Dissertação de Mestrado em Engenharia Informática - Computação Móvel

Serviços Empresariais Em Windows Azure

Edgar José Casaleiro dos Santos

Dissertação de Mestrado realizado sob a orientação do Doutor Patrício Rodrigues Domingues e do Doutor Sílvio Priem Mendes, Professores da Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Leiria.

Leiria, Setembro de 2012

Esta página foi intencionalmente deixada em branco

A todos os que fizeram parte da minha vida académica e me tornaram melhor pessoa e melhor profissional.

Resumo

O entusiasmo criado em torno do paradigma da computação na nuvem leva as instituições a considerar a sua inclusão em novos projetos e em projetos existentes. As empresas necessitam contudo, de conhecer em antemão as especificidades do novo paradigma e determinar se o negócio beneficia com o uso de serviços de *cloud-computing*. Do mesmo modo, é importante perceber se a nuvem se encontra preparada para o sector empresarial e qual a curva de aprendizagem e esforço implicados na sua adoção. Nesta dissertação, analisaram-se todos estes pontos através de uma das maiores plataformas de *cloud-computing*, o Microsoft Windows Azure. Para se aferir o esforço seguiu-se uma abordagem prática, na qual se desenvolveu uma aplicação empresarial que posteriormente foi migrada para a nuvem. O custo de uma solução alojada localmente foi comparado ao custo de uma solução na nuvem tendo-se recorrido a dados de servidores reais para dimensionar o serviço. Foi concluído que não existe ganho financeiro para uma instituição se a mudança para o paradigma da computação na nuvem não tiver em vista o aproveitamento de uma das suas características diferenciadoras como por exemplo, o associativismo de custo, a elasticidade ou a rapidez de aprovisionamento. Por outro lado, os resultados mostram uma poupança na ordem dos 40% caso se faça uso da elasticidade da *cloud*.

Palavras chave: Computação na nuvem, serviços empresariais, Windows Azure

Esta página foi intencionalmente deixada em branco

Abstract

The hype created around the cloud-computing paradigm makes the companies consider its use in future and ongoing projects. However, companies need a prior understanding of the specific characteristics of the new paradigm in order to know the benefits that come with it. Likewise, it is important to understand wheter the cloud is prepared for the enterprise and what is the learning curve and the effort associated with such adoption. All these topics are covered throughout the analysis, in which Microsoft Windows Azure, one of the major cloud-computing plataforms, was used. To measure the effort, an enterprise application was developed and then migrated to the cloud. The cost of running an on-premises solution was compared to the cost of running an equivalent solution in the cloud. To achieve a fair comparison, data from production servers has been used to properly size the solutions. Results show that there is no financial gain by using the cloud without taking advantage of any of its specificities. On the other hand, the results show a 40% saving if the company takes advantage of elasticity, one of the main characteristics of the cloud.

Keywords: Cloud computing, enterprise services, Windows Azure

Esta página foi intencionalmente deixada em branco

Índice de Ilustrações

Esta página foi intencionalmente deixada em branco

Índice de Tabelas

Esta página foi intencionalmente deixada em branco

Índice de Listagens

Esta página foi intencionalmente deixada em branco

Lista de Siglas

ACID	Atomicity, Consistency, Isolation, Durability
AMI	Amazon Machine Image
API	Application Programming Interface
ARPANET	Advanced Research Projects Agency Network
AWS	Amazon Web Services
BD	Base de Dados
BLOB	Binary Large Object
CAPEX	Capital Expenditure
CDN	Content Delivery Network
CPU	Central Processing Unit
DARPA	Defense Advanced Research Project Agency
DER	Diagrama de Entidade Relacionamento
DLL	Dynamic Link Library
DNS	Domain Name System
DU	Database Unit
EBS	Elastic Block Store
EC2	Elastic Compute Cloud
EUA	Estados Unidos da América
EUR	Euro
FTP	File Transfer Protocol
GAE	Google App Engine
GB	Gigabyte
GCE	Google Compute Engine
GHz	Gigahertz
GUI	Graphical User Interface
HIT	Human Intelligence Tasks
HPC	High-Performance Computing
HTC	High-Throughput Computing
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IaaS	Infrastructure as a Service
IDE	Integrated Development Environment
IIS	Internet Information Services

IP	Internet Protocol
IPTO	Information Processing Techniques Office
IT	Information Technology
ITU-T	International Telecommunications Union
KB	Quilobyte
LINQ	Language Integrated Query
MB	Megabyte
NAT	Network Address Translation
noSQL	Not Only SQL
NTFS	New Technology File System
OLAP	Online Analytical Processing System
OPEX	Operation Expenditure
P2P	Peer-to-Peer
PaaS	Platform as a Service
PC	Personal Computer
PUE	Power Usage Effectiveness
PXE	Preboot eXecution Environment
RAM	Random Access Memory
RDBMS	Relational Database Management System
RDP	Remote Desktop Protocol
RDS	Relational Database Service
REST	Representational State Transfer
RF	Requisito Funcional
RFC	Request For Comments
RNF	Requisito Não Funcional
S3	Simple Storage Service
SaaS	Software as a Service
SDK	Software Development Kit
SLA	Service-Level Agreement
SMS	Short Message Service
SO	Sistema Operativo
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SQS	Amazon Simple Queue Service
SSD	Solid State Disk
SSL	Secure Sockets Layer
SSMS	SQL Server Management Studio
TB	Terabyte

TCO	Total Cost of Ownership
TCP	Transmission Control Protocol
TI	Tecnologias da Informação
TLS	Transport Layer Security
T-SQL	Transact-SQL
UDP	User Datagram Protocol
UPS	Uninterruptible Power Supply
URL	Uniform Resource Locator
USD	United States Dollar
VHD	Virtual Hard Disk
VLAN	Virtual Local Area Network
VM	Virtual Machine
VPC	Virtual Private Cloud
VPN	Virtual Private Network
WASABi	Autoscaling Application Block
XML	Extensible Markup Language
XSS	Cross-Site Scripting

Esta página foi intencionalmente deixada em branco

Índice

Esta página foi intencionalmente deixada em branco

1. Introdução

1.1. Enquadramento e Motivação

O conceito de computação como um serviço não é original mas a interpretação moderna desse conceito é muito diferente do que se assistiu na década de 70 com os sistemas de *time-sharing*. Nas suas diferentes formas a nuvem tem, hoje em dia, o propósito de: democratizar o acesso a recursos computacionais (*Infrastructure as a Service*, *IaaS*); concentrar as equipas de Tecnologias da Informação *(TI)* no desenvolvimento de aplicações minimizando tarefas de manutenção e administração de infraestrutura (*Platform as a Service*, *PaaS*); e disponibilizar soluções *Web* com o mesmo tipo de funcionalidades que seriam expectáveis numa aplicação *desktop* (*Software as a Service*, *SaaS*). Estas valências despertaram o interesse de empresas das mais variadas áreas de negócio [1] que mostraram predisposição no desenvolvimento de aplicações baseadas no novo paradigma da computação na nuvem. Por certo, o uso de um novo paradigma exige às empresas um esforço adicional, que se acredita poder ser minimizado pela leitura do presente documento.

O intuito deste trabalho é apresentar o paradigma para que uma empresa perceba se a nuvem é adequada ao seu negócio. Ao ler este trabalho deve ser possível compreender os benefícios reais da nuvem, conhecer os principais fornecedores de serviço e aferir o esforço e custo exigidos pela mudança.

1.2. Definição do Problema

O entusiasmo e o facilitismo de acesso aos recursos da *cloud* pode levar a que as instituições abordem o novo paradigma sem terem uma estratégia definida. Sendo um paradigma recente é necessário determinar antecipadamente quer a adequação do negócio à nuvem, quer a adequação da própria nuvem ao setor empresarial. Julgou-se por isso relevante expor os benefícios reais da nuvem, dar a conhecer os principais fornecedores de serviço e aferir qual a curva de aprendizagem e o esforço implicados na sua adoção.

1.3. Estrutura

O documento está dividido nos capítulos que passam a ser descritos:

Capítulo 1: Capítulo introdutório. É apresentado o problema e a estrutura do documento.

Capítulo 2: Este capítulo apresenta o conceito de computação como serviço e as arquiteturas e tecnologias que o tornaram possível.

Capítulo 3: Este capítulo mostra as diferentes categorias da computação na nuvem. Mostra também as características do paradigma que o tornam aliciante para as empresas e os principais entraves à sua adoção.

Capítulo 4: Neste capítulo são identificados os componentes comuns às soluções de *cloud computing*. De seguida, são apresentados os serviços disponibilizados pelos maiores fornecedores de computação na nuvem para que o leitor possa averiguar qual a solução que melhor se adequa ao seu caso particular.

Capítulo 5: Foi criada uma aplicação que posteriormente foi migrada para a nuvem de modo a aferir o esforço implicado na tarefa. Ao longo do capítulo são indicados quais os problemas encontrados assim como, as opções tomadas para tornar a aplicação conforme com o novo paradigma.

Capítulo 6: Este capítulo indica algumas opções que permitem otimizar o custo de uma aplicação na nuvem.

Capítulo 7: Neste capítulo é feita uma análise critica a algumas características associadas à computação na nuvem que não se mostraram verdadeiras ou que ficaram aquém do esperado.

Capítulo 8: Foram efetuados cálculos que permitem comparar o custo de uma solução *on-premises* com uma solução na nuvem. Os cálculos têm por objetivo averiguar a viabilidade do modelo do ponto de vista financeiro.

Capítulo 9: Fecha o documento com as conclusões tiradas ao longo da dissertação.

2. A Computação como Serviço

2.1. Introdução

O termo *cloud computing* é recente no entanto a visão da computação como um serviço público ao nível da rede de eletricidade, água ou telefone data da década de sessenta, época em que foram criadas as bases para a *Internet*.

Estávamos no período da Guerra Fria e o espaço era usado para a afirmação tecnológica e ideológica dos países, em 1957 a União Soviética lançava o satélite *Sputnik* colocando-se assim na frente da corrida espacial. Este facto gerou um tremendo impacto politico e psicológico na América e evidenciou lacunas no plano científico e tecnológico americano. Como consequência, menos de um ano depois foi fundada a *DARPA* (*Defense Advanced Research Projects Agency*) tendo como objetivo principal a criação de tecnologias e sistemas que colocassem a Defesa Americana em vantagem [2]. Em 1962 foi contratado Joseph Carl Robnett Licklider para liderar o departamento *IPTO* (*Information Processing Techniques Office*) e foi deste departamento que saiu a primeira rede de comutação de pacotes a *ARPANET* (*Advanced Research Projects Agency Network*) [3]. J.C.R. Licklider ficou conhecido como uma das pessoas mais influentes da história das *TI* e a sua investigação criou as bases daquilo a que viríamos a chamar *Internet*, atribuindo-se a ele a primeira referência ao *cloud computing* [4]. Também John McCarthy e Douglas Parkhill previram o surgimento deste paradigma ainda nos anos sessenta, o primeiro sugeriu que, se os sistemas que defendia se tornassem os computadores do futuro, então "um dia a computação poderia ser disponibilizada como um serviço público, tal como o telefone é um serviço público" [5], o segundo foi ainda mais longe ao publicar um livro denominado "The Challenge of the Computer Utility" [6] no qual explora grande parte das características que atribuímos hoje à nuvem tais como: a disponibilização da computação como um serviço, o conceito de elasticidade e o facto dos recursos disponibilizados parecerem infinitos aos olhos do utilizador.

O *cloud computing* amadureceu desde os anos sessenta e ressurgiu agora, beneficiando de novos paradigmas como a computação distribuída e de novas tecnologias como a virtualização e os *Web Services*. O aumento da largura de banda nos acessos à *Internet* fixa e móvel desempenhou também um papel crucial ao abrir caminho para uma *Web* mais rica, mais social e mais cooperativa, a *Web* como plataforma [7]. Uma das categorias de *cloud*

computing o *SaaS* surgiu com esta mudança pois, começaram a ser disponibilizadas aplicações *Web*, pagas mediante a utilização e com o mesmo tipo de funcionalidades que seriam espetáveis numa aplicação *desktop*. Estas aplicações passaram a ser vistas como serviços que estão sempre atualizados e que estão sempre disponíveis onde quer que o utilizador esteja e independentemente do dispositivo de acesso. A nuvem permitiu que o utilizador acedesse às aplicações em *PC*s (*Personal Computer*), *smartphones* e *tablets* disponibilizando uma experiência de utilização uniforme entre dispositivos e delegando para si grande parte do processamento das aplicações.

A procura por serviços *Web* 2.0 nas mais variadas plataformas é uma realidade que tende a aumentar no futuro próximo estimando-se que o volume de tráfego de dados móvel aumente 10000% até 2015 [8]. Esta tendência faz com que muitas empresas olhem para a nuvem como um meio para desenvolver o seu negócio.

O surgimento da computação como um serviço nos dias de hoje foi portanto, potenciada pela melhoria das condições de acesso para a qual contribuiu o aumento da largura de banda nas ligações à *Internet*, pelo aproveitamento da janela de oportunidade criada pelo *Web* 2.0 e pela proliferação de dispositivos móveis com ligações de alto débito. Beneficiou ainda de um conjunto de novos paradigmas e tecnologias dos quais podemos destacar:

- Sistemas de *Time-Sharing*
- Sistemas Distribuídos
- Virtualização
- *Web Services*

2.2. Sistemas de Time-Sharing

O conceito *de time-sharing* significa a partilha de um recurso por um determinado período de tempo. Na computação o *CPU* (*Central Processing Unit*) é partilhado por um conjunto de tarefas de forma a aproveitar da melhor maneira os recursos disponíveis. Para isso, o *CPU* atende cada tarefa por um determinado período de tempo sendo que, se o período de tempo for suficientemente curto (da ordem dos milissegundos) cria nos utilizadores a ilusão de que as tarefas são executadas simultaneamente [9]. Os sistemas de *time-sharing* tiveram sucesso na década de 70, permitindo que múltiplos utilizadores interagissem concorrentemente com a mesma máquina. Aos utilizadores que doutro modo não teriam acesso a tamanha capacidade de computação, era fornecido um ambiente que lhes permitia

correr os seus programas habituais sendo que em troca deveriam pagar um valor fixo pelo aluguer do equipamento ao que acrescia um valor pelo número de horas que estavam ligados, pelo tempo de *CPU* gasto (segundos) e pelo espaço ocupado em disco (*KB*/mês) [10].

Hoje em dia a computação está a mover-se novamente para fora das instalações das empresas, a principal diferença entre este movimento e o *time-sharing* reside no facto da arquitetura centralizada ser substituída por uma arquitetura distribuída em que os clientes se ligam a vários servidores ao mesmo tempo, alguns deles partilhando também informação entre si [11].

2.3. Sistemas Distribuídos

Podemos definir um sistema distribuído como "um conjunto de computadores independentes que se apresenta ao utilizador como um sistema único e consistente" [12]. O principal benefício deste tipo de sistema é o de ligar recursos computacionais de uma forma transparente, eficiente, fiável e escalável através de uma rede de dados (entenda-se por recursos computacionais quer recursos físicos, *hardware*, quer recursos virtuais, *software* e dados).

Existem várias representações das camadas que compõe um sistema distribuído sendo que, a apresentada na Ilustração 1 é uma versão simplificada do modelo definido pela *International Telecommunications Union* (*ITU-T*) no documento *Distributed Computing: Utilities, Grids & Clouds* [13]. O modelo apresentado é usualmente utilizado na *Web*.

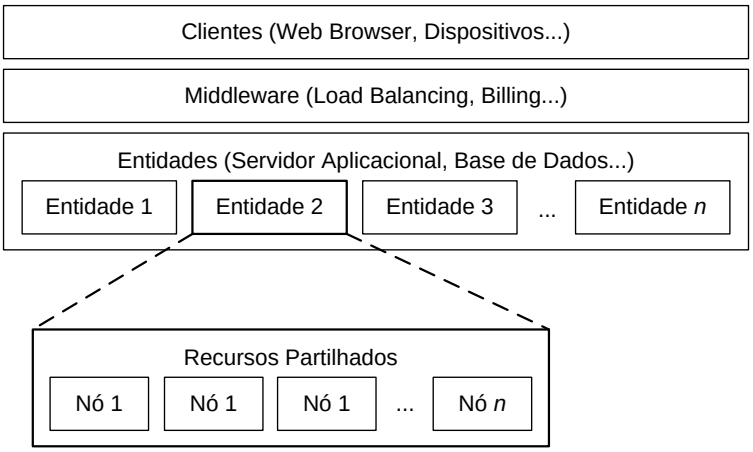


Ilustração 1 - Camadas que compõem um Sistema Distribuído

A primeira camada é a responsável pela interação com os utilizadores.

A camada de *middleware* é responsável pela alocação e admissão de recursos, pelo balanceamento de carga, pela monitorização de todas as atividades do sistema e por recolher estatísticas que mais tarde podem ser usadas para *billing* [13]. Esta camada tem por isso que ser robusta e estar sempre disponível.

A terceira camada é composta pelas entidades. Uma entidade não é mais do que um conjunto de nós que podem ou não estar geograficamente dispersos e que são apresentados ao utilizador como um sistema único e consistente.

As camadas são válidas para as diferentes formas que os sistemas distribuídos podem assumir tais como o *grid computing* e o *cloud computing*. Na verdade, existem tantos pontos em comum entre os modelos de computação distribuída que, há até quem defenda que é difícil delinear fronteiras entre eles [13].

2.4. Virtualização

A virtualização é a criação de uma versão virtual de algo. Uma aplicação, um sistema operativo ou mesmo um servidor são recursos passíveis de serem virtualizados [14].

Porventura, a forma mais comum de virtualização é o particionamento de discos em que o mesmo disco rígido físico é separado em unidades lógicas mais pequenas. Estas unidades são apresentadas ao utilizador como se se tratassem de recursos reais e distintos.

O interesse pela virtualização vai ao encontro da recente tendência pelos sistemas com administração autónoma e pela computação como um serviço [14]. A virtualização permitiu "mascarar" recursos como servidores, processadores, discos rígidos, etc. dos utilizadores, melhorando o *workload*, robustez e escalabilidade dos sistemas assim como facilitar a administração dos parques informáticos e baixar os custos de *TI*.

2.5. Web Services

Os *Web Services* são uma tecnologia que permite disponibilizar funcionalidades de uma aplicação na *Internet*. Ao contrário do que é usual não é disponibilizada uma *interface* gráfica (*GUI*) para a interação com o utilizador, sendo que em vez disso é disponibilizada

uma interface que expõe a lógica, dados e processos de negócio a outras aplicações, o chamado *machine-to-machine* [15]. O funcionamento dos *Web Services* é bastante simples:

- O fornecedor do serviço aplicacional define o formato para os pedidos que recebe e para as respostas geradas;
- Um processo remoto, em execução num outro computador faz um pedido para a aplicação através da *Internet*;
- A aplicação envia a resposta à tarefa solicitada.

As funcionalidades expostas podem ser tarefas complexas ou pequenos utilitários como gravar uma reunião num calendário, efetuar o câmbio entre duas moedas ou validar o número de um cartão de crédito.

Garantir a interoperabilidade é o principal objetivo dos *Web Services* por isso, todos os serviços são disponibilizadas usando protocolos abertos, quer para a chamada, quer para a transferência de dados. A linguagem de programação usada no cliente não é relevante assim como a plataforma, desde que seja respeitada a interface com o *Web Service* e que se usem tipos primitivos passíveis de ser entendidos pelo serviço consumidor. A ponte entre o cliente e o serviço é assegurada pelo protocolo mais usado na *Internet*, o *Hypertext Transfer Protocol* (*HTTP*, RFC#2616) o que garante uma quase universalidade no acesso dado que o protocolo *HTTP* raramente está bloqueado em *firewalls*. Para a representação dos dados é usada a linguagem *XML* (*Extensible Markup Language*), formato que permite representar estruturas complexas de dados de forma hierárquica e computacionalmente legível [16].

2.6. Síntese

Neste capítulo foram abordadas as arquiteturas e as tecnologias mais importantes que tornaram o *cloud computing* possível. Nenhum destes conceitos é novo, contudo a nuvem consegue sê-lo pois integra todos os modelos acima descritos [17].

3. A Nuvem

3.1. Introdução

O termo *cloud computing* refere-se quer às aplicações que são disponibilizadas como um serviço, quer ao *hardware* e *software* que suporta estes serviços em *datacenters* acessíveis através da internet. Às aplicações disponibilizadas dá-se o nome de *Software as a Service* enquanto que, ao *datacenter* acessível através da *Internet* dá-se o nome de nuvem [18]. Se a nuvem for pública e o acesso for pago mediante os recursos utilizados, tal qual a rede elétrica, chamamos ao serviço disponibilizado *Infrastructure as a Service*.

Seguidamente será mostrada uma maneira de classificar os diferentes serviços da nuvem. Irão ser destacadas as principais vantagens da nuvem assim como os principais entraves à sua adoção. Por fim, será feito um comparativo às soluções comerciais existentes e serão apresentados os componentes comuns a todas as ofertas.

De uma forma geral podemos definir *cloud computing* como uma infraestrutura altamente escalável desenhada para hospedar aplicações e paga mediante a utilização [19]. A infraestrutura deve ser abstraída e o pedido de recursos por parte do utilizador deve ser feito de forma independente através de uma *Application Programming Interfaces* (*API*).

3.2. Classificação de Serviços na Nuvem

Os serviços da nuvem podem ser categorizados mediante os utilizadores a que estão destinados tal como é apresentado na Tabela 1.

Nível	**Categoria**	**Sigla**
Utilizador	Software as a Service	SaaS
Programador	Platform as a Service	PaaS
IT	Infrastructure as a Service	IaaS

Tabela 1 - Diferentes categorias de cloud computing

3.2.1. SaaS

Software as a Service (*SaaS*) refere-se às aplicações que são hospedadas na nuvem e que são acedidas pelos utilizadores através da *Internet* [17]. O serviço disponibilizado é uma aplicação *Web* com as mesmas funcionalidades de uma aplicação local.

Um exemplo de *SaaS* é o Google Docs, um conjunto de aplicações alojadas nos servidores da Google que disponibiliza uma experiência semelhante ao Microsoft Office [17]. Todas as operações do Google Docs são efetuadas na *Web*, sendo os documentos guardados na nuvem. Este tipo de aplicações garante aos utilizadores a mesma experiência de utilização independentemente do dispositivo de acesso e garante ainda, que a aplicação e dados estão disponíveis onde quer que o utilizador esteja.

3.2.2. PaaS

Platform as a Service (*PaaS*) refere-se à possibilidade dos programadores projetarem, desenvolverem e testarem aplicações para a nuvem [17]. As aplicações são depois disponibilizadas ao utilizador sobre a infraestrutura do fornecedor de serviço [17].

Dois exemplos de *PaaS* são o Windows Azure e o Google App Engine, plataformas que permitem aos programadores desenvolver e hospedar aplicações nos *datacenters* da Microsoft e da Google, respetivamente. Estas plataformas permitem que os programadores se centrem nas soluções que pretendem desenvolver e não na infraestrutura. Para isso o fornecedor de serviço fica responsável por grande parte das tarefas de manutenção e administração ao mesmo tempo que promete uma plataforma que escala automaticamente e que é tolerante a falhas.

3.2.3. IaaS

Infrastructure as a Service (*IaaS*) refere-se à possibilidade das equipas de *IT* obterem processamento, armazenamento ou base de dados (*BD*) através na internet, pagando mediante a quantidade de recursos utilizados.

Um exemplo de *IaaS* é o serviço Amazon *Elastic Compute Cloud* ou simplesmente Amazon *EC2* [20]. O Amazon *EC2* disponibiliza ao administrador de sistemas a possibilidade de criar uma máquina virtual com um *Sistema Operativo* (*SO*) com o mínimo esforço, através de uma simples chamada a um *Web Service*. A máquina pode ser completamente customizada e fica acessível tal como se se tratasse de um servidor dedicado hospedado na infraestrutura da empresa. O administrador fica deste modo, livre de grande parte das tarefas de manutenção.

3.3. Principais Características de uma Solução na Nuvem

De seguida serão descritas as principais características da nuvem que a distinguem de um infraestrutura convencional.

3.3.1. Ilusão de disponibilizar recursos infinitos

A ilusão de disponibilizar recursos infinitos é uma das características da nuvem e advém do facto de todos os recursos serem virtualizados. A maneira como eles são multiplexados e partilhados é escondida do programador [18] o que lhe retira perceção do limite ou das dimensões reais da nuvem.

3.3.2. Eliminação do compromisso inicial

Nem todas as empresas têm ao seu dispor um ou mais servidores livres para iniciar um novo projeto. Na maioria dos casos os recursos têm que ser adquiridos e por isso passam por um longo processo de aprovação. Mesmo que tenham sido aprovados, os recursos têm que ser instalados e configurados até serem postos à disposição das equipas de desenvolvimento. Todas estas etapas consomem recursos e tempo à empresa, esforço este que deve ser canalizado para a aplicação e não para a plataforma de suporte. Como a nuvem responde a aumentos de carga o utilizador da nuvem não se sente obrigado a fazer um provisionamento a longo prazo, podendo começar com poucos recursos e aumentando-os à medida das suas necessidades, eliminando-se assim o compromisso inicial [21].

Esta característica é particularmente importante quando não se sabe de antemão qual o sucesso que determinada aplicação vai ter (ver Instagram no ponto 3.3.5) e por isso não há uma previsão dos recursos que se devem comprar.

3.3.3. Pagar mediante a utilização

O *cloud computing* permite que os recursos sejam pagos mediante a sua utilização. Por exemplo, se o recurso for processamento o utilizador paga somente as horas em que a instância esteve alocada. A distribuição pode ser não linear, o utilizador paga 48 horas quer sejam utilizadas 48 horas de *CPU* num dia (correspondendo a dois *CPU*s durante 24 horas) e 0 no dia seguinte ou 24 horas em cada um dos dias [21].

3.3.4. Associativismo de custo

Associativismo de custo refere-se ao facto de 1000 máquinas a correrem por uma hora, custarem o mesmo do que uma máquina a correr por 1000 horas [21]. Esta característica faz com que a comunidade cientifica se interesse pela nuvem e a veja como uma alternativa para o processamento em lote [21]. Prova disto é o amplo uso do sistema de *HTC* (*High-throughput computing*) Condor[1] na nuvem [22] e o aparecimento de soluções de *HPC (High-performance computing)* baseadas no paradigma do *cloud computing* tais como o CycleCloud[2].

3.3.5. Elasticidade

Elasticidade refere-se à capacidade dos recursos se dimensionarem automaticamente mediante a carga. Um serviço elástico é aquele em que os recursos disponibilizados correspondem aos recursos exigidos ao longo do tempo, um serviço não elástico pode por sua vez, apresentar um de três comportamentos possíveis [18].

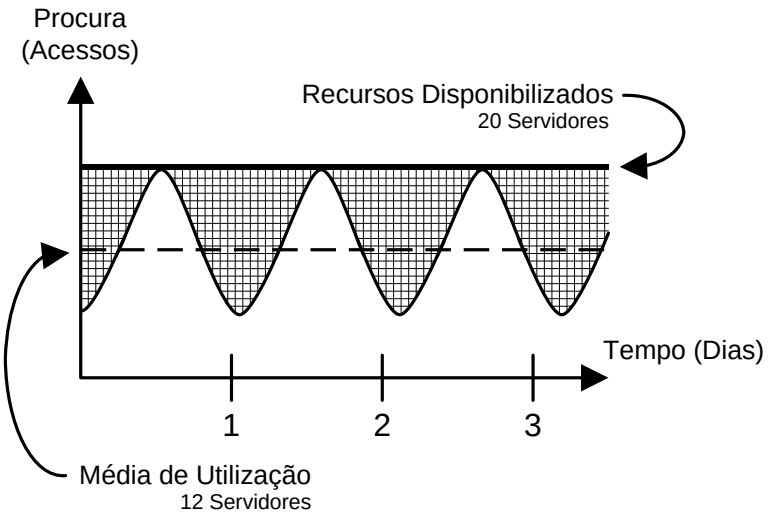


Ilustração 2 - Serviço não elástico, cenário I [18]

As Ilustrações 2-5 mostram a procura de um serviço ao longo do tempo. O serviço apresenta o seu pico de utilização à tarde, sendo que a linha horizontal corresponde aos recursos disponibilizados/provisionados, a linha preta corresponde aos recursos exigidos no momento e a área quadriculada corresponde ao excesso entre os recursos disponibilizados e os recursos exigidos.

[1] http://research.cs.wisc.edu/condor/, acedido em 4 de Setembro de 2012.
[2] http://www.cyclecomputing.com/cyclecloud/overview, acedido em 4 de Setembro de 2012.

Na Ilustração 2 temos um serviço que requer em média 12 servidores e que apresenta um pico de tráfego previsível durante a tarde. Para responder a este pico são necessários 20 servidores por isso o sistema é dimensionado para 20 servidores * 24 horas ou seja, 480 horas de computação por dia. O provisionamento é adequado para suportar os períodos de maior procura pelo serviço no entanto no restante tempo existe um subaproveitamento de recursos, pois das 480 horas provisionadas, apenas 288 são necessárias tendo em conta a média de utilização do serviço ou seja, o pico excede a média em cerca de 70%.

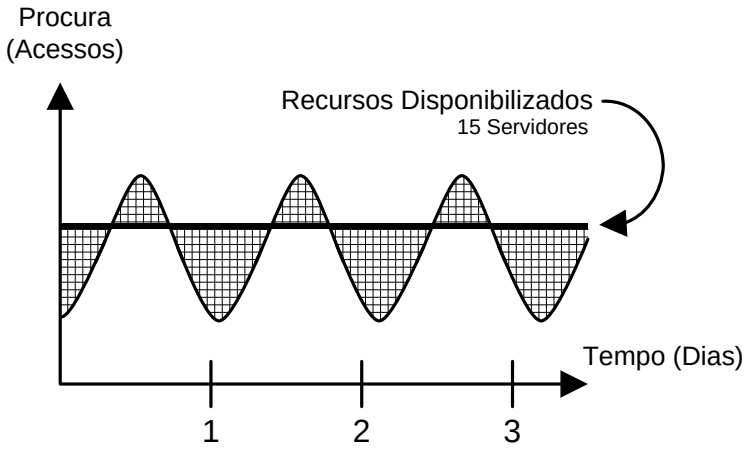


Ilustração 3 - Serviço não elástico, cenário II [18]

Esta situação agrava-se quando o dimensionamento é feito por excesso mas o que será que acontece se, pelo contrário, houver menos recursos do que é exigido (Ilustração 3)? Quando o pico de tráfego é mal calculado ou simplesmente imprevisível, os utilizadores são confrontados com problemas técnicos que inviabilizam ou afetam a sua experiência de utilização. O mau dimensionamento pode conduzir os utilizadores a abandonarem o serviço, fazendo com que a empresa perca receitas e clientes sem sequer se aperceber [18]. Embora impossíveis de contabilizar, as perdas geradas pelo subdimensionamento dos recursos são potencialmente mais graves que as geradas pelo subaproveitamento dos mesmos [18].

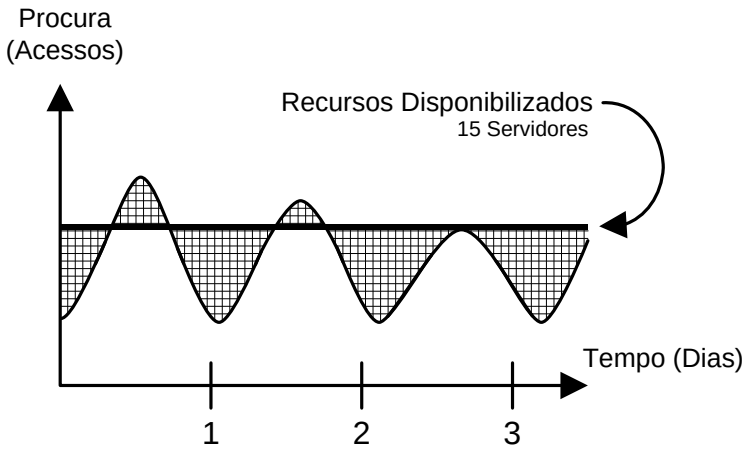


Ilustração 4 - Serviço não elástico, cenário III [18]

Um cenário curioso pode ocorrer fruto da redução gradual da procura por parte dos utilizadores, os recursos outrora escassos podem passar a ser suficientes sem que tenha havido qualquer intervenção no serviço como se pode observar pela Ilustração 4. Contudo isto corresponde à perda de negócio, sendo pois indesejável.

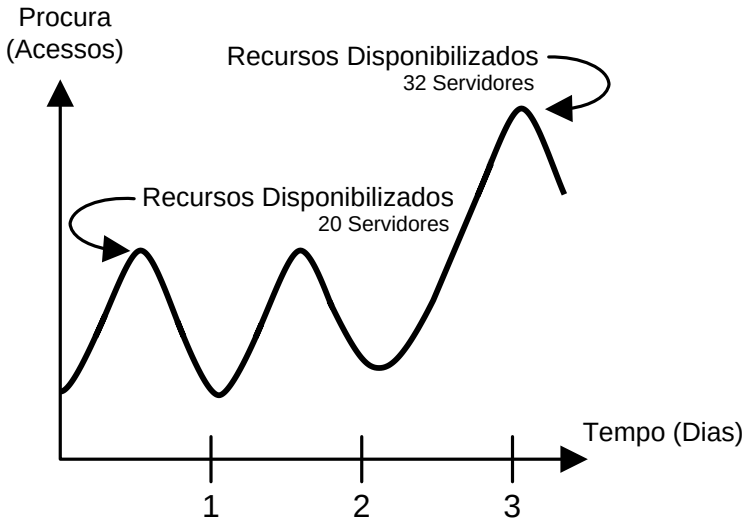


Ilustração 5 - Serviço elástico

Os serviços elásticos não só respondem a todos os cenários acima referidos, como também são a solução para serviços com picos de tráfego sazonal ou pontual e para serviços para os quais é impossível prever a procura [18]. O fator diferenciador de um serviço elástico é o facto de ser flexível e se adaptar à carga exigida no momento (Ilustração 5). Quando existe mais procura são pedidos mais recursos à nuvem, e quando deixam de ser necessários são

libertados [17]. Este novo modelo faz com que a instituição só pague pelo que é de facto consumido não havendo sub alocação nem sobre alocação de recursos.

Para algumas empresas nomeadamente *startups* a elasticidade é mesmo um requisito operacional [18], tomemos como exemplo o Instagram[3] ou o Animoto[4]. Ambos são serviços com uma forte componente social e ambos tiveram uma tremenda aceitação e exposição.

É difícil prever o sucesso que um determinado serviço possa vir a ter e foi isso que aconteceu com o Instagram, uma aplicação de partilha de imagens que foi lançada na App Store da Apple dia 6 de Outubro de 2010. O Instagram registou 20000 utilizadores na quarta-feira dia do seu lançamento, um número que não estava na cabeça do cofundador Mike Krieger que disse num evento para membros e funcionários da rede Airbnb[5]: "De algum modo conseguimos manter o serviço por 3 dias mas era claro que o serviço não se ia aguentar no fim de semana" [23]. A migração foi feita para o Amazon *EC2* e não só o serviço se manteve no fim de semana como também chegou à marca dos 100 mil utilizadores numa semana e um milhão de utilizadores dois meses depois[6].

Os criadores do Animoto, uma aplicação para a criação de vídeos a partir de conteúdos como imagens e sons, também optaram pelo *EC2*. Em Abril de 2008 o Animoto passou por um pico de procura que coincidiu com a libertação da sua aplicação para o Facebook passando de 50 servidores de *rendering* para 3500 em apenas 3 dias [24]. Estes crescimento num tão curto espaço de tempo teria sido impraticável caso a *startup* não tivesse usado a nuvem [24].

3.3.6. Rapidez de aprovisionamento

Possibilidade de aprovisionar servidores, armazenamento ou aplicações em minutos em vez de semanas [25]. A rapidez de aprovisionamento está ainda aliada à facilidade de acesso aos recursos, que são pedidos através da chamada a uma *API* de gestão ou através de um portal *Web*.

[3] http://instagram.com/, acedido em 4 de Setembro de 2012.
[4] http://animoto.com/, acedido em 4 de Setembro de 2012.
[5] https://www.airbnb.com/, acedido em 8 de Setembro de 2012.
[6] http://money.cnn.com/2012/04/09/technology/facebook_acquires_instagram/, em Abril o Instagram foi adquirido pela rede social Facebook pelo valor de 1 bilião de euros.

3.4. Obstáculos à adoção

Seguidamente serão apresentados um conjunto de obstáculos que impedem algumas empresas de tomar em consideração a nuvem.

3.4.1. Uso de Tecnologias Proprietárias

As *API*s dos fornecedores são proprietárias o que faz com que as empresas tenham medo de ficar "presas" a determinada solução. O custo de migrar um serviço de um fornecedor para outro é muito grande o que faz com que as empresas por um lado, receiem que a continuidade do seu negócio esteja dependente de quem disponibiliza a nuvem e por outro, receiem ficar vulneráveis a aumentos de preço fruto de uma posição dominante de quem controla a nuvem [21]. Uma solução para este problema seria normalizar as *API*s de modo a que as empresas pudessem distribuir os seus dados e aplicações por várias nuvens, baixando a dependência do cliente e aumentando ao mesmo tempo a disponibilidade e a robustez do serviço [21].

O uso de *API*s normalizadas permite que as infraestruturas privadas falem a mesma linguagem que a nuvem, dando origem a modelos híbridos como o projeto de código aberto Eucalyptus [26]. O Eucalyptus permite transformar o infraestrutura de determinada instituição num *IaaS* que, ao ser compatível com a *API* da Amazon possibilita a migração de projetos/dados da nuvem privada para a nuvem pública e vice-versa [26]. A normalização permite ainda que todas as ferramentas e aplicações conformes com a *AWS* (*Amazon Web Services*) *API* sejam utilizadas na nuvem privada ou seja, o utilizador pode fazer a mesma chamada a um *Web Service* quer queira adicionar uma máquina virtual na infraestrutura local ou no *datacenter* da *Amazon*. Todas estas funcionalidades são importantes, no entanto, a principal razão pelo qual o projeto Eucalyptus foi criado [27] é o facto de permitir que os recursos locais sejam utilizados em conjunto com os recursos de terceiros [26]. À junção da nuvem privada com a nuvem pública dá-se o nome de nuvem híbrida.

3.4.2. Garantia de Confidencialidade e Segurança

Confidencialidade e segurança são possivelmente os fatores mais importantes quando uma empresa pondera acerca da viabilidade do recurso ao *cloud computing* [21].

Um utilizador da nuvem está exposto quer a ameaças internas quer às mesmas ameaças externas que afetam grandes datacenters [21]. O utilizador da nuvem tem a responsabilidade de proteger a sua solução ao nível aplicacional, o fornecedor de serviço por seu lado é responsável pela segurança física dos equipamentos e pelas *firewalls* de fronteira com a nuvem [21]. A responsabilidade sobre as restantes camadas é partilhada entre o utilizador e o fornecedor sendo que, quanto menor for o nível de abstração do serviço maior é a responsabilidade que recai sobre o utilizador. Um utilizador do Amazon *EC2* é mais responsável que um utilizador do Windows Azure sendo que este, é por sua vez mais responsável que um cliente do Google AppEngine [21]. Por exemplo, um utilizador do Amazon *EC2* deve ele próprio ter os sistemas operativos funcionais nas máquinas virtuais o que não acontece na oferta da Microsoft ou da Google.

O facto da nuvem poder ser pública cria desconfiança acerca da segurança interna da mesma. Os mesmos recursos podem ser partilhados por várias empresas, empresas essas que no extremo podem ser concorrentes. Como garantir, assim, que os dados não são expostos quando os serviços estão alojados no mesmo servidor físico e usam os mesmos canais de comunicação? Tecnologias como cifração, *Virtual Local Area Networks* (*VLAN*s), *firewalls* e a própria virtualização dão resposta a boa parte destas questões mas deixam em aberto o desconforto sentido pelas empresas ao saberem que o seu serviço está alojado algures no mundo.

Fornecedores de serviço como a Amazon e a Microsoft possibilitam que o utilizador escolha a sub-região que pretende, sendo que o utilizador pode selecionar, por exemplo, entre Europa Ocidental e Europa Setentrional. Contudo, por razões de segurança, não lhe é facultada a localização exata do serviço [28]. Como a lei que vigora é a do país onde os dados estão alojados [29] se o país for desconhecido a lei é igualmente desconhecida. O desconhecimento do país onde os dados estão guardados não é contudo, a única ameaça relativamente à sua confidencialidade e segurança. Que lei prevalece se uma instituição processar os dados no Reino Unido, os guardar na Irlanda e os enviar para França onde tem uma subsidiária [29]? Neste caso não é claro que lei específica prevalece numa disputa legal [29], sendo que esta incerteza cria desconfiança e faz com que a adoção ao *cloud computing* por parte das instituições seja mais lenta [29].

Outro facto que torna as instituições reticentes ao uso de uma solução de *cloud computing* é a possibilidade dos dados serem divulgados às autoridades dos Estados Unidos da

América (*EUA*) através do Patriot Act [30]. Ao abrigo do Patriot Act, as autoridades dos Estados Unidos podem exigir a divulgação de dados sem ser necessário consentimento prévio por parte do proprietário evocando para o efeito luta antiterrorismo. Estão abrangidos por esta lei, não só dados alojados nos Estados Unidos, mas também dados que, embora estejam fora dos *EUA* sejam detidos por uma empresa do país. São ainda abrangidos utilizadores que recorram a qualquer tipo de solução Americana para o armazenamento e processamento de dados [31]. Este facto, entra em conflito com as politicas de privacidade da União Europeia [29] que proíbem o acesso aos dados sem o que o utilizador dê o seu consentimento. Na prática, são as diretivas dos Estados Unidos que prevaleceriam se existisse uma disputa legal referente à proteção de dados por isso, alguns utilizadores evitam serviços como o Amazon *EC2*, o Google App Engine ou o Windows Azure sob pena de estarem abrangidos por esta lei [32]. À data de escrita, a União Europeia está a finalizar a sua estratégia para a computação na nuvem na qual serão endereçadas questões relativas à proteção de dados e à propriedade intelectual. Segundo palavras da própria União, estas questões são o principal motivo para o atraso da Europa na área do *cloud computing* situação que pretende alterar com a definição de regras que defendam os utilizadores [33].

3.4.3. Congestionamentos/Limitações na Transferência de Dados

A velocidade associada à transferência de dados para a nuvem pode constituir um problema para aplicações que fazem uso intensivo de dados. Considere-se um cenário hipotético que demonstra este problema: um laboratório gera 1 *Terabyte* (*TB*) de dados em bruto por mês. Estes dados necessitam de ser transformados em informação útil, sendo por isso enviados para a nuvem da Amazon a fim de serem processados. Segundo a Amazon [34] a transferência de dados demoraria 13 dias se o laboratório tivesse uma ligação à internet a 10Mbps e se a largura de banda disponível para a operação nunca estivesse abaixo dos 80%.

O problema do tempo excessivo na transferência de dados foi mitigado pelos fornecedores de serviço através da criação de serviços que possibilitem a expedição física de discos rígidos via transportadora. Tendo em conta que a velocidade de leitura e escrita de um disco rígido é aproximadamente 100*MB*/s, demoraria aproximadamente 3 horas para que o disco fosse escrito, 3 horas para que fosse lido e 3 horas para que fosse processado o que

totaliza 9 horas [34]. Em termos monetários um disco de 2*TB* custa aproximadamente 120 *United States Dollars* (*USD*), sendo que a Amazon cobra um valor fixo de 80 *USD* por processar o disco e 2,49 *USD* por cada hora a carregar os dados, neste caso 6*2,49 ou seja 14,94 *USD*, o que perfaz o total de aproximadamente 215 *USD*.

Aos custos estimados em cima falta ainda acrescentar o valor do transporte do disco para um dos *datacenters* da Amazon, bem como o tempo de transporte. Se tomarmos como exemplo o envio do mesmo disco de 2*TB* de Portugal para a Amazon da Irlanda é expectável que a entrega demore entre 2 a 3 dias[7] e que o transporte custe cerca de 30 *USD*[8] o dobro caso seja ida e volta. À data de escrita, o Windows Azure não tem nenhuma oferta semelhante.

3.5. Síntese

Neste capítulo foi apresentado o conceito de nuvem e os vários níveis de serviço associados ao conceito de nuvem. Foram ainda apresentadas as principais características da nuvem que a tornam uma das prioridades das empresas ao nível tecnológico [1]. O capítulo termina com uma análise aos principais obstáculos associados à nuvem e que são obstáculos à adoção das instituições aos serviços na nuvem.

[7] http://www.chronopost.pt, acedido em 17 de Julho de 2012.
[8] http://awsimportexport.s3.amazonaws.com/aws-import-export-calculator.html, acedido em 17 de Julho de 2012.

4. Soluções Comerciais

4.1. Introdução

Ao longo dos anos, empresas como a Amazon foram ganhando perícia a administrar grandes *datacenters* [17]. Sendo a principal loja de comércio eletrónico a nível mundial, o *datacenter* é a espinha dorsal do seu negócio, por isso a Amazon escalou-o tendo em conta o cenário mais exigente ou seja, todo o *datacenter* é dimensionado para picos de tráfego. É importante para a Amazon ter recursos suficientes para "aguentar" um pico de tráfego anormal caso contrário arriscar-se-ia a perder negócio e clientes. No entanto, isto faz com que a utilização dos servidores esteja muito aquém dos limites em períodos normais. Este cenário é encontrado na Amazon como ainda na maioria dos *datacenters*. Estima-se que a média de utilização de um servidor esteja entre os 5 e os 20% sendo que em período de pico a carga exigida é 2 a 10 vezes superior à média de utilização [21].

A Amazon apercebeu-se disto e, aproveitando o conhecimento que já tinha, fez crescer a nuvem e criou um novo modelo de negócio. Em Outubro de 2007 a Amazon abriu as portas aos seus *datacenters* e foi pioneira ao vender o acesso à sua infraestrutura como um serviço [17].

4.2. Solução Típica

Várias soluções comerciais surgiram após o lançamento do *AWS*. No entanto todas elas partilham um conjunto de funcionalidades que são consideradas "padrão". Este conjunto de funcionalidades vai ao encontro do que é pedido por qualquer aplicação: um modelo de computação, um modelo de armazenamento e um modelo de comunicação [21]. Os modelos podem ser utilizados em conjunto ou de forma independente.

4.2.1. Modelo de Computação

Independentemente do nível de abstração, qualquer fornecedor disponibiliza uma plataforma que permite hospedar as aplicações criadas pelos utilizadores. A solução disponibilizada é potenciada pelo uso de tecnologias de virtualização, podendo tomar a forma de um servidor aplicacional ou até de uma máquina virtual totalmente customizável. A virtualização permite que uma aplicação seja independente de outras hospedadas na mesma máquina física. Permite ainda que várias aplicações residentes no mesmo servidor

físico não concorram entre si. Outra característica da plataforma é o facto de não manter estado (*stateless*) favorecendo a escalabilidade e a robustez das soluções por si hospedadas. A computação é usualmente cobrada em *CPU*/hora como se pode observar na Tabela 2. A Tabela 2 mostra o custo de um servidor virtualizado também designado por instância no Microsoft Windows Azure (a solução comercial escolhida para o âmbito desta dissertação).

Tamanho da instância	**Número de núcleos de processamento disponibilizados para a instância**	**Velocidade de relógio do CPU**	**Quantidade de memória**	**Custo/Hora em USD**	**Fator relativo a uma instância "Small"**
Extra Small	Partilhado	1.0 GHz	768 MB	0,02	1/6 hora
Small	1	1.6 GHz	1.75 GB	0,12	1 hora
Medium	2	1.6 GHz	3.5 GB	0,24	2 horas
Large	4	1.6 GHz	7 GB	0,48	4 horas
Extra Large	8	1.6 GHz	14 GB	0,96	8 horas

Tabela 2 - Custo/Hora das instâncias no Microsoft Windows Azure [35]

As instâncias encontram-se divididas consoante o tamanho, o que faz com que o valor cobrado seja aferido através da multiplicação do número de horas de computação pelo valor de referência (custo/hora de uma instância *Small*) vezes o fator multiplicativo. Por exemplo o custo de 2 horas de uma instância *Extra Large* é 2 * 0,12 * 8 ou seja 1,92 *USD*.

4.2.2. Modelo de Armazenamento

Para complementar o modelo de computação é disponibilizado uma plataforma de armazenamento *statefull*. Normalmente os dados armazenados localmente numa instância da nuvem não são persistentes, ou seja, os dados são perdidos caso a instância seja reiniciada num novo *hardware*, o que pode acontecer por falha de *hardware* ou por uma simples tarefa de manutenção que implique a reorganização do *datacenter* [36]. A plataforma de armazenamento vem não só preencher esta lacuna ao permitir gravar os dados de forma persistente, mas também possibilitar que os dados sejam acessíveis quer a múltiplas instâncias da aplicação quer a aplicações externas [36].

O armazenamento pode ser feito em base de dados ou em *BLOB*s (*Binary Large Object*) sendo que o serviço é cobrado pela quantidade de dados armazenados, usualmente em *GB*/mês.

Os valores da Tabela 3 mostram o preço de uma base de dados *T-SQL* (*Transact-Structured Query Language*) no Microsoft Windows Azure. Tal como no modelo de computação também existe um valor de referência que neste caso é 9,99 *USD* [35]. A este valor de referência dá-se o nome de *Database Unit* (*DU*).

Tamanho da base de dados			**Custo/Mês em USD**		**Fator relativo a um DU**	
			Custo do tamanho mínimo	**GB seguintes**	**Custo do tamanho mínimo**	**GB seguintes**
0 MB	a	100 MB	4,995	-	0,5 DU	-
100 MB	a	1 GB	9,990	-	1 DU	-
1 GB	a	10 GB	9,990	3,996	1 DU	0,4 DU
10 GB	a	50 GB	45,954	1,998	4,6 DU	0,2 DU
50 GB	a	150 GB	125,874	0,999	12,6 DU	0,1 DU

Tabela 3 - Custo/Mês de um base de dados SQL no Microsoft Windows Azure [35]

A partir de 1 *GB* é cobrado um valor fixo pelo limite inferior de determinado "tamanho", representado na Tabela 3 pelas colunas "Custo do tamanho mínimo". Se o tamanho da base de dados ultrapassar o limite inferior acresce uma fração do valor do *DU* por cada *GB* adicional. Por exemplo, o custo por mês do armazenamento de uma base de dados com 25,8 *GB* é calculado da seguinte forma: 4,6 *DU*s pelos primeiros 10 *GB* + 0,2 *DU*s pelos restantes 16 *GB* o que perfaz o total de 7,8 *DU*s ou 77,922 *USD*.

Apesar do custo ser apresentado ao utilizador como um valor mensal, este valor é calculado diariamente de modo a refletir as possíveis mudanças de tamanho na base de dados ao longo do mês [35]. A Tabela 4 mostra por sua vez o custo mensal dos *BLOB*s no Microsoft Windows Azure.

Capacidade armazenada			**Custo/Mês por GB em USD**
0 GB	a	1 TB	0,125
1 TB	a	50 TB	0,110
50 TB	a	500 TB	0,095
500 TB	a	1000 TB	0,090
1000 TB	a	5000 TB	0,080

Tabela 4 - Custo/Mês do armazenamento no Microsoft Windows Azure, BLOBs, Tabelas, Queues e Discos Virtuais [35]

Ao contrário da base de dados, o custo por *GB* dos *BLOB*s não tem uma tarifa fixa mediante o tamanho. O custo é calculado através do somatório de várias tarifas diferentes que se vão alterando à medida que a capacidade requisitada aumenta. Por exemplo, o custo por mês de 3300 *GB* de armazenamento é calculado somando 1000 *GB* * 0,125 *USD* a

2300 *GB* * 0,110 *USD* o que totaliza 378 *USD*. Há ainda que ter em conta o custo de 0,01 *USD* por cada 100000 transações (operações de leitura/escrita) efetuadas aos dados armazenados [35].

4.2.3. Modelo de Comunicação

Uma solução típica é composta ainda por um modelo de comunicação. Para o efeito é disponibilizada uma fila de mensagens vulgarmente designada por *queue*. Uma *queue* pode ter dois propósitos principais:

- Permitir a troca de informação entre componentes de uma solução distribuída, garantido a entrega mesmo que um desses componentes fique temporariamente indisponível [37];
- Armazenar tarefas que esperam na fila pela sua vez até serem processadas.

As *queue*s são apropriadas quando se lidam com tarefas pequenas e isoladas como é o caso do registo de votos num sistema de votações. Uma possível aplicação neste cenário é a introdução dos votos numa *queue* que somente é processada quando chega aos 100 registos, deste modo os votos só seriam registados de 100 em 100 o que permitiria que os acessos à base de dados fossem reduzidos num fator de 1/100.

As *queues* promovem a resiliência e a elasticidade da aplicação. O facto de guardarem as mensagens em fila até que um pedido possa ser atendido faz com que por um lado, a solução resista a falhas num dos componentes envolvidos e por outro, aguente picos de procura onde o débito dos pedidos é muito superior ao débito com que as mensagens conseguem ser processadas [37]. Esta detalhe intrínseco ao funcionamento da *queue* faz com que as mensagens sejam corretamente balanceadas [37]. No modelo *Pull* se dois servidores desempenharem a mesma tarefa mas o segundo tiver uma capacidade de débito três vezes superior ao primeiro, é expectável que lhe seja entregue o triplo das mensagens, uma vez que cabe aos servidores pedirem novas mensagens à *queue* assim que tiverem acabado de processar.

É cobrado pela ocupação da *queue* no Microsoft Windows Azure o mesmo valor mensal aplicável aos *BLOB*s. Os custos inerentes à transferência dos dados são imputados ao utilizador caso os dados saiam do *datacenter* em direção à *Internet* ou em direção a um *datacenter* numa sub-região diferente. Se a *queue* ligar dois componentes da mesma sub-

região este valor não se aplica caso contrário os valores a aplicar são os apresentados na Tabela 5 [35].

Dados transferidos			**Custo/Mês por GB em USD**	
			Zona 1 Europa Setentrional Europa Ocidental Estados Unidos da América	**Zona 2** Este Asiático Sudoeste Asiático
0 GB	a	10 TB	0,12	0,19
10 TB	a	50 TB	0,09	0,15
50 TB	a	150 TB	0,07	0,13
150 TB	a	500 TB	0,05	0,12

Tabela 5 - Custo/Mês da transferência de dados no Microsoft Windows Azure [35]

Existem duas zonas com preços distintos, a zona 1 engloba as sub-regiões da Europa e Estados Unidos da América, enquanto a zona 2 engloba as sub-regiões da Ásia [35].

Para além das *queues* é disponibilizada uma *cache* de memória que permite reduzir a latência no acessos a dados [37]. Ao introduzir uma *cache* entre a aplicação e um servidor de base de dados é possível reduzir a carga do servidor pois, todas as consultas que já se encontrem em memória não são efetuadas na base de dados [37]. O tempo de acesso será inferior no caso da informação já se encontrar disponível na *cache* (*cache hit*) uma vez que o acesso a uma memória *Random Access Memory* (*RAM*) que se encontra próxima da aplicação é muito inferior ao tempo de uma consulta numa base de dados armazenada em disco. Supondo que na página principal do sistema de votações (apresentado anteriormente) são mostrados os resultados de 10 painéis de votação e que a página é acedida por 10 utilizadores antes dos dados serem atualizados na base de dados. Numa aplicação tradicional teriam sido feitas 100 (10x10) operações na base de dados, com a *cache* os resultados das *queries* podem estar todos pré-carregados na memória do sistema ou seja, cada um dos resultados dos 10 painéis de votação resultaria num *cache hit*. Esta arquitetura permite que 100 acessos à base de dados possam eventualmente passar a 0 desde que, não seja feita nenhuma operação que torne os dados em memória inconsistentes com os da base de dados e que faça com que expirem. Se houvesse uma atualização no intervalo bastaria a um dos clientes ir à base de dados para que o resultado fosse atualizado em memória.

O preço mensal da *cache* é fixo e só está dependente do tamanho escolhido aquando da sua criação (Tabela 6).

Tamanho da cache	Custo/Mês em USD
128 MB	45,00
256 MB	55,00
512 MB	75,00
1 GB	110,00
2 GB	180,00
4 GB	325,00

Tabela 6 - Custo/Mês da cache no Microsoft Windows Azure [35]

Por exemplo, uma *cache* de 128 *MB* tem o custo mensal de 45 *USD* enquanto uma *cache* de 256 *MB* tem o custo de 55 *USD* (Tabela 6).

4.3. Principais Fornecedores de Serviço

4.3.1. Amazon

Em 2007 a Amazon foi a primeira a apresentar uma solução comercial e a criar furor em torno da nuvem [17]. Aproveitando a experiência na gestão da maior loja online do mundo, a Amazon fez crescer os seus *datacenters* e criou serviços que permitiram aproveitar todo o poder computacional disponível [17]. Os serviços criadas foram agrupadas sob a designação de *Amazon Web Services* numa clara alusão à forma como eram acedidos.

4.3.1.1. Principais Serviços

O serviço mais conhecido do *AWS* é o Amazon *Elastic Compute Cloud* ou simplesmente *EC2*. Este serviço permite que o utilizador tenha acesso a uma máquina virtual numa questão de minutos, máquina esta que lhe oferece quase o mesmo nível de controlo que teria num servidor físico. O *EC2* insere-se portanto na categoria de *IaaS*, providenciando uma máquina virtual altamente customizável isolada da estrutura física através de virtualização de *hardware* (XEN *hypervisor*) [18]. Este nível de customização é ao mesmo tempo uma vantagem e uma desvantagem: por um lado o utilizador é livre de escolher toda a plataforma desde o sistema operativo, as bibliotecas de software e as aplicações, por outro torna-se mais difícil administrar funcionalidades relacionadas com redundância e escalabilidade porque as tarefas de replicação e manutenção de estado são dependentes da aplicação em si, não sendo colocadas quaisquer restrições ao desenvolvimento por parte da Amazon [21]. Em algumas ofertas de *PaaS* a redundância e a escalabilidade das aplicações são completamente automatizadas [21] mas são impostas um conjunto de restrições que

garantem que as aplicações são conformes com o ambiente controlado definido pelo fornecedor de serviço.

O utilizador é também livre de escolher o tipo de máquina pretendida consoante as necessidades da sua aplicação, pode necessitar de mais *CPU*, fazer uso intensivo da memória, ou necessitar de muito espaço em disco. Enquanto as duas últimas características são fáceis de quantificar, o processamento é muito dependente da arquitetura do *CPU* e não só da sua velocidade de relógio. Por isso a Amazon criou uma unidade para normalizar e simplificar a comparação entre duas instâncias denominada "*EC2 compute unit*"[9]. O tipo de máquina, ou instância, define o preço hora sendo que o valor pago é o resultado da multiplicação deste valor pelo número de horas que a instância esteve a correr [20] (Anexo I).

Todas estas opções são feitas através de uma interface *Web* ou através de uma chamada a um *Web Service* numa questão de minutos. Depois de o utilizador ter uma conta no *AWS* basta-lhe escolher a região geográfica onde a instância vai ser criada, o tipo de máquina pretendida, a *Amazon Machine Image* (*AMI*) ou seja a imagem do sistema operativo e aplicações a serem carregadas, e o número de instâncias pretendidas [38]. Pouco depois é-lhe facultado um *ID* único para a instância e um *DNS* (*Domain Name System*) *name* que pode ser prontamente usado para aceder à máquina [38]. Por omissão as máquinas são *stateless* contudo é possível contornar o comportamento ao escolher uma *AMI* com armazenamento persistente denominado de *Elastic Block Store* (*EBS*).

O *EBS* é apresentado como um disco físico e é uma das opções de armazenamento da Amazon. Outras opções são o *Simple Storage Service* (*S3*) e as bases de dados SimpleDB, DynamoDB e *Relational Database Service* (*RDS*) [39].

O *S3* foi desenhado para armazenar objetos com tamanho compreendido entre 1 Byte e 5*TB*, dando a ilusão de recursos infinitos que caracteriza a nuvem. O serviço faz a replicação dos dados por múltiplos *datacenters*, prevenindo erros físicos e erros lógicos. Assim uma operação de *PUT* ou *COPY* só retorna sucesso após o armazenamento em múltiplos sites [40]. Para além das características já referenciadas como a escalabilidade, alta disponibilidade e tolerância a falhas, o *S3* providencia controlo de acesso e controlo de

[9] Uma unidade de processamento equivale a um processador Intel Opteron ou Xeon de 2007 com uma velocidade de relógio compreendida entre os 1.0 e os 1.2 GHz.

versões. Ao utilizador cabe pagar uma determinada quantia mediante a quantidade de dados armazenada (Anexo I).

Uma das limitações apontadas às soluções de armazenamento na nuvem era a falta de base de dados relacionais [41]. De facto, no início os fornecedores disponibilizavam somente bases de dados *noSQL* (*not only SQL*) que se baseiam em tuplos chave-valor [42]. Em vez de se usar a abordagem normalizada, tabelas com linhas e colunas e com relações entre elas, numa base de dados *noSQL* existem somente chaves que apontam para um valor. Deixam de haver chaves primárias e secundárias, *schemas*, tipos de dados ou operações de *JOIN* e *UNION*. Este tipo de base de dados simplifica a estrutura e retira grande parte do custo computacional associado. Estas características garantem uma escalabilidade e um tempo de resposta mais rápido do que a média. Uma base de dados *noSQL* pode ser usada sozinha, quando não é estritamente necessário que a base de dados seja relacional ou que cumpra com as propriedades *ACID* (Atomicidade, Consistência, Isolamento e Durabilidade ISO/IEC#10026-1:1992 secção 4) , ou em complemento a uma base de dados convencional [43]. Em troca à consistência estrita, uma base de dados *noSQL* providencia escalabilidade horizontal, garantindo uma baixa latência e um elevado débito independentemente do número de servidores que suportam a base de dados e do tamanho da mesma [44].

Um dos motores de base de dados disponibilizado pela Amazon é o SimpleDB, uma base de dados *noSQL* acessível através de *Web Services "Web Service as a database"* [45]. O SimpleDB oferece todos os benefícios de uma base de dados na nuvem sem que o utilizador se tenha que preocupar com nenhum detalhe de administração. De raiz a base de dados oferece alta disponibilidade e flexibilidade, sendo que os dados são replicados por múltiplos *datacenters* e a base de dados pode expandir quer em tamanho (até 10*GB*) quer em número de servidores que a suportam, como acontece de resto, com o *S3*. O nível de consistência pode também ser alterada consoante as necessidades do utilizador [45]. Ao invés de consistência "eventual" em que um dado pode ou não ser consistente com a última atualização à base de dados, pode ser forçada uma consistência "estrita" em que qualquer leitura a um objeto está sempre de acordo com a última atualização.

Outra das ofertas da Amazon é o DynamoDB, uma solução semelhante ao SimpleDB mas que leva o conceito de escalabilidade ainda mais à frente. Enquanto que no SimpleDB é previsível que o serviço suporte por volta de 25 escritas por segundo [46], no DynamoDB é o utilizador que escolhe o débito de leitura/escrita e o serviço adapta-se. Para ir ao

encontro do desempenho desejado o serviço divide os dados pelo número de servidores que julgar conveniente. Para além disso, todos os dados são armazenados em *Solid State Disks* (*SSD*) garantindo um desempenho previsível e com uma velocidade de acesso superior aos discos rígidos convencionais. O modelo de pagamento é ligeiramente diferente e baseia-se na capacidade de leitura e escrita reservada por hora [47].

O *Relational Database Service* (*RDS*) é tal como o nome indica a oferta da Amazon para base de dados relacional. Esta opção permite que o utilizador migre mais facilmente aplicações desenhadas *on-premises* para a nuvem, utilizando um motor familiar (MySQL, Oracle ou Microsoft SQL Server) e as ferramentas habituais [48]. O *RDS* encarrega-se das atualizações de *software* bem como das salvaguardas dos dados. É oferecida também a possibilidade do utilizador restaurar a base de dados para um ponto definido no tempo sendo que o período a salvaguardar é configurável [48].

O *AWS* dispõe ainda do serviço Amazon *Simple Queue Service* (*SQS*) que disponibiliza uma fila de mensagens entre componentes de um sistema distribuído [49] e o Amazon ElastiCache [50], uma *cache* de memória 100% compatível com o memcached [51] que permite reduzir o tempo de acesso aos dados, introduzindo uma *cache* de memória entre a aplicação e as bases de dados armazenadas em discos rígidos.

4.3.1.2. Outros Serviços Disponibilizados

Elastic Load Balancer, CloudWatch e Auto Scaling - A elasticidade, um dos principais benefícios da nuvem, só é verdadeiramente alcançado na Amazon quando são usadas um conjunto de serviços (Ilustração 6) como o Elastic Load Balancer o CloudWatch [52] e o Auto Scaling [53].

Embora as soluções sejam desenhadas de raiz para serem elásticas, o aumento ou diminuição de recursos é feita pelo utilizador quer manualmente quer através do desenho de soluções próprias para a monitorização e reação a picos de carga nas instâncias existentes. O CloudWatch é a opção da Amazon para a monitorização dos recursos das instâncias e para a monitorização de métricas customizadas criadas pelo utilizador especificamente para os seus serviços ou aplicações. Quando devidamente configurado o CloudWatch permite que o utilizador recolha métricas, ganhe conhecimento do estado da instância, aplicação ou serviço e reaja se assim o desejar através do Auto Scaling [52].

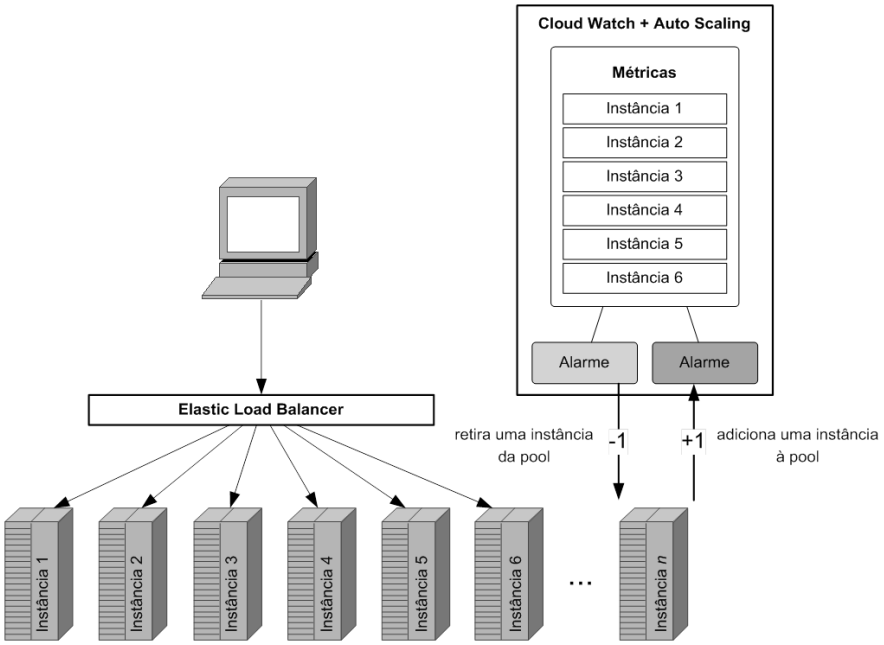


Ilustração 6 - Solução para garantir elasticidade de forma automática no EC2

Aquando de um pico anormal de tráfego num sistema composto por 6 instâncias (Ilustração 6) o Auto Scaling pode lançar um ou mais servidores aplicacionais caso seja notificado que os recursos da *pool* estão todos acima dos 90% de carga. Quando a carga baixa o Auto Scaling pode do mesmo modo retirar a máquina da *pool* sendo que, assim, o utilizador só tem as máquinas que necessita consoante a carga exigida [53]. O Elastic Load Balancer permite que um único ponto de contacto seja usado para balancear o serviço pelo número de instâncias existentes na *pool* de recursos detetando instâncias com problemas e desviando o tráfego para outras de modo a não haver afetação.

Amazon Mechanical Turk – permite o acesso programático a uma plataforma onde é possível submeter tarefas que necessitam de intervenção humana para serem completadas. Na plataforma estas tarefas são designadas por *Human Intelligence Tasks* (*HIT*). Um dos casos de sucesso deste serviço é o CastingWords[10], um serviço *online* que efetua a transcrição de ficheiros áudio através de um *workflow* que utiliza o Mechanical Turk [54].

Amazon Virtual Private Cloud – *Virtual Private Cloud* (*VPC*) é um serviço que permite redefinir todo o desenho da rede na nuvem, incluindo endereçamento, encaminhamento, controlo de acesso etc. [55]. Adicionalmente é possível criar uma *Virtual Private Network* (*VPN*) *site-to-site* permitindo ligar a infraestrutura local de uma instituição à nuvem.

[10] http://castingwords.com, acedido em 8 de Setembro de 2012.

Amazon CloudFront - *Content Delivery Network* (*CDN*) da Amazon que permite fazer *caching* de conteúdos respondendo ao utilizador através de um *datacenter* mais próximo da sua localização geográfica de modo a reduzir a latência e a aumentar o débito na transferência de dados [56].

4.3.2. Google

A Google é conhecida por ter sido responsável por uma autêntica revolução nos motores de busca. O que é menos conhecido é que para este fim, revolucionou igualmente o conceito de *datacenter* [17]. Em vez de usarem uma arquitetura baseada em máquinas com uma elevada densidade de processamento e com um preço que as tornava proibitivas, a Google optou por usar máquinas mais baratas, com características modestas mas em grande número. A Google fez uma clara aposta na escalabilidade horizontal sendo que hoje acredita-se que tenha cerca de 1 milhão de servidores [57] espalhados por onze localizações geográficas distintas.

Em 2008 a Google disponibilizou poder computacional aos utilizadores apresentando o *Google App Engine* (*GAE*), uma plataforma para desenvolver e hospedar aplicações *Web*.

4.3.2.1. Principais Serviços

O *Google App Engine* (*GAE*) é tal como foi dito anteriormente, uma plataforma inserida no Google Cloud Platform para desenvolver e hospedar aplicações *Web*. O *GAE* insere-se na categoria de *PaaS*, permitindo que o utilizador se preocupe somente com a aplicação e delegando tudo o que tenha a ver com a infraestrutura. O lema da Google para este serviço é "*Focus on your app, leave the rest to us*"[11]. Este nível de abstração impõe um conjunto de restrições como por exemplo o facto de só se poderem desenvolver aplicações numa das duas linguagens suportadas (Python ou Java) [58] e o facto da comunicação com outras máquinas estarem limitadas a pedidos *HTTP* ou *Hypertext Tranfer Protocol Secure* (*HTTPS*, RFC#2818) [59]. Outra das limitações é o facto de um processo criado no servidor para responder a um pedido *HTTP* não poder durar mais de 60 segundos [59]. Os processos criados através de uma *queue* de tarefas ou através do *cron* também são automaticamente destruídos se se estenderem por mais de 10 minutos [60]. Se este tempo não for adequado e a tarefa necessitar de correr por mais tempo a Google providencia um tipo especial de instância denominada *Backend Instance* [60]. Existem portanto dois tipos

[11] https://developers.google.com/appengine/, acedido em 20 de Julho de 2012.

de instância *Frontend* e *Backend*, subdivididos em classes consoante os recursos da máquina (*CPU* e memória). Tal como acontece com o *EC2* e com o Windows Azure o custo/hora é definido pelo tipo e classe da instância, sendo que um sistema com 512*MB* de *RAM* é substancialmente mais caro do que um sistema com apenas 128*MB*, por exemplo (Anexo II). Também à semelhança dos serviços concorrentes, o utilizador paga pelo número de horas que as instâncias estão a correr. Devido à elasticidade estar implantada de raiz, o *GAE* pode lançar novas instâncias a qualquer momento sem que para isso seja necessária uma solução semelhante à apresentada na Ilustração 6. Cada instância de uma aplicação tem uma fila de pedidos que é monitorizada automaticamente pelo App Engine. Se o *GAE* detetar que o número de pedidos para a aplicação é demasiado longo é criada uma nova instância para suportar a carga [61]. Se entretanto a carga baixar, o número de instâncias também baixa até ao especificado pelo utilizador no parâmetro *"minimum number of idle instances"*. Este parâmetro garante um mínimo de instâncias que estão sempre disponíveis e que respondem ao utilizador com a mínima latência possível.

Para se desenvolver para o *GAE* é necessário usar a *API* da Google que está disponível para Python e Java. A Google fornece também um *plugin* para estender o *Integrated Development Environment* (*IDE*) Eclipse de forma a integrar quer o App Engine quer o Google Web Toolkit [62]. O utilizador não é obrigado a criar a aplicação no fornecedor até que esta esteja pronta, uma vez que tem à sua disposição um ambiente de emulação que permite testar a aplicação ao longo do desenvolvimento. Quando a aplicação estiver pronta, o programador pode utilizar o Eclipse para fazer a instalação (*deployment*) para a nuvem bastando para isso que tenha registado previamente a aplicação na consola do App Engine e lhe tenha sido atribuído o respetivo *ID* [62].

No dia 28 de Junho de 2012 a Google apresentou um serviço que possibilita correr máquinas virtuais Linux, entrando assim no mercado do *IaaS* [63]. O acesso ao serviço de seu nome *Google Compute Engine* (*GCE*) encontra-se, à data de escrita, limitado, tendo os utilizadores que preencher um formulário de admissão que será avaliado pela Google.

Quanto a opções de armazenamento a Google disponibiliza ao utilizador a mesma tecnologia que usa nos seus serviços como a indexação de páginas, o Analytics ou o Earth. O App Engine Datastore, mais conhecido como BigTable é fruto de anos de experiência do fornecedor de serviço tendo respondido com sucesso às diferentes solicitações das aplicações da Google quer em termos de tamanho de dados quer em termos de latência. O

BigTable é uma base de dados não relacional feita à medida das necessidades e da arquitetura da Google, uma arquitetura altamente dinâmica e escalável baseada em computadores com recursos modestos dispersos fisicamente [64]. O facto de ter sido desenhada de raiz para apresentar uma baixa latência, ser tolerante a falhas e estar dispersa por um elevado número de máquinas permite que cresça para milhares de servidores num tamanho acumulado que pode chegar à ordem dos Petabytes [64]. A base de dados não é baseada em tuplos chave-valor mas sim em tabelas multidimensionais compostas por linhas, colunas e células, o tamanho das colunas não é fixo podendo variar entre linhas contíguas [65]. As células podem por sua vez ter vários *timestamps* o que possibilita a salvaguarda do histórico de determinado valor (Ilustração 7).

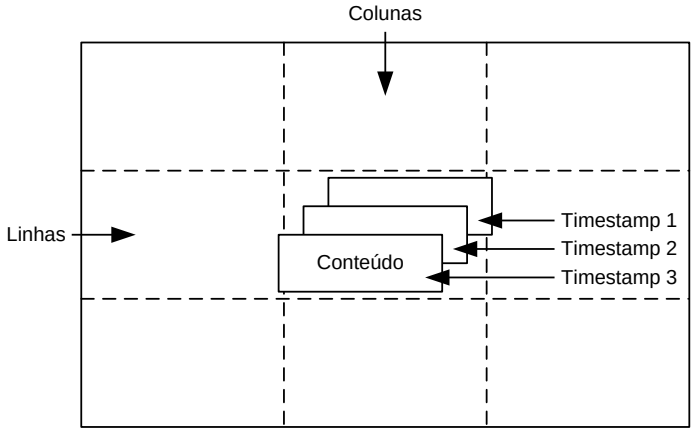


Ilustração 7 - Modelo básico de dados do Amazon BigTable [65]

Um valor da base de dados é referenciado através da combinação da linha, da coluna e de um *timestamp* sendo que o indicador da linha e da coluna são *strings* assim como todos os valores armazenados nas células [65].

Outras das opções de armazenamento disponibilizadas são o Google Cloud Storage e o Google Cloud SQL, o primeiro é um serviço de armazenamento de objetos com uma interface *Representational State Transfer* (*REST*) sendo, o segundo a oferta da Google em termos de base de dados relacionais. O Cloud Storage não é muito diferente do *S3*, na verdade está exposto da mesma maneira e apresenta o mesmo conceito de *buckets* e objetos. Os *buckets* são repositórios de objetos que servem para estruturar a informação e para facilitar a atribuição de diferentes níveis de acesso, os objetos são os ficheiros em si [66]. Os objetos podem ir até aos Terabytes de tamanho o que torna imperativo ter um serviço que permita retomar o *upload* ou o *download* de um ficheiro aquando de um

problema de rede por exemplo. A Google oferece esta possibilidade usando para isso o Google Data Resumable Protocol, oferece também alta disponibilidade ao replicar os dados automaticamente por múltiplos *datacenters* e escalabilidade [67]. O utilizador paga pela quantidade de dados armazenada, em *GB* (Anexo II).

Em relação a base de dados relacionais a Google apresenta um serviço que não necessita de administração e que se baseia no motor mySQL. O serviço permite que os programadores portem facilmente aplicações *on-premises* para a nuvem, dando uso a um *Relational Database Management System* (*RDBMS*) familiar [68]. O Google Cloud SQL apresenta como principais vantagens seguir as propriedades *ACID* e colocar ao dispor do programador a flexibilidade da linguagem de consulta estruturada *SQL*. Apesar destas vantagens o utilizador deve ponderar o facto das bases de dados estarem limitadas a 10*GB* de tamanho [69].

O Google Cloud Platform dispõe ainda de dois tipos de fila de mensagens (*queues*) "*Push Queues*" e "*Pull Queues*" que têm como principal diferença a forma como as tarefas são consumidas. No primeiro tipo de fila os pedidos presentes na *queue* são despachados segundo um débito definido pelo utilizador sendo que, para cumprir com esse débito o App Engine pode lançar novas instâncias [70]. O serviço entrega automaticamente os pedidos às instâncias e apaga-os à medida que é retornado um código *HTTP* 2xx indicando que o pedido foi corretamente processado. Apesar de todas as suas vantagens, este tipo de fila encontra-se limitado ao App Engine, não possibilitando a interação com sistemas externos. Para a interação com sistemas externos, o utilizador deve utilizar o segundo tipo de fila "*Pull Queues*" [70] que usa o mesmo modelo implementado pela Amazon no *SQS*. Nas "*Pull Queues*" para que os pedidos sejam processados é necessário que o utilizador tenha um conjunto de máquinas a fazer *pooling* à fila de modo a detetar novas entradas e a responder adequadamente. Um pedido é reclamado por uma instância do App Engine ou por uma máquina externa por determinado período de tempo, findo este período outra instância/máquina pode reclamar a tarefa caso a primeira falhe. Ao contrário das "*Push Queues*" cabe ao utilizador responder a picos de carga e controlar quantas máquinas tem para consumir as tarefas. Está também dependente do utilizador a remoção das tarefas da *queue* após serem processadas. O Google Cloud Platform dispõe ainda, de um serviço de *cache* denominado Google Memory Caching Service [71]

4.3.2.2. Outros Serviços Disponibilizados

BigQuery – permite realizar consultas sobre grandes quantidades de dados na ordem dos Terabytes em poucos segundos [72]. Esta análise é usualmente morosa mas a infraestrutura da Google permite que o utilizador extraia dados úteis através de uma sintaxe semelhante ao *SQL* quase em tempo real. Questões como a indexação ou o particionamento dos dados são abstraídas do utilizador sendo que, a Google disponibiliza quer a infraestrutura quer a tecnologia (a mesma que é usada nos anúncios e no motor de pesquisa) para tornar as consultas possíveis [62]. O BigQuery não deve ser contudo confundido com uma base de dados pois apesar de permitir fazer consultas não permite modificar os dados. Segundo a Google o serviço é um *Online Analytical Processing System* (*OLAP*) [73].

4.3.3. Microsoft

Tal como a Amazon e a Google também a Microsoft fez crescer a sua infraestrutura, não só aumentando o número de servidores mas também tornando os seus *datacenters* mais eficientes e amigos do ambiente. A evolução dos *datacenters* passou por diversas iterações: em 2007 foi lançado o primeiro *datacenter* de 2ª geração em Quincy, *EUA* que é 100% alimentado por energia hidroelétrica [74].

A escolha de uma boa localização geográfica para um *datacenter* é crucial sendo motivada por um conjunto de variáveis como o custo da eletricidade, o clima (que influencia as necessidade de refrigeração), o custo da mão de obra, custo dos terrenos, impostos e largura de banda disponível. De todos estes fatores o custo da eletricidade e da refrigeração podem só por si representar 1/3 dos custos do *datacenter* [18] o que torna uma localização próxima de uma barragem, como é o caso do *datacenter* de Quincy, apetecível.

Para reduzir os custos operacionais e para encurtar o tempo de construção (usualmente de 18 a 24 meses) a Microsoft reinventou o seu conceito de *datacenter* apostando numa arquitetura modular em que os servidores são entregues em contentores que por sua vez, são alojados no *site*. Esta arquitetura de 3ª geração foi utilizada na conceção do *datacenter* de Chicago em 2009 e nas gerações subsequentes, permitindo que um contentor com 2400 servidores seja instalado em apenas 9 horas [74]. Os contentores são levados para dentro do *datacenter* de camião sendo que, após estarem no local destinado são ligados ao circuito elétrico, ao circuito de refrigeração e à rede. Este trabalho que demora cerca de 9 horas demoraria semanas uma vez que os bastidores teriam de ser instalados um a um.

No mesmo ano, a abordagem modelar foi melhorada no sentido de trazer uma maior eficiência energética ao *datacenter* de 3ª geração de Dublin na Irlanda. Em Dublin o ar de fora do edifício é puxado e utilizado para arrefecer o *datacenter* sendo filtrado e humidificado/desumidificado consoante as necessidades numa técnica a que se dá o nome de *free-cooling* [75]. As poupanças deste tipo de refrigeração são tais que o consumo de água destas instalações é 1% do que seria expectável num *datacenter* convencional [74] sendo que a eficácia energética é de 1,25 *PUE* (*Power Usage Effectiveness*) contra 2,5 de um *datacenter* convencional [76].

O *PUE* é dado pelo rácio do consumo de energia das instalações como por exemplo *UPS*s (*Uninterruptible Power Supply*) e sistemas de refrigeração pela energia consumida pelos recursos de *TI* como servidores, *routers* e *switches* (Listagem 1). A eficiência energética é maior à medida que o *PUE* se aproxima de 1,0.

$$PUE = \frac{Consumo\ de\ energia\ das\ instalações\ (W)}{Energia\ consumida\ pelo\ equipamento\ de\ TI(W)}$$

Listagem 1 - Cálculo da eficácia energética de um datacenter

Neste momento a Microsoft já se encontra na 4ª geração de *datacenters* motivada a reduzir cada vez mais o custo decorrente do consumo de eletricidade e da refrigeração bem como os custos operacionais através da sua arquitetura modelar. Os *datacenters* podem hospedar um conjunto de serviços Microsoft como o Bing, o Windows Live Messenger e o Outlook.com não estando limitados ao Windows Azure. A Tabela 7 apresenta as regiões geográficas que hospedam de facto serviços do Azure [77], o nome das regiões foi mantido na tabela para estar coerente com as diversas ferramentas usadas para interagir com o sistema.

Continente	**Região**	**Localização geográfica**
Ásia	East Asia	Hong Kong
	South East Asia	Singapura
América	North-Central US	Estado do Illinois
	South-Central US	Estado do Texas
	East US	Estado da Virginia
	West US	Estado da California
Europa	North Europe	Irlanda
	West Europe	Holanda

Tabela 7 - Regiões geográficas dos datacenters que hospedam serviços do Windows Azure

No dia 1 de Fevereiro de 2010 [78] a Microsoft lançou a sua oferta sob o nome de Microsoft Windows Azure e disponibilizou a sua infraestrutura para hospedar aplicações. O Azure tornou-se um dos três serviços mais populares de *cloud computing* ao lado do *AWS* da Amazon e do *GAE* da Google, o Google Trend apresenta-o mesmo como o serviço mais procurado (Ilustração 8).

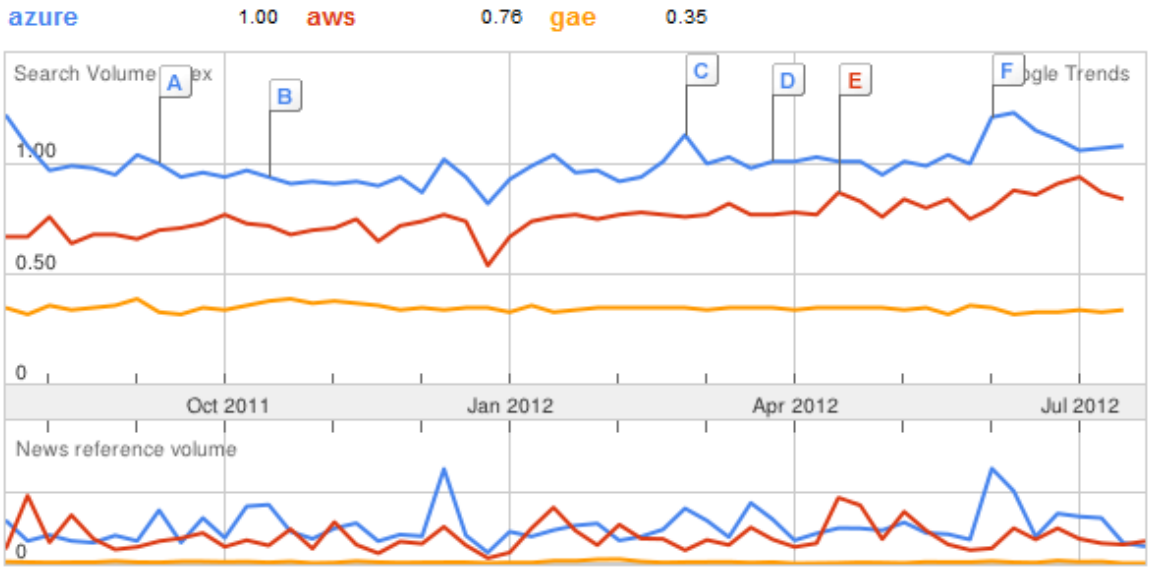


Ilustração 8 - Google Trend resultados da pesquisa por Azure, AWS, GAE[12]

No dia 7 de Junho de 2012 foi lançado a versão 1.7 do *Software Development Kit* (*SDK*) que trouxe um conjunto de novidades como a possibilidade de correr máquinas virtuais Windows e Linux na infraestrutura da Microsoft (*IaaS*) [79]. A versão usada ao longo deste documento é a versão 1.5 no entanto, dado que o *SDK* 1.7 é o maior lançamento desde que o Azure entrou no mercado [79] serão abordadas as principais novidades no Ponto 4.3.3.3.

4.3.3.1. Principais Serviços

Windows Azure é o nome dado à solução de *PaaS* da Microsoft que permite desenvolver, gerir e hospedar aplicações na nuvem. O *PaaS* é uma solução de continuidade [18] para quem quer desenvolver, gerir e hospedar aplicações Microsoft na nuvem. A solução não se limita contudo a aplicações desenvolvidas em .NET suportando um conjunto de linguagens de programação nomeadamente, Java, Python ,PHP e Node.js.

[12] http://www.google.com/trends/?q=azure,+aws,+gae&ctab=0&geo=all&date=ytd&sort=0, acedido em 27 de Julho de 2012.

Qualquer aplicação desenvolvida numa destas linguagens é hospedada numa máquina virtual que usa uma versão do Windows Server 2008 R2 [80] e que está isolada de outras através da tecnologia de virtualização Hyper-V. As aplicações e os servidores hospedeiros encontram-se acessíveis através da *Internet* e o utilizador pode aceder-lhes através de *HTTP*/*HTTPS* (única opção no *GAE*), através de *Remote Desktop Protocol* (*RDP*) ou *sockets TCP* (*Transmission Control Protocol*) e *UDP* (*User Datagram Protocol*). Ao usar o Remote Desktop o utilizador depara-se com uma máquina Windows aparentemente comum. Na prática, a principal diferença entre usar um serviço de *PaaS* em lugar de uma máquina virtual a correr Windows Server 2008 num serviço de *hosting* reside no facto de toda a manutenção ficar a cargo do fornecedor de serviço. Todas as tarefas de manutenção relativas ao servidor que hospeda a aplicação ficam a cargo da Microsoft tais como: atualizações ao sistema operativo, correções de segurança, monitorização do "estado de saúde" e resposta em caso de serem detetados problemas [80]. O facto da Microsoft monitorizar o estado das instâncias e atuar caso sejam encontrados problemas pode levar a que uma instância seja trocada de servidor. Deste modo, as aplicações não podem guardar estado no disco local, sendo necessárias outras opções de armazenamento para garantir a persistência dos dados. O fornecedor de serviço fica ainda responsável pela preparação da máquina com todos os requisitos necessários para que o utilizador corra a aplicação, permitindo que o utilizador se preocupe apenas com o desenvolvimento da aplicação e com a sua adaptação à nuvem.

Para utilizar o *PaaS* o utilizador deve ter um Windows Live ID que vai associar à sua subscrição no Windows Azure e um cartão de crédito onde vão ser cobrados os seus gastos de utilização. De seguida, o utilizador deve criar um novo serviço, dar-lhe um nome que irá servir de prefixo para o *Uniform Resource Locator (URL)* <nome do serviço>.cloudapp.net e criar um certificado que permita autenticar as operações na subscrição [81]. O utilizador tem à sua disposição três tipos de instâncias *Web Role*, *Worker Role* e *VM Role* com propósitos distintos, sucintamente descritos de seguida:

Web Role – Este tipo de instância foi concebida para hospedar aplicações *Web*. Um *Web Role* corre o Windows Server 2008 R2 tal como o *Worker Role*, a principal diferença reside no facto do primeiro ter o servidor *Web Internet Information Services* (*IIS*) integrado [80].

Worker Role – Um *Worker Role* foi concebido para hospedar todo o tipo de aplicações sendo prática comum usar este tipo de instância para correr tarefas em *background* que demorem algum tempo a processar [82]. *Web* e *Worker Roles* podem ser usados em conjunto. Por exemplo, enquanto um *Worker Role* pode redimensionar imagens ou aplicar a transcodificação de vídeo, um *Web Role* pode estar encarregue da interação com o utilizador. As instâncias do tipo *Worker/Web Role* têm por objetivo providenciar um serviço fiável, flexível e que requer pouca administração [80].

VM Role – O *VM Role* permite ao utilizador construir a sua imagem customizada do sistema operativo partindo de uma de duas versões do Windows Server 2008 (R2 Enterprise ou R2 Standard). Um cenário típico de aplicação do *VM Role* é quando um utilizador tem um processo de instalação demorado que necessita de ser executado para que a máquina fique pronta [82]. Ao preparar uma imagem em que o ambiente de execução está pronto é possível diminuir o tempo de arranque da instância. Quando existe este tipo de instalação ou quando o *setup* não pode ser automatizado deve ser usado um *VM Role*.

As instâncias encontram-se ainda agrupadas mediante o tamanho que varia entre o *Extra Small* e o *Extra Large*. O tamanho indica a capacidade da instância sendo que uma instância do tipo *Extra Small* tem um núcleo partilhado a 1.0*GHz* (*Gigahertz*) e 768*MB* de *RAM* enquanto uma instância do tipo *Extra Large* dispõe de 8 núcleos a 1.6*GHz* e 14*GB* de *RAM* (Tabela 2). A uma instância de maior tamanho está associada uma capacidade e custo superior. O custo pode ser aferido através da multiplicação do número de horas de computação pelo valor de referência (custo/hora de uma instância *Small*) vezes o fator multiplicativo (número de horas que seriam necessárias numa instância *Small* para fazer 1 hora no tamanho de instância para a qual se está a fazer o cálculo) (Tabela 2).

Todos os três tipos de instância são *stateless* portanto, o utilizador não pode contar com o armazenamento da máquina para guardar informação que necessita de ser persistente. Esta é uma característica necessária para possibilitar a gestão das máquinas por parte da Microsoft e para facilitar a escalabilidade. Outra imposição é a de que o utilizador tenha pelo menos duas instâncias para que lhe seja garantido o *Service-Level Agreement* (*SLA*) de 99,95% de *uptime* o que corresponde a um *downtime* mensal de 21 minutos e 36 segundos. Se a aplicação tiver duas ou mais instâncias é assegurado pela Microsoft que a aplicação usa pelo menos dois domínios de falha (*Fault Domain*) e dois domínios de upgrade (*Upgrade Domain*) (Tabela 8).

	Domínio de Falha 1	**Domínio de Falha 2**
Domínio de Upgrade 1	Instância 1 (i1)	
Domínio de Upgrade 2		Instância 2 (i2)

Tabela 8 - Domínios de Falha/Upgrade no Microsoft Windows Azure[13]

Os domínios de falha indicam *hardware* separado ou seja, duas instâncias (i1 e i2) em domínios de falha distintos nunca vão partilhar um único ponto de falha ou seja, não vão estar hospedados no mesmo servidor físico, no mesmo bastidor ou partilhar o mesmo *switch* [83]. Se o servidor que hospeda a instância i1 tiver uma falha de hardware, este problema fica isolado a uma das instâncias da aplicação. O Windows Azure retira a instância i1 do sistema de balanceamento de carga, de seguida encarrega-se de lançar uma nova instância num servidor considerado saudável enquanto a instância i2 assegura todo o serviço. Quando a aplicação voltar a ter duas instâncias a alta disponibilidade é recuperada.

Os domínios de falha isolam a aplicação de falhas de *hardware* e os domínios de *upgrade* isolam por seu lado a aplicação em processos de atualização [84]. Se o utilizador quiser atualizar a aplicação e tiver somente uma instância ou um único domínio de *upgrade*, o Windows Azure vai parar a instância, atualizar a aplicação e voltar a colocar a aplicação *on-line*. Este processo causa quebra de serviço sendo uma das razões pelas quais a Microsoft não consegue garantir o seu *SLA* neste tipo de configuração. Se existirem duas instâncias em dois domínios de upgrade distintos esta atualização pode ser faseada numa operação a que se dá o nome de *rolling upgrades*. Num *rolling upgrade* o tráfego é desviado do primeiro domínio, depois as instâncias desse domínio são atualizadas e reiniciadas [83]. Quando a operação estiver concluída as instâncias são colocadas novamente ao serviço e a atualização é feita ao domínio de *upgrade* seguinte [83]. Deste modo não há quebra de serviço. O Windows Azure dispõe ainda de outra opção para atualizar as aplicações hospedadas com um impacto de serviço mínimo a que se dá o nome de *static upgrades* [83]. As aplicações podem ser enviadas quer para o ambiente de produção quer para o ambiente de *staging*. Aquando de um *static upgrade* o utilizador pode instalar a aplicação no ambiente de *staging*, correr uma série de testes para se certificar de que a aplicação se está a comportar conforme esperado e depois passá-la para o ambiente produção. A troca do ambiente de testes para produção é quase instantânea [84]

[13] http://blog.toddysm.com/2010/04/upgrade-domains-and-fault-domains-in-windows-azure.html, acedido em 28 de Julho de 2012.

bastando ao utilizador comutar o *IP* virtual do ambiente de produção para o de *staging* o que pode ser feito clicando num botão no portal de administração (Ilustração 9).

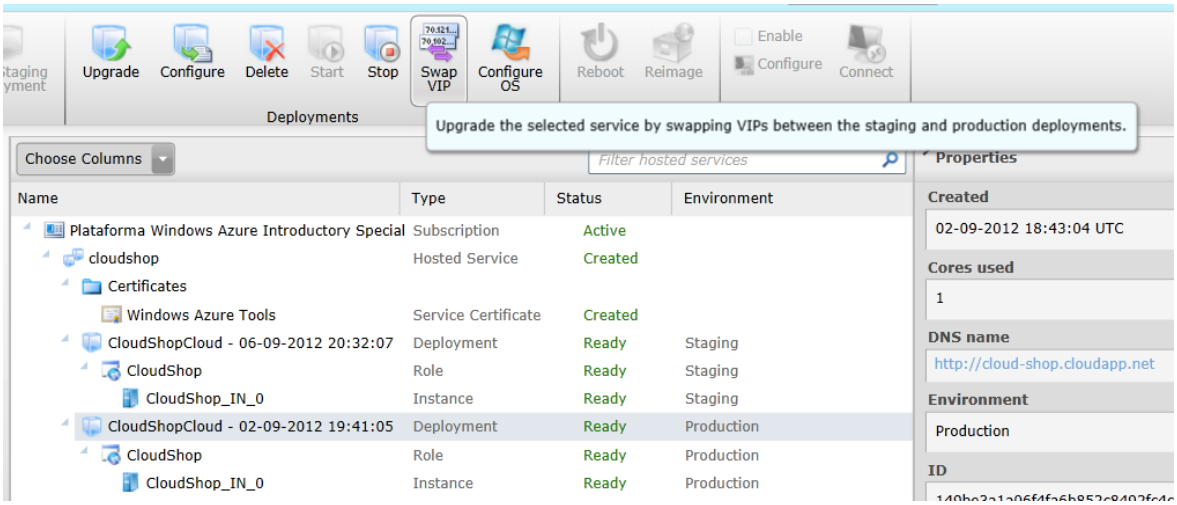


Ilustração 9 - Comutação do IP virtual no Windows Azure

Para contornar o facto do armazenamento local ser volátil a Microsoft disponibiliza o Windows Azure Drive, um disco rígido virtual (*VHD*, *Virtual Hard Drive*) com o sistema de ficheiros *New Technology File System* (*NTFS*) que está implementado sobre um *BLOB* [85]. Este tipo de armazenamento garante persistência mas não pode ser partilhado uma vez que um *VHD* só pode estar montado numa única instância. Outras opções de armazenamento que podem ser partilhadas são o SQL Azure, as Tabelas e os *BLOB*s.

O SQL Azure é a oferta da Microsoft para bases de dados relacionais. O serviço usa a linguagem *T-SQL* e as mesmas ferramentas usadas no SQL Server tais como o Management Studio. As semelhanças com o SQL Server permitem que um utilizador familiarizado com esta tecnologia consiga passar facilmente para a nuvem, beneficiando de alta disponibilidade, tolerância a falhas, abstração do *hardware* e manutenção do *software* da base de dados e do sistema operativo [80]. Para assegurar alta disponibilidade os dados são replicados por três servidores físicos na mesma região, estando os dados sempre atualizados (consistência estrita) o que permite que o Windows Azure comute a qualquer momento para outro servidor em caso de falha. A comutação para uma das cópias redundantes é transparente para o utilizador.

Tal como acontece com outras bases de dados relacionais como o Amazon *RDS* ou o Google Cloud SQL a escalabilidade não é providenciada automaticamente. A escalabilidade ao nível das bases de dados relacionais é completamente diferente da que

pode ser encontrada nas instâncias *stateless* uma vez que estas guardam informação persistente que se encontra relacionada. Como as bases de dados mantêm relacionamentos, a adição ou remoção de um nó nunca pode ser tratada isoladamente ao contrário do que acontece com as instâncias *stateless* que funcionam autonomamente. O aumento de recursos como resposta a um pico de procura ao serviço é relativamente fácil na camada de apresentação ou na camada de lógica de negócio [86] uma vez que basta lançar um número arbitrário de *Web* e/ou *Worker Roles* para suster o pico de procura, mas é mais difícil de conseguir na camada de acesso a dados. A escalabilidade vertical (aumento da capacidade de um nó) continua a ser a solução mais comum utilizada para diminuir a latência e maximizar o débito de uma base de dados. Partindo do princípio que a cada dois anos a procura ao servidor não duplica, o administrador de base de dados estará protegido pela lei de Moore caso migre a base de dados para um novo servidor com o dobro da capacidade em igual período [87]. Este é o caminho mais simples para aumentar o desempenho da base de dados e por isso é também o mais utilizado. No Windows Azure não é possível alterar as características da máquina que assegura o SQL Azure, estando a escalabilidade vertical limitada ao aumento do tamanho máximo da base de dados. Quando o limite de determinada base de dados é alcançado, o Windows Azure retorna o código 40544 [88], o utilizador deve agir de modo a libertar espaço ou a aumentar o tamanho máximo da base de dados. Se o utilizador optar por aumentar o tamanho máximo da base de dados o SQL Azure disponibiliza vários tamanhos agrupados em dois tipos WEB e BUSINESS. O primeiro disponibiliza um tamanho máximo de 1 e 5*GB*, enquanto o segundo disponibiliza um tamanho máximo de 10, 20, 30, 40, 50, 100 e 150*GB*. O aumento de capacidade pode ser feito numa questão de segundos tal como exemplificado no comando *T-SQL* presente na Listagem 2.

```
ALTER DATABASE CloudShop MODIFY (EDITION='WEB', MAXSIZE=5GB)
```

Listagem 2 - Comando T-SQL para aumentar a capacidade da base de dados SQL Azure

A limitação dos 150*GB* de capacidade pode constituir um problema para algumas aplicações mas, mais importante ainda, são as limitações de desempenho que podem advir do uso de um único servidor [86]. Para resolver este problema pode ser utilizado um padrão denominado de *sharding* [89] que consiste no particionamento de uma base de dados de grandes dimensões em partes mais pequenas denominadas de *shards*. Os *shards* são distribuídos por vários servidores de modo a dividir a carga e a obter-se a

escalabilidade desejada [86] (escalabilidade horizontal). A divisão dos dados e o processo de encaminhar os pedidos para o *shard* correto introduzem complexidade na aplicação, complexidade essa que é atenuada pelo uso do SQL Azure Federations.

O SQL Azure Federations é um serviço que permite aplicar o padrão de *sharding* a uma base de dados SQL Azure, diminuindo a complexidade do processo e evitando qualquer quebra de serviço [90]. Existem várias formas de particionar uma base de dados sendo que a empregue no SQL Azure Federations é o particionamento de linhas ou particionamento horizontal. Um exemplo de particionamento horizontal é a divisão de uma tabela por código de cliente, linhas em que os clientes tenham o *ID* compreendido entre 0 e 1000000 vão para uma tabela alojada num servidor e clientes de 1000000 a 2000000 vão para outra tabela alojada num servidor distinto. A uma base de dados à qual é aplicado *sharding* dá-se o nome de base de dados federada. De seguida irá ser feita uma breve exposição da arquitetura de uma base de dados federada no SQL Azure Federations (Ilustração 10).

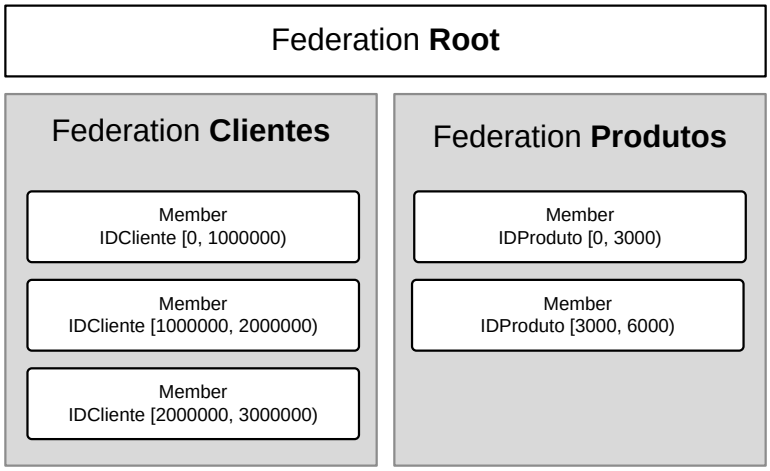


Ilustração 10 - Arquitetura de uma base de dados federada [91]

Federation Root – corresponde à base de dados central que contém a informação sobre o particionamento dos dados [86]. A *Federation Root* armazena os metadados relativos às *Federations* bem como os intervalos de valores armazenados pelos *Federations Members* portanto, deve ser a primeira base de dados a ser criada. Todas as *Federations* vão herdar as propriedades desta tabela tais como a codificação de caracteres, o tipo de base de dados (WEB ou BUSINESS) e o tamanho máximo [91].

Federation – define que dados vão ser particionados e como [86]. Para criar uma *Federation*, o utilizador liga-se à *Federation Root* e define que campo inequívoco vai ser

usado para dividir as linhas pelos *Federation Members* [91] e qual o método de distribuição. Na Ilustração 10 o campo inequívoco para a "Federation Clientes" é o "IDCliente" o que indica que a tabela vai ser distribuídas por partes mais pequenas (*shards*) com base no *ID* do cliente. Numa base de dados federada este campo é conhecido como *Distribution Key*. Importa notar que neste momento, o método de distribuição se encontra limitado ao particionamento horizontal conhecido como *RANGE* no SQL Azure Federations.

Federation Member – um *Federation Member* ou *shard* corresponde a uma partição dos dados de uma *Federation*. Um *shard* é uma base de dados hospedada num servidor distinto que contém um intervalo de valores. Na Ilustração 10 a "Federation Produtos" tem 2 *shards* sendo que o primeiro armazena os produtos com o *ID* 0 a 3000 e o segundo *shard* armazena os produtos com o *ID* compreendido entre 3000 e 6000.

As operações de *sharding* podem ser feitas no portal de administração do Windows Azure, utilizando ferramentas como o SQL Azure Migration Wizard [91] ou diretamente através de comandos *T-SQL* [86]. Qualquer que seja a opção tomada pelo utilizador, o serviço encarrega-se da distribuição automática dos dados pelos *Federation Members* sem que haja interrupção no serviço.

O custo do serviço de base de dados (federada ou não) é calculado através da quantidade de dados armazenados por mês (*GB*/mês segundo as tarifas que se encontram na Tabela 3) ao que acrescem os custos de transferência de dados presentes na Tabela 5. No caso das tabelas e dos *BLOB*s, os custos de armazenamento também são transferidos, do mesmo modo, para o utilizador (Tabela 4). A única diferença para a base de dados relacional reside no facto das tabelas e dos *BLOB*s terem um custo de 0,01 *USD* por cada 100000 transações (operações de leitura/escrita) efetuadas aos dados armazenados [35].

As tabelas no Windows Azure não são mais do que bases de dados não-relacionais *noSQL* que se baseiam em tuplos chave-valor. O uso das tabelas é indicado para se guardarem dados estruturados onde não são necessários operações de *JOIN* ou chaves estrangeiras [92]. Se a aplicação não tiver esta necessidade, o utilizador pode ganhar escalabilidade horizontal de forma automática. As tabelas no Windows Azure oferecem um serviço ao estilo do SimpleDB da Amazon, providenciando alta disponibilidade, tolerância a falhas e flexibilidade, sendo que os dados são replicados por três servidores físicos e a base de

dados pode expandir quer em tamanho, não tendo qualquer limite associado [93], quer em número de servidores que a suportam. A base de dados providencia consistência "estrita" e pode ser acedida através de *REST* ou através das tecnologias Microsoft ADO.NET e *LINQ* (*Language Integrated Query*) [94].

Assim como as tabelas do Windows Azure são semelhantes ao SimpleDB da Amazon, também a implementação dos *BLOB*s é semelhante aos dois serviços concorrentes o *S3* da Amazon e o Google Cloud Storage. O objetivo deste serviço é o de guardar grandes quantidades de dados não estruturados como ficheiros de texto ou ficheiros binários como imagens, áudio e vídeo. Para organizar os objetos armazenados e para lhes dar diferentes níveis de acesso, o utilizador pode criar repositórios que no *S3* e no Google Cloud Storage têm o nome de *buckets*, mas que no Windows Azure se designam de *containers* [95]. Para além dos objetivos já apresentados, os *BLOB*s podem também servir para virtualizar um disco rígido "Windows Azure Drive" que traz às instâncias armazenamento persistente na forma de um sistema de ficheiros *NTFS* [80].

O Microsoft Windows Azure dispõe ainda de uma *queue* e de um serviço de *caching*. O principio de funcionamento de uma *queue* é simples: uma aplicação guarda uma mensagem em fila, e uma segunda aplicação consome a mensagem e retira-a da fila. A introdução de uma *queue* num sistema promove a resiliência e a elasticidade do mesmo. O facto de guardar as mensagens em fila até que um pedido possa ser atendido faz com que por um lado, a solução resista a falhas num dos componentes envolvidos e por outro, aguente picos de procura onde o débito dos pedidos é muito superior ao débito com que as mensagens conseguem ser processadas [37]. Um cenário típico para o uso de *queues* é um sítio que faz transcodificação de vídeo [80]. O sítio, que poderia por exemplo ter sido programado em PHP, recebe um vídeo e muda o seu formato para que possa ser lido em dispositivos móveis. A transcodificação de vídeo é uma tarefa computacionalmente exigente e consequentemente demorada e por isso não é apropriada para um *Web Role*. A aplicação é então apoiada por um *Worker Role* que realiza a tarefa de forma assíncrona, mas como pode ser feita a comunicação entre as duas instâncias? É aqui que entra a *queue*. O sítio recebe o vídeo e pode guardá-lo num *BLOB*, de seguida envia uma mensagem para a *queue* que contém o *URL* do objeto. Um ou mais *Worker Roles* encarregues de fazer a transcodificação de vídeo acedem à *queue* para verificarem se há alguma mensagem (modelo "*Pull*"). Quando detetam uma mensagem nova, o *Worker Role* reclama-a ficando

invisível para outras instâncias que acedam à *queue* (isto impede que duas instâncias estejam a fazer o mesmo trabalho), acede ao vídeo através do *URL* providenciado e realiza a transcodificação. Quando a tarefa tiver sido completada, a instância remove a mensagem da fila. Este desenho torna a aplicação mais escalável, promove a independência das instâncias e constitui um meio para a realização de tarefas assíncronas [80]. É cobrado pela ocupação da *queue* no Microsoft Windows Azure o mesmo valor mensal aplicável às tabelas e aos *BLOB*s.

No Windows Azure existe a possibilidade de subscrever um serviço de *caching* que permite colocar uma memória *RAM* de tamanho fixo próximo da aplicação por forma a reduzir a latência e a diminuir a carga da base de dados. Existem pelo menos três usos que podem ser dados à *cache* de memória [96]. Usá-la para guardar informação volátil de uma fonte externa que não precisa de estar atualizada, um exemplo disso pode ser o armazenamento dos *feeds* do Twitter[14] relacionados com um evento musical que está a ser acompanhado por um sítio na *Internet*. Um segundo uso é a introdução da *cache* entre a aplicação e uma base de dados, a *cache* neste caso tem o intuito de reduzir a latência no acesso [37] respondendo com tempos uniformes à medida que o número de pedidos aumenta, ao mesmo tempo que diminui a carga da base de dados. Quando os pedidos são encaminhados diretamente para a base de dados, os tempos de resposta uniformes podem não se verificar uma vez que, a partir de um certo número de utilizadores, as bases de dados podem começar a saturar [96] o que se pode dever a limites de I/O ou de processamento. Como dois pedidos consecutivos na nuvem podem ser respondidos por duas instâncias diferentes, não é garantido que uma variável de sessão guardada num servidor *Web* se encontre disponível. O terceiro caso de uso tem em vista a resolução deste cenário [96] (Ponto 5.3.3.6).

Consoante o caso de uso aplicável o acesso à *cache* pode ser feito através de um ASP.NET Provider ou através da *API* criada para o efeito. O preço mensal da *cache* é fixo e só está dependente do tamanho escolhido aquando da sua criação (Tabela 6).

4.3.3.2. Outros Serviços Disponibilizados

Service Bus – serviço que disponibiliza mecanismos que possibilitam a troca de mensagens de forma segura entre componentes de um sistema distribuído quer eles estejam

[14] https://twitter.com, acedido em 8 de Setembro de 2012.

on-premises ou na nuvem [37]. Os mecanismos colocados à disposição do utilizador pelo Windows Azure Service Bus facilitam a composição [97] numa arquitetura orientada a serviços. O Azure Service Bus providencia atualmente dois modelos de troca de mensagens *Relayed Messaging* e *Brokered Messaging* que são usados num barramento de serviço para interligar os diversos intervenientes da comunicação. No primeiro modelo, dois intervenientes da comunicação abrem uma ligação para o endereço do Service Bus. O Service Bus responde abrindo uma conexão no sentido inverso o que resulta num canal de comunicação bidirecional entre os dois extremos. Esta técnica é a mesma usada em programas de *Peer-to-Peer* (*P2P*) e em jogos *online* permitindo contornar *firewalls* e mecanismos de *Network Address Translation* (*NAT*) [37] tornando uma máquina acessível sem que seja feita nenhuma alteração na mesma ou na rede. No segundo modelo, o barramento é denominado de *broker* e existem uma ou mais filas de mensagens que suportam dois cenários distintos. No cenário mais simples, o Service Bus tem uma fila de mensagens ou *queue* na qual um ou mais produtores colocam mensagens e múltiplos consumidores competem para as receber. Cada consumidor processa as mensagens ao seu ritmo fazendo com que as mensagens sejam corretamente balanceadas [37] e com que os consumidores nunca sejam sobrecarregados.

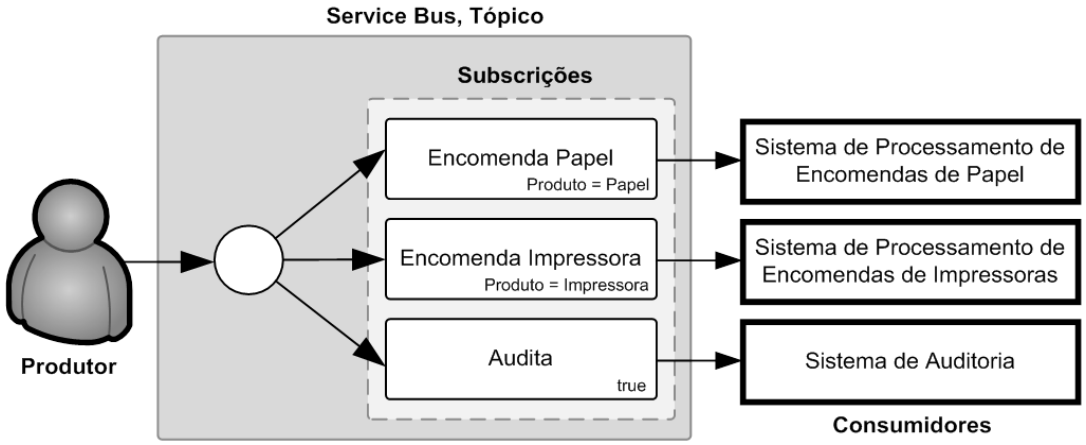


Ilustração 11 - Tópico no Azure Service Bus [37]

No segundo cenário um ou vários produtores criam mensagens para um tópico ao qual estão associadas subscrições (Ilustração 11). As mensagens que determinada subscrição recebe dependem de um conjunto de regras que filtram as mensagens pelo conteúdo. No exemplo da Ilustração 11 existem três subscrições: uma recebe as encomendas de papel outra recebe as encomendas de impressoras e a terceira regista as encomendas no sistema de auditoria. A subscrição que regista as encomendas no sistema de auditoria não tem

regras de filtragem de modo a receber uma cópia das mensagens que surgem em qualquer uma das outras subscrições. Este cenário permite especificar com granularidade que tipo de mensagens determinado consumidor recebe, acrescentando flexibilidade ao barramento de serviço.

CDN - *Content Delivery Network* (*CDN*) é o nome dado a uma rede de *datacenters* espalhados pelo mundo que permitem fazer *caching* de *BLOB*s [80]. Supondo que uma instituição sediada nos Estados Unidos da América lança uma atualização de *software* para o seu produto mais popular. É provável que ocorra uma elevada procura, de variadas partes do mundo. É ainda expectável que utilizadores distantes dos Estados Unidos tenham uma latência superior e um débito inferior na transferência de dados. Para mitigar esta situação, a empresa ativa o suporte ao *CDN* no *container* que aloja o ficheiro da atualização. Quando um utilizador de qualquer parte do mundo transfere a atualização é criada uma cópia no *datacenter* mais próximo. Assim, o utilizador seguinte que partilhar a mesma localização geográfica acede à cópia em *cache* em vez de aceder ao *datacenter* de origem, melhorando desta forma o desempenho no acesso aos dados [80]. Outros fornecedores de serviço apresentam ofertas semelhantes como o CloudFront da Amazon (Ponto 4.3.1.2).

4.3.3.3. Novidades do SDK 1.7

Um dos fatores que torna o *SDK* 1.7 o maior lançamento desde que o Azure entrou no mercado [79] é a possibilidade de correr *Virtual Machines* (*VM*s) na infraestrutura da Microsoft. Este lançamento marcou a entrada da Microsoft no mercado dos fornecedores de *IaaS* abrindo portas para novos cenários na nuvem, como a possibilidade de correr um servidor de Active Directory numa *VM* ou a possibilidade de integrar uma aplicação a correr num *Web/Worker Role* com uma base de dados *noSQL* como o Apache Cassandra a correr numa *VM* [80]. O maior grau de customização está associado a um custo operacional superior relativamente aos *Web/Worker* e *VM Role* que no *SDK* 1.7 passam a ser designados de *Cloud Services*. Ao contrário das instâncias dos *Cloud Services* as *VM*s são persistentes ou seja, todos os dados e todos os programas instalados nas *VM*s são mantidos ao longo do tempo. Esta característica advém da *drive* do sistema ser armazenada num *BLOB* como acontece com o Windows Azure Drive. Em relação aos sistemas suportados, a Microsoft disponibiliza no portal de administração algumas imagens preparadas tais como o Microsoft SQL Server 2012, o Windows Server 2012 ou mesmo

sistemas Linux como o SUSE, CentOS ou Ubuntu [79]. O utilizador tem ainda a opção de construir o seu próprio *VHD* e de o migrar para a nuvem, bem como a possibilidade, de voltar a utilizar o *VHD* localmente, retomando o estado que tinha aquando da retirada do Microsoft Windows Azure [80].

Outra das novidades é o Azure Web Sites, uma oferta de alojamento *IIS* para sítios menos complexos. Com o Azure Web Sites o utilizador não tem que alterar a sua aplicação para adaptá-la à nuvem como num *PaaS*. Numa *VM* a aplicação também não necessitaria de ser alterada mas a gestão de todo o ambiente de execução ficaria a cargo do utilizador. O Azure Web Sites constituem uma solução apropriada para hospedar aplicações *Web* com uma arquitetura menos complexa ou que se baseie numa *framework Web* como é o caso do WordPress, Joomla ou Drupal [80]. O alojamento pode ser partilhado ou reservado sendo que a publicação neste tipo de alojamento se processa em poucos segundos [98]. O *File Transfer Protocol* (*FTP*) é disponibilizado como uma das opções de publicação tal como é usual noutras opções de alojamento. À semelhança do que acontece também com outros serviços de alojamento é disponibilizada uma base de dados MySQL providenciada através de uma parceria da Microsoft com a ClearDB [80].

O serviço de *caching* do Windows Azure também foi alvo de algumas alterações entre elas o suporte ao protocolo *memcached* [51] e a possibilidade de usar uma percentagem de memória dos Cloud Services (*Web e Worker Roles*) para formar uma memória distribuída [98] que tem como principal beneficio a redução de custos face a uma *cache* dedicada. Se o utilizador tiver quatro *Web Roles* e reservar em cada um deles 300*MB* de memória é criada uma *cache* distribuída com o tamanho de 1,2*GB*. Para as aplicações a *cache* é única e o facto de os dados poderem estar presentes em qualquer uma das quatro instâncias é transparente. O método de acesso à *cache* mantém-se também inalterado. À data de escrita, este serviço está em fase de pré-lançamento.

Antes da versão 1.7 do *SDK*, as *VPN*s entre a infraestrutura local de uma instituição e a nuvem estavam limitadas ao serviço Windows Azure Connect. O serviço permitia a ligação de máquinas locais a aplicações da nuvem no entanto, esta ligação só era possível em máquinas Windows e recorria a um agente de *software* que necessitava de ser instalado [80] na máquina cliente. O serviço Virtual Network veio colmatar essa lacuna permitindo não só redefinir todo o desenho da rede na nuvem, incluindo endereçamento, encaminhamento e controlo de acesso mas também, permitir a criação de *VPN*s *site-to-site*.

Não menos importante foi o suporte oficial dado aos comandos de PowerShell [99] que permitem automatizar um conjunto de tarefas de gestão através de *scripts*.

4.4. Fornecedor de Serviço Escolhido

O Windows Azure foi escolhido para esta dissertação não só pela sua popularidade (Ilustração 8) mas também, por ser uma oferta que está num patamar intermédio de compromisso entre a flexibilidade e o nível de customização [18]. A plataforma suporta um vasto leque de aplicações desenvolvidas num conjunto de linguagens diferentes, ao contrário do *GAE* que suporta somente aplicações *Web* escritas em Python ou Java. Embora o utilizador do Azure tenha acesso ao *SO* e ao ambiente de execução, não tem o mesmo nível de controlo que teria se usa-se o *AWS*, nem a resiliência a falhas e escalabilidade que teria no *GAE* sem necessitar de qualquer configuração. É por isso que o Microsoft Windows Azure fica entre uma oferta de *PaaS* como a da Google e uma oferta de *IaaS* como a da Amazon. Outro factor que pesou na escolha do Windows Azure foi ter-se apresentado como uma das ofertas mais completas ao nível de serviços (Tabela 9).

4.5. Síntese

Cada serviço que faz parte das ofertas da Amazon, Google ou Microsoft poderia ter um capítulo a si dedicado. Neste capítulo foi feita uma análise aos principais serviços de cada uma das ofertas sendo que, sempre que se julgou necessário foi descrita a tecnologia em que o serviço se baseia e foram dados alguns exemplos de aplicação. Ao abordar os principais serviços disponibilizados por cada um dos maiores fornecedores de *cloud computing* este capítulo apresenta-se como um catálogo a que se pode recorrer para escolher a melhor oferta consoante as necessidades do utilizador. Neste capítulo encontra-se também a justificação porque foi utilizado o serviço da Microsoft para o desenvolvimento da componente prática. Por fim é deixada uma tabela de referência em que estão presentes os serviços dos três fornecedores (Tabela 9).

	Amazon Web Services		Google Cloud Platform		Microsoft Windows Azure	
PaaS	✘	-	✓	Google App Engine (GAE)	✓	Cloud Services
IaaS	✓	Elastic Compute Cloud (EC2)	✓	Google Compute Engine (GCE)	✓	Virtual Machines
Alojamento Web	✘	-	✘	-	✓	Web Sites
VHD	✓	Elastic Block Store (EBS)	✓	Persistent Block Device usado no Google Compute Engine	✓	Windows Azure Drive usado nos Cloud Services. As Virtual Machines e os Web Sites já são persistentes por natureza
BLOB	✓	Simple Storage Service (S3)	✓	Google Cloud Storage	✓	BLOB
Base de Dados Relacional	✓	Relational Database Service (RDS) o utilizador pode optar por uma base de dados com o motor MySQL, Oracle ou Microsoft SQL Server	✓	Google Cloud SQL baseado no motor MySQL	✓	SQL Database, também designado por SQL Azure baseado no motor Microsoft SQL Server
Base de Dados Não Relacional	✓	SimpleDB, DynamoDB	✓	App Engine Datastore (BigTable)	✓	Tables
Queue	✓	Amazon Simple Queue Service (SQS)	✓	Task Queue	✓	Queues, Service Bus
Serviço de Caching	✓	Amazon CloudFront CDN, Amazon Simple Queue Service (SQS) compatível com o protocolo memcached	✓	Google Memory Caching Service	✓	Content Delivery Network (CDN), Caching compatível com o protocolo memcached a partir da versão 1.7
VPN Site-to-Site	✓	Amazon Virtual Private Cloud	✘	-	✓	Virtual Network disponível com o SDK 1.7, dantes as VPNs estavam limitadas a machine-to-machine através do Windows Azure Connect

✓ = Serviço existe e está disponível ao público	**✓ = Serviço encontra-se em fase de acesso limitado e ainda não está disponível ao público**	**✘ = Serviço não existe**

Tabela 9 - Comparativo dos serviços oferecidos pelos fornecedores de cloud computing

5. Migração de Solução On-Premises para o Windows Azure

5.1. Introdução

Para aferir o esforço de migrar uma aplicação *Web on-premises* para a nuvem, foi criado um sitio que representasse uma aplicação típica de uma empresa de pequena/média dimensão. De modo a que o processo de migração fosse fidedigno não foi tomado nenhum cuidado especial durante a conceção da aplicação que se pretendia posteriormente adaptar à nuvem. O objetivo é obter um cenário próximo de uma situação real, com as mesmas dificuldades expetáveis num processo de migração real. No final do capítulo são apresentadas recomendações para a migração de aplicações *Web* no Microsoft Windows Azure fruto, quer da recolha das melhores práticas quer da experiência obtida no capítulo.

5.2. Aplicação On-Premises a Migrar

Para efeitos de cenário de testes, foi elaborada uma loja de comércio eletrónico utilizando a linguagem ASP.NET e uma arquitetura de três camadas. O nome do sítio é *CloudShop* e apresenta-se como uma loja de comércio de material informático por via eletrónica.

5.2.1. Requisitos

Os requisitos enumerados estão subdivididos em requisitos funcionais (*RF*) e não funcionais (*RNF*). Todos os requisitos devem ser cumpridos para que a solução desenvolvida providencie uma base fidedigna para se averiguar qual o esforço necessário à migração de uma aplicação para a nuvem. Neste ponto o verbo Gerir é usado para se referir à conjugação das funcionalidades de Criar, Alterar e Remover. Gerir produtos nos requisitos funcionais refere-se portanto, à funcionalidade de Criar, Alterar e Remover produtos.

5.2.1.1. Requisitos Funcionais

Os requisitos funcionais do sítio *CloudShop* são os seguintes:

- RF1 - Permitir que os utilizadores se registem no sítio;
- RF2 - Autenticar utilizadores usando a base de dados;
- RF3 - Adicionar e remover artigos do carrinho de compras;

- RF4 - Fazer *checkout* do carrinho de compras;
- RF5 - Gerir produtos;
- RF6 - Gerir marcas de produtos;
- RF7 - Gerir categorias de produtos;
- RF8 - Gerir utilizadores;
- RF9 - Ativar e desativar contas de utilizador;
- RF10 - Fazer *reset* à palavra passe dos utilizadores.

5.2.1.2. Requisitos Não-Funcionais

Os requisitos não funcionais estão relacionados com o uso do sítio *CloudShop* ao nível da segurança:

- RNF1 – Só os administradores podem aceder à área de administração;
- RNF2 – O utilizador necessita de estar registado para completar o processo de *checkout*.

5.2.2. Ferramentas Utilizadas

O desenvolvimento de uma aplicação ASP.NET 4 pareceu a solução natural depois de se ter optado pela plataforma de *PaaS* da Microsoft. A linguagem de programação empregue foi o C# (C Sharp) embora pudesse ter sido utilizada qualquer outra das linguagens suportadas pela plataforma .NET.

A ferramenta de desenvolvimento escolhida foi o Microsoft Visual Studio 2010[15], uma escolha óbvia quando se desenvolve em .NET. Optou-se por não se utilizar o depurador em conjunto com o servidor *Web* do Visual Studio o "Cassini" em vez disso, o depurador foi ligado ao servidor *Web* escolhido o *IIS* 7.5[16]. O motor de base de dados escolhido foi, por sua vez, o Microsoft SQL Server 2008 R2[17] com a ferramenta Microsoft *SQL Server Management Studio* (*SSMS*). Foi ainda empregue a ferramenta soapUI[18] para auxiliar na integração com o motor de pagamentos.

5.2.3. Arquitetura de Três Camadas

Uma arquitetura de três camadas tem, tal como o nome sugere, três camadas lógicas para segmentar a aplicação: (1) Camada de Apresentação, (2) Camada de Lógica de Negócio e

[15] http://www.microsoft.com/visualstudio/en-us, acedido em 2 de Agosto de 2012
[16] http://www.iis.net/, acedido em 2 de Agosto de 2012
[17] http://www.microsoft.com/sqlserver/en/us/default.aspx, acedido em 2 de Agosto de 2012
[18] http://www.soapui.org/, acedido em 10 de Agosto de 2012

(3) Camada de Acesso a Dados. A separação em camadas lógicas aumenta a robustez e a escalabilidade da aplicação para além de ajudar a estruturar o código facilitando a perceção do fluxo da informação e promovendo a reusabilidade [100].

Podemos prever que uma aplicação estruturada num modelo de três camadas seja mais facilmente migrada para a nuvem. O facto de estar bem estruturada faz com que seja mais fácil isolar os blocos lógicos que a constituem.

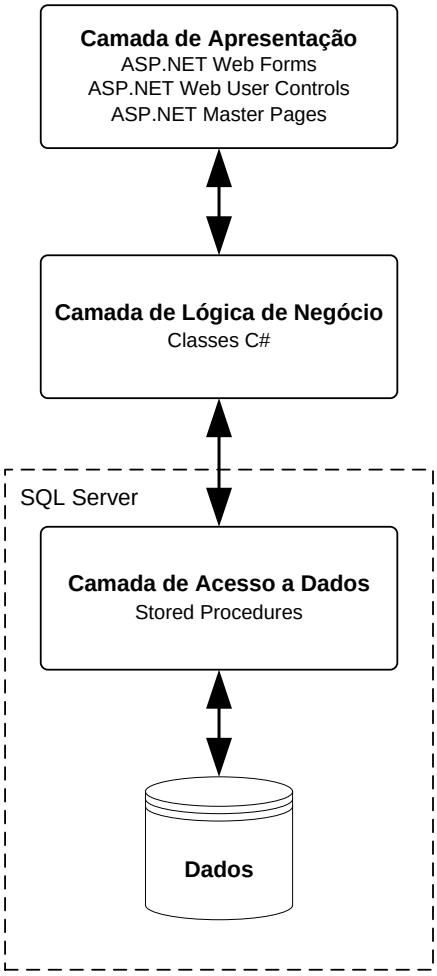


Ilustração 12 - Arquitetura de três camadas utilizando tecnologias Microsoft [100]

As três camadas da arquitetura são as seguintes (Ilustração 12):

Camada de Apresentação – Parte da aplicação que está exposta ao utilizador, contém todos os elementos gráficos do sítio e toda a lógica de interação com o utilizador. A camada de apresentação é composta pelas páginas dinâmicas de extensão *aspx* e pelos componentes definidos pelo utilizador (extensão *ascx*). Os componentes definidos pelo utilizador (*Web User Control*) permitem criar componentes personalizados que podem ser

reaproveitados em várias páginas do sítio. Um exemplo deste tipo de componentes é o *Pager*. O *Pager* permite navegar por uma listagem de produtos quando esta é muito extensa.

Camada de Lógica de Negócio – Recebe pedidos da Camada de Apresentação, processa-os e retorna os resultados. Quase todos os eventos que ocorrem na Camada de Apresentação resultam numa chamada a esta camada. Esta camada intermédia é também responsável por abstrair as chamadas à base de dados.

Camada de Acesso a Dados – Esta camada é responsável pelo armazenamento dos dados. Foram empregues *stored procedures* em vez de consultas construídas dinamicamente. Os *stored procedures* não são mais do que consultas compiladas no servidor de base de dados, tendo como principal benefício o desempenho acrescido.

O fluxo de informação deve seguir uma ordem sequencial ou seja, a Camada de Apresentação nunca pode aceder à Camada de Acesso a Dados diretamente. O fluxo tem que ser sempre Camada de Apresentação →Camada de Lógica de Negócio→Camada de Acesso a Dados e vice-versa para o pleno aproveitamento dos benefícios desta arquitetura.

5.2.4. Principais Pormenores de Implementação

Destacam-se como principais pormenores de implementação:

Camada de Abstração da Base de Dados - Foi criada uma classe de acesso a dados [100] que permite que as chamadas feitas à base de dados sejam independentes do motor utilizado pela aplicação. Em vez de ser empregue o namespace *System.Data.SqlClient* para um servidor *SQL Server* ou *System.Data.Oracle* para um servidor Oracle, é usado o namespace *System.Data.Common* que contém classes independentes do motor de base de dados utilizado. Graças ao polimorfismo, quando for chamado um método de acesso à base de dados este conterá uma instância do método específico para o motor de *BD* utilizado pela aplicação.

Mecanismo de Tratamento de Exceções da Aplicação - Num sítio ASP.NET cada vez que uma exceção não tratada se propaga pela aplicação é gerada uma página que, para além de ser alarmante, mostra a linha onde foi lançada a exceção e o *Stack Trace*. Esta página pode revelar dados sensíveis relativos à implementação do sítio. Por estas razões foi implementado o método *Application_Error* do ficheiro *Global.asax* [101] que é chamado

sempre que uma exceção não tratada se propaga pela aplicação. Deste modo, é apresentada uma mensagem customizada e enviado um email com os detalhes do erro para o administrador.

Classe de Acesso ao(s) Ficheiro(s) de Configuração - Foi criada na aplicação *CloudShop* uma classe que serve de ponto de contacto para o(s) ficheiro(s) de configuração. Nesta fase da implementação a classe serve somente para aceder ao ficheiro de configuração *Web.Config*. No entanto, no futuro a classe poderá servir para aceder a múltiplos ficheiros. A classe tem o nome *CloudShopConf* e é composta por propriedades estáticas que possibilitam a leitura dos valores dos ficheiros de configuração (*GET*) sem que a classe tenha que ser instanciada.

Autenticação e Autorização - Num sítio nem todas as páginas devem ser públicas por isso, é necessário usar um mecanismo de segurança que providencie autenticação e autorização. Autenticação é o processo de identificar os utilizadores, sendo o método mais comum a solicitação de um nome de utilizador e uma palavra-passe. Autorização é o processo de identificar quais os recursos a que um utilizador tem acesso. No sítio *CloudShop* a autenticação e autorização são necessárias para se cumprir com o requisito não-funcional RNF1 e RNF2 (Ponto 5.2.1.2). Ambos os métodos recorrem à mesma base de dados que armazena a restante informação do sítio (os detalhes da implementação encontram-se no Anexo III).

Carrinho de Compras - Para que o carrinho de compras seja implementado é necessário que os *items* persistam. No sítio *CloudShop* foi utilizada uma combinação de *cookies* e base de dados, de forma a assegurar a persistência necessária. O *cookie* guarda somente um *ID* único (Guid) que identifica o carrinho de compras no sistema. Deste modo o utilizador pode retomar as compras se o *cookie* estiver presente no navegador. O armazenamento dos *items* fica a cargo da base de dados que usa o Guid como chave da tabela vendas. A junção destes dois métodos permite que o carrinho de compras persista mesmo que o utilizador feche o navegador e permite por outro lado, que os detalhes dos *items* não possam ser modificados indevidamente uma vez que estão armazenados em base de dados. Como se optou por: (1) permitir que o utilizador adicionasse *items* ao carrinho sem estar autenticado e (2) que o carrinho de compras persistisse ao término da sessão, é possível que um segundo utilizador consiga ver os *items* do carrinho de compras do primeiro caso partilhem o mesmo navegador. Os detalhes dos produtos são atualizados somente no *checkout* para

prevenir possíveis inconsistências entre o preço aquando da introdução no carrinho de compras e o preço à data da conclusão da venda. Quando a venda é terminada o *cookie* é expirado.

Integração com Motor de Pagamentos - Os métodos mais populares para a compra por via eletrónica são o PayPal e o uso de cartões de crédito. No sítio CloudShop foram habilitados os dois métodos através da integração com um só motor de pagamentos, uma vez que o PayPal permite pagamentos quer pelo uso de crédito da conta PayPal quer através de cartão [102]. Optou-se nesta aplicação por usar a *API SOAP* (*Simple Object Access Protocol*) e implementar o "Express Checkout" (os detalhes da implementação encontram-se no Anexo IV).

5.2.5. Limitações da Solução

As seguintes funcionalidades não foram desenvolvidas por não se acharem relevantes ao propósito da aplicação:

- Processar as encomendas depois do processo de *checkout*. Quando o utilizador completa o processo de *checkout* é enviado um *email* com o *ID* da encomenda e é colocada uma *flag* na base de dados a indicar que o processo foi realizado com sucesso. A partir deste momento não existem mais operações sobre a encomenda.

As seguintes funções apresentam limitações conhecidas:

- As imagens dos produtos podem ser enviadas para o sítio e alteradas consoante a necessidade do administrador. No entanto, cada vez que o administrador quiser associar uma imagem a um produto terá que fazer o *upload* da mesma independentemente do facto de ela já estar presente no servidor ou não;

- A integração com o motor de pagamentos foi feita mas usa um ambiente de testes, a *Sandbox* do Paypal. Seria imperativo trocar o ambiente de testes pelo ambiente de produção caso a aplicação fosse usada para fins comerciais;

- O esquema de autenticação usado no sítio (Forms) é vulnerável a ataques de *reply*. Para tornar o método de autenticação mais robusto deve ser utilizado *SSL/TLS* [103];

- A página de administração dos produtos permite introduzir texto formatado em *HTML* para se poderem formatar parágrafos, introduzir enumerações etc. A introdução deste tipo de dados e posterior envio num formulário *HTML* pode ser visto pelo servidor como uma tentativa de ataque *Cross-Site Scripting* (*XSS*) [104] e

por isso está barrado no ASP.NET. A opção correta para implementar esta funcionalidade e ao mesmo tempo respeitar as medidas de segurança do ASP.NET, é a de integrar um editor *HTML* como o CKEditor [19] ou usar uma sintaxe customizada que faça a conversão entre o texto do utilizador e *tags HTML* [105]. No entanto, para maior brevidade e dado que a página tem controlo de acesso optou-se por desativar a validação de *XSS* para esta página em específico.

- É possível apagar produtos já vendidos, por isso a tabela da base de dados LinhaVenda não está relacionada com a tabela Produto (*Diagrama de Entidade Relacionamento, DER* no Anexo V). A boa prática seria verificar se o produto já tinha sido vendido alguma vez, e em caso afirmativo, colocar-se-ia uma *flag* a indicar que o produto tinha sido "apagado". Se o produto nunca tivesse sido vendido ou seja, não houvesse nenhum registo na LinhaVenda referente àquele produto então podíamos mesmo retirar o produto da base de dados.

5.3. Processo de Migração

Nesta fase procedeu-se à migração do sítio para o Microsoft Windows Azure, a solução de *cloud computing* escolhida. Nesta secção serão apresentadas as principais alterações efetuadas ao sítio para que ele pudesse estar alojado na nuvem.

5.3.1. Criação de Conta no Windows Azure

Para que possamos usar o serviço de *PaaS* da Microsoft necessitamos de efetuar o registo em *https://www.windowsazure.com*. Os pré-requisitos para concluir o registo são:

- Possuir um Windows Live ID[20] que não é mais do que uma combinação de *email* e palavra-passe usada para aceder a outros serviços Microsoft como o Outlook.com[21];

- Ter um número de telefone do país em que foi efetuado o registo para que possa ser enviado um código de confirmação via *Short Message Service* (*SMS*);

- Ter um cartão de crédito VISA, Mastercard ou American Express para que seja cobrado o serviço consoante a utilização.

[19] http://ckeditor.com/, acedido em 1 de Agosto de 2012.
[20] www.passport.net, acedido em 12 de Agosto de 2012.
[21] www.outlook.com, acedido em 7 de Setembro de 2012.

O processo de registo é simples e no final o utilizador é redirecionado para o portal de administração do serviço, onde tem acesso a todos os serviços contratados e às informações de consumo e faturação.

5.3.2. Ferramentas Utilizadas

Para além das ferramentas mencionadas no Ponto 5.2.2 foi empregue o Windows Azure *SDK* 1.5 obtido através da página oficial e o Azure Storage Explorer[22]. O Windows Azure *SDK* permite que o utilizador desenvolva para a solução de *cloud computing* da Microsoft integrando-se automaticamente com o Visual Studio. Providenciando ainda um emulador que permite correr as aplicações localmente como se estivessem a correr numa instância da nuvem. O Azure Storage Explorer é por sua vez uma *GUI* que auxilia na manipulação de tabelas, *BLOB*s e *Queues*. A instalação do *software* foi simples. No final foi necessário configurar a ligação do emulador com o SQL Server com o comando DSInit uma vez que tabelas, *BLOB*s e *Queues* são emuladas através da base de dados.

5.3.3. Migração do Site ASP.NET

A principal diferença entre usar um serviço de *PaaS* em lugar de uma máquina virtual num serviço de *hosting* reside no facto de toda a manutenção ficar a cargo do fornecedor de serviço. O facto da Microsoft monitorizar o estado das instâncias e atuar caso sejam encontrados problemas pode fazer com que uma instância seja trocada de servidor. Deste modo, as aplicações não podem guardar o estado no disco local, tendo de ser usadas outras opções de armazenamento para seja garantida a persistência dos dados. O fornecedor de serviço fica ainda responsável pela preparação da máquina com todos os requisitos necessários para que o utilizador corra a aplicação. Assim, o utilizador deve preocupar-se somente com o desenvolvimento da aplicação e com a sua adaptação à nuvem. Esta secção descreve as principais alterações efetuadas ao sítio *CloudShop* para o adaptar à nuvem.

5.3.3.1. Web Site VS Web Application

Como foi visto no Ponto 4.3.3.1 o serviço de *PaaS* coloca à disposição do utilizador três tipos de instância com propósitos distintos: *Web Role*, *Worker Role* e *VM Role*. O primeiro é o que mais se adequa à loja de comércio eletrónico *CloudShop*. Contudo, o serviço de *PaaS* só suporta projetos do tipo *Web Application* e o projeto da loja é do tipo *Web Site*.

[22] http://azurestorageexplorer.codeplex.com, acedido em 12 de Agosto de 2012.

O projeto Web Site e Web Application partilham o mesmo objetivo principal (criar aplicações *Web*) mas têm características distintas:

Web Site – Num projeto do tipo Web Site os ficheiros são compilados dinamicamente e são publicados no servidor *Web* em aberto (tal como acontece por exemplo com o PHP). Um Web Site é simplesmente um conjunto de ficheiros a publicar não existindo sequer um ficheiro de projeto (extensão *csproj*) associado [106]. Outro pormenor que é necessário conhecer é o facto das classes necessitarem de estar num diretório especial denominado *App_Code* para que possam ser acedidas por várias páginas ASP.NET. Este tipo de projeto tem como principal vantagem tornar o processo de atualização ao sítio mais simples dado que as páginas podem ser modificadas individualmente sem necessitarem sequer de sair do servidor *Web* [106]. Esta característica foi explorada no desenvolvimento da solução *on-premises* uma vez que se ligou o ferramenta de desenvolvimento diretamente ao servidor *IIS*.

Web Application – Num projeto do tipo Web Application o código é pré-compilado numa *DLL* (*Dynamic Link Library*) que é publicada no servidor *Web*. Uma Web Application tem um ficheiro de projeto associado à semelhança de uma aplicação de consola ou de um projeto Windows Forms o que possibilita definir que ficheiros estão incluídos ou não no projeto e, por consequência, que ficheiros vão ser ou não compilados [106]. O tipo de projeto dispõe ainda de um ficheiro *designer.cs* que contém os controlos ASP.NET da página. Ao contrário de um Web Site as classes podem ser colocadas em qualquer diretório sem que isso afete a sua visibilidade para o resto da aplicação.

Optou-se por converter um tipo de projeto para outro. Para o efeito, foi criada uma nova solução no Visual Studio com dois projetos: um Web Application que resultou da conversão do Web Site (documentada no Anexo VI) e um Windows Azure Cloud Service.

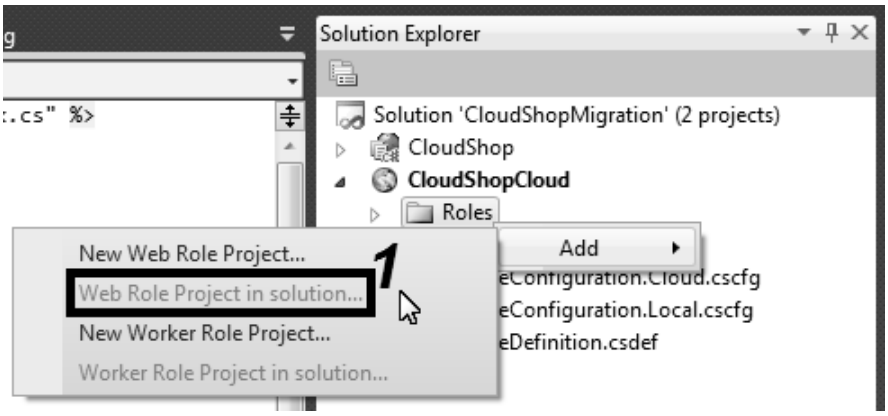


Ilustração 13 - Associar o Windows Azure Cloud Service à Web Application

Por fim foi associado um projeto ao outro através da opção "*Add New Web Role Project in solution...*" (Ilustração 13).

5.3.3.2. Migração da Base de Dados

Inicialmente foi averiguado se existia alguma limitação no SQL Azure que implicasse uma alteração na base de dados (*DER* no Anexo V) [107]. Não foi encontrada nenhuma limitação que afetasse o sítio portanto, partiu-se para a exportação da estrutura e dos dados através do *SSMS*. O *SSMS* dispõe de suporte ao serviço de base de dados na nuvem a partir da versão 2008 R2 [108] a mesma versão em que o motor se baseia. Este suporte mostra o esforço feito pela Microsoft no sentido de tornar o serviço de base de dados na nuvem semelhante à sua solução *on-premises* o que lhe confere uma curva de aprendizagem menor.

Para se exportar a base de dados foi gerado um ficheiro *SQL* que posteriormente foi importado na nuvem (Ilustração 14), o ficheiro *SQL* pode ser criado especificamente para o SQL Azure através da alteração da opção "*Script for the database engine type*" (2).

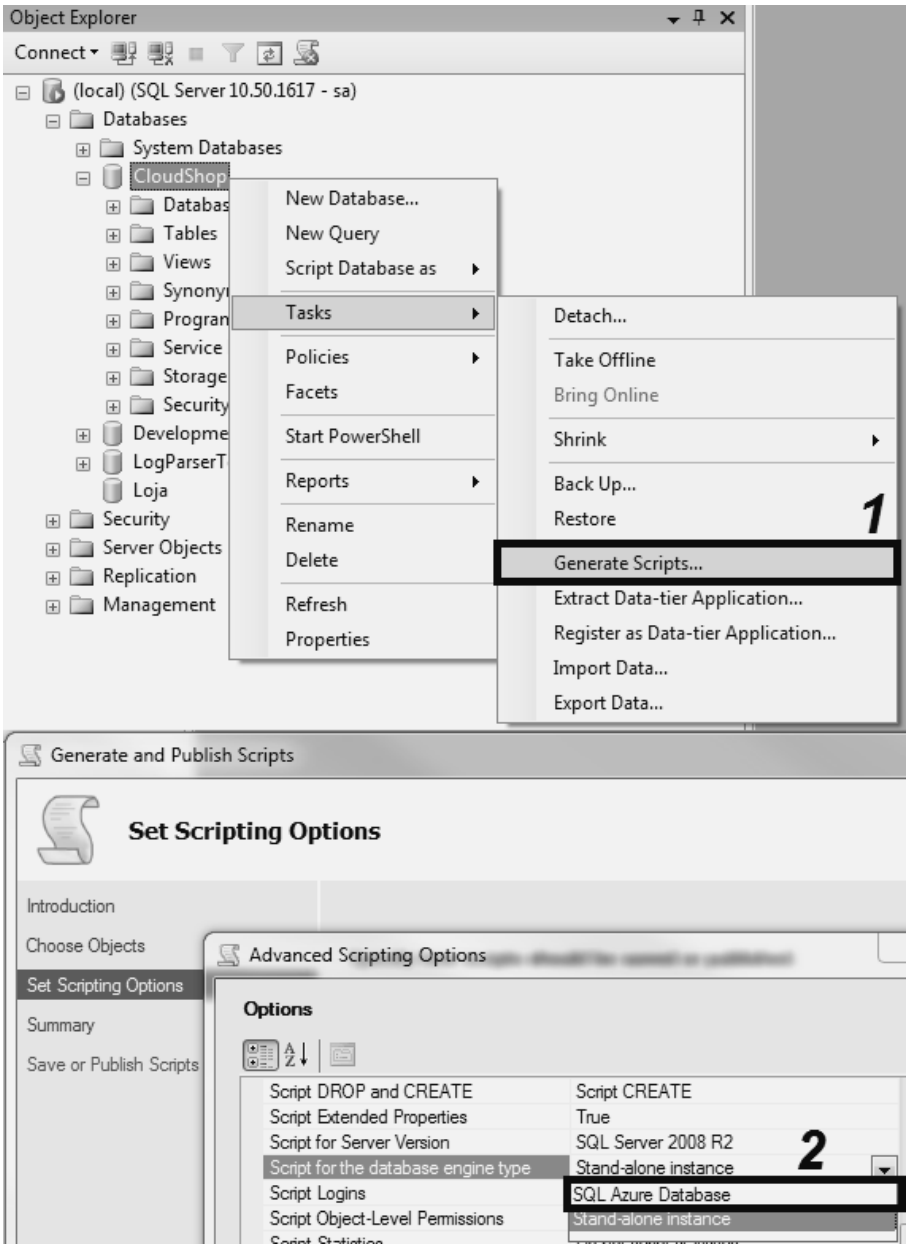


Ilustração 14 - Gerar ficheiro SQL de criação da base de dados através do SSMS

Finda a criação do ficheiro foi averiguado o tamanho da base de dados *on-premises* para que pudesse ser criada uma base de dados na nuvem com um tamanho máximo adequado. O tamanho da base de dados do sítio é, à data de escrita, 4*MB* tendo 3 clientes e 10 produtos agrupados em 7 marcas e 19 categorias e sub-categorias. Esta dimensão é facilmente acomodada na base de dados mais pequena do SQL Azure: uma base de dados do tipo WEB com o tamanho máximo de 1*GB*.

Seguidamente são enumerados os passos necessários para a criação do servidor de *BD* no Windows Azure através do portal de administração:

- Criar novo servidor de base de dados e definir a região no qual vai estar alojado (foi escolhido *West Europe*);

- Definir credenciais de acesso (utilizador e palavra-passe);

- Configurar regras de *firewall* de modo a habilitar o acesso ao servidor por parte de um determinado *IP* (Internet Protocol, RFC#791) ou gama de *IP*s. Somente os pedidos provenientes deste(s) *IP*(s) público(s) são atendidos pelo servidor;

- Definir se outros serviços dentro da nuvem têm acesso ao servidor.

Quando estes passos forem concluídos é devolvido um endereço com o formato apresentado na Listagem 3 que será usado para a ligação do *SSMS* à nuvem.

```
<nome gerado para o servidor>.database.windows.net
```

Listagem 3 - Formato dos endereços gerados para acesso aos servidores SQL Azure

A ligação do *SSMS* a este endereço permitiu que a base de dados fosse criada e que a estrutura e os dados fossem importados através do ficheiro *SQL* gerado. Para completar a migração da base de dados bastou alterar a *ConnectionString* que é usada na classe *GenericDataAccess*.

5.3.3.3. Mudança de Ficheiro de Configuração

Uma aplicação local necessita unicamente de um ficheiro de configuração, o *Web.config*. Um projeto do tipo Windows Azure Cloud Service introduz dois ficheiros de configuração adicionais: *ServiceDefinition.csdef* e *ServiceConfiguration.cscfg*. O primeiro define os tipos de instância usados pela aplicação e os endereços e protocolos de acesso à mesma (vulgarmente designados de *endpoints*) [109]. Define ainda os elementos de configuração que estão associados à instância e que podem ser atribuídos no segundo ficheiro, *ServiceConfiguration.cscfg*. O ficheiro *ServiceConfiguration.cscfg* para além de atribuir um valor de configuração aos elementos definidos, define o número de instâncias de cada tipo [109].

Os ficheiros *ServiceDefinition.csdef* e *Web.config* não podem ser alterados enquanto a aplicação está em execução [83] na nuvem. Configurações como o número de produtos por página (*CloudShopConf.ProductsPerPage*) ou o número de carateres da descrição (*CloudShopConf.ProductShortDescLength*) só fazem sentido se puderem ser afinados ao

longo da execução do sítio. Por esta razão os valores foram migrados do ficheiro *Web.config* para o ficheiro *ServiceConfiguration.cscfg*, o que permite que sejam alterados através do portal de administração ou através do comando *CSRun* enquanto a aplicação está em execução. Se as configurações fossem, todavia, mantidas no ficheiro *Web.config* seria necessário efetuar uma nova instalação da aplicação (*deployment*) para alterar os valores.

A existência da classe de acesso aos ficheiros de configuração (Ponto 5.2.4) tornou a aplicação agnóstica à origem dos valores. Deste modo, quando foi efetuada a troca de ficheiros de configuração as alterações ficaram circunscritas à classe *CloudShopConf*. De modo a minimizar o esforço caso a aplicação voltasse a correr localmente, os valores de configuração não foram retirados do ficheiro *Web.config* mas sim duplicados. Antes da propriedade ser lida, o programa verifica se está a correr no Windows Azure através da verificação da propriedade *RoleEnvironment.IsAvailable*. Em caso afirmativo, retorna a configuração do novo ficheiro (*ServiceConfiguration.cscfg*) caso contrário retorna a configuração do ficheiro *Web.config* (Listagem 4).

```
public static int ProductsPerPage { get { return RoleEnvironment.IsAvailable ?
Int32.Parse(RoleEnvironment.GetConfigurationSettingValue("ProductsPerPage")) :
Int32.Parse(ConfigurationManager.AppSettings["ProductsPerPage"]); } }
```

Listagem 4 - Propriedade da classe de acesso aos ficheiros de configuração

5.3.3.4. Redireccionamento de URLs para o Load Balancer

Na arquitetura do Windows Azure nenhum pedido chega diretamente da *Internet* para as instâncias que suportam a aplicação [83]. De facto, todos os pedidos passam pelo serviço de distribuição de carga (*load balancer*) que tem como principal função distribuir a carga pelas máquinas que fazem parte da *pool*. O *load balancer* tem um papel crucial na alta disponibilidade e na tolerância a falhas da aplicação uma vez que, retira instâncias da *pool* quando estas estão a ser atualizadas ou quando têm problemas e desvia o tráfego para outras que asseguram o serviço.

Um dos primeiros problemas com que a aplicação *CloudShop* foi confrontada no processo de migração está relacionado com o facto das instâncias não estarem diretamente expostas à *Internet*. Existe na aplicação uma classe de nome *LinkFactory* que é responsável pela criação de *links* dinamicamente. Os *links* criados através desta classe apontavam para o porto que está à escuta na instância e não no *load balancer* (Ilustração 15). Quando o

utilizador solicitava um desses *links* era pedida a página ao *load balancer* num porto não reconhecido o que levava o *load balancer* a retornar um erro *HTTP* 400. Este problema foi observado logo nos testes com o emulador tendo sido resolvido pela chamada à propriedade *HttpContext.Current.Request.Headers["Host"]* em vez da chamada à propriedade *HttpContext.Current.Request.Url.Host* e *HttpContext.Current.Request.Url.Port* na classe *LinkFactory*.

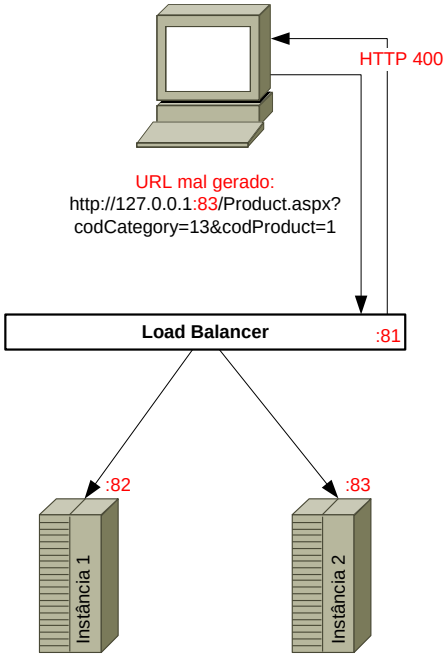


Ilustração 15 - URL mal gerado na classe LinkFactory

5.3.3.5. Ficheiros Locais

Como o armazenamento local é volátil devem ser utilizados métodos alternativos para garantir a persistência da informação. Estes métodos devem assegurar não só a persistência dos dados mas também que todas as instâncias da aplicação tenham visibilidade sobre os mesmos. Na aplicação *CloudShop* as imagens dos produtos necessitam de ser armazenadas de forma persistente. Antes da migração as imagens dos produtos eram enviadas para o sítio segundo o fluxograma apresentado na Ilustração 16.

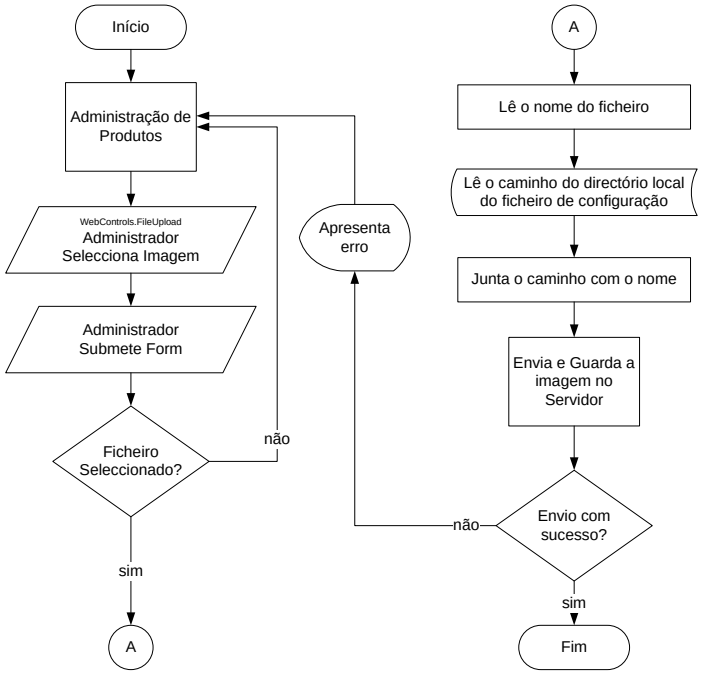


Ilustração 16 - Fluxograma do envio de imagens antes da migração

O processo era simples o administrador usava um controlo do ASP.NET para selecionar a imagem que pretendia, submetia o formulário e a imagem era enviada para o servidor. O caminho do directório local era dado pela propriedade *ProductImagesDirectory* da classe de acesso ao ficheiro de configuração. O diretório para se guardarem as imagens *full-size* e *thumbnails* era único, facto que foi mudado aquando da migração para a nuvem.

Para tornar o armazenamento das imagens persistente optou-se por usar o serviço de *BLOB*s do Windows Azure. Para isso foi criada uma conta de armazenamento através do portal de administração [110] do Windows Azure tendo sido escolhido West Europe para a sub-região e *cloudshopdatastore* para o nome da conta. Os *URL*s para se acederem aos serviços são baseados no nome da conta e no tipo de serviço tendo o formato apresentado na Listagem 5.

```
<http|https>://<nome da conta de armazenamento>.<nome do serviço>.core.windows.net
```

Listagem 5 - Formato dos URLs gerados pela conta de armazenamento

Para que uma aplicação seja integrada com a conta de armazenamento devem ser providenciados quer o nome da conta, quer as chaves de acesso que podem ser obtidas no portal de administração. Esta informação deve ser guardada num ficheiro de configuração para que possa ser alterada com facilidade no futuro. Assim, a informação de acesso à

conta de armazenamento pode ser acedida na aplicação *CloudShop* através da propriedade *MyDATAConnString* da classe de acesso ao ficheiro de configuração.

O envio das imagens passou a ser efetuado segundo o fluxograma apresentado na Ilustração 17.

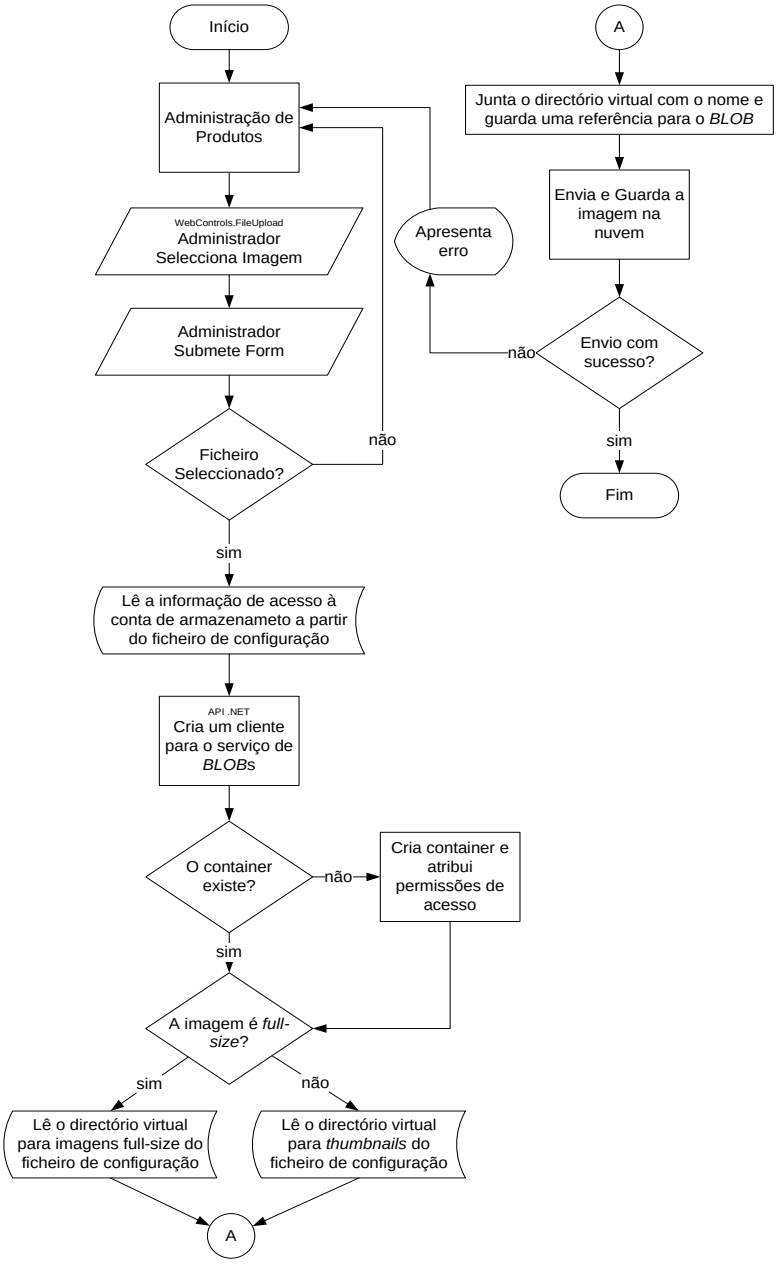


Ilustração 17 - Fluxograma do envio de imagens depois da migração

Quando o formulário é submetido é criado um cliente do serviço de *BLOB*s do Windows Azure recorrendo à informação de acesso armazenada no ficheiro de configuração e à API .NET. Através do cliente é verificado se o *container* (abstração que permite agrupar os

*BLOB*s) existe, se não existir é criado e são-lhe atribuídas permissões de acesso. Por omissão, um *container* é privado o que não se adequa ao propósito que queremos. As permissões de acesso público ao *container* devem ser alteradas sendo que os valores possíveis estão presentes na enumeração *BlobContainerPublicAccessType* [111]:

- **Blob –** Com este tipo de permissão é possível a um utilizador anónimo ler o conteúdo e os metadados associados a qualquer *BLOB* dentro de um determinado *container*. É contudo negada a leitura dos metadados associados ao *container* bem como a listagem dos seus conteúdos.

- **Container –** Todas as permissões do anterior sem as restrições de listagem e da leitura dos metadados ao nível do *container*.

- **Off –** Não são permitidos acessos anónimos.

O valor escolhido para o armazenamento de imagens do sítio foi *container*. Dentro de um *container* não é possível criar diretórios no entanto, é possível criar uma estrutura hierárquica ao acrescentar um caminho virtual ao nome do ficheiro [112]. Por exemplo, na aplicação *CloudShop* é acrescentado o caminho dado pela propriedade *FSImagesVirtualFolder* ao nome da imagem, se a imagem se chamar "*image001.png*" o nome do *BLOB* a gravar no *container* será "*FSimages/image001.png*", na base de dados é somente guardado o nome original "*image001.png*". Esta técnica permite filtrar as imagens dentro do *container* em imagens *full-size* (guardadas no diretório virtual *FSimages*) e *thumbnails* (guardadas no diretório virtual *TNimages*). O facto de se guardar o nome original na base de dados permite que a alteração da conta de armazenamento fique circunscrita ao ficheiro de configuração e à alteração da informação de acesso. Para completar o processo a imagem é enviada para a nuvem passando a estar disponível em todas as instâncias da aplicação e a beneficiar da escalabilidade, alta disponibilidade e tolerância a falhas que caracterizam o serviço.

Para se evitar realizar operações na *API* sem ser necessário instanciar um cliente da conta de armazenamento sempre que é apresentada uma imagem, foram criados dois métodos na classe *LinkFactory*: *ToProductImageFS* e *ToProductImageTN*. Estes métodos permitem construir o *URL* completo para o *BLOB* usando para o efeito a propriedade *BLOBImageDirectory* que armazena o caminho para o serviço de armazenamento de *BLOB*s segundo a convenção apresentada na Listagem 5.

5.3.3.6. Transferência de Informação entre Páginas

O protocolo *HTTP* é um protocolo *stateless*, pelo que, por omissão, todos os pedidos feitos ao servidor *Web* são independentes e não guardam estado acerca dos pedidos anteriores. Para tornar a informação persistente entre pedidos deve ser empregue um dos métodos da Tabela 10.

Método	Compatível com o Windows Azure	Comentários
Cookies	Sim	Como os cookies são guardados no cliente e requisitados pelo servidor mesmo que a instância mude, o valor continua a conseguir ser lido.
Variáveis de Sessão	Não	As variáveis de sessão encontram-se guardadas na memória do servidor: se num pedido for chamado um servidor diferente daquele onde foi guardado o valor, o valor encontra-se indisponível.
Query String	Sim	Os valores são enviados junto com o URL da página sob a forma variável=valor.
Cross-Page Posting	Sim	Este método usa a propriedade PostBackUrl do botão de submeter o formulário. Os valores do formulário são enviados por POST para a página de destino.
Server.Transfer	Sim	Se for utilizado o evento Click do botão de submeter e o pedido for redireccionado para outra página é importante ter em conta que deve ser usado o método Server.Transfer em vez do Response.Redirect que não preserva o formulário. Para preservar o formulário, o segundo parâmetro do Server.Transfer deve ser true.

Tabela 10 - Métodos para fazer persistir informação entre páginas ASP.NET

Durante os testes efetuados verificou-se que as variáveis de sessão não funcionavam no Windows Azure quando era usada mais do que uma instância. De facto, quando um utilizador acede ao sítio através de um navegador é-lhe atribuído um *ID* de sessão que é enviado em pedidos posteriores através de um cookie. O *ID* de sessão serve para fazer o mapeamento entre o utilizador e as variáveis guardadas no servidor a que damos o nome de variáveis de sessão. Por omissão (Listagem 6) as variáveis de sessão são guardadas na memória do processo responsável por hospedar a aplicação ASP.NET [83] ou seja, as variáveis só estão acessíveis na instância em que foram definidas.

```
<sessionState mode="InProc">
</sessionState>
```

Listagem 6 - Configuração do armazenamento das variáveis de sessão, ficheiro Web.config

Como dois pedidos consecutivos podem ser encaminhados pelo *load balancer* para duas instâncias diferentes é importante que estas possam aceder a armazenamento partilhado para que tenham acesso às mesmas variáveis de sessão. No sítio *CloudShop* optou-se pelo uso de base de dados para o armazenamento das variáveis de sessão.

O ASP.NET já dispunha de um *Provider* integrado para o armazenamento de sessões em bases de dados SQL Server que embora pudesse ser utilizado com o SQL Azure não era oficialmente suportado e apresentava alguns problemas como não apagar automaticamente as sessões expiradas [113]. Em 2011 foi lançado um conjunto de *Providers* sob o nome de *System.Web.Providers* com o propósito de estender o suporte oficial ao SQL Azure e ao SQL Compact Edition. Foram estes os *Providers* utilizados na aplicação *CloudShop*. A versão utilizada foi a 1.2 que pode ser obtida através do gestor de pacotes do Visual Studio [114], findo o download do pacote basta adicionar uma referência a "*System.Web.Providers*" para que a ferramenta de desenvolvimento se encarregue da adequação dos ficheiros de configuração.

```
<sessionState mode="Custom" customProvider="DefaultSessionProvider">
   <providers>
      <add name="DefaultSessionProvider"
type="System.Web.Providers.DefaultSessionStateProvider, System.Web.Providers,
Version=1.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35"
connectionStringName="SQLServerConnectionString" applicationName="CloudShop"/>
   </providers>
</sessionState>
```

Listagem 7 - Configuração do armazenamento das variáveis de sessão, definição da ConnectionString no ficheiro Web.config

O utilizador deve providenciar a *ConnectionString* para a base de dados (Listagem 7), assim da próxima vez que a aplicação for executada é criada a tabela *Sessions* permitindo assim, o uso de variáveis de sessão na base de dados tal qual elas estivessem armazenadas na memória do servidor.

5.3.4. Integração com o Visual Studio e Processo de Deployment

Primeiro é necessário criar os serviços e configurá-los no portal de administração e só de seguida são ligados à ferramenta de desenvolvimento. Para hospedar a loja de comércio eletrónico foi criado um serviço com o nome *cloudshop* e foi escolhido um prefixo o mais semelhante possível ao nome escolhido. O *URL* gerado (Listagem 8) foi *cloud-*

shop.cloudapp.net sendo este o caminho que os eventuais utilizadores da aplicação devem usar para lhe aceder. A região seleccionada foi a mesma escolhida para a base de dados e para a conta de armazenamento pois, para além da latência ser menor dentro da mesma região os dados transferidos entre serviços não são taxados o que permite reduzir custos.

```
<prefixo escolhido>.cloudapp.net
```

Listagem 8 - Formato dos URLs gerados pela conta de computação

Optou-se por não efetuar nenhuma instalação (*deploy*) para a nuvem aquando da criação pois esta vai ser feita quando o Windows Azure estiver integrado com a ferramenta de desenvolvimento. Para se integrar a ferramenta de desenvolvimento com o Windows Azure é necessário criar um par de chaves, o uso da criptografia assimétrica permite que a *API* de Gestão verifique a autenticidade e integridade dos pedidos feitos pelo Visual Studio (Anexo VII). No final da integração basta usar o menu de contexto do projeto Windows Azure Cloud Service para que as instâncias sejam criadas de acordo com a configuração e a aplicação seja instalada na nuvem (*deploy*). As instâncias criadas passam também a aparecer no "*Server Explorer*" sendo que, é possível aceder-lhes por *RDP* mediante a configuração de uma conta de acesso [115].

5.4. Custo da Solução na Nuvem

Quando a solução ficou pronta foi calculado o seu custo mensal em *USD*. Todavia, o custo não é fixo variando mediante a procura à aplicação. Uma das premissas da nuvem é a de só se pagar pelos recursos utilizados assim, se o número de acessos aumenta os recursos disponibilizados também aumentam sendo o custo associado transferido para o utilizador. Este é um modelo justo pois o acréscimo do número de acessos também significa um acréscimo no número de potenciais clientes e na receita gerada pela aplicação. A Tabela 11 resume os cálculos feitos para se chegar ao custo mensal da aplicação. Não estão abrangidos no cálculo os custos da transferência de dados para a Internet (0,12 *USD* por *GB*, Tabela 5) e o custo das operações na conta de armazenamento (0,01 *USD* por 100000 operações). Está ainda de fora a possibilidade do número de instâncias variar devido à implementação da escalabilidade automática.

Qty	Descrição	Horas	Dias	Custo /Hora	Custo /Dia	Custo /Mês	Sub-Total
2	Medium Web Role	720		0,240		172,80	345,600
1	Base de Dados 1GB					9,990	9,990
1	Conta de Armazenamento 2GB *					0,250	0,250
2	Instâncias Monitorizadas no AzureWatch (Ponto 7.1)		30		0,330	9,900	19,800
						Total:	**375,515**

* Custo de armazenamento mensal inclui:
500MB para *logs* que são armazenados em tabelas para a escalabilidade automática (Max. 666666 registos de métricas [116])
500MB para as imagens dos produtos que são armazenadas em *BLOB*s
1GB para a comunicação com a *thread* responsável pelas tarefas de *background* (Ponto 6.2) em que são usadas tabelas e *queues* (Max. 400000 mensagens [116])

Tabela 11 - Custo da aplicação CloudShop no Windows Azure

5.5. Recomendações

Neste secção efetuam-se algumas recomendações orientadas para a migração de aplicações para a nuvem. As recomendações são fruto quer da experiência adquirida ao migrar a aplicação *CloudShop* quer da recolha das melhores práticas.

- Se o projeto usado é do tipo Web Site convertê-lo para Web Application (Ponto 5.3.3.1);

- Usar a propriedade *RoleEnvironment.IsAvailable* sempre que se pretenda isolar código específico da nuvem e a propriedade *RoleEnvironment.IsEmulated* para isolar código específico do emulador;

- Utilizar o emulador para certificar que a aplicação está a funcionar corretamente [117] ;

- Certificarmo-nos que a aplicação é compatível com uma arquitetura de 64 bits como a do Windows Azure [117];

- Configurar pelo menos duas instâncias no emulador para detetar eventuais problemas relacionados com o facto da aplicação estar balanceada por vários servidores (Ponto 5.3.3.4);

- Mover as configurações que necessitam de ser alteradas enquanto a aplicação está em execução do ficheiro *Web.config* para o ficheiro *ServiceConfiguration.cscfg* (Ponto 5.3.3.3);

- Evitar o uso de armazenamento local migrando a informação para armazenamento persistente como base de dados, tabelas ou *BLOB*s (Ponto 5.3.3.5);

- Se o projeto *on-premises* usa uma base de dados SQL Server, usar a versão 2008 R2 e o Management Studio para exportar a estrutura e os dados num formato compatível com o SQL Azure (Ponto 5.3.3.2). Se for usada uma versão diferente do SQL Server o script exportado tem que ser alvo de uma série de modificações para que se torne compatível com o SQL Azure [83];

- Usar os *System.Web.Providers* em alternativa aos *Providers* integrados no ASP.NET se se pretender usar a base de dados SQL Azure (Ponto 5.3.3.6);

- O uso do *caching* do ASP.NET deve ser substituído pelo serviço de *caching* do Windows Azure [117];

- Se o *Web Role* comunicar com um *Worker Role* essa comunicação deve feita recorrendo a *queues* [117];

- Se a aplicação necessitar de executar tarefas bloqueantes ou tarefas em *background* [117] utilizar um *Worker Role*;

- Todas as aplicações hospedadas terminam em *cloudapp.net* não estando associadas ao domínio da instituição a que pertencem. Para esse efeito adicionar um registo *CNAME* ao servidor de *DNS* a apontar para o *URL cloudapp.net*;

- Configurar tarefas de arranque no ficheiro *ServiceDefinition.csdef* se for necessário instalar componentes ou executar quaisquer outras ações que tenham em vista a preparação do servidor virtual para hospedar a aplicação [117] ;

A conceção da aplicação *CloudShop* demorou cerca de três meses ao passo que a migração demorou um mês. O tempo de migração foi 1/3 do tempo de desenvolvimento no entanto, há a convicção de que este rácio poderia ser melhorado caso as recomendações acima apresentadas fossem conhecidas antecipadamente. Tratou-se também da primeira abordagem do autor à temática do *cloud computing* e ao uso do Windows Azure em particular.

6. Otimização de Custos em Ambiente Windows Azure

Este capítulo foca algumas das boas práticas para o uso eficiente de recursos e consequente controlo de custos no ambiente Windows Azure.

6.1. Guardar Variáveis de Sessão na Base de Dados

Como foi visto no ponto 5.3.3.6 por omissão as variáveis de sessão só estão acessíveis na instância em que foram definidas. Para contornar este comportamento é necessário utilizar armazenamento partilhado por todas as instâncias da aplicação. No que toca ao sítio *CloudShop*, existiam várias opções de armazenamento partilhado que poderiam ter sido utilizadas. O serviço de *caching* do Windows Azure pareceu, à partida, a opção mais favorável por ser previsivelmente mais rápido. O facto de se basear numa *cache* de memória fazia prever que se destacasse de outras opções baseadas em base de dados, tabelas ou *BLOB*s. Para o comprovar foram executados testes do qual resultou o gráfico da Ilustração 18.

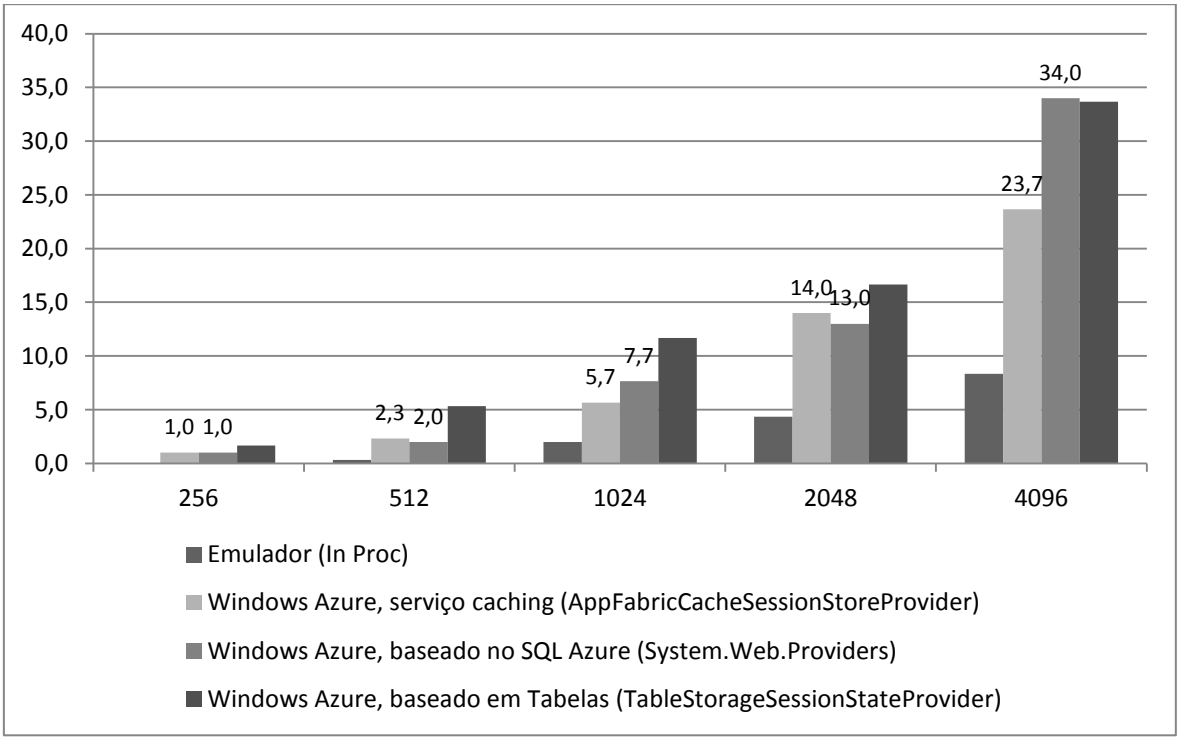


Ilustração 18 - Tempos de acesso às variáveis de sessão usando vários tipos de armazenamento

O teste consistiu no registo do tempo de escrita e leitura de 256, 512, 1024, 2048 e 4096 variáveis de sessão. Os tempos registados em milissegundos representam a média de 3 execuções consecutivas usando cada uma das opções de armazenamento (memória do processo que hospeda a aplicação, serviço de caching [118], base de dados[114] e tabelas [119] do Windows Azure). Como se pode verificar o serviço de *caching* foi de facto a solução de armazenamento partilhado mais rápida, sendo contudo a mais dispendiosa (Tabela 6). Acresce-se que a diferença de desempenho não foi tão notória como seria de esperar, apenas se fazendo notar no caso de 4096 variáveis de sessão em simultâneo. Deste modo, o uso da *cache* de memória só é justificado caso a aplicação faça uso intensivo de variáveis de sessão ou use a memória para outros efeitos. Se a aplicação não se enquadra neste perfil de utilização e já dispõe de uma base de dados SQL Azure então o uso do *System.Web.Providers* providencia um desempenho semelhante sem introduzir quaisquer custos adicionais à solução (Ponto 5.3.3.6).

6.2. Worker Role e Web Role na mesma Máquina

Algumas tarefas da aplicação *CloudShop* necessitam de processamento em *background* (e.g. envio de emails), pelo que se pensou em criar uma instância do tipo *Worker Role*. Contudo, o facto de se necessitar de pelo menos duas instâncias de cada tipo para a garantia do *SLA* leva a que sejam procuradas outras alternativas.

Tal como foi mencionado no Ponto 4.3.3.1 a principal diferença entre um *Web Role* e um *Worker Role* é que um tem o servidor *IIS* integrado e o outro não. O facto de serem semelhantes levou a crer que fosse possível hospedar as tarefas de *background* no mesmo servidor onde estava alojado o sítio [120]. Deste modo, a lógica foi implementada num *Web Role* da mesma forma que seria implementada num *Worker Role* ou seja, através da criação de um ciclo infinito no método *Run*. Embora o método *Run* não esteja presente no modelo de projeto *Web Role* pode ser implementado da mesma forma [121] sendo chamado pela plataforma Windows Azure no processo de arranque. A implementação deste método permitiu que fosse criado um falso *Worker Role* que corre num processo distinto do sítio [122] (o *RoleEntryPoint* corre no processo *WaIISHost.exe* ao passo que o sítio corre no processo *w3wp.exe*). O funcionamento do código implementado é o seguinte (Ilustração 19):

Quando existe uma mensagem de email para enviar o sítio guarda os campos da mensagem numa tabela (1), depois coloca uma mensagem na *queue* contendo o tipo de operação a realizar (Enumeração *TypeOfAction*) e a chave do registo na tabela (2). O uso da *queue* permite notificar o falso *Worker Role* de que existem tarefas para realizar. Como compete ao falso *Worker Role* fazer o *pulling* dos dados há ainda o beneficio acrescido das tarefas serem corretamente balanceadas pelas múltiplas instâncias da aplicação.

Quando o falso *Worker Role* encontra uma tarefa na *queue* (3) verifica o tipo de operação pretendida, de seguida retira da mensagem a chave que lhe permite encontrar o registo associado. Os campos necessários são retirados da tabela (4) e o email é enviado (5), no final os dados são removidos. A *thread* faz nova iteração ao ciclo e verifica se existem mais tarefas para processar. Se existirem o processo repete-se caso contrário, a *thread* dorme pelo período especificado pelo algoritmo de *backoff*. O algoritmo de *backoff* permite minimizar o número de acessos uma vez que todas as operações de acesso à *queue* são contabilizadas.

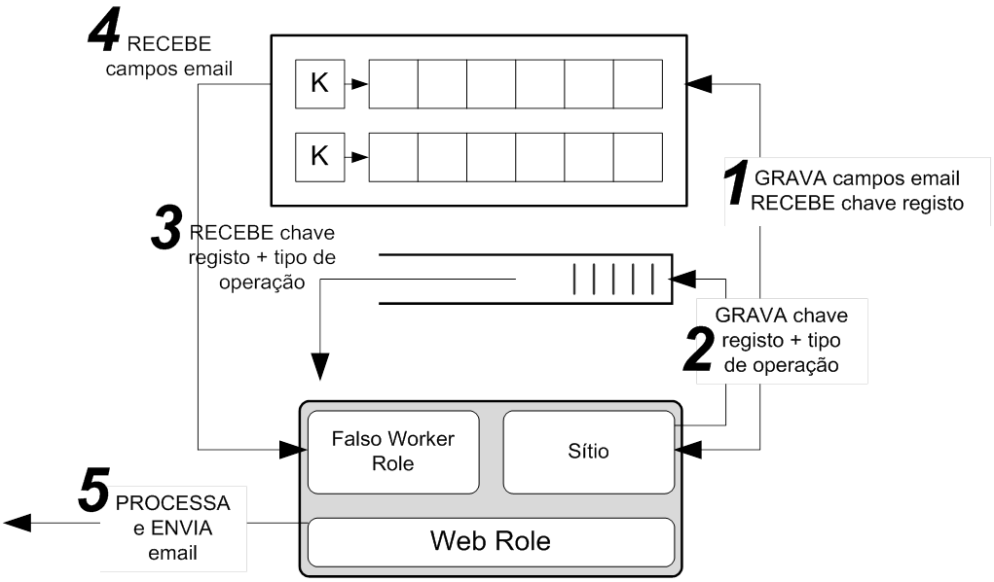


Ilustração 19 - Funcionamento do falso Worker Role

Conseguiu-se desta forma que as tarefas de *background* fossem enviadas para um processo isolado que não bloqueia o sítio, a própria forma como foi feita a interligação entre o sítio e o falso *Worker Role* promoveu a independência das duas funcionalidades. Assim, se no futuro, o utilizador necessitar de mais instâncias que corram as tarefas de *background* mas não precisar de mais instâncias do sítio pode facilmente, migrar o código para um *Worker*

Role real e passar a dimensionar as duas funcionalidades de forma independente. Até lá o utilizador pode beneficiar da economia de ter somente duas instâncias em vez de quatro.

6.3. Zonas Geográficas

A quantidade de dados transferidos é um dos custos associados à nuvem (Tabela 5). Este custo aplica-se inclusive a dados transferidos entre dois serviços do Windows Azure se estes necessitarem de "atravessar" a *Internet* [35]. Uma medida óbvia para não se incorrer em custos de transferência entre componentes da solução é alojá-los no mesmo *datacenter*. Se o utilizador escolher a mesma sub-região para todos os serviços que utiliza, os serviços serão alojados no mesmo *datacenter*, eliminando-se o custo da transferência, para além do beneficio de uma menor latência.

Para além de poder escolher a sub-região em que o serviço é inserido o utilizador pode criar um Affinity Group. O Affinity Group deve ser utilizado sempre que esteja disponível, dado permite não só, que o serviço seja alojado no mesmo *datacenter*, como também que seja alojado o mais próximo possível de outros serviços do utilizador [123], possivelmente no mesmo contentor e no mesmo bastidor.

7. Falácias da Computação na Nuvem

Algumas das características que tornam a computação na nuvem distinta de uma infraestrutura convencional não corresponderam ao esperado. Este capítulo apresenta uma análise critica a essas características.

7.1. Elasticidade Automática

É verdade que a nuvem coloca ao dispor do utilizador quer escalabilidade horizontal quer escalabilidade vertical mas, o facto da aplicação ser migrada para a nuvem não garante por si só que esta seja elástica. Os benefícios da elasticidade só são realmente alcançados através de um desenho cuidado da aplicação que alcance a independência entre os componentes da mesma, que evite tarefas bloqueantes e preveja mecanismos de prevenção e resposta à saturação do acesso a dados.

A adequação do número de instâncias face a picos de utilização necessita de intervenção manual por isso a nuvem deve ser vista como um potenciador da elasticidade e não como uma solução. No Windows Azure quando um servidor está a chegar aos limites o utilizador pode lançar um servidor adicional sem interromper o serviço através da alteração do número de instâncias no ficheiro *ServiceConfiguration.cscfg* (escalabilidade horizontal). Do mesmo modo, o utilizador pode optar por escolher uma instância de tamanho superior (Tabela 2) no ficheiro *ServiceDefinition.csdef* (escalabilidade vertical) se a sua aplicação não estiver preparada para ser distribuída. Estas são duas formas de aumentar os recursos da aplicação mas, ambas requerem intervenção manual por parte do utilizador, o que pode não ser viável. O facto da escalabilidade necessitar de intervenção manual mete em causa dois dos principais benefícios associadas à nuvem: a adequação dos recursos à carga exigida, também designada de elasticidade, e a redução do esforço em tarefas de manutenção.

Embora a escalabilidade automática não esteja implementada de raiz no Windows Azure é passível de ser alcançada. O primeiro passo para a escalabilidade automática é recolher um conjunto de métricas das instâncias que, depois de tratadas indicam se a aplicação está sob carga. A recolha dessas métricas pode ser feita através do Windows Azure Diagnostics Monitor[124], um agente que permite a recolha de informação de diagnóstico, tais como, *logs* do *IIS* ou eventos do Windows. O agente *MonAgentHost.exe* é a solução empregue

para que o processo de diagnóstico de uma aplicação na nuvem se aparente ao processo de diagnosticar *on-premises*. Para o efeito, o agente copia os *logs* guardados no armazenamento local das máquinas e centraliza-os numa conta de armazenamento providenciada pelo utilizador. Os *logs* são exatamente os mesmos encontrados num servidor *on-premises*. Deste modo, o utilizador do Windows Azure pode analisá-los da mesma forma. O Diagnostics Monitor já estava a ser utilizado pela aplicação CloudShop para encaminhar as mensagens geradas pelo *System.Diagnostics.Trace* para a tabela "*WADLogsTable*" da conta de armazenamento porém não estavam a ser recolhidas quaisquer métricas relativas à máquina. Para se ativar a recolha das métricas é necessário adicionar o trecho de código (Listagem 9) ao método *OnStart* do tipo de instância usado, neste caso *Web Role*. O código cria uma instância da configuração por omissão do Windows Azure Diagnostics Monitor [124], depois define a métrica a recolher e a frequência de amostragem. É ainda definida a periodicidade do envio dos dados recolhidos para a conta de armazenamento.

```
//Use diagnostic monitor initial configuration, nothing is transferred by default
all is buffered locally
DiagnosticMonitorConfiguration dmc =
DiagnosticMonitor.GetDefaultInitialConfiguration();

(...)

//Performance counter, we're able to get more counter names using the command
typeperf.exe /q
PerformanceCounterConfiguration pcc = new PerformanceCounterConfiguration();
pcc.CounterSpecifier = @"\Processor(_Total)\% Processor Time";
pcc.SampleRate = System.TimeSpan.FromSeconds(5);
dmc.PerformanceCounters.DataSources.Add(pcc);
dmc.PerformanceCounters.ScheduledTransferPeriod = TimeSpan.FromSeconds(15);
```

Listagem 9 - Código para habilitar a recolha de métricas na instância

O código da Listagem 9 despoleta a exportação das métricas seleccionadas para a tabela "*WADPerformanceCountersTable*" com a periodicidade definida pela propriedade *ScheduledTransferPeriod*. O segundo passo para a escalabilidade automática é analisar as métricas armazenadas e desencadear ações consoante essa mesma análise. Para adequar os recursos à carga exigida, o utilizador deve usar a *API* de Gestão do Windows Azure. Contudo, o esforço necessário para criar uma aplicação deste tipo é muito grande [83] pois, requer o desenvolvimento de um conjunto de funcionalidades. As funcionalidades a implementar incluem, entre outras: a definição das métricas a recolher; a monitorização da

conta de armazenamento para onde as métricas são exportadas; a agregação e análise dos dados recolhidos; e a alteração dos recursos disponibilizados à aplicação de acordo com a análise feita ou mediante agendamento. O utilizador pode optar por utilizar ferramentas já desenvolvidas como o *WASABi* (*Autoscaling Application Block*) da Microsoft [125] ou ferramentas de terceiros no modelo *SaaS* como o RightScale[23], Opstera[24] ou o Paraleap AzureWatch[25]. Qualquer uma destas ferramentas necessita de estar em execução para que detete picos de carga e altere os recursos disponibilizados. Deste modo, não são desencadeadas ações caso a ferramenta esteja indisponível porque, apesar das métricas continuarem a ser exportadas para a conta de armazenamento não são alvo de nenhuma análise. Se for escolhido o *WASABi* é o utilizador que tem a responsabilidade de o hospedar *on-premises* ou na nuvem. A escolha do *WASABi* implica também um grande esforço de configuração [126] sendo que a ferramenta não disponibiliza o mesmo tipo de interface encontrada nas soluções de terceiros que dispõem de gráficos em tempo real, relatórios das métricas e formulários que auxiliam na definição das métricas recolhidas e das regras que desencadeiam ações de resposta.

Para a aplicação *CloudShop* foi escolhido o *SaaS* da Paraleap, o preço do serviço é de 0,33 *USD*/hora por instância monitorizada[26]. O processo de integração do AzureWatch revelou-se simples e intuitivo (Anexo VIII), concedendo mais tempo para o que realmente traz mais valia à aplicação, a definição de métricas e regras. Como a definição da recolha de novas métricas está presente na aplicação, a configuração da recolha de uma métrica em antecedência não é obrigatória (Listagem 9) servindo apenas para verificar a criação da tabela "*WADPerformanceCountersTable*" na conta de armazenamento. As métricas que foram configuradas na ferramenta são as apresentadas na Tabela 12 [127].

Nome da métrica	**Descrição**
\ASP.NET\Requests Rejected	Número de pedidos que foram rejeitados porque a queue de pedidos se encontra cheia. Um valor diferente de 0 indica que o servidor está sobrecarregado.
\ASP.NET\ Requests Queued	O número de pedidos que aguardam para ser despachados. Se o número de pedidos exceder o limite da queue os pedidos são rejeitados.
\Memory\Available MBytes	Espaço livre em memória. Deve ser prestada atenção a leituras consistentes em que o

[23] http://www.rightscale.com/, acedido em 16 de Agosto de 2012.
[24] http://www.opstera.com/, acedido em 16 de Agosto de 2012.
[25] http://www.paraleap.com/AzureWatch, acedido em 16 de Agosto de 2012.
[26] http://www.paraleap.com/AzureWatch/Pricing, acedido em 18 de Agosto de 2012.

	tamanho da memória livre seja inferior a 20% da memória total do sistema.
\Memory\Page Reads-sec	Indica que a memória requerida pela aplicação é superior à memória física do sistema e por isso está a ocorrer paginação. Deve ser prestada atenção a leituras consistente acima de 5.
\ASP.NET Applications(_Total_)\Requests-Sec	Débito da aplicação ASP.NET. Número de pedidos que estão a ser processados simultaneamente pela aplicação.
\Processor(_Total)\% Processor Time	Percentagem de CPU utilizado. Picos de CPU momentâneos não são um problema mas deve ser prestada atenção a leituras consistentes acima dos 85%.

Tabela 12 - Métricas configuradas no AzureWatch

Algumas destas métricas não têm significado se forem recolhidas isoladamente e não tiverem histórico associado. Um pico de *CPU* momentâneo não é alarmante mas uma média de utilização acima dos 80% nos últimos 10 minutos já o é. O AzureWatch possibilita que os valores sejam agregados ao longo do tempo sendo que a média, total, valor mínimo, valor máximo e última leitura são as operações passíveis de ser realizadas.

Uma mesma métrica pode ser agregada em períodos diferentes o que torna possível a criação de regras mais complexas como por exemplo, lançar uma nova instância se a média de utilização do *CPU* nos últimos 10 minutos for 25% superior à média de utilização na última hora. As regras são uma das etapas mais importantes da configuração da ferramenta da Paraleap, afinal para que servem os dados recolhidos se não os utilizarmos? Para testarmos o correto funcionamento da integração da ferramenta com o sítio *CloudShop* foi criada uma regra que adiciona mais uma instância se a condição "Average10CPUTime > 80" se verificar ou seja, uma nova instância é lançada se a média de utilização do CPU nos últimos 10 minutos for superior a 80%. Deste modo, a elasticidade automática é alcançada e os utilizadores da aplicação na nuvem não devem sequer notar que os recursos estiveram perto da exaustão. O administrador da aplicação é por sua vez notificado por email cada vez que o número de instâncias é alterado.

Outras opções do AzureWatch são a definição de um teto mínimo e máximo para o número de instâncias até ao qual a ferramenta pode escalar, a possibilidade de criar regras para o ajuste do número de instâncias em determinada período do dia para cenários onde esse período é previsível e a possibilidade de só desencadear ações entre o minuto x e y de determinada hora. Esta última opção pode ser utilizada não para reduzir custos, mas para a

otimização da aplicação face aos custos. Podemos por exemplo optar por baixar o número de instâncias somente a partir do minuto 50. Visto que o serviço é taxado à hora o utilizador tem todo o interesse em utilizar horas completas pois paga o mesmo caso a instância saia da *pool* ao minuto 5 ou ao minuto 50.

Mesmo que a aplicação tire partido da escalabilidade automática a disponibilização dos recursos não é instântanea como podemos ver pela Ilustração 20 em que foram registados os tempos de arranque de instâncias do tipo *Web* de tamanho médio.

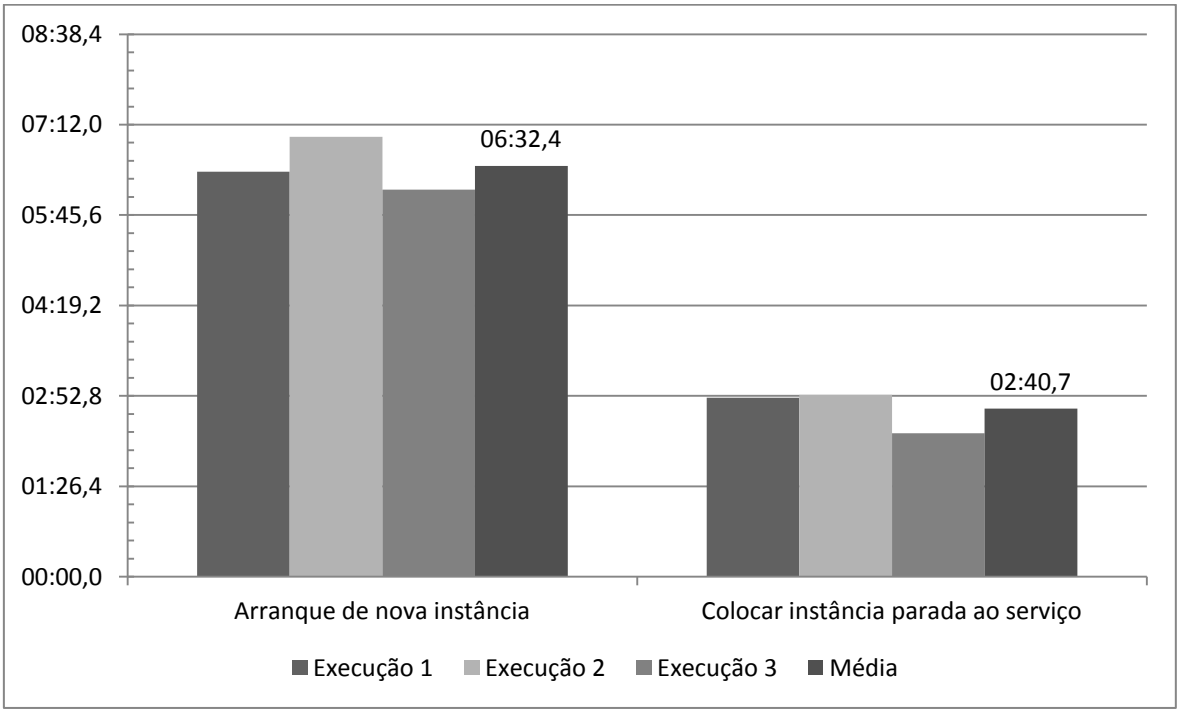


Ilustração 20 - Tempos de arranque de instâncias Web de tamanho médio

Uma nova instância do tipo *Web* demora em média 6 minutos e 32 segundos a estar pronta pois necessita de ser instalada na nuvem (*deploy*), enquanto uma instância parada (no estado *Stopped*) demora 2 minutos e 40 a entrar ao serviço. Estes tempos fazem com que seja necessário configurar um valor de *trigger* abaixo do que seria expectável para que as instâncias estejam prontas com antecedência. Por exemplo, lançar uma nova instância da aplicação quando a média de carga do *CPU* nos últimos 5 minutos for superior a 90% pode não ser adequado uma vez que os 10% que faltam para a exaustão podem não ser suficientes para suportar o tempo de arranque de uma instância adicional.

Abaixo estão enumerados todos os passos que são necessários realizar para que uma nova instância da aplicação seja hospedada num servidor físico [83]:

- O servidor físico procura uma imagem do sistema operativo na rede usando o protocolo *PXE* (*Preboot eXecution Environment*). Descarrega-a e arranca a imagem;

- O servidor físico passa a correr o ambiente de Pré-instalação do Windows (Windows PE) que é usado para ligar o servidor à plataforma Windows Azure. A plataforma envia instruções à máquina para descarregar uma imagem do Windows Server 2008 Server Core para o disco local, esta imagem é um disco virtual (*VHD*) do sistema operativo hospedeiro do Hyper-V. De seguida o servidor é instruído para arrancar essa mesma imagem;

- Depois de a máquina ter arrancado com o sistema operativo hospedeiro é descarregado o sistema operativo "convidado" no caso das instâncias *Web* é descarregado um *VHD* com uma versão customizada do Windows Server 2008 R2 com o IIS 7.5 instalado. A imagem do sistema operativo "convidado" é gravada somente com permissões de leitura de modo a ser partilhada por várias instâncias;

- Um segundo *VHD* é ligado à instância, criada a partir da imagem do sistema operativo "convidado", para guardar as alterações especificas àquele servidor virtual. A aplicação do utilizador é enviada para o segundo *VHD* e o servidor virtual arranca;

- Quando o servidor virtual a que designamos de instância arranca o seu estado é validado. Se tudo estiver conforme, a plataforma é notificada procedendo à configuração de rede (alocação de *IP*s, configuração de *VLAN*s, *DNS* etc.) e à adição do servidor virtual à *pool* de instâncias da aplicação.

Alguns destes passos são efetuados em antecedência [83] mas é de esperar que o tempo de lançar uma nova instância seja semelhante ao apresentado no gráfico da Ilustração 20.

7.2. Pagar Somente os Recursos Utilizados

Uma das bandeiras dos serviços de computação na nuvem é a de que o utilizador só paga pelos recursos utilizados num modelo semelhante ao da água ou eletricidade. A maior parte dos serviços do Windows Azure como é o caso dos *BLOB*s, tabelas ou *Queues* respeitam esta premissa sendo pagos mediante a quantidade de dados armazenados (Ponto 4.2.2). O mesmo não acontece contudo, com as instâncias de computação em que o utilizador paga o mesmo quer o servidor virtual esteja parada (no estado *Stopped*) quer esteja perto dos limites de utilização.

Na realidade, as instâncias de computação são pagas por estarem reservadas e não consoante o processamento. Este modelo pode fazer com que os utilizadores fiquem reticentes a tirar partido de cenários que tenham instâncias que não estão a processar. Um destes cenários são os *static upgrades* em que é utilizado um ambiente de testes para fazer atualizações à aplicação. Neste cenário, as aplicações são atualizadas num ambiente de *staging* no qual, é executado uma bateria de testes. Se a aplicação passar nos testes é colocada em produção. O envio da aplicação do ambiente de testes para o ambiente de produção é feito através da comutação de um *IP* virtual (Ilustração 9), deste modo, a *pool* de instâncias de *staging* passa a ser a *pool* de produção e vice-versa.

A opção mais sensata a tomar quando se executa um *static upgrade* é a de manter a antiga *pool* de produção como salvaguarda. Se algo de imprevisto ocorrer devido à atualização basta comutar o *IP* para recuperar o serviço. Consoante a política da empresa a salvaguarda pode ser mantida por uma semana ou mais sendo que, neste período o valor pago por uma instância da salvaguarda é igual ao valor pago por uma instância de produção. Parar a instância através do portal de administração também não surte qualquer efeito porque o servidor virtual está reservado na mesma.

7.3. Síntese

Foi verificado neste capítulo que algumas das características associadas com a computação na nuvem não podem ser tomadas como garantidas. Tomemos como base a definição de *cloud computing* apresentada no Ponto 3.1:

> "De uma forma geral podemos definir *cloud computing* como uma infraestrutura altamente escalável desenhada para hospedar aplicações e paga mediante a utilização [19]"

Ao confrontar a definição com a análise feita neste capítulo foi verificado que o Windows Azure disponibiliza, de facto, recursos que podem ser solicitados a qualquer momento, mas também foi verificado que essa solicitação necessita de intervenção manual ou recorre a código externo que monitoriza as instâncias e que atua em conformidade. A escalabilidade automática é alcançável mas necessita de um esforço adicional que não era expectável. Foi ainda verificado que nem todos os recursos são pagos mediante a utilização. Se alguns recursos são realmente pagos consoante a utilização como é o caso dos *BLOB*s, outros como as instâncias de computação são pagos por estarem reservados, estando ou não a ser utilizados.

8. Análise da Solução On-Premises VS Nuvem

Este capítulo compara, do ponto de vista dos custos, uma solução *on-premises* versus uma solução na nuvem. Para o efeito, são empregues dados estatísticos reais (e anónimos) da maior loja de comércio eletrónico ("*Loja A*") alojada por um determinado serviço de *web hosting*.

8.1. Dimensionamento

As características do servidor que hospeda a *Loja A* encontram-se na Tabela 13.

Virtualização	Parallels Virtuozzo[27]
Processador	Intel Xeon E5420 @ 2,50GHz
Número de *Cores*	4
Memória *RAM*	8GB
Armazenamento	4 discos 1TB (Raid 1 + Hotspare + Backup)

Tabela 13 - Características do servidor on-premises

Como se pretende que sejam comparadas soluções equivalentes, o tamanho das instâncias da aplicação *CloudShop* necessita de ser afinado para que o desempenho se assemelhe o mais possível com o servidor da *Loja A*. Foi utilizada a versão 7 da aplicação Passmark Performance Test[28] para medir o desempenho computacional de cada uma das instâncias da nuvem (Tabela 2): *Small*, *Medium*, *Large* e *Extra Large*. Para a solução *on-premises* foram utilizados os resultados de um servidor equiparado ao da *Loja A* presentes na base de dados de *benchmarks* da Passmark. Do comparativo resultou a Ilustração 21.

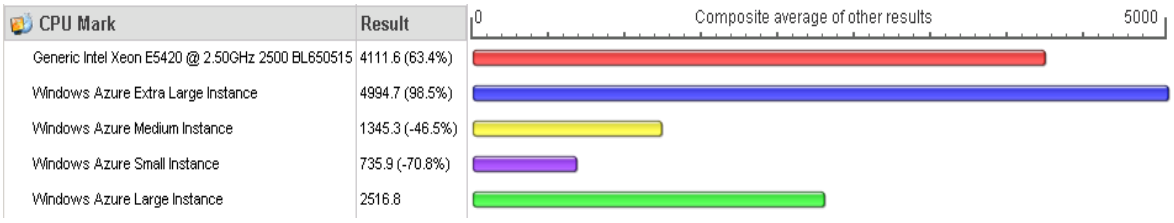


Ilustração 21 - Comparativo de desempenho entre as instâncias do Windows Azure e um servidor equiparado ao da Loja A

O comparativo revela que o desempenho do *CPU* das instâncias do Windows Azure é inferior ao desempenho do *CPU* presente no servidor *on-premises*, à exceção da instância

[27] http://www.parallels.com/products/pvc, acedido em 9 de Setembro de 2012
[28] http://www.passmark.com/products/pt.htm, acedido em 9 de Setembro de 2012

de tamanho *Extra Large*, que apresenta um desempenho 35,1% superior. Não obstante a este facto, foi escolhida para a aplicação *CloudShop* uma instância de tamanho *Large* uma vez que os 64,3% de diferença de desempenho podem ser compensados por se usarem duas instâncias em vez de uma. Para comprovar que a solução na nuvem está bem dimensionada e consegue suster a procura da solução *on-premises*, foram efetuados testes de carga que se basearam no tráfego usual da *Loja A*. A *Loja A* regista uma média de 496 pedidos *HTTP* por dia ou seja, 21 pedidos por hora. Como foi observado no Ponto 3.3.5 a procura a um serviço é variável ao longo do tempo. Também na *Loja A* se notam picos de procura bem definidos no tempo que se traduzem num incremento da ocupação do *CPU* sendo que, nos períodos em que a *Loja A* regista maior procura a percentagem de ocupação do *CPU* é sensivelmente três vezes e meia superior à média. Partindo do pressuposto que o aumento de pedidos *HTTP* no período de maior procura é proporcional ao aumento da ocupação do *CPU*, é expectável que se registem, aproximadamente, 75 pedidos por hora.

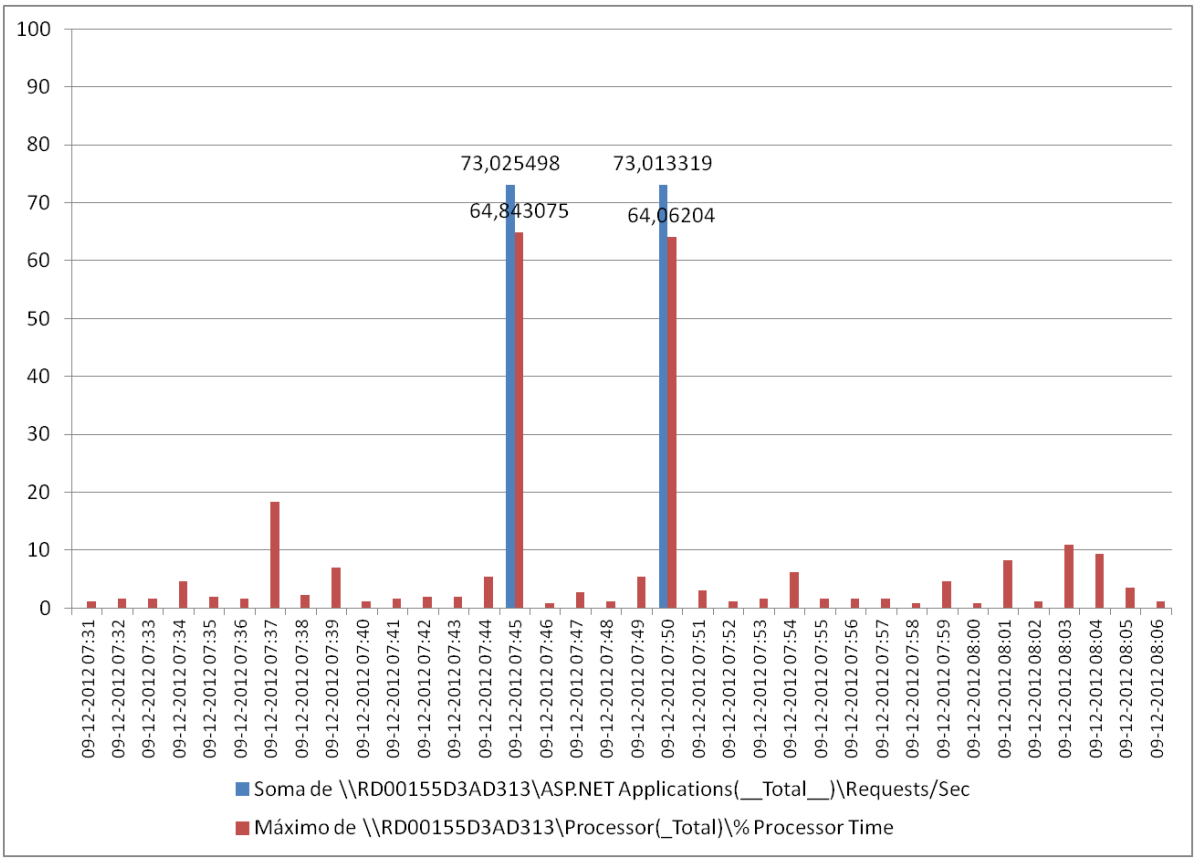


Ilustração 22 - Testes de carga realizados na aplicação CloudShop

Deste modo, se a aplicação *CloudShop* conseguir suster os 75 pedidos *HTTP* então, as instâncias foram dimensionadas corretamente. Para o teste de carga foi criada uma

aplicação de consola que simula o acesso de clientes ao sítio. A aplicação foi executada em duas máquinas distintas que acederam ao sítio para descarregar a listagem de produtos (em *http://cloud-shop.cloudapp.net/Catalog.aspx*). Aquando do teste o sítio *CloudShop* encontrava-se a correr numa única instância que conseguiu suster 75 pedidos num minuto, com uma carga máxima de CPU na ordem dos 65% (Ilustração 22) e com um tempo médio de resposta de 337,44 milissegundos.

8.2. Comparação de Custos

Foi calculado numa primeira fase, os custos associados a ter um servidor como o da *Loja A* alojado localmente. Foram englobados dois tipos de custo, o custo de aquisição do equipamento e licenciamento (custo não corrente ou *CAPEX*, Capital Expenditure) e os custos de operação e manutenção do equipamento (custo corrente ou *OPEX*, Operational Expenditure).

Para se calcular o custo do equipamento, nomeadamente do servidor e do bastidor, foi utilizado o sítio da Hewlett-Packard[29] e o sítio do distribuidor ibérico INEC[30]. O custo do hardware de rede incluindo cablagem, foi definido de acordo com a calculadora de comparação de custo da Amazon[128] correspondendo a 20% do total do investimento com servidores. Para se chegar ao preço do *software* foram utilizadas as cotações da CPCDI, Companhia Portuguesa de Computadores e Distribuição de Produtos Informáticos SA[31] ao abrigo do programa de licenciamento Microsoft Open License[32].

Nos custos correntes foram englobados os gastos com eletricidade e manutenção do equipamento. Para se aferir o custo com eletricidade foi estimado um consumo mensal de 252 kWh tendo por base uma potência de 350 W 24 horas por 7. Para a manutenção foi considerado que a instituição tem um contrato de assistência técnica para o seu parque informático no qual, está englobado o servidor. Como o contrato é partilhado apenas foi considerado 50% do valor total do contrato que, no caso do "Contrato PME" da ICD[33] corresponde a 371 USD mensais. Se a manutenção envolver a substituição de *hardware* foi considerado que o custo do *hardware* de substituição numa base anual é 10% do valor do equipamento, mais uma vez a percentagem definida está de acordo com a calculadora de comparação de custo da Amazon [128]. Outros custos que são partilhados pelos equipamentos que compõem o parque informático são o acesso à *Internet* e o ar condicionado. No primeiro foi considerado 50% de um acesso com uma largura de banda de 100MB de *download* e 10MB de *upload* presente no sítio da PT Negócios[34], o segundo não foi contabilizado pelo que a estimativa do custo mensal é uma estimativa por baixo.

[29] http://www8.hp.com/pt/pt/home.html
[30] http://www.inec.pt
[31] http://www.cpcdi.pt
[32] http://www.microsoft.com/Licensing/licensing-options/open-license.aspx
[33] http://www.icd.pt
[34] http://www.ptnegocios.pt

Depois de averiguados os valores associados a ter um servidor como o da *Loja A* alojado localmente, foram calculados os custos de uma solução na nuvem tendo em conta o dimensionamento efetuado na secção anterior (Ponto 8.1). Numa solução na nuvem só existem custos correntes não existindo qualquer investimento inicial devido à compra de equipamento. Para o custo da solução na nuvem foram utilizadas 2 instâncias *Large* e uma base de dados e espaço de armazenamento de acordo com as dimensões do servidor da *Loja A*. No custo total não consta o custo da transferência de dados para a Internet (0,12 *USD* por *GB*) e o custo das operações na conta de armazenamento (0,01 *USD* por 100000 operações). Está ainda de fora a possibilidade do número de instâncias variar devido à implementação da escalabilidade automática com a ferramenta Paraleap AzureWatch.

A Tabela 14 mostra o *TCO* (*Total Cost of Ownership*) de uma solução *on-premises* relativamente a uma solução na nuvem num período de 3 anos (o custo detalhado das soluções pode ser consultado no Anexo IX). O *TCO* é uma estimativa usada para medir a viabilidade de um investimento financeiro e que engloba todos os custos diretos e indiretos que estão envolvidos em determinada aquisição [129]. No cálculo da estimativa da *Loja A*, todos os valores que se encontravam em euros (*EUR*) foram convertidos em dólares americanos (*USD*) sendo que, os valores bem como a taxa de câmbio correspondem a 14 de Setembro de 2012. A comparação tem por base o desenvolvimento de uma nova solução, que seria criada de raiz quer para ser alojada localmente, quer para ser alojada na nuvem. Deste modo, o custo de desenvolvimento não foi contabilizado por se achar equivalente. Num cenário de migração para a nuvem, teria de ser contabilizado o esforço de adaptação da aplicação existente ao novo paradigma.

Pelos cálculos percebe-se que o custo entre a solução *on-premises* e a solução na nuvem está equiparado chegando a uma diferença de dois dígitos a meio do terceiro ano. Nesse período o custo da solução na nuvem acaba mesmo, por ultrapassar o custo da solução local apresentado no final dos 3 anos, um total acumulado de 27700 *USD* contra 25924 *USD* (mais 7%).

Cálculo TCO da *Loja A* On-Premises vs Nuvem

Custos On-Premises

Custos	Investimento Inicial (CapEx)	Ano 1	Ano 2	Ano 3	Total no Fim do Período
Custo de Hardware					
Servidor 1 processador 4 cores	$3.390	$0	$0	$0	$3.390
UPS Online 3300VA	$2.030	$0	$0	$0	$2.030
Custo hardware de rede (20% do investimento com servidores)	$678	$0	$0	$0	$678
Custo bastidor	$275	$0	$0	$0	$275
Custo manutenção hardware (10% do custo total de hardware) *	$0	$68	$271	$271	$609
Sub-Total do "Custo de Hardware"	$6.373	$68	$271	$271	$6.983
Custo de Software					
Microsoft Windows Server Standard Edition OLP NL (1 licença)	$1.467	$0	$0	$0	$1.467
SQL Server Standard Edition OLP NL (1 licença)	$1.491	$0	$0	$0	$1.491
Sub-Total do "Custo de Software"	$2.958	$0	$0	$0	$2.958
Custo das Instalações					
Eletricidade (720 horas @ 350W = 252kWh)	$0	$552	$552	$552	$1.656
Ligação à Internet (proporcional de 50%)	$0	$321	$321	$321	$962
Ar condicionado **	$0	$0	$0	$0	$0
Sub-Total do "Custo das Instalações"	$0	$873	$873	$873	$2.618
Outros Custos					
Contrato de assistência técnica (proporcional de 50%)	$0	$4.455	$4.455	$4.455	$13.365
Sub-Total "Outros Custos"	$0	$4.455	$4.455	$4.455	$13.365
CUSTO TOTAL	$9.331	$5.396	$5.599	$5.599	$25.924

Custos Nuvem

Custos	Investimento Inicial (CapEx)	Ano 1	Ano 2	Ano 3	Total no Fim do Período
Custo dos Serviços na Nuvem					
2 Instâncias Web Role de tamanho Large	$0	$8.294	$8.294	$8.294	$24.883
Base de Dados 10GB	$0	$551	$551	$551	$1.654
Conta de Armazenamento 100GB	$0	$150	$150	$150	$450
Sub-Total "Custo dos Serviços na Nuvem"	$0	$8.996	$8.996	$8.996	$26.988
Outros Custos					
2 Instâncias Monitorizadas no AzureWatch	$0	$238	$238	$238	$713
Sub-Total "Outros Custos"	$0	$238	$238	$238	$713
CUSTO TOTAL	$0	$9.233	$9.233	$9.233	$27.700

* Só foi considerado o custo de hardware de rede nos 3 anos e da UPS no 2º e 3º anos. O servidor está coberto por garantia nos primeiros 3 anos com "Next Business Day Parts Replacement" e por isso não foi considerado, a UPS por sua vez, está coberta por garantia que inclui a substuição de hardware no 1º ano.

** Não foi contabilizado, estimativa por baixo

Tabela 14 - Cálculo TCO da Loja A On-Premises vs Nuvem

Estes resultados demonstram a exatidão dos preços praticados pelos fornecedores de *cloud-computing* que têm por objetivo transferir os custos para o utilizador. O paradigma não se mostrou aliciante do ponto de vista financeiro para o cenário da *Loja A* o que se deve ao facto, de não se estar a tirar partido da elasticidade e se estar a utilizar um dimensionamento acima das reais necessidades da aplicação. Todos os cálculos feitos até agora utilizam duas instâncias de tamanho *Large*, este número de instâncias é uma condição necessária para que seja assegurado o *SLA* (pelo menos duas instâncias de cada tipo) mas, resulta num dimensionamento excessivo fora do período de pico.

Com o objetivo de estimar qual o impacto financeiro de um serviço elástico na nuvem, foi ignorado o SLA, efetuando-se uma estimativa mais precisa que tem em conta a procura diária ao sítio. A *Loja A* apresenta um padrão diário que se caracteriza por um aumento da procura das 14 às 21 horas na qual são necessárias duas instâncias, no restante tempo uma única instância é suficiente. Ao contrário da estimativa anterior, não foram utilizadas diariamente 48 horas de uma instância *Large* mas antes uma instância a tempo inteiro e uma instância adicional no período de pico. As anteriores 48 horas passaram a 24 horas mais 7 o que perfaz um total de 31 horas de computação por dia. O benefício financeiro da adequação dos recursos à carga exigida está evidenciado na Ilustração 23 permitindo uma poupança na ordem dos 40% num período de 3 anos.

Ilustração 23 - Custo cumulativo da Loja A ao longo dos 3 anos

Os resultados levam a concluir que num cenário em que o número de instâncias é fixo ou em que o número de instâncias necessárias "fora do período de pico" é inferior a dois o paradigma do *cloud-computing* não é atrativo do ponto de vista financeiro. Mesmo que haja diferença esta deve ser suficientemente grande para que justifique a adoção de um novo paradigma. A Saugatuck Technology[35], uma empresa de consultadoria e investigação vocacionada para o *cloud-computing* nas suas diferentes formas (*SaaS*, *PaaS* ou *IaaS*), defende que a diferença do *TCO* entre uma solução *on-premises* e uma solução na nuvem nunca deve ser inferior a 20% para que o aspeto financeiro justifique, por si só, a adoção de um novo paradigma [130]. Uma instituição deve, por isso, averiguar a viabilidade financeira da nuvem antes de optar por introduzir o novo paradigma na sua estratégia de negócio. Os cálculos devem ser feitos com rigor e com a maior quantidade de dados possível para que a estimativa do *TCO* seja próxima da realidade. Para o efeito a instituição pode utilizar o seu próprio modelo de cálculo ou utilizar um modelo de uma empresa independente [130], tal como apresentado na Tabela 14 que utiliza o modelo da Infotech[36]. Os modelos dos fornecedores podem ser utilizados como complemento mas, como são "patrocinados" podem induzir a resultados demasiados favoráveis para o fornecedor de serviço [131]. Deste modo, não têm validade quando utilizados isoladamente.

[35] http://saugatucktechnology.com, acedido em 19 de Setembro de 2012.
[36] http://www.infotech.com, acedido em 19 de Setembro de 2012.

9. Conclusão e Trabalho Futuro

9.1. Conclusão

Na dissertação foi feita uma análise às ofertas dos três principais fornecedores de serviço, a Amazon, a Google e a Microsoft. Embora a base seja sensivelmente a mesma, cada uma das ofertas tem especificidades que podem justificar a sua escolha em detrimento de outra. A oferta que é mais adequada para uma instituição pode não ser a mais adequada para outra por isso, a seleção terá de ser feita caso a caso. Ao longo da análise foi observado que os fornecedores de serviço demonstram uma clara aposta no sector empresarial. Potenciando por exemplo, o desempenho das bases de dados de grandes dimensões através da oferta de bases de dados *noSQL* e através da disponibilização de serviços que aplicam o padrão de *sharding* ou facilitando por exemplo, a composição de serviços através de um *Service Bus*. A Microsoft apresenta a oferta mais ampla a nível de serviços e foi escolhida para os testes práticos às plataformas de *cloud-computing*. A análise teve enfoque no Windows Azure no entanto, acredita-se que grande parte das conclusões é resultado do novo paradigma e não é dependente da plataforma usada.

Verificaram-se algumas divergências entre o que é publicitado e o que é realmente oferecido pelo fornecedor de serviço nomeadamente, ao nível da elasticidade e do modelo de pagamento. A elasticidade, uma das bandeiras do *cloud-computing*, não é garantida pelo simples facto de se usar a nuvem. Esta característica só é alcançada através de um desenho cuidado da aplicação que alcance a independência entre os componentes da mesma, que evite tarefas bloqueantes e preveja mecanismos de prevenção e resposta à saturação do acesso a dados. As características que distinguem a nuvem de uma infraestrutura convencional nas quais se insere a elasticidade, podem e devem ser utilizadas. Aliás, foi concluído que não existe nenhum ganho financeiro para uma instituição se a mudança para o paradigma do *cloud-computing* não tiver em vista o aproveitamento de alguma destas características diferenciadoras. Deste modo, num cenário em que o número de instâncias é fixo ou em que o número de instâncias necessário "fora do período de pico" é inferior a dois, o paradigma não é atrativo do ponto de vista financeiro. A exatidão dos preços praticados pelos fornecedores de serviço é tal, que o custo cumulativo de uma solução na nuvem foi sensivelmente o mesmo que o calculado para uma solução *on-premises* quando considerado um período de 3 anos. Logo, as instituições devem ter uma estratégia definida

e entender se o seu negócio se adequa à nuvem, não devendo basear a sua decisão no entusiasmo criado em torno da mesma e no facilitismo de acesso aos recursos disponibilizados pelos fornecedores de serviço.

Se o novo paradigma se mostrar adequado a barreira de aprendizagem não constituirá um obstáculo e poderão ser alcançadas poupanças substanciais. A evolução do paradigma permite que hoje em dia, a nuvem esteja preparada para as empresas e corresponda às expectativas criadas.

9.2. Trabalho Futuro

A análise de custos efetuada neste trabalho baseia-se exclusivamente numa aplicação, uma loja de comércio eletrónico. Entende-se que seria conveniente analisar mais cenários, com aplicações de diferentes dimensões, com propósitos distintos e com diferentes perfis de utilização. Deste modo, a análise seria mais representativa e os resultados seriam mais robustos.

Julga-se também, que a escalabilidade horizontal na camada de acesso a dados merece um estudo aprofundado onde sejam empregues bases de dados relacionais, não relacionais e mecanismos de *caching*. O estudo de várias soluções para a saturação no acesso a dados seria com certeza um tópico relevante dada a sua atualidade.

Esta página foi intencionalmente deixada em branco

10. Bibliografia

[1] Gartner, “Analyst Q&A with Conference Chair Mike Chuba,” 2010. [Online]. Available: http://www.gartner.com/technology/summits/na/data-center/analyst-qa.jsp. [Acedido em 17 Julho 2012].

[2] R. Atta, “DARPA History,” 2008. [Online]. Available: http://www.darpa.mil/WorkArea/DownloadAsset.aspx?id=2553. [Acedido em 10 Julho 2012].

[3] M. Waldrop, “DARPA History,” 2008. [Online]. Available: http://www.darpa.mil/WorkArea/DownloadAsset.aspx?id=2554. [Acedido em 10 Julho 2012].

[4] A. Mohamed, “A History of cloud computing,” 27 Março 2009. [Online]. Available: http://www.computerweekly.com/feature/A-history-of-cloud-computing. [Acedido em 10 Julho 2012].

[5] S. Garfinkel, Architects of the Information Society: Thirty-Five Years of the Laboratory for Computer Science at MIT, Massachusetts: The MIT Press, 1999.

[6] D. Parkhill, The Challenge of the Computer Utility, Addison-Wesley, 1966.

[7] T. O'Reilly, “What Is Web 2.0, Design Patterns and Business Models for the Next Generation of Software,” 30 Setembro 2005. [Online]. Available: http://oreilly.com/pub/a/web2/archive/what-is-web-20.html. [Acedido em 10 Junho 2012].

[8] Nokia Siemens Networks, “Mobile data from smart devices to increase 10,000 percent by 2015,” 15 Fevereiro 2010. [Online]. Available: http://www.nokiasiemensnetworks.com/sites/default/files/document/NokiaSiemensNetworks_2010_02_15_en_V1_MWC_2010.pdf. [Acedido em 20 Março 2012].

[9] E. Raymond, The New Hacker's Dictionary, Massachusetts: The MIT Press, 1996.

[10] Wikipedia, “Time-sharing,” 2012. [Online]. Available: http://en.wikipedia.org/wiki/Time-sharing. [Acedido em 28 Agosto 2012].

[11] B. Hayes, “Cloud Computing,” *Communications of the ACM*, Julho 2008.

[12] A. Tanenbaum e M. Steen, Distributed Systems, Principles and Paradigms, Amsterdam: Prentice Hall, 2006.

[13] M. Adolph, “Distributed Computing: Utilities, Grids & Clouds,” International Telecommunication Union, 2009.

[14] TechTarget, “Server Virtualization Infrastructure and Architecture, Introduction to virtualization and how-tos,” Dezembro 2010. [Online]. Available: http://searchservervirtualization.techtarget.com/definition/virtualization. [Acedido em 27

Março 2012].

[15] TechTarget, "SOA Resources, Web Services (application services)," Março 2007. [Online]. Available: http://searchsoa.techtarget.com/definition/Web-services. [Acedido em 11 Julho 2012].

[16] W3Schools, "Web Services Basic," 2012. [Online]. Available: http://www.w3schools.com/webservices/default.asp. [Acedido em 25 Março 2012].

[17] A. Weiss, "ACM," *Computing In the Clouds,* Dezembro 2007.

[18] M. e. a. Armbrust, "Above the Clouds: A Berkeley View of Cloud Computing," Electrical Engineering and Computer Sciences University of California at Berkeley, Berkeley, California, 2009.

[19] J. Staten, "Is Cloud Computing Ready For The Enterprise," Forrester, 2008.

[20] Amazon Web Services, "Amazon Elastic Compute Cloud (Amazon EC2)," 2012. [Online]. Available: http://aws.amazon.com/ec2/. [Acedido em 20 Julho 2012].

[21] M. e. a. Armbrust, "A View of Cloud Computing," *Communications of the ACM,* Abril 2010.

[22] D. Kondo, B. Javadi, P. Malecot, F. Cappello e D. Anderson, "Cost-Benefit Analysis of Cloud Computing versus Desktop Grids," INRIA, France, 2009.

[23] M. Krieger, "AirBnB Tech Talk 2012: Scaling Instagram," 11 Abril 2012. [Online]. Available: http://lanyrd.com/2012/airbnb-mike-krieger/. [Acedido em 13 Julho 2012].

[24] J. Hsiao, "Amazon.com CEO Jeff Bezos on Animoto," 21 Abril 2008. [Online]. Available: http://animoto.com/blog/company/amazon-com-ceo-jeff-bezos-on-animoto/. [Acedido em 13 Julho 2012].

[25] JP Sá Couto, "Cloud Computing Oportunidades e Desafios," *JP Sá Couto news,* pp. 14-22, 29 Julho 2012.

[26] Eucalyptus, "What is Eucalyptus," 2009. [Online]. Available: http://www.eucalyptus.com/learn/what-is-eucalyptus. [Acedido em 16 Julho 2012].

[27] Eucalyptus, "The Eucalyptus Story," 2009. [Online]. Available: http://www.eucalyptus.com/about/story. [Acedido em 16 Julho 2012].

[28] Forrester Research, Inc., "Is Cloud Computing Ready For The Enterprise?," Março, 7, 2008.

[29] EurActiv, "Cloud computing: A legal maze for Europe," 5 Abril 2012. [Online]. Available: http://www.euractiv.com/innovation-enterprise/cloud-computing-legal-maze-europ-linksdossier-511262. [Acedido em 17 Julho 2012].

[30] E. Knapp, "Patriot Act Threatens American Cloud Computing," 24 Janeiro 2012. [Online]. Available: http://wallstcheatsheet.com/breaking-news/patriot-act-threatens-american-cloud-computing.html/. [Acedido em 4 Setembro 2012].

[31] B. Irene, "EU Data Protection Law and the Patriot Act in the Cloud," 29 Março 2012. [Online]. Available: http://www.webanalyticsworld.net/2012/03/eu-data-protection-law-and-the-patriot-act-in-the-cloud.html. [Acedido em 4 Setembro 2012].

[32] EurActiv, "EU-US data privacy storm blows cloud off course," 30 Novembro 2011. [Online]. Available: http://www.euractiv.com/specialreport-cloud-computing/eu-us-data-privacy-storm-blows-c-news-509134. [Acedido em 5 Setembro 2012].

[33] EurActiv, "Brussels to unveil EU cloud computing strategy," 19 Julho 2012. [Online]. Available: http://www.euractiv.com/infosociety/brussels-unveil-eu-cloud-computi-news-514012. [Acedido em 5 Setembro 2012].

[34] Amazon Web Services, "AWS Import/Export," 2012. [Online]. Available: http://aws.amazon.com/importexport/. [Acedido em 17 Julho 2012].

[35] Microsoft, "Pricing Details," 2012. [Online]. Available: https://www.windowsazure.com/en-us/pricing/details/. [Acedido em 18 Julho 2012].

[36] Microsoft, "Data Storage Offerings on the Windows Azure Platform," 25 Junho 2012. [Online]. Available: http://social.technet.microsoft.com/wiki/contents/articles/1674.data-storage-offerings-on-the-windows-azure-platform.aspx. [Acedido em 18 Julho 2012].

[37] Microsoft, "Windows Azure Learning Series, Service Bus," 17 Dezembro 2011. [Online]. Available: http://blogs.msdn.com/b/appfabric/archive/2011/05/17/new-service-bus-and-appfabric-videos-available.aspx. [Acedido em 19 Julho 2012].

[38] C. Richardson, "Deploying Java Applications on Amazon EC2, Presentation at QCon," 10 Agosto 2010. [Online]. Available: http://www.infoq.com/presentations/Deploying-on-Amazon-EC2. [Acedido em 20 Julho 2012].

[39] Amazon Web Services, "AWS Solutions," 2012. [Online]. Available: http://aws.amazon.com/solutions/aws-solutions/. [Acedido em 20 Julho 2012].

[40] Amazon Web Services, "Amazon Simple Storage Service (Amazon S3)," 2012. [Online]. Available: http://aws.amazon.com/s3/. [Acedido em 20 Julho 2012].

[41] P. Hofmann, "Cloud Computing: The Limits of Public Clouds for Business Applications," *Internet Computing, IEEE,* pp. 90-93, 2010.

[42] TechTarget, "Database management system (DBMS) architecture, design and stategy, NoSQL (Not Only SQL)," Oututbro 2011. [Online]. Available: http://searchdatamanagement.techtarget.com/definition/NoSQL-Not-Only-SQL. [Acedido em 20 Julho 2012].

[43] TechTarget, "Microsoft SQL Server Tools and Utilities, ACID (atomicy, consistency, isolation and durability)," Julho 2006. [Online]. Available: http://searchsqlserver.techtarget.com/definition/ACID. [Acedido em 20 Julho 2012].

[44] R. Cattell, "Scalable SQL and NoSQL data stores," *ACM SIGMOD Record,* vol. 39, pp. 12-27, Dezembro 2010.

[45] Amazon Web Services, "Amazon SimpleDB," 2012. [Online]. Available: http://aws.amazon.com/simpledb/. [Acedido em 20 Julho 2012].

[46] Amazon Web Services, "DynamoDB FAQ, How does Amazon DynamoDB differ from Amazon SimpleDB? Which should I use?," 2012. [Online]. Available: http://aws.amazon.com/dynamodb/faqs/#How_does_Amazon_DynamoDB_differ_from_Amazon_SimpleDB_Which_should_I_use. [Acedido em 20 Julho 2012].

[47] Amazon Web Services, "Amazon DynamoDB," 2012. [Online]. Available: http://aws.amazon.com/dynamodb/. [Acedido em 20 Julho 2012].

[48] Amazon Web Services, "Amazon Relational Database Service (Amazon RDS)," 2012. [Online]. Available: http://aws.amazon.com/rds/. [Acedido em 20 Julho 2012].

[49] Amazon Web Services, "Amazon Simple Queue Service (Amazon SQS)," 2012. [Online]. Available: http://aws.amazon.com/sqs/. [Acedido em 20 Julho 2012].

[50] Amazon Web Services, "Amazon ElastiCache," 2012. [Online]. Available: http://aws.amazon.com/elasticache/. [Acedido em 20 Julho 2012].

[51] B. Fitzpatrick, 2003. [Online]. Available: http://memcached.org/. [Acedido em 20 Julho 2012].

[52] Amazon Web Services, "Amazon CloudWatch," 2012. [Online]. Available: http://aws.amazon.com/cloudwatch/. [Acedido em 20 Julho 2012].

[53] Amazon Web Services, "Auto Scaling," 2012. [Online]. Available: http://aws.amazon.com/autoscaling/. [Acedido em 20 Julho 2012].

[54] Amazon Web Services, "CastingWords makes audio searchable by teaming with Mechanical Turk on audio transcription," 2012. [Online]. Available: https://requester.mturk.com/case_studies/cs/castingwords. [Acedido em 20 Julho 2012].

[55] Amazon Web Services, "Amazon Virtual Private Cloud (Amazon VPC)," 2012. [Online]. Available: http://aws.amazon.com/vpc/. [Acedido em 20 Julho 2012].

[56] Amazon Web Services, "Amazon CloudFront," 2012. [Online]. Available: http://aws.amazon.com/cloudfront/. [Acedido em 30 Julho 2012].

[57] R. Miller, "Report: Google Uses About 900,000 Servers," 1 Agosto 2011. [Online]. Available: http://www.datacenterknowledge.com/archives/2011/08/01/report-google-uses-

about-900000-servers/. [Acedido em 28 Abril 2012].

[58] Google, “Google App Engine General Questions, What languages are supported by Google App Engine?,” 26 Junho 2012. [Online]. Available: https://developers.google.com/appengine/kb/general#language. [Acedido em 20 Julho 2012].

[59] Google, “What Is Google App Engine? The Sandbox,” 11 Julho 2012. [Online]. Available: https://developers.google.com/appengine/docs/whatisgoogleappengine. [Acedido em 20 Julho 2012].

[60] Google, “Queues and Backends,” 2012. [Online]. Available: http://googcloudlabs.appspot.com/codelabexercise8.html. [Acedido em 20 Julho 2012].

[61] Google, “Instances in the Admin Console, How Applications Scale, Scaling in Instances,” 26 Junho 2012. [Online]. Available: https://developers.google.com/appengine/docs/adminconsole/instances. [Acedido em 20 Julho 2012].

[62] Google, “Google App Engine, Using the Google Plugin for Eclipse,” 12 Junho 2012. [Online]. Available: https://developers.google.com/appengine/docs/java/tools/eclipse. [Acedido em 20 Julho 2012].

[63] C. McLuckie, “Google Compute Engine: Computing without limits,” 28 Junho 2012. [Online]. Available: http://googledevelopers.blogspot.pt/2012/06/google-compute-engine-computing-without.html. [Acedido em 20 Julho 2012].

[64] F. Chang, J. Dean, S. Ghemawat, W. Hsieh, D. Wallach, M. Burrows, T. Chandra, A. Fikes e R. Gruber, “Bigtable: A Distributed Storage System for Structured Data,” Google, Inc., 2006.

[65] J. Dean, “BigTable: A Distributed Structured Storage System, Lecture at the University of Washington,” 2006. [Online]. Available: http://video.google.com/videoplay?docid=7278544055668715642. [Acedido em 22 Julho 2012].

[66] Google, “Google Cloud Storage, Developer-Guide,” 2012. [Online]. Available: https://developers.google.com/storage/docs/developer-guide. [Acedido em 22 Julho 2012].

[67] Google, “Google Cloud Storage, Getting Started,” 2012. [Online]. Available: https://developers.google.com/storage/docs/getting-started. [Acedido em 22 Julho 2012].

[68] Joneja, Navneet, “Google Cloud SQL: your database in the cloud,” 6 Outubro 2011. [Online]. Available: http://googlecode.blogspot.pt/2011/10/google-cloud-sql-your-database-in-cloud.html. [Acedido em 22 Julho 2012].

[69] Google, “Google Cloud SQL FAQ, How large a database can I use with Google Cloud SQL?,” 27 Junho 2012. [Online]. Available: https://developers.google.com/cloud-

sql/faq#howlargedataset. [Acedido em 22 Julho 2012].

[70] Google, "Google App Engine, Task Queue Java API Overview," 4 Junho 2012. [Online]. Available: https://developers.google.com/appengine/docs/java/taskqueue/overview. [Acedido em 22 Julho 2012].

[71] Google, "Google App Engine, Memcache Java API Overview," 26 Junho 2012. [Online]. Available: https://developers.google.com/appengine/docs/java/memcache/overview. [Acedido em 23 Julho 2012].

[72] Google, "Google BigQuery," 2012. [Online]. Available: https://developers.google.com/bigquery/docs/overview. [Acedido em 23 Julho 2012].

[73] TechTarget, "Database management system (DBMS) architecture, design and strategy, OLAP (online analytical processing)," Junho 2007. [Online]. Available: http://searchdatamanagement.techtarget.com/definition/OLAP. [Acedido em 23 Julho 2012].

[74] Microsoft State & Local Government Azure Team, "Azure Data Center Long Tour, Video," 20 Abril 2012. [Online]. Available: https://twitter.com/slgazure/status/193398938894737409. [Acedido em 27 Julho 2012].

[75] TechTarget, "Data center cooling, free cooling," Agosto 2011. [Online]. Available: http://searchdatacenter.techtarget.com/definition/free-cooling. [Acedido em 27 Julho 2012].

[76] TechTarget, "x86 commodity rackmount servers, power usage effectiveness (PUE)," Abril 2009. [Online]. Available: http://searchdatacenter.techtarget.com/definition/power-usage-effectiveness-PUE. [Acedido em 27 Julho 2012].

[77] Microsoft, "Trust center, Privacy," 2012. [Online]. Available: https://www.windowsazure.com/en-us/support/trust-center/privacy/. [Acedido em 27 Julho 2012].

[78] Microsoft, "Windows Azure Platform Now Generally Available in 21 Countries," 1 Fevereiro 2010. [Online]. Available: http://blogs.msdn.com/b/windowsazure/archive/2010/02/01/windows-azure-platform-now-generally-available-in-21-countries.aspx. [Acedido em 24 Julho 2012].

[79] Microsoft, "Keynote from the June 7 Meet Azure Event, Scott Guthrie," 7 Junho 2012. [Online]. Available: http://www.meetwindowsazure.com/Conversations#ScottGuthrieMeet. [Acedido em 27 Julho 2012].

[80] Microsoft, "Develop, Introducing Windows Azure," 2012. [Online]. Available: https://www.windowsazure.com/en-us/develop/net/fundamentals/intro-to-windows-azure/. [Acedido em 27 Julho 2012].

[81] Microsoft, "Develop, How to Create a Certificate for a Role," 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/windowsazure/gg432987.aspx. [Acedido em 27

Julho 2012].

[82] Microsoft, “Develop, Overview of Creating a Hosted Service for Windows Azure,” 13 Abril 2011. [Online]. Available: http://msdn.microsoft.com/en-us/library/windowsazure/gg432976.aspx. [Acedido em 27 Julho 2012].

[83] C. Hay e B. H. Prince, Azure In Action, Stamford: Manning, 2011.

[84] Microsoft, “Develop, Improving Application Availability in Windows Azure,” 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/windowsazure/gg236576.aspx. [Acedido em 28 Julho 2012].

[85] Microsoft, “Windows Azure Drive,” Fevereiro 2010. [Online]. Available: http://go.microsoft.com/?linkid=9710117. [Acedido em 28 Julho 2012].

[86] V. Tomáz, “SQL Azure Fedarations,” *PROGRAMAR*, n.º 32, pp. 24-28, Dezembro 2011.

[87] TechTarget, “Hardware, Moore's Law,” Abril 2005. [Online]. Available: http://whatis.techtarget.com/definition/Moores-Law. [Acedido em 28 Julho 2012].

[88] Microsoft, “Develop, General Guidelines and Limitations (Windows Azure SQL Database),” 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/windowsazure/ee336245.aspx. [Acedido em 28 Julho 2012].

[89] TechTarget, “Cloud management and monitoring, sharding,” Dezembro 2011. [Online]. Available: http://searchcloudcomputing.techtarget.com/definition/sharding. [Acedido em 28 Julho 2012].

[90] Microsoft, “Develop, Federations in Windows Azure SQL Database (formerly SQL Azure),” 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/windowsazure/hh597452.aspx. [Acedido em 28 Julho 2012].

[91] G. Huey, “Scaling Out with SQL Azure Federation,” *MSDN Magazine*, Fevereiro 2012.

[92] Microsoft, “Develop, How to Use the Table Storage Service,” 2012. [Online]. Available: https://www.windowsazure.com/en-us/develop/net/how-to-guides/table-services/. [Acedido em 29 Julho 2012].

[93] Microsoft, “Windows Azure Table - Dec 2008.docx,” Dezembro 2008. [Online]. Available: http://download.microsoft.com. [Acedido em 29 Julho 2012].

[94] TechTarget, “Windows Presentation Foundation, LINQ (Language Integrated Query),” Fevereiro 2008. [Online]. Available: http://searchwindevelopment.techtarget.com/definition/LINQ. [Acedido em 29 Julho 2012].

[95] Microsoft, “Windows Azure Blob - Dec 2008.docx,” Dezembro 2008. [Online]. Available: http://download.microsoft.com. [Acedido em 29 Julho 2012].

[96] Microsoft, “New Service Bus and AppFabric Videos Available,” 17 Maio 2011. [Online]. Available: http://blogs.msdn.com/b/appfabric/archive/2011/05/17/new-service-bus-and-appfabric-videos-available.aspx. [Acedido em 29 Julho 2012].

[97] Erl, Thomas, “Service Composition,” 2012. [Online]. Available: http://www.soaglossary.com/service_composition.php. [Acedido em 30 Julho 2012].

[98] S. Guthrie, “Meet the New Windows Azure,” 7 Junho 2012. [Online]. Available: http://weblogs.asp.net/scottgu/archive/2012/06/07/meet-the-new-windows-azure.aspx. [Acedido em 29 Julho 2012].

[99] Microsoft, “Develop, What's New in Windows Azure,” 7 Junho 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/windowsazure/gg441573.aspx. [Acedido em 7 Setembro 2012].

[100] C. Darie e K. Watson, Beginning ASP.NET E-Commerce in C#, Nova Iorque: Apress, 2009.

[101] Microsoft, “KB306355, How to create custom error reporting pages in ASP.NET by using Visual C# .NET,” 13 Junho 2012. [Online]. Available: http://support.microsoft.com/kb/306355. [Acedido em 8 Agosto 2012].

[102] PayPal, “PayPal Express Checkout Integration Guide,” Julho 2011. [Online]. Available: https://cms.paypal.com/cms_content/US/en_US/files/developer/PP_ExpressCheckout_IntegrationGuide.pdf. [Acedido em 10 Agosto 2012].

[103] Microsoft, “Chapter 10 - Building Secure ASP.NET Pages and Controls, Session Hijacking,” Janeiro 2006. [Online]. Available: http://msdn.microsoft.com/en-us/library/ff648635. [Acedido em 9 Agosto 2012].

[104] TechTarget, “Software Quality Resources, cross-site scripting (XSS),” Setembro 2010. [Online]. Available: http://searchsoftwarequality.techtarget.com/definition/cross-site-scripting. [Acedido em 1 Julho 2012].

[105] Microsoft, “Microsoft Patterns & Practices, How To: Prevent Cross-Site Scripting in ASP.NET,” Maio 2005. [Online]. Available: http://msdn.microsoft.com/en-us/library/ms998274.aspx. [Acedido em 7 Agosto 2012].

[106] T. Dykstra, “ASP.NET Web Application Projects vs. Web Site Projects in Visual Studio,” 14 Dezembro 2009. [Online]. Available: http://blogs.msdn.com/b/aspnetue/archive/2009/12/14/asp-net-web-application-projects-vs-web-site-projects-in-visual-studio.aspx. [Acedido em 13 Agosto 2012].

[107] Microsoft, “Develop, General Guidelines and Limitations (Windows Azure SQL Database),” 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/windowsazure/ee336245.aspx. [Acedido em 13 Agosto 2012].

[108] Microsoft, “Develop, Tools and Utilities Support (Windows Azure SQL Database),” 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/windowsazure/ee621784.aspx.

[Acedido em 13 Agosto 2012].

[109] Microsoft, "Develop, Configuring a Windows Azure Project," 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/windowsazure/ee405486.aspx. [Acedido em 14 Agosto 2012].

[110] Microsoft, "How to Create a Storage Account for a Windows Azure Subscription," 26 Outubro 2011. [Online]. Available: http://msdn.microsoft.com/en-us/library/windowsazure/gg433066.aspx. [Acedido em 8 Agosto 2012].

[111] Microsoft, "Windows Azure Managed Library, BlobContainerPublicAccessType Enumeration," 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/microsoft.windowsazure.storageclient.blobcontainerpublicaccesstype.aspx. [Acedido em 8 Agosto 2012].

[112] T. Redkar, Windows Azure Platform, Nova Iorque: Apress, 2009.

[113] W. W. Berry, "Using SQL Azure for Session State," 4 Agosto 2010. [Online]. Available: http://blogs.msdn.com/b/sqlazure/archive/2010/08/04/10046103.aspx. [Acedido em 18 Agosto 2012].

[114] Microsoft, "ASP.NET Universal Providers For SqlExpress," 31 Maio 2012. [Online]. Available: http://nuget.org/packages/System.Web.Providers. [Acedido em 18 Agosto 2012].

[115] Microsoft, "Develop, Using Remote Desktop with Windows Azure Roles," 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/windowsazure/gg443832.aspx. [Acedido em 15 Agosto 2012].

[116] Microsoft, "Understanding Windows Azure Storage Billing – Bandwidth, Transactions, and Capacity," MSDN, Windows Azure Storage Team Blog, 8 Julho 2010. [Online]. Available: http://blogs.msdn.com/b/windowsazurestorage/archive/2010/07/09/understanding-windows-azure-storage-billing-bandwidth-transactions-and-capacity.aspx. [Acedido em 19 Agosto 2012].

[117] S. Varghese, "Tips and Important Steps for Migrating Apps to Windows Azure," 29 Janeiro 2012. [Online]. Available: http://weblogs.asp.net/shijuvarghese/archive/2012/01/29/tips-and-important-steps-for-migrating-apps-to-windows-azure.aspx. [Acedido em 18 Agosto 2012].

[118] Microsoft, "Develop, How to: Configure the ASP.NET Session State Provider (Windows Azure Shared Caching)," 7 Junho 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/windowsazure/gg278339.aspx. [Acedido em 19 Agosto 2012].

[119] Microsoft, "Develop, Windows Azure ASP.NET Providers Sample," 29 Julho 2011. [Online]. Available: http://code.msdn.microsoft.com/windowsazure/windows-azure-aspnet-03d5dc14. [Acedido em 19 Agosto 2012].

[120] S. Mattia, "http://stackoverflow.com/questions/10932291/background-thread-in-azure-web-role," 7 Junho 2012. [Online]. Available:

http://stackoverflow.com/questions/10932291/background-thread-in-azure-web-role. [Acedido em 19 Agosto 2012].

[121] F. Cory, "Develop, Real World: Startup Lifecycle of a Windows Azure Role," Windows Azure, 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/windowsazure/hh127476.aspx. [Acedido em 19 Agosto 2012].

[122] Microsoft, "New Full IIS Capabilities: Differences from Hosted Web Core," 2 Dezembro 2010. [Online]. Available: http://blogs.msdn.com/b/windowsazure/archive/2010/12/02/new-full-iis-capabilities-differences-from-hosted-web-core.aspx. [Acedido em 20 Agosto 2012].

[123] N. Godinho, "Importance of Windows Azure Affinity Groups," 4 Março 2012. [Online]. Available: http://social.technet.microsoft.com/wiki/contents/articles/7916.importance-of-windows-azure-affinity-groups.aspx. [Acedido em 20 Agosto 2012].

[124] Microsoft, "Develop, Enabling Diagnostics in Windows Azure," 2012. [Online]. Available: https://www.windowsazure.com/en-us/develop/net/common-tasks/diagnostics/. [Acedido em 15 Agosto 2012].

[125] Microsoft, "Microsoft Patterns & Practices, Autoscaling and Windows Azure," Junho 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/hh680945%28v=pandp.50%29.aspx. [Acedido em 16 Agosto 2012].

[126] Melnik, Grigori, "Autoscaling Windows Azure applications," 12 Setembro 2011. [Online]. [Acedido em 16 Agosto 2012].

[127] J. D. Meier, S. Vasireddy, A. Babbar, R. Mariani e A. Mackman, "Microsoft Patterns & Practices, Chapter 15 - Measuring .NET Application Performance," Maio 2004. [Online]. Available: http://msdn.microsoft.com/en-us/library/ms998579.aspx. [Acedido em 17 Agosto 2012].

[128] Amazon, "AWS Economics Center," 2012. [Online]. Available: http://aws.amazon.com/en/economics/. [Acedido em 19 Setembro 2012].

[129] TechTarget, "Mainframe operating systems and management, TCO (total cost of ownership)," Setembro 2005. [Online]. Available: http://searchdatacenter.techtarget.com/definition/TCO. [Acedido em 18 Setembro 2012].

[130] Saugatuck Technology, "The Cloud Bottom Line: Calculating TCO and ROI for Business Applications in the Cloud with Bruce Guptill," 6 Outubro 2011. [Online]. Available: http://www.enterpriseefficiency.com/webinar.asp?webinar_id=29737&caschk=yes. [Acedido em 19 Setembro 2012].

[131] Saugatuck Technology, "The Cloud Bottom Line: Calculating TCO and ROI for Business Applications in the Cloud with Bill Kirwin," 6 Outubro 2011. [Online]. Available: http://www.enterpriseefficiency.com/webinar.asp?webinar_id=29737&caschk=yes. [Acedido em 19 Setembro 2012].

Esta página foi intencionalmente deixada em branco

11. Anexos

Anexo I – Custo da solução da Amazon

Dado que o custo dos serviços varia consoante a região geográfica escolhida as tabelas deste anexo são referentes ao(s) *datacenter(s)* da Amazon na Irlanda. Os preços indicados (Tabela 15-18) estão em dólares americanos e correspondem a 20 de Julho de 2012.

	Linux/UNIX Usage	**Windows Usage**
Standard On-Demand Instances		
Small (Default)	$0.085 per Hour	$0.115 per Hour
Medium	$0.170 per Hour	$0.230 per Hour
Large	$0.340 per Hour	$0.460 per Hour
Extra Large	$0.680 per Hour	$0.920 per Hour
Micro On-Demand Instances		
Micro	$0.020 per Hour	$0.020 per Hour
High-Memory On-Demand Instances		
Extra Large	$0.506 per Hour	$0.570 per Hour
Double Extra Large	$1.012 per Hour	$1.140 per Hour
Quadruple Extra Large	$2.024 per Hour	$2.280 per Hour
High-CPU On-Demand Instances		
Medium	$0.186 per Hour	$0.285 per Hour
Extra Large	$0.744 per Hour	$1.140 per Hour
Cluster Compute Instances		
Quadruple Extra Large	N/A*	N/A*
Eight Extra Large	$2.700 per Hour	$2.970 per Hour
Cluster GPU Instances		
Quadruple Extra Large	N/A*	N/A*
High-I/O On-Demand Instances		
Quadruple Extra Large	$3.410 per Hour	$3.580 per Hour
* Cluster GPU Instances are not available in all regions		

Tabela 15 - Tabela de preço do EC2[37]

	Standard Storage
First 1 TB / month	$0.125 per GB
Next 49 TB / month	$0.110 per GB
Next 450 TB / month	$0.095 per GB
Next 500 TB / month	$0.090 per GB
Next 4000 TB / month	$0.080 per GB
Over 5000 TB / month	$0.055 per GB

Tabela 16 - Tabela de preço do S3[38]

	Pricing
PUT, COPY, POST, or LIST	$0.01 per 1,000 requests

[37] http://aws.amazon.com/ec2/pricing/
[38] http://aws.amazon.com/s3/pricing/

Requests	
GET and all other Requests *	$0.01 per 10,000 requests
* No charge for delete requests	

Tabela 17 - Custo das transações de armazenamento no AWS[39]

	Pricing
Data Transfer IN	
All data transfer in	$0.000 per GB
Data Transfer OUT	
First 1 GB / month	$0.000 per GB
Up to 10 TB / month	$0.120 per GB
Next 40 TB / month	$0.090 per GB
Next 100 TB / month	$0.070 per GB
Next 350 TB / month	$0.050 per GB

Tabela 18 - Custo da transferência de dados no AWS[40]

[39] http://aws.amazon.com/s3/pricing/
[40] http://aws.amazon.com/s3/pricing/

Anexo II – Custo da solução da Google

Os preços indicados (Tabela 19-22) estão em dólares americanos e estão atualizados a 20 de Julho de 2012.

	Memory limit	**CPU limit**	**Cost per hour per instance**
Frontend class			
F1 (default)	128MB	600MHz	$0.08
F2	256MB	1.2GHz	$0.16
F4	512MB	2.4GHz	$0.32
Backend class			
B1	128MB	600MHz	$0.08
B2 (default)	256MB	1.2GHz	$0.16
B4	512MB	2.4GHz	$0.32
B8	1024MB	4800MHz	$0.64

Tabela 19 - Tabela de preços do GAE, Frontends[41] e Backends[42]

Monthly Usage	**Price (per GB)**
First 0 - 1TB	$0.12
Next 9TB	$0.105
Next 90TB	$0.095
Next 400TB	$0.085

Tabela 20 - Tabela de preço do Google Cloud Storage[43]

	Pricing
PUT, POST, GET bucket*, GET service* Requests (per 1,000 requests/month)	$0.01
GET, HEAD Requests (per 10,000 requests/month)	$0.01
DELETE Requests	Free
* GET bucket requests are API calls to list objects and GET service requests are API calls to list buckets	

Tabela 21 - Custo das transações de armazenamento na Google Cloud Platform[44]

Monthly Usage	**Network (Egress) - Americas and EMEA* (per GB)**	**Network (Egress) - Asia-Pacific (per GB)**	**Network (Ingress)**
0 - 1TB	$0.12	$0.21	Free
Next 9TB	$0.11	$0.18	
Next 90TB	$0.08	$0.15	
*Europe, the Middle East, and Africa (EMEA)			

Tabela 22 - Custo da transferência de dados na Google Cloud Platform[45]

[41] https://developers.google.com/appengine/docs/adminconsole/performancesettings
[42] https://developers.google.com/appengine/docs/java/backends/overview
[43] https://developers.google.com/storage/docs/pricingandterms
[44] https://developers.google.com/storage/docs/pricingandterms

Anexo III - Autenticação e Autorização

Num sítio nem todas as páginas devem ser públicas sendo pois necessário usar um mecanismo de segurança que providencie Autenticação e Autorização.

A escolha de um esquema de autenticação é feita alterando o atributo "*mode*" do elemento "*/configuration/system.web/authentication*" no ficheiro *Web.config*. A configuração relacionada com a autorização também é feita no mesmo ficheiro nomeadamente a definição dos recursos a que determinado utilizador tem acesso baseado nas suas valências e a definição do código responsável por verificar essas mesmas valências. As valências no ASP.NET têm o nome de *Roles*. *Roles* são perfis de utilizadores que contêm um conjunto de permissões de acesso. Os *Roles* são adicionados ou retirados aos utilizadores e com isso damos ou retiramos permissões de acesso aos utilizadores daquele *Role*.

As permissões de acesso a determinado recurso (Autorização) podem ser dadas a uma página individual ou a uma localização no servidor *Web*. Na Listagem 10 é mostrado como a pasta *Admin* apenas é acessível aos Administradores do sítio.

```
<location path="Admin">
  <system.web>
    <authorization>
      <allow roles="Admin" />
      <!-- Deny all users -->
      <deny users="*" />
    </authorization>
  </system.web>
</location>
```

Listagem 10 - Exemplo de autorização de acesso a um recurso do sítio

O esquema selecionado para autenticação no sítio *CloudShop* foi o Forms. Tanto o método que lida com a autenticação como o método que lida com a autorização usam a mesma base de dados que armazena o resto da informação da aplicação. A autenticação é realizada via o evento *Authenticate* do *WebControl Login*. A autorização é implementada através de um *Role Provider*. A classe responsável pela autorização (ARoleProvider) descende da classe RoleProvider, graças à herança a classe é compatível com os controlos ASP.NET e com o controlo de acesso exemplificado na Listagem 10. O diagrama de sequência da Ilustração 24 mostra como é tratado o pedido a uma página restrita por parte de um

[45] https://developers.google.com/storage/docs/pricingandterms, acedido a 20 de Julho de 2012.

utilizador anónimo e pretende mostrar como funciona todo o mecanismo de Autenticação e Autorização implementado.

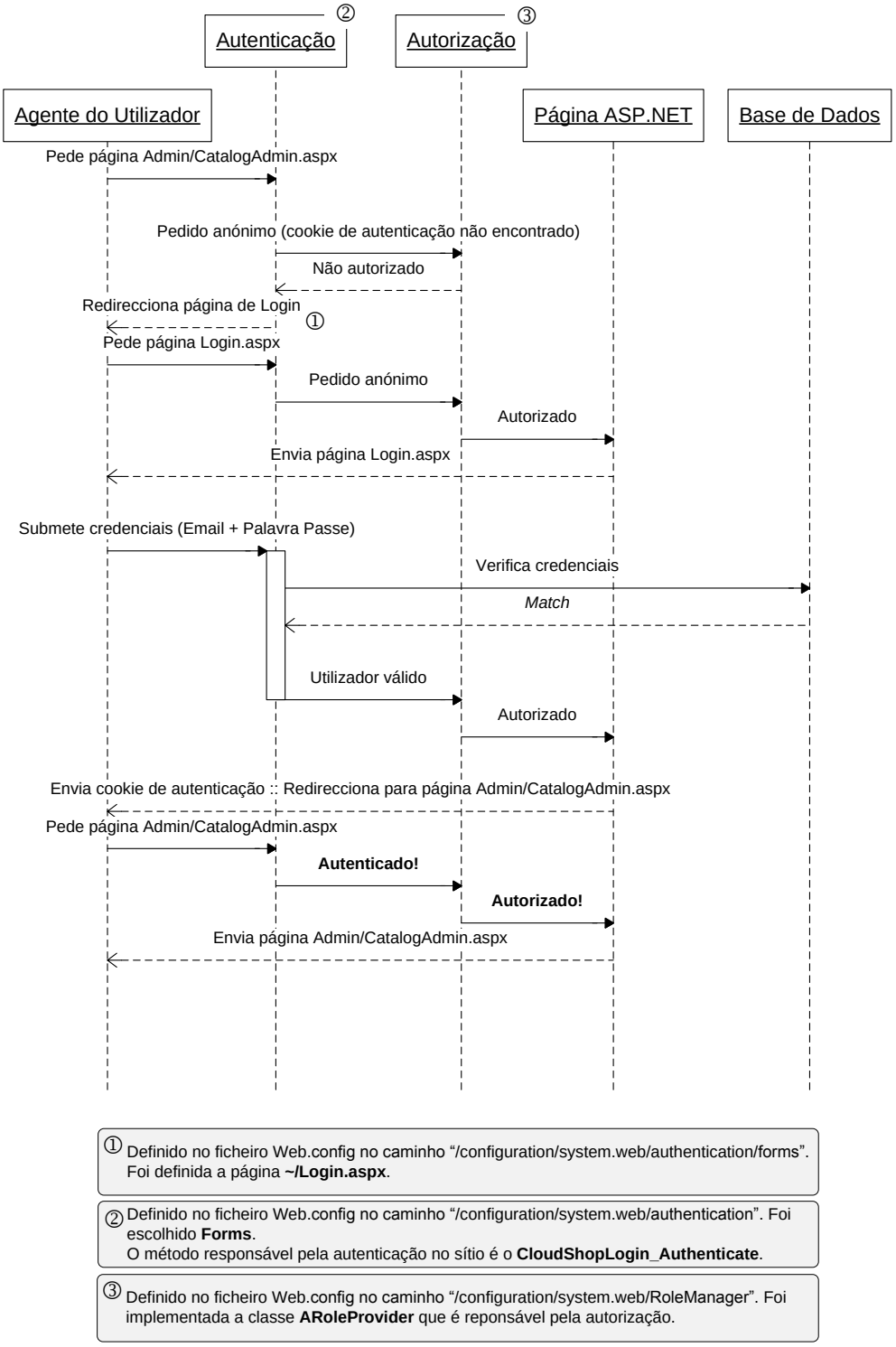


Ilustração 24 - Diagrama de sequência do processo de Autenticação e Autorização[46]

[46] http://www.asp.net/web-forms/tutorials/security/introduction/security-basics-and-asp-net-support-cs, acedido em 9 de Agosto de 2012.

Anexo IV - Integração com Motor de Pagamentos

Optou-se neste projeto por usar a *API SOAP* e implementar o "Express Checkout". O "Express Checkout" é uma das opções de integração com o PayPal tendo como principal vantagem permitir que o programador implemente o seu próprio carrinho de compras e controle o processo de pagamento[47]. Para uma correta implementação devem ser chamados por ordem, os seguintes três métodos da *API*: *SetExpressCheckout*, *GetExpressCheckoutDetails* e *DoExpressCheckoutPayment*, como pode ser observado pelo diagrama retirado do manual de integração (Ilustração 25).

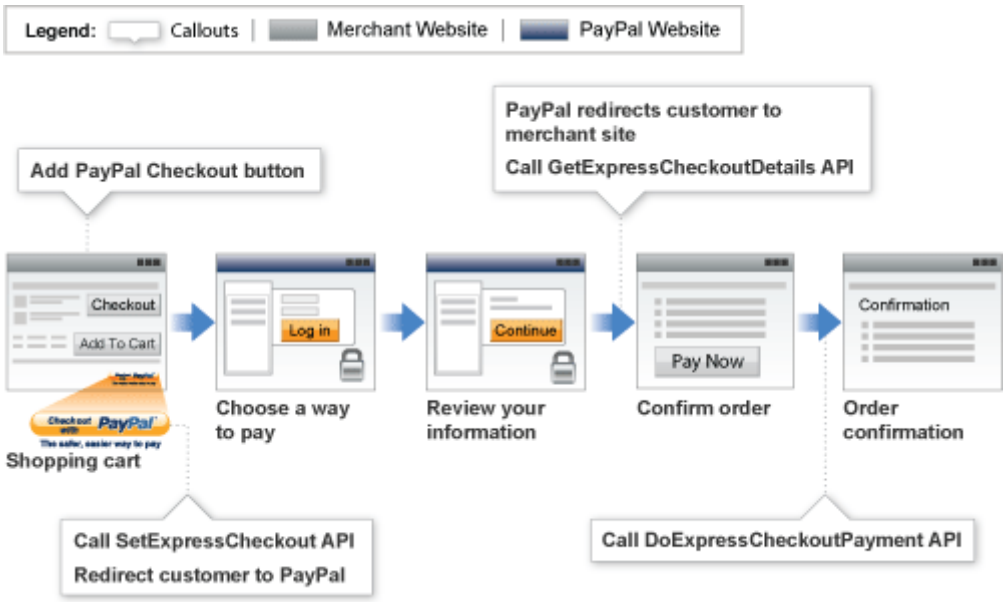


Ilustração 25 - Diagrama conceptual do Express Checkout[47]

É recomendado também que o programador ligue o "Express Checkout" à *Sandbox*. A *Sandbox* é um ambiente que simula o PayPal e que permite validar o correto funcionamento da aplicação sem se enviar dinheiro real. Por forma a testar a aplicação o programador deve ter duas contas uma para se autenticar como vendedor e outra para se autenticar como comprador. Antes que possa utilizar qualquer uma delas o programador deve estar autenticado na página *https://developer.paypal.com*[47]. No final do programador confirmar o funcionamento do "Express Checkout" com a *Sandbox* basta alterar para os servidores de produção mantendo o código intacto. No diagrama da Ilustração 26 é

47 https://cms.paypal.com/cms_content/US/en_US/files/developer/PP_ExpressCheckout_IntegrationGuide.pdf, acedido em 10 de Agosto de 2012.

mostrado a sequência de ações relativas ao *checkout* de um carrinho de compras por parte de um utilizador autenticado como cliente.

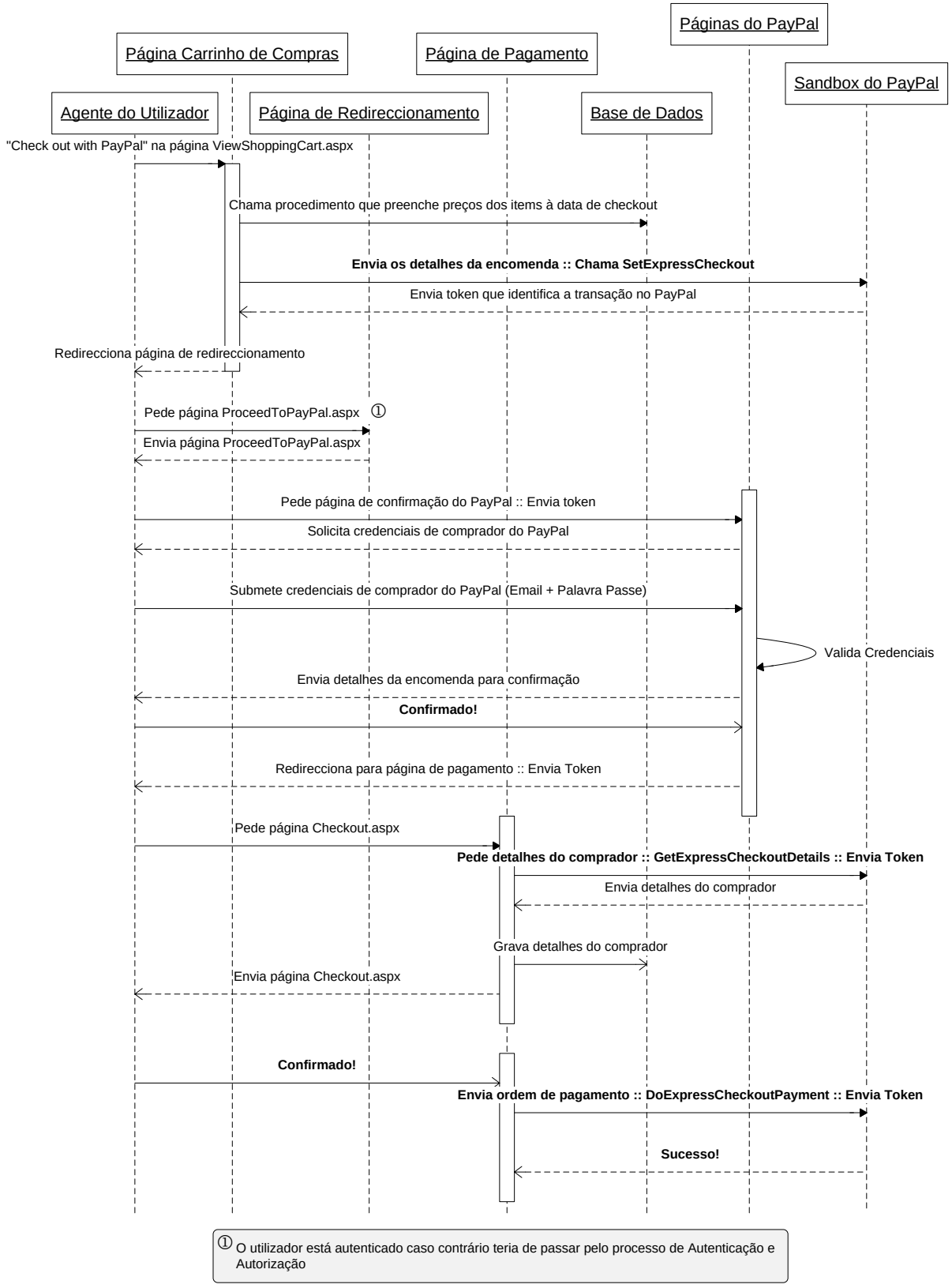


Ilustração 26 - Diagrama de sequência do processo de checkout[47]

Anexo V - Diagrama de Entidade Relacionamento

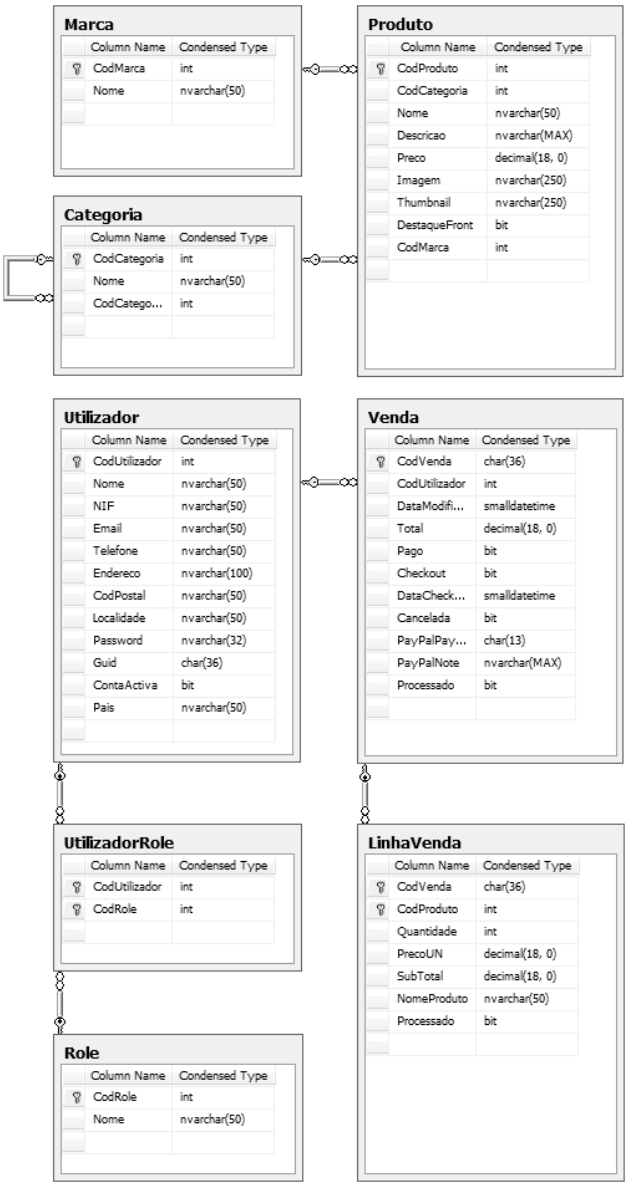


Ilustração 27 - Diagrama de Entidade Relacionamento (DER) do sítio CloudShop

A Ilustração 27 apresenta o *DER* da aplicação. A utilidade da maior parte das tabelas e colunas é fácil de aferir. Os elementos menos óbvios são explicados de seguida:

Tabela Categoria – Armazena categorias e subcategorias de produtos. Para esse efeito a tabela está relacionada com ela própria.

Tabela LinhaVenda – O Carrinho de Compras está implementado de modo a que o preço dos artigos seja coerente à data do *checkout*. Para que isto seja possível, o preço dos artigos

não é guardado quando eles são adicionados ao carrinho e as colunas relativas a preços (PrecoUN e SubTotal) somente são preenchidas quando é feito o *checkout*. Quando é feito o *checkout* o sistema itera pelos registos da LinhaVenda referentes ao mesmo CodVenda. Cada vez que o preço de um artigo é preenchido a *flag* Processado é colocada a 1 indicando que aquela registo já foi processado.

Tabela Marca – Armazena as marcas dos produtos.

Tabela Produto – A coluna Imagem e a coluna Thumbnail da tabela Produto armazenam o caminho para a imagem correspondente. A coluna DestaqueFront é uma *flag* que serve para colocar o produto em destaque no catálogo. Deste modo, quando o utilizador aceder ao catálogo sem ter definido nenhuma categoria os produtos em destaque ser-lhe-ão apresentados.

Tabela Role – Define as diferentes valências que um utilizador pode ter no sítio. O sítio define duas valências: Admin (administrador) e Client (cliente).

Tabela Utilizador – A coluna Password armazena uma *hash* da palavra-passe do utilizador. Se por algum motivo alguém não autorizado tiver acesso a esta coluna não deverá ser capaz de averiguar a palavra-passe que gerou a *hash* armazenada. A coluna Guid armazena o *ID* único gerado aquando do registo do utilizador no sítio, este *ID* é enviado por email para o endereço indicado pelo utilizador. A conta só é ativada quando o utilizador validar o endereço de email seguindo o *URL* que contém o *ID* e o CodUtilizador. A coluna ContaActiva é uma *flag* que serve para Activar/Desactivar a conta de utilizador.

Tabela Utilizador Role – Tabela de relacionamento entre a tabela Utilizador e a tabela Role.

Tabela Venda – A coluna DataModificacao indica a data da última alteração à venda e é utilizada para apagar as vendas antigas que não foram completadas, aquelas em que o utilizador não chegou a fazer checkout. A *flag* Checkout indica que o processo de checkout foi completado com sucesso na data indicada pela coluna DataCheckOut, se a *flag* estiver a 1 o utilizador já pagou pelos artigos tendo completado todo o processo de checkout. A *flag* Cancelada indica que a venda foi cancelada por vontade do utilizador. Os campos PayPalPayerID e PayPalNote são relativos à integração com o motor de pagamentos. A *flag* Processado indica que o Total da venda foi calculado.

Anexo VI – Converter projeto de Web Site para Web Application

- Criar um projeto do tipo "ASP.NET Empty Web Application" no Microsoft Visual Studio;

- Abrir a pasta do projeto recentemente criado e copiar todos os ficheiros do Web Site para o diretório;

- Habilitar a visualização de todos os ficheiros presentes na pasta do projeto (Ilustração 28);

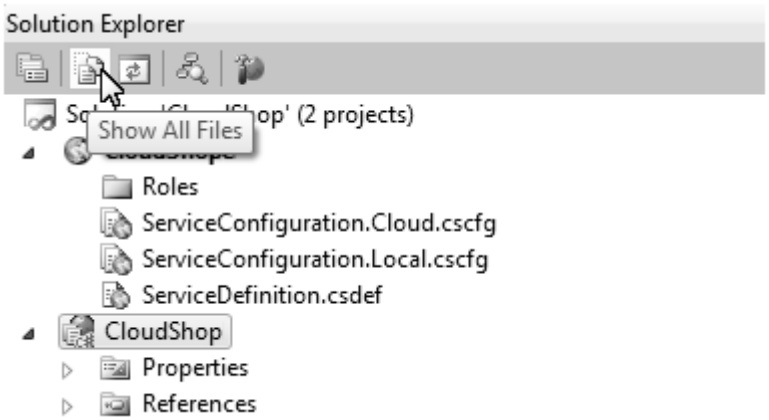


Ilustração 28 - Mostrar todos os ficheiros da pasta do projeto no Visual Studio

- Incluir os ficheiros que aparecem na Web Application através da opção do menu de contexto "Include In Project" (Ilustração 29);

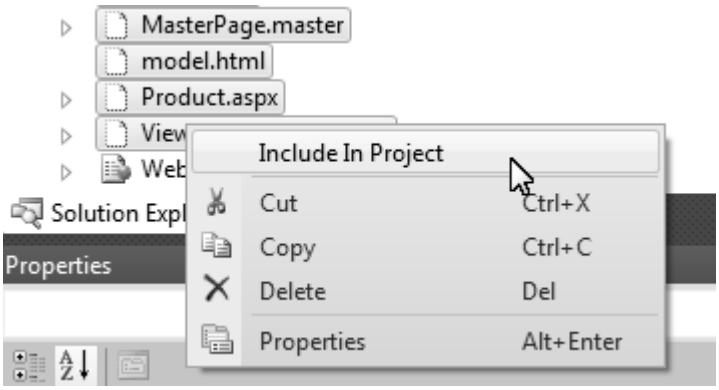


Ilustração 29 - Incluir ficheiros do projeto Web Site na Web Application

- Para finalizar clicar com o botão do lado direito no projeto e escolher a opção "Covert to Web Application" do menu de contexto.

Anexo VII – Gerar certificados para a interação do Visual Studio com a API de Gestão do Windows Azure

A criação do certificado permite que as ferramentas interajam com a API de Gestão do Windows Azure, conferindos-lhe a possibilidade de criar, apagar e modificar serviços e contas de armazenamento[48]. O certificado a ser criado é um certificado X.509 auto-assinado que contém quer a chave privada quer a chave pública, a chave privada irá ser instalada no repositório pessoal de certificados do utilizador ao passo que, a chave pública irá ser exportada para o Windows Azure[49]. O uso da criptografia assimétrica permite que a API de Gestão verifique a autenticidade e integridade dos pedidos feitos pelo Visual Studio. Para a criação do certificado utilizou-se a própria ferramenta de desenvolvimento que oferece esta opção aquando do primeiro envio da aplicação para a nuvem (*deploy*):

- Para se fazer o *deploy* usa-se o menu de contexto do projeto Windows Azure Cloud Service (Ilustração 30);

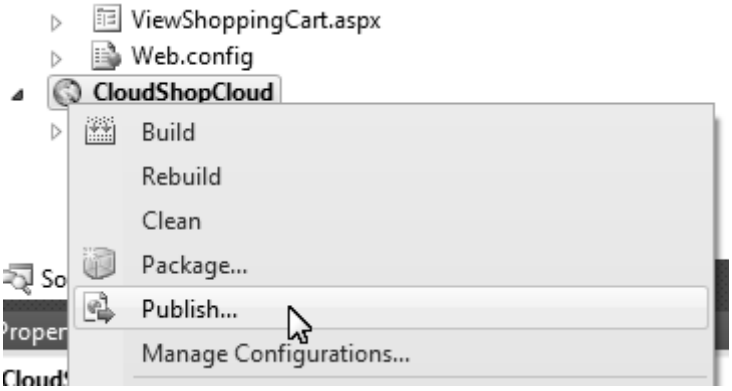


Ilustração 30 - Envio da aplicação para a nuvem através do Visual Studio

- Na janela de configuração de acesso à API de gestão foi escolhida a opção "<Create...>" para que fosse criado um novo certificado X.509 (Ilustração 32) sendo-lhe dado um nome que o permite identificar entre os certificados presentes no repositório pessoal de certificados do utilizador;

[48] http://msdn.microsoft.com/en-us/library/windowsazure/gg551722.aspx, acedido em 15 de Agosto de 2012.
[49] http://msdn.microsoft.com/en-us/library/windowsazure/gg551722.aspx, acedido em 15 de Agosto de 2012.

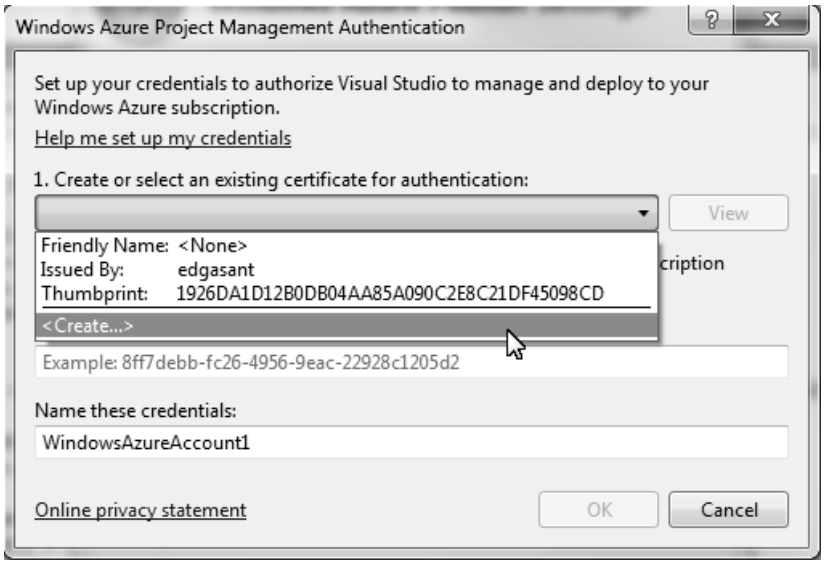


Ilustração 31 - Criar um novo certificado X.509 para o acesso à API de Gestão do Windows Azure

- Foi copiado para a área de transferência o caminho para o certificado que contém a chave pública (ponto 2 da Ilustração 32). Este certificado foi exportado através do certificado X.509 criado no ponto 1 sendo utilizado para associar o Visual Studio com a subscrição do Windows Azure que pretendemos gerir;

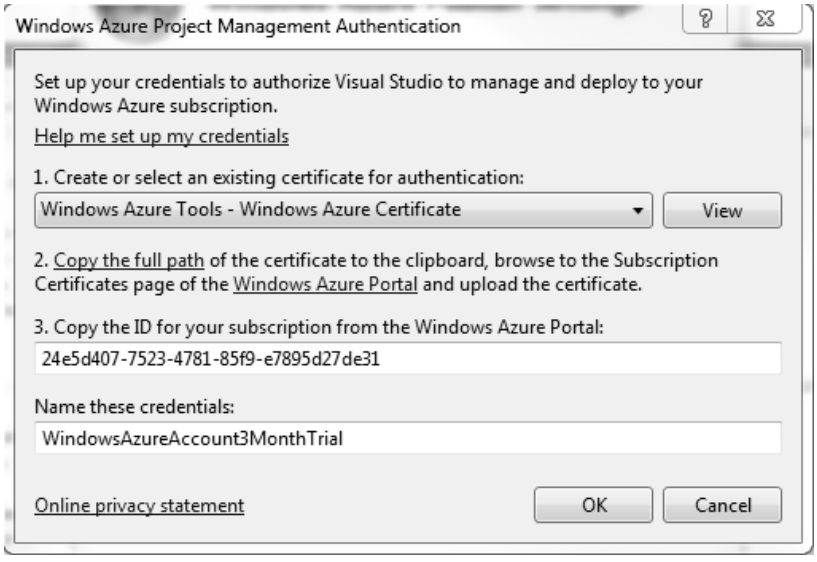
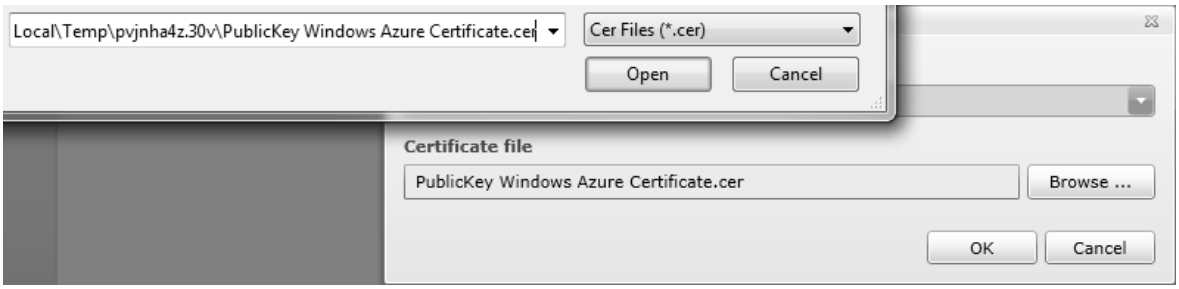


Ilustração 32 - Janela para a criação e associação do certificado X.509 ao Windows Azure

- De seguida foi aberto o portal de administração e na secção Hosted Services, Storage Accounts & CDN > Management Certificates foi importado o certificado com a chave pública;



- Por fim foi copiado e colado o ID da subscrição (ponto 3 da Ilustração 32) e foi clicado em OK.

Anexo VIII – Integração do Paraleap AzureWatch com a subscrição do Windows Azure

Para integrar o Paraleap AzureWatch é necessário que a ferramenta tenha acesso à API de gestão da subscrição. Adicionalmente é necessário que a ferramenta tenha acesso à conta de armazenamento onde estão armazenadas as métricas recolhidas através do Windows Azure Diagnostics Monitor[50]. Para integrar o Paraleap AzureWatch é necessário:

- Efetuar o registo na página da Paraleap[51];
- Descarregar e Instalar o "AzureWatch Control Panel" (ferramenta onde são definidas as métricas e as regras, gerados gráficos etc.) e o "AzureWatch Agent" (um agente que mostra o estado da monitorização das instâncias);
- Abrir o "AzureWatch Control Panel" e providenciar as credenciais de registo (Ilustração 33);

Ilustração 33 – Introdução das credenciais de acesso no "AzureWatch Control Panel"

- De seguida será apresentada uma janela onde será pedido o ID da subscrição do Windows Azure, o certificado de acesso à API de gestão (o mesmo do Anexo VII) e a conta de armazenamento onde estão guardadas as métricas (Ilustração 34). É muito importante marcar a opção "Monitor from AzureWatch servers", o que esta opção faz é monitorizar as instâncias a partir dos servidores da Paraleap, se esta

[50] https://www.windowsazure.com/en-us/develop/net/common-tasks/diagnostics/, acedido em 16 de Agosto de 2012
[51] http://www.paraleap.com, acedido em 16 de Agosto de 2012

opção não fosse usada a monitorização só ocorria enquanto o utilizador tivesse a aplicação *desktop* a correr;

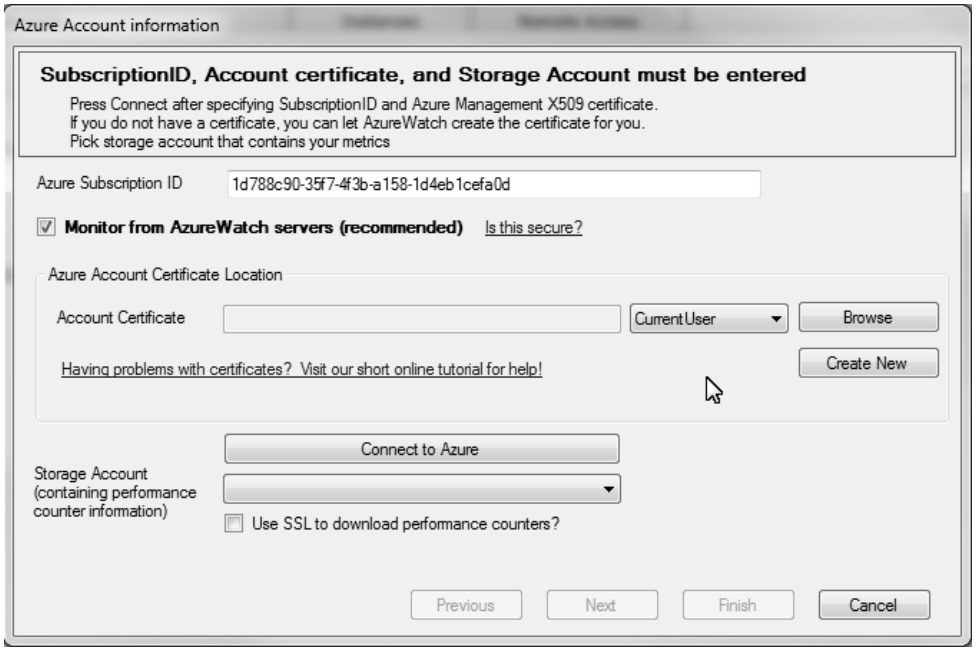


Ilustração 34 - Ligação à API de gestão e à conta de armazenamento no "AzureWatch Control Panel"

- Por fim é definida uma conta de email para notificações (Ilustração 35);

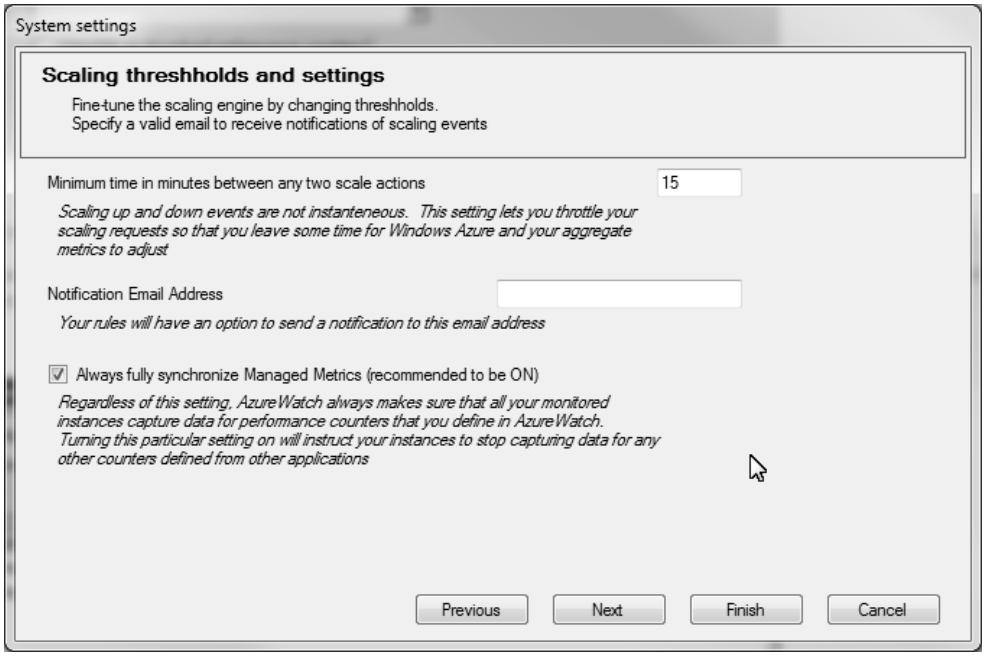


Ilustração 35 - Definir conta de email para notificações

Anexo IX – Custo detalhado das soluções usadas para o cálculo do TCO

Os valores que se encontravam em euros (*EUR*) foram convertidos em dólares americanos (*USD*) sendo que, os valores bem como a taxa de câmbio correspondem a 14 de Setembro de 2012 (Tabela 23-25).

	Qty	Descrição	Preço	Sub-Total EUR	Sub-Total USD ***
Licenças					
	1	Microsoft Windows Server Standard Edition OLP (Open Licensing) NL (No Level) 2 Processors P/N: P73-05762	1119,99	1119,99	1461,027
	1	SQL Server Standard Edition OLP NL P/N: 228-09884	1138,81	1138,81	1485,578
Equipamento					
	1	UPS Online HP R/T3000 G2 Potência: 3300VA ** Factor de Forma: 2U P/N: AF468A	1550,00	1550,00	2012,975
	1	Painel frontal preto 19" com guia de cabos de 5 ganchos P/N: RF 3301	26,29	26,29	34,295
	1	Estante aberta 19" chão 600x600 22U preta P/N: PHP 7022	183,60	183,60	239,506
	4	Disco rígido HP 1 TB 3G SATA 7,2 K RPM P/N: 454146-B21	307,50	1230,00	1604,535
	2	Memória 2GB PC3-106000 (DDR-1333) Registered CAS-9 P/N: 500670-B21	49,20	98,40	128,363
	1	HP ProLiant DL180 G6 Processador: Intel Xeon E5606 (4 cores, 2.13GHz) * Memória: 2 x 2GB PC3-106000 (DDR3-1333) Registered CAS-9 Rede: HP NC362i	1260,75	1260,75	1644,648

		Alimentação: Factor de Forma: P/N:	Integrated Dual Port Gigabit Server Adapter HP 460W Common Slot Gold Hot Plug Power Supply Kit 2U 641363-425			
* O processador presente no servidor da Loja A foi lançado no quarto trimestre de 2007 e já não se encontra disponível por isso, foi escolhido um equivalente[52] ** 40 minutos de autonomia com uma carga exigida de 20% (600W)[53] *** Taxa de câmbio de referência diária publicada pelo Banco de Portugal 1 EUR = 1,3095 USD						

Tabela 23 - Custo não corrente da solução on-premises

Qty	Descrição	Custo/ Mês	Sub-Total EUR	Sub-Total USD ***
1/2	Contrato de Assistência Técnica "Contrato PME" Prazo máximo de resposta: 6 horas (remoto) 12 horas (presencial) Horas de assistência incluídas: 12 horas Deslocações: Gratuitas	567,03	283,52	369,852
252	kWh (720 @ 350W) *	35,10	35,10	45,788
1/2	Ligação à Internet "Sapo Fibra 100MB" Largura de banda download: 100MB Largura de banda upload: 10MB	41,00	20,50	26,74
* 350W durante 720 horas segundo a tarifa simples da EDP, Serviço Universal SA 1 kWh = 0,1393 EUR ** Taxa de câmbio de referência diária publicada pelo Banco de Portugal 1 EUR = 1,3095 USD *** Taxa de câmbio de referência diária publicada pelo Banco de Portugal 1 EUR = 1,3095 USD				

Tabela 24 - Custo corrente da solução on-premises

Qty	Descrição	Horas	Dias	Custo /Hora	Custo /Dia	Custo /Mês	Sub-Total USD
2	Large Web Role	720		0,48		345,60	691,20
1	Base de Dados 10GB *					45,954	45,954

[52] http://ark.intel.com/compare/33927,52583
[53] http://h20195.www2.hp.com/v2/GetPDF.aspx/c02889155.pdf

1	Conta de Armazenamento 100GB *					12,50	12,50
2	Instâncias Monitorizadas no AzureWatch (Ponto 7.1)		30		0,330	9,900	19,800
* Dimensionado de acordo com a base de dados e o espaço de armazenamento do servidor da Loja A							

Tabela 25 - Custo da solução no Windows Azure