloud-platform/, obiskano 16. oktober 2009.

[11] Oxite, http://www.codeplex.com/oxite/, obiskano 16. oktober 2009.

[12] Oxite TheLab, http://nathan.heskew.com/Oxite-Skins/TheLab, obiskano 16. oktober 2009.

[13] ASP.NET MVC, http://www.asp.net/mvc/whatisaspmvc/, obiskano 16. oktober 2009.

[14] Unity Application Block, http://msdn.microsoft.com/en-us/library/dd203104.aspx, obiskano 16. oktober 2009.

[15] BlogML, http://www.codeplex.com/BlogML, obiskano 16. oktober 2009.

[16] AntiXSS, http://www.codeplex.com/AntiXSS, obiskano 16. oktober 2009.

[17] MetaWeblog, http://www.xmlrpc.com/metaWeblogApi, obiskano 16. oktober 2009.

9 PRILOGE

9.1 Seznam slik