MASARYK UNIVERSITY

FACULTY OF INFORMATICS

Analysis of differences in web system development for on-premises and cloud computing

BACHELOR THESIS

Martin Novák

May, 2012

Brno, Czech Republic

Declaration

Hereby I declare that this paper is my original authorial work, which I have worked out by my own. All sources, references and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

Thesis supervisor: Ing. RNDr. Barbora Bühnová, Ph.D.

Acknowledgement

I wish to thank to people who helped with the professional realization of my thesis. My gratitude belongs to *Ing. RNDr. Barbora Bühnová, Ph.D.* for supervising my thesis and her patient help, *Bc. Petr Kunc* and *Bc. Pavel Smolka* for their comments on logging and diagnostics implementation and last but not least *doc. RNDr. Tomáš Pitner, Ph.D.* for his support and trust during whole time of my studies.

Abstract

The thesis covers analysis of differences in development of on-premises and cloud computing systems with focus on cloud computing platform Windows Azure. The analysis deals with data persistence, failovers, testing, backups, security and availability.

Practical knowledge of performance is tested during development and deployment of a tool for performance logging and diagnostics in an open source system Celebrio Kernel.

Keywords

Cloud computing, platform as a service, on-premises, web system, Windows Azure, Celebrio Kernel

Contents

Chapter 1

Introduction

1.1 Background and motivation

Cloud computing is a large datacenter computing power offered as a service through its resources being shared among a number of consumers and available over a network. Cloud computing offers infrastructure, platform and/or software as a service. Companies which were developing their web systems on-premises can be in many cases more cost effective by moving their systems to the cloud and reaching for seemingly unlimited resources available on-demand.

Although the cloud computing providers claim that a common web system can be moved to a cloud nearly without any changes in its code, the real-life experience shows that to be able to fully exploit its advantages, the web system must be designed or modified for the specific environment.

1.2 Aim of the thesis

The thesis aims to analyze the differences between the development of a common web system on-premises and cloud computing with the focus on local development, data backup, performance diagnostics, testing, security, data persistence and connection among subservices.

As cloud computing services are being paid on demand it is very important for the web system designers to be able to analyze it through various diagnostics. The thesis takes Windows Azure platform as a service to demonstrate a web system design for cloud computing.

Celebrio Kernel is a free software technology used to test the thesis conclusions in real life through a new diagnostics module usable in both on-premises and cloud computing environment.

1.3 Thesis outline

The thesis consists of seven chapters.

The original work of the thesis is contained in chapters 4-7.

Chapter 1 Introduction describes background and motivation of the work, introduces goals and shortly discusses the thesis outline.

Chapter 2 State of the art covers the current on-premises options and technology available in cloud computing.

Chapter 3 Context outlines the Windows Azure platform technology including services such as Compute, SQL Azure, storage and others.

Chapter 4 Analysis discusses the differences between on-premises and cloud computing environments covering server distribution, development process, security, availability, failovers, backups and testing.

Chapter 5 Diagnostics describes the requirements and development of a performance logging module for an open-source system Celebrio Kernel.

Chapter 6 Evaluation covers the results of performance testing of an on-premises server, Windows Azure emulator and Windows Azure cloud.

Chapter 7 Conclusion summarizes an analysis of various features of on-premises and cloud computing environments.

Chapter 2

State of the art

2.1 On-premises

On-premises web systems run on the premises (in the building) of the company or organization it belongs to. The company usually manages everything itself including the hardware, connectivity, security, operating system, storage, database, backup, updating and failovers.

These solutions typically include only one server and are not expected to deal with a distribution among higher number of instances.

The lasting advantage of such approach is a total control over the whole hardware and software environment.

2.2 Server housing

In this case the company or organization has its servers in a building of a server-housing provider. The advantage lies in an absence of the need to deal with the hardware but still having total control over the software.

The company has to handle itself the software environment security and manage operating system, storage, database and usually also the backups and failovers.

The service user pays full price for all of the used commercial server software.

Server housing has very limited options of scaling compared to cloud computing.

2.3 Cloud computing

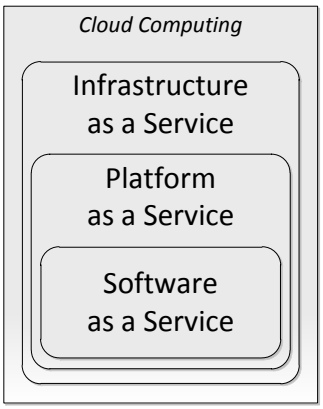

Figure 1: Cloud computing service model

Cloud computing is a form of distributed computing introducing utilization for remote provisioning of scalable resources with main benefits of reduced investment and proportional costs, increased scalability, availability and reliability. [1]

Cloud references to abstraction and virtualization. Abstraction hides the details of system specifications from users and developers. Virtualization pools and shares scalable resources than can be provisioned with enabled multi-tenancy. [2]

By a service model cloud computing solutions are divided into "infrastructure as a service", "platform as a service" or "software as a service". [2]

2.3.1 Infrastructure as a service (IaaS)

The provider offers computing power, storage, basic solution for backup, server failovers, load balancing and various additional services.

The service user manages all the software themselves which makes this service similar to server housing approach including an obligation to pay for software licenses. [2]

Examples: Amazon Web Services[1], Rackspace[2]

2.3.2 Platform as a service (PaaS)

The service provider offers similar services as in the case of IaaS but everything comes preconfigured as a whole platform so the developers need only to make their own software system compatible with the target platform. [2]

In this approach the development can be much faster and easier without the need of own server specialist but the user has less control over the environment. Also all of the server software is a part of the service and there is no need to pay any software licenses.

Also some PaaS can be built on top of IaaS, for example Red Hat's OpenShift[3] runs on top of Amazon Web Services.

Examples: Google App Engine[4], Windows Azure[5]

2.3.3 Software as a service (SaaS)

The service provider rents its software as a service rather than as a product so the user pays only for the usage without any long-term commitment and doesn't need to care about anything but the software itself.

Examples: Google Docs[6], Windows Live[7]

[1] http://aws.amazon.com/
[2] http://www.rackspace.com/
[3] https://openshift.redhat.com/
[4] https://developers.google.com/appengine/
[5] http://www.windowsazure.com/
[6] https://docs.google.com/
[7] https://home.live.com/

Chapter 3

Context

3.1 Windows Azure

3.1.1 Overview

Windows Azure is a cloud computing platform as a service from Microsoft, offering their common on-premises server software through the cloud. Microsoft itself uses Windows Azure to run its solutions such as Office 365[8], Bing[9], Windows Live and others. [3]

Windows Azure is convenient especially for .NET programming framework[10] developers used to Microsoft Windows Server[11] and Microsoft SQL Server[12] but it also allows development using programming languages supported by Internet Information Services[13] such as Java[14] or PHP[15].

3.1.2 Compute

Compute[16] is a Windows Azure hosted service that runs the developer's code on virtual machines of three types called "roles". [4]

- A Web role provides Internet Information Services to run a web system based for example on ASP.NET[17] or PHP.
- A Worker role runs a .NET framework code in a cycle to run backend applications for example for processing data retrieved through a web role.
- A Virtual Machine role allows deploying Windows Server image to Windows Azure to allow a higher level of server customization.

3.1.3 Database and storages

Windows Azure offers database server, local storage and centralized storages. [4]

- SQL Azure[18] is a database server as a service based on MS SQL Server

[8] http://office365.microsoft.com
[9] http://www.bing.com/
[10] http://www.microsoft.com/net
[11] http://www.microsoft.com/windowsserver/
[12] http://www.microsoft.com/sql/
[13] http://www.iis.net/
[14] http://www.java.com/
[15] http://php.net/
[16] http://www.windowsazure.com/en-us/home/features/compute/
[17] http://www.asp.net/
[18] http://www.windowsazure.com/en-us/home/features/sql-azure/

- Local storage allows developers to save their data locally on the server instance in a case that you can handle that the data will not be fully persistent and will not be distributed among instances.
- Centralized storages[19] are Tables, Queue, BLOB storage and Windows Azure Drive:
    - Tables store structured data without relations organized into partitions and rows
    - Queue is a simple queue storage designed mainly for communication among web and worker roles
    - BLOB storage is designed to store large digital data from documents to multimedia
    - Windows Azure Drive allows roles to mount NTFS Virtual Hard Drive

3.1.4 Other services

Windows Azure offers many other services such as caching shared among Compute roles, content delivery network, identity, access control and others[20]. [5]

3.2 Celebrio Kernel

Celebrio Kernel is free software that is being actively developed by Celebrio Software, s.r.o. and Lab Software Architectures and Information Systems at Faculty of Informatics at Masaryk University.

Celebrio Kernel is a general web system modular core that provides platform-independent modules for example for database, storages, logging, user management and full access control list. It can run all on-premises on GNU Linux or Windows server or in cloud computing environment of Windows Azure.

[19] http://www.windowsazure.com/en-us/home/features/storage/
[20] http://www.windowsazure.com/en-us/home/features/overview/

Chapter 4

Analysis

4.1 Single server against distributed environment

On-premises system is usually developed for a single server that provides both the computing power and storage space. It is easy to develop as the developers have full control over server customization and everything is available at one place.

In the cloud there is a distributed scaling environment of variable number of virtual server machines and developers have to deal with data centralization and persistency. The characteristics of design needs of server clusters come down to relatively small web systems.

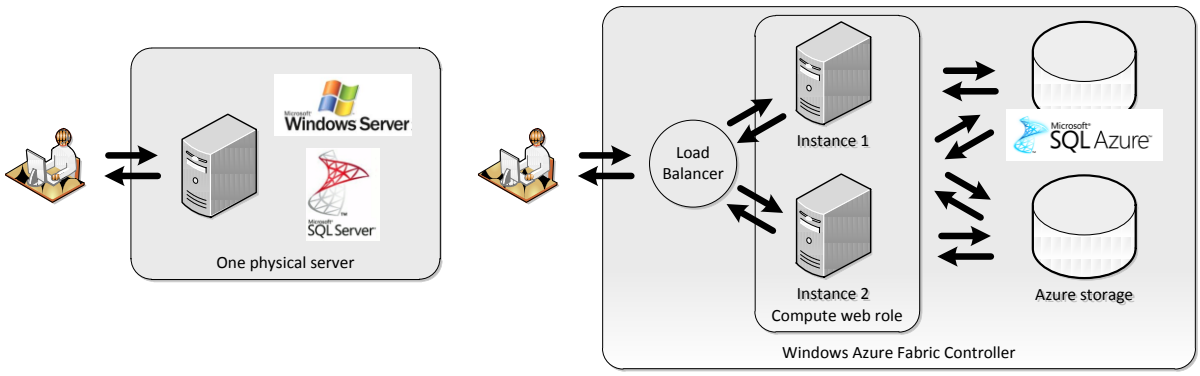

Figure 2: On-premises Environment & Cloud Windows Azure Environment

4.1.1 Dynamic number of distributed servers

The outside world accesses servers always through a load balancer and never directly, therefore it is unpredictable where the next request is going and the system has to be prepared for that by data centralization. [4]

In a common web systems development a server is seen as an environment where everything is stored but in cloud you have to see instances purely only as a compute power. That means that instances cannot store any data that may be needed between two server request and that includes user sessions and error reports.

The server instances are organized by Windows Azure Fabric Controller that manages load balancing, resource allocation and complete application lifecycle. It can also shut any server instance anytime there is something wrong or for example interrupt the request if there is no output from the web role instance for more than 60 seconds (which is important if the developed system is using output buffering and does not send out data during request processing).

Microsoft provides a 99.95% monthly service level agreement[21] for Compute services if the web role uses two or more instances which means high standard of availability for small companies running their software on Windows Azure. But there were cases when Windows Azure ran into serious trouble such as for example on February 29, 2012 when its software did not anticipate a leap year. [6] [7]

4.1.2 Data persistency and data centralization

In the distributed environment of cloud computing all the data must be centralized in order to be available for all server instances. No data is persistent on instance level and one instance does not have access to files stored on other instances. It is possible to require local disc space on instances which is permanent for the one instance and can be used for local-file designed caching or storing temporary files. [4]

For storing sessions there is a solution based on Azure storage Table. It is important that the developed system has to manage both session storing and erasing all data. [4] There can be also used Windows Azure Caching[22] service to provide faster access. [8]

Error logs cannot be stored on local file system of instances as the system manager would never see the full report. The Windows Azure SDK uses Table for storing logs [9] and there is Windows Azure Diagnostics service that manages low level errors and saves them into a BLOB storage. [10]

On-premises data is stored usually on the same server as the application using common file system and small companies do not have available resources to purchase servers for live replication. In case of scaling the data would have to be moved to some central storage and there would be a problem with migrating. But a small application has much easier data management.

4.1.3 Caching

Windows Azure applications can use caching on instance level using common server tools (local file system to store processed data or in memory cache using IIS) or use Azure Caching service among instances. [8]

[21] http://www.windowsazure.com/en-us/support/sla/
[22] http://www.windowsazure.com/en-us/home/features/caching/

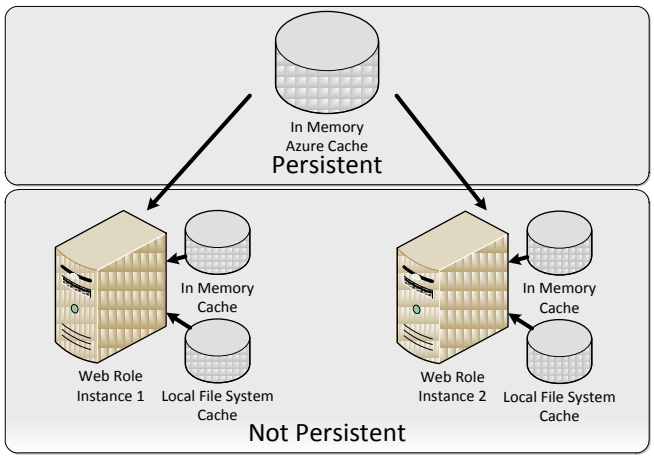

Figure 3: Caching on Windows Azure

Azure Caching is in memory solution that provides faster access then using slower disk based storages which can be taken advantage of especially for "write once read many" scenarios such as various catalog data. Microsoft offers cache size from 128 MB to 4 GB. [8]

For example Celebrio Kernel uses file based cache for automatic class loading and storing processed views (from MVC model[23]). On an instance level there is an IIS extension for PHP that allows in memory cache of optcode[24] and files. And Azure Caching is used for fast access to sessions and configuration. With every software upgrade, cache stored on instances is erased and only Azure Caching data remains. The data stored in Azure Caching would need to be placed on-premises in the IIS extension in memory cache. The advantage of Azure Caching lies only in its availability among more server instances.

4.2 Local development and development process

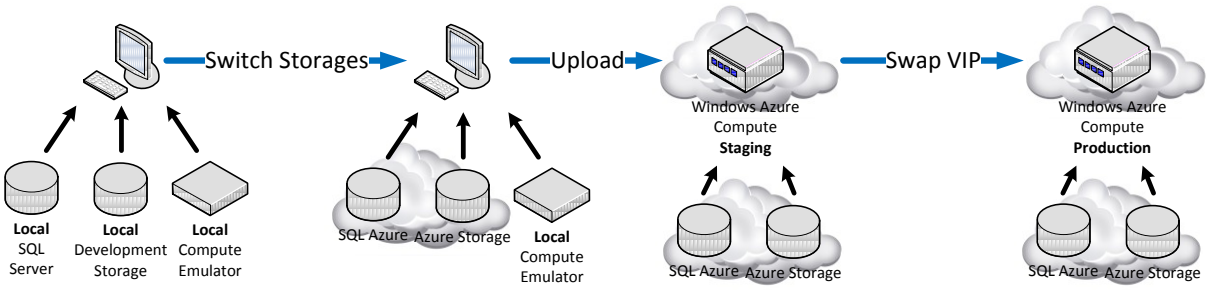

Figure 4: Windows Azure Development Steps

Windows Azure Compute roles can be developed using local emulators. There is a compute emulator, storage emulator and one can use Microsoft SQL Server Express (which is also used by the storage emulator) to emulate SQL Azure.

[23] Model-View-Controller, a design pattern dividing an application into three levels
[24] Processed byte-code of PHP scripts

Microsoft Visual Studio and Eclipse have extensions that allow easy deployment to both emulators and cloud.

Deploying to a local emulator usually takes only few seconds. Deploying to cloud takes up to half an hour as one has to wait for the deployment package to be created, uploaded and then for the instances to be started by Windows Azure Fabric Controller.

The best practice is to develop the application using local emulators and then test it by switching from local storages to cloud storages and then upload it to cloud. In cloud the upload should be made into staging environment for testing and if no errors or issues occur it can be switched to production which shows on the application domain (through virtual IP address swapping). Always the whole deployment package has to be created and uploaded for updates and there is no direct access to files unless a Virtual Machine Compute role is used and set up to be accessible directly.

On-premises the system is developed on a local test server running usually on the developer's computer and then files are uploaded to the live server typically using FTP[25] or similar service. The development is faster and easier but more prone to errors.

4.3 Security

Windows Azure datacenters and systems are highly secured and certified or compliant with:

- Safe Harbor[26]
- ISO 27001[27]
- SAS70 Type II[28]
- FISMA[29]

Companies running their software on top of Windows Azure may take advantage of these certificates that a small company would usually not be able to obtain.

It is important to take into consideration a law in a country where the business is made with Windows Azure based solution. For example the law of the Czech Republic prohibits storing bank account data or credit card data on any server situated outside of the country.

The access to Azure Management Portal which allows complete control over an Azure account including erasing all data is secured only by a Live ID login which is based on a user name and password.

[25] File Transfer Protocol – protocol used for uploading and/or downloading data on web

[26] http://safeharbor.export.gov/companyinfo.aspx?id=12409

[27] http://go.microsoft.com/fwlink/?LinkId=213084&clcid=0x409

[28] http://cdn.globalfoundationservices.com/documents/InformationSecurityMangSysforMSCloudInfrastructure.pdf

[29] http://blogs.technet.com/b/gfs/archive/2010/12/02/microsoft-s-cloud-infrastructure-receives-fisma-approval.aspx

Common on-premises system made by a small company does not achieve such level of security.

4.4 Availability and failovers

Data saved in Azure storage and SQL Azure is considered safe as the platform uses triple replication of all storages. Additionally Tables and BLOBS have replication between two datacenters at least 100 miles away from each other to be immune even against major natural disasters. SQL Azure can be also replicated by using Data Sync service. [11]

Windows Azure Fabric Controller monitors the environment and in case of a failure starts an automatic recovery process.

The service level agreement ensures for example: [12]

- external connectivity at least 99.95 % of the time to the Compute roles
- 99.9 % of detecting Compute role process inactivity
- 99.9 % successfully processing actions on Azure storage
- 99.9 % availability of SQL Azure

Common on-premises solutions do not have failover processes and do not offer such level of availability.

4.5 Backups

Compute roles have failover processes ensured by Windows Azure Controller Fabric and there is no need for backing up the data. Failover is also ensured for storages and SQL Azure.

SQL Azure can be backed up using Data Sync service so you can have your own safe differential time backup of the database. [13]

Azure storage BLOB has own solution for differential time backup based on changed blocks or pages but it is saved together with the BLOB and there has to be created own usage method.

There are no official tools for Azure storage local backup and Microsoft itself promotes backing up the data into separate Azure account in a different geo location. But the REST API makes it easy to develop own backup solution for Windows Azure if needed.

On-premises there are standard tools developed for data backed up but small companies usually do not have failover processes or live replication.

4.6 Testing

Standard application testing tools can be used for testing cloud systems regarding unit testing and behavioral testing. In case of Windows Azure environment it can be tested using local emulators but they do not cover all features and do not fully emulate Azure server. The Main differences are:

- A local emulator does not lock files against changes as a Compute service
- Cloud runs in 64 bit environment with modified Windows Azure server [14]
- Windows Azure Fabric Controller is not locally fully emulated
- Load balancer behaves differently in cloud
- Latency among Compute instances and storages is bigger in cloud
- Local emulator does not emulate Compute performance
- SQL Azure has different rules than Microsoft SQL Server [15]

Chapter 5
Diagnostics

5.1 Celebrio Kernel

Celebrio Kernel is a general web system core for building web applications. It provides database module supporting Microsoft SQL Server, Azure SQL, MySQL, Postgre SQL, contains user management module providing full access control list and Celebrio Virtual File System that enables simple work with various types of storages such as local file systems and Windows Azure storage BLOB.

The system was chosen because of its support of both on-premises and Windows Azure environments and open source free license.

Celebrio Kernel was used to build Celebrio[30], computer system for the elderly, by Celebrio Software, s.r.o. and Celebrio was used for testing the developed diagnostics tools in real-life environment.

5.2 Performance Logging Module for Celebrio Kernel

There was a need to create a new logging module for Celebrio Kernel in order to be able to easily monitor, log and view performance time indicators.

5.2.1 Non-functional Requirements

- Use standard PHP logging framework Apache log4php
- Extend log4php to log performance data
- Integrate performance overview in Celebrio Kernel logging module

Performance module is required to gather time based information about processing duration anywhere in Celebrio Kernel to be able to answer questions such as how it takes to create a new user or which pages are generated slowly.

5.2.2 Functional Requirements

- Support Windows Azure storage
- Support local file system storage
- Optimization for large amount of data

[30] http://www.celebriosoftware.com/

5.3 Apache log4php

Apache log4php[31] is a standard PHP logging framework similar to log4j (Java) or log4net (.NET) which is integrated into Celebrio Kernel as a main logging tool.

Example of error logging code in Celebrio Kernel:

\Celebrio\Logging\Logger::getLogger("errors")->error("test error - nothing actually happened", new \Exception("testing exception"));

5.3.1 Windows Azure support

Apache log4php does not include any support for the Windows Azure platform so there was a need to create new logger appender class as a descendant of log4php *LoggerAppender* class.

There were two options for target storage: Table or BLOB. BLOB storage is used by standard Windows Azure diagnostics [10] to store IIS logs and Compute instance performance logs and Table is used by standard logging class in Windows Azure PHP SDK [9]. Developed log4php appender *Celebrio\Diagnostics\LoggerAppenderAzureTable* uses Table and its implementation is inspired by Windows Azure PHP SDK logger and log4php MongoDB appender. The implementation directly uses Windows Azure PHP SDK to handle Windows Azure storage.

Appender was designed to store logged events in memory and commit them to Azure Table using batch processing at the end of web request processing. Advantage is in performance gain as direct writing per log would deal with a big amount of added up latency. Every target appender can specific its target table within storage so error logs do not mix with debug logs, etc. There cannot be used a different approach based for example on indexes as Azure storage tables do not support relationships and are designed for fast retrieval based on partition and row key combination but are slow on every kind of direct search within the logs data.

The developed appender supports exceptions logging and inner exceptions logging.

Azure storage table entities are structured by a partition key and a row key as indexes. Partition key optimizes data and load distribution, row key is the unique index within a partition and a combination of partition key and row is unique for every entity. The developed appender uses date YYYYmmdd as a partition key and a UNIX microtime for a row key. This allows appropriate load balancing and index uniqueness optimized for log reading per day ordered by a time.

[31] http://logging.apache.org/log4php/

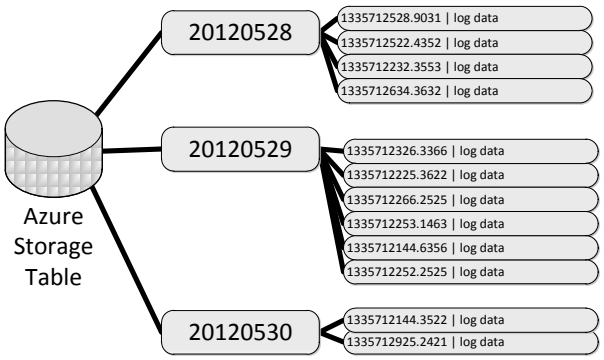

Figure 5: Azure storage Table structure for log4php logs

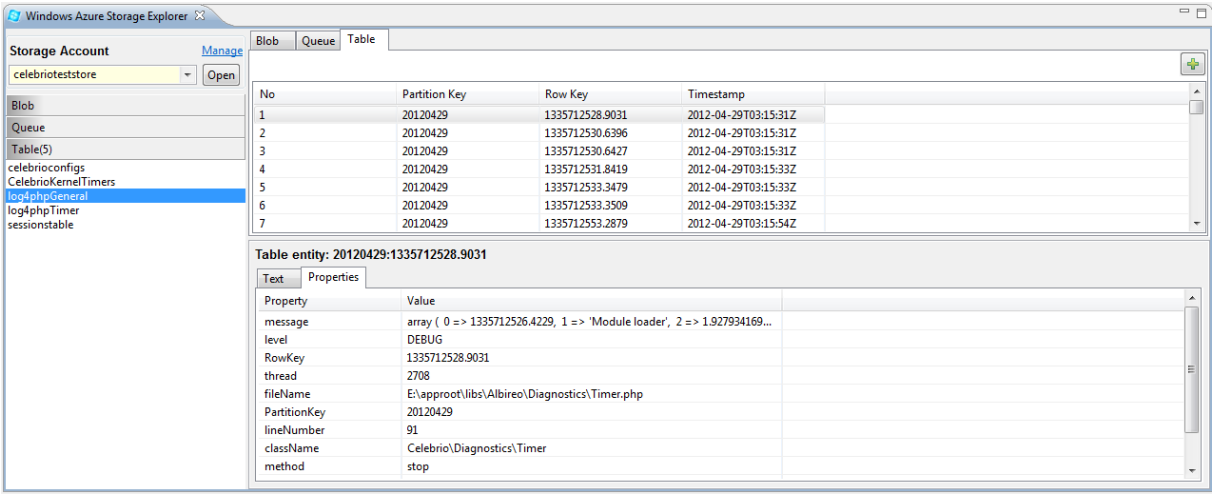

Figure 6: Windows Azure Storage Explorer in Eclipse showing log4php logs

5.3.2 Extending log4php for performance logging

Performance logging and diagnostics are expected to process a large amount of data and therefore it was decided to use optimized log formatting and appending to meet requirements. Using standard general logging would mean storing large amount of redundant data which is not optimal as performance diagnostics are based on a large number of separate logs. Partition key and row key in Azure storage tables needed to be optimized by storing per request and measured blocks to allow search based on those two variables which would not be possible with the standard log appender.

Logs contain unique web request UNIX microtime identification, measured processing time and measured block name including a class identification of measured block placement if available.

There was created a new class *Celebrio\Diagnostics\LoggerLayoutTimer* that extends standard log4php *LoggerLayout* class to format layout for local files logging using log4php class *LoggerAppenderFile*.

Example logged line:

1335712526.4229 - 1.9279341697693 – Performance logs processing (KernelModule/LogsModule/PerformanceModule/DisplayPresenter)

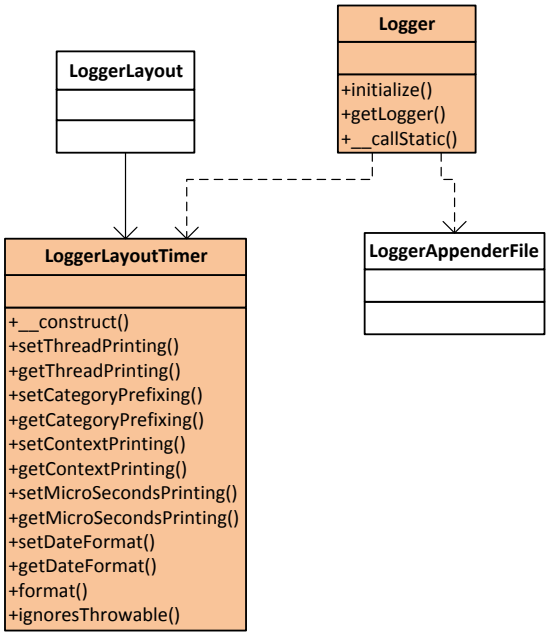

Figure 7: Class diagram for file based performance logging

A new logger appender class *Celebrio\Diagnostics\LoggerAppenderAzureTimer* was created extending *Celebrio\Diagnostics\LoggerAppenderAzureTable* to optimize logs for a different structure using web request identification as a partition key and measured block identification as a row key to empower diagnostics based on both particular web request and blocks.

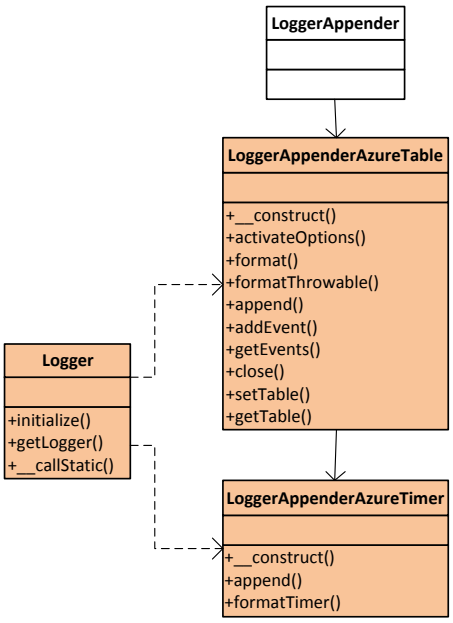

Figure 8: Class diagram for Windows Azure storage table based performance logging

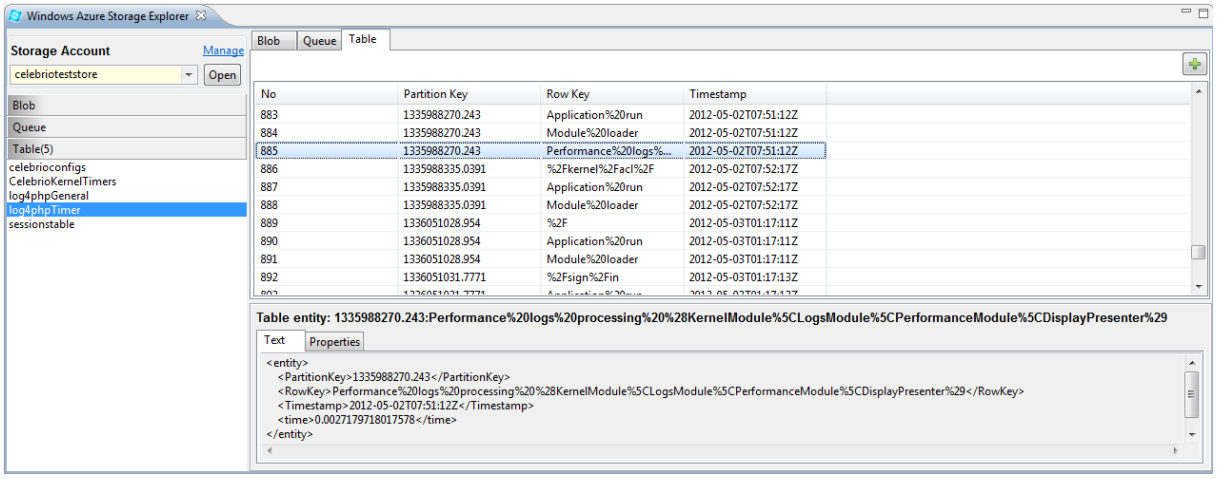

Figure 9: Windows Azure Storage Explorer in Eclipse showing performance logs

5.4 Diagnostics Tool

There was a need for a simple class designed to measure various blocks of code with different features. Class *Celebrio\Diagnostics\Timer* uses static methods to start and stop measuring and log final data using performance logging and supports three types of details: REQUEST, OVERVIEW and DETAILED; so the system administrator can control the amount of logged data.

Example of class usage:

```
Celebrio\Diagnostics\Timer::start("Module loading",
Celebrio\Diagnostics\TimerLevel::OVERVIEW);
// time consuming operation
Celebrio\Diagnostics\Timer::stop("Module loading");
```

5.5 Performance Overview

A performance log module was created for Celebrio Kernel to process and show diagnostics for created performance logs supporting both local file logs and Windows Azure storage Table logs.

5.5.1 Performance logs processing

A log processing on local files uses *Celebrio\FileSystem\TextFile* class to simplify work with files. Windows Azure processing uses Windows Azure SDK for PHP to handle storage.

During the processing we measure consumed time with limit of 50 seconds (divided into logical blocks) to ensure that the processing time will not overflow Windows Azure Fabric Controller time limit.

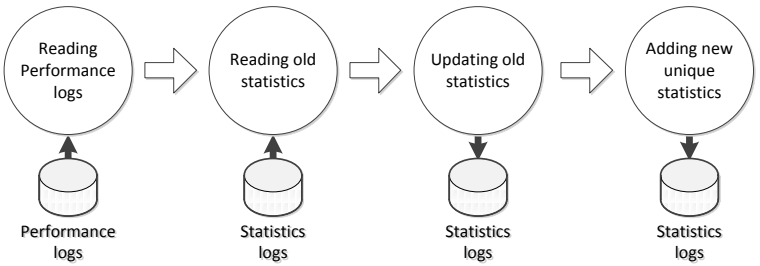

Figure 10: Performance logs processing

At the beginning new logs are read beginning from the last read byte in a file or from the last processed partition. Number of read bytes or last processed partition is stored using standard Celebrio Kernel configuration method. At the beginning the number is set to zero so in case of an error the next processing will start from total beginning effectively avoiding data corruption.

In the next step old diagnosed statistics are read in memory to update them and count average measured processing time based on total time and number of measurements.

In the last step there are new unique diagnosed blocks added.

On Windows Azure a batch processing is used to optimize performance and an entity number limit is watched to ensure seamless processing. If processing time limit is reached, the algorithm stores last processed logs position and starts from the point in next processing request.

During the development there a bug in Windows Azure PHP SDK was found and reported to its developers: http://phpazure.codeplex.com/workitem/6901

5.5.2 Celebrio Kernel integration

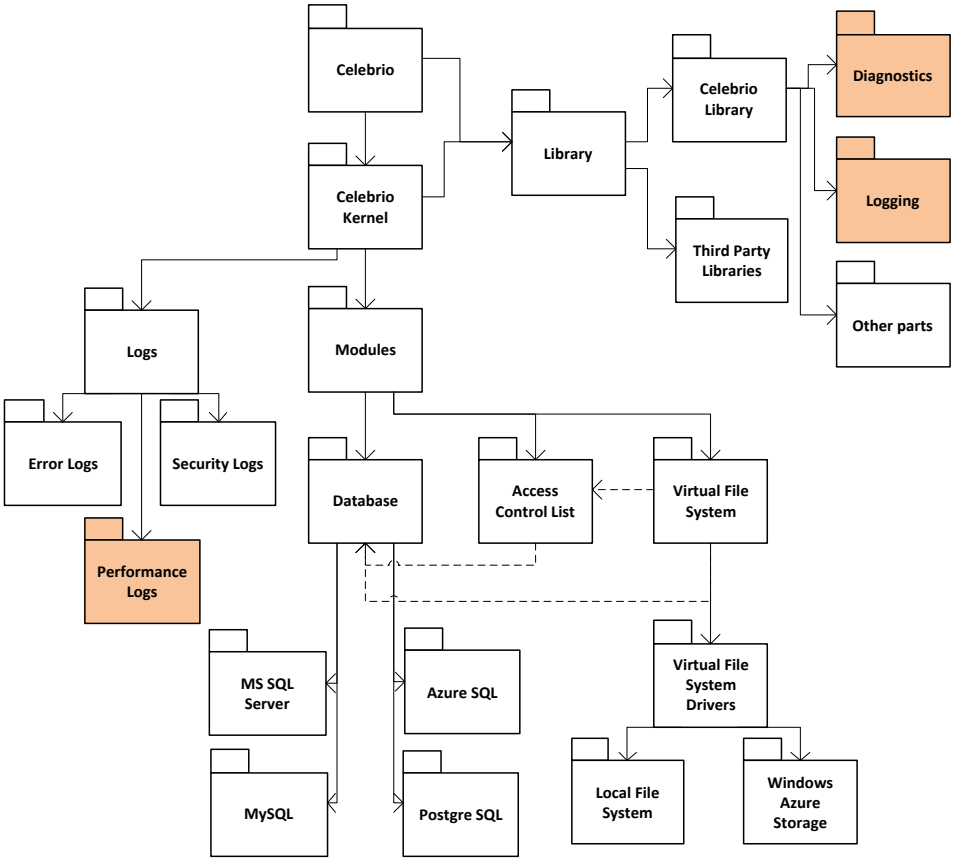

Figure 11: Celebrio Kernel diagram

The created tools are divided between Celebrio Kernel itself and Celebrio Library. Celebrio Library contains all the tools that are used through the code for logging and measuring blocks of code and the Performance Logs module (orange in the figure 11) that processes and displays data.

The module (Performance Logs in the figure 11) was designed using Model-View-Controller method as a new part of Celebrio Kernel Logs Module.

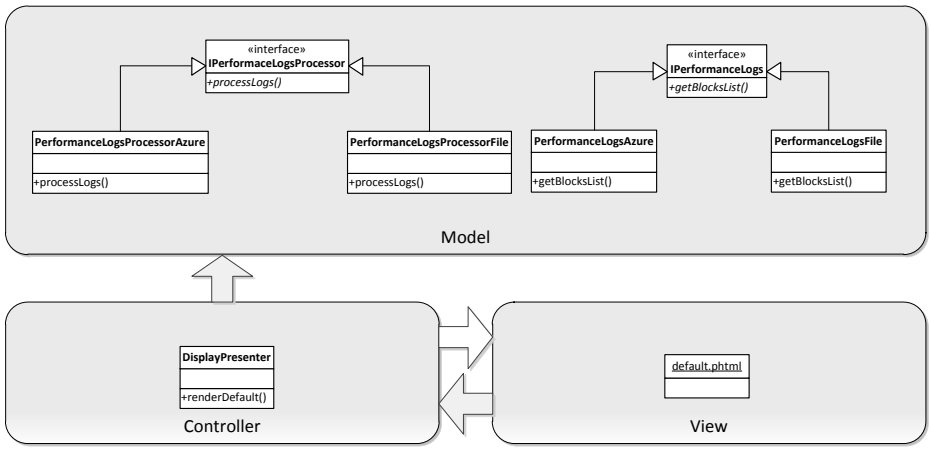

Figure 12: Celebrio Kernel Performance Logs Module diagram

Model uses an algorithm to process performance logs and retrieve statistics.

Controller controls data processing and retrieval from module and creates output for view.

View presents output data in a form of a Celebrio Kernel template.

5.6 Next development

The work was focused on working with performance data and diagnostics but although there were created general logging tools as well, at the moment there is no native Celebrio Kernel module that would read them and they have to be accessed through the Windows Azure Storage Explorer or read opened as a file on the local file system.

Chapter 6

Evaluation

6.1 Performance Evaluation

All diagnostics were made on fully installed Celebrio system with permission from Celebrio Software, s.r.o.

1,000 performance logs are used for diagnostics.

6.1.1 Local on-premises server

Tested on machine: Intel Core i5 processor, 2.3 GHz with 4 CPU cores, 4 GB RAM.

Local on-premises server shows to be the most efficient solution.

Block	Average Microtime
/kernel/	0.45298890272776
/kernel/modules/	0.65337546666464
/kernel/performance-logs/?process=1	0.99005603790284
Application run	0.17181092930823
/	1.0223519076472
/app/chat/	0.36315298080444

Table 1: Performance statistics on local on-premises server

6.1.2 Local Windows Azure emulator

Using 2 small instances in emulator tested on machine: Intel Centrino Duo processor, 2.0 GHz with 2 CPU cores, 4 GB RAM.

Locally emulated Windows Azure shows to be slower than a local on-premises server.

Block	Average Microtime
/kernel/	0.47446888371518
/kernel/modules/	0.95292387406033
/kernel/performance-logs/?process=1	2.9037259022395
Application run	0.414773345987
/	2.4059007228949
/app/chat/	0.70887640118599

Table 2: Performance statistics on local Windows Azure Emulator

6.1.3 Cloud Windows Azure

Using 2 small instances in cloud in a datacenter placed in Dublin, Ireland. One small instance configuration: 1 CPU core 1.6 GHz, 1.75 GB RAM.

Performance of cloud Windows Azure solution seems to be comparable with tested locally emulated Windows Azure.

Block	Average Microtime
/kernel/	1.6157951116562
/kernel/modules/	1.7324267625809
/kernel/performance-logs/?process=1	4.2073950767517
Application run	0.58849500150097
/	1.8688276368518
/app/chat/	1.3217213153839

Table 3: Performance statistics on cloud Windows Azure

Chapter 7
Conclusion

7.1 Learning and development

It is easier to stay with common on-premises system development and use well known practices to develop small systems. Cloud computing builds on a foundation of on-premises solutions but adds other elements such as various new storages, distributed environment, etc.

It might be discussed when it is the right moment to start focusing on cloud development and when an on-premises system is sufficient enough. The trend seems to be moving all web applications to the cloud and common on-premises systems might slowly retire. [2] There is a free offering for Windows Azure testing and companies in Microsoft BizSpark[32] program that gives access to free Windows Azure services sufficient for uninterrupted run of two small Compute instances.

7.2 Solution advantages

Cloud computing has an advantage in providing a more long term sustainable solution. Once the system is developed and becomes optimized for the target cloud platform it becomes very predictable and thanks to scaling capabilities it can easily grow.

Fast developed on-premises systems are usually not optimized for high performance and large number of users and therefore once they reach their performance limit, scaling the solution to meet higher needs can be very difficult.

7.3 Cost efficiency

For a very small system with low number of users an on-premises system can run on a cheap server with a low cost investment but if the developers want to achieve higher standard that would include high security, data safety and availability the upfront expenses will become consuming and not easily predictable.

Cloud system acquires high level of data safety, availability and certified security from the service provider without any expenses and there are no upfront payments.

Cost is also influenced by software licenses which have to be paid on on-premises solution for commercial software but cloud platform comes without such costs.

[32] http://www.microsoft.com/bizspark/default.aspx

7.4 Performance

On-premises solution works faster thanks to low latency even without high level of optimization. Optimized cloud solution usually runs slower.

Very important fact is that cloud solution keeps the same level of performance as it grows and is not as prone to slowing down as on-premises solutions. Once current hardware configuration is insufficient in the cloud it is possible to just obtain faster machine or balance the load among higher number of instances. On-premises solutions are more difficult to scale because obtaining more server power is connected to data and software migration and operations with the hardware. Once one server is not enough, the on-premises software must be rewritten for distributed environment.

Bibliography

1. ERL, T. *SOA Governance: Governing Shared Services On-Premise and in the Cloud*. SOA Systems Inc. 2011. 978-0-13-8156675-6.
2. SOSINSKY, B. *Cloud Computing Bible*. Wiley Publishing, Inc. 2011. ISBN 978-0-470-90356-8.
3. MICROSOFT. Cloud Resources. In: *Microsoft Global Foundation Services* [online]. 2012 [cit. 2012-05-07]. http://www.globalfoundationservices.com/cloud-resources.aspx
4. BETTS, D. *Moving Applications to the Cloud: on the Microsoft Windows Azure Platfom*. Microsoft, 2010. ISBN 9780735649675.
5. MICROSOFT. Features. *Windows Azure* [online]. 2012 [cit. 2012-05-07]. http://www.windowsazure.com/en-us/home/features/overview/
6. LAING, B. MSDN Blogs. In: *Windows Azure Service Disruption Update* [online]. 29. 2. 2012 [cit. 2012-05-06]. http://blogs.msdn.com/b/windowsazure/archive/2012/03/01/windows-azure-service-disruption-update.aspx
7. LAING, B. Windows Azure Service Disruption Resolved. *MSDN Blogs* [online]. 1. 3. 2012 [cit. 2012-05-07]. http://blogs.msdn.com/b/windowsazure/archive/2012/03/01/window-azure-service-disruption-resolved.aspx
8. MICROSOFT. Windows Azure Caching. *Windows Azure* [online]. 2012 [cit. 2012-05-07]. http://www.windowsazure.com/en-us/home/features/caching/
9. CENTRE, R. M. C. Source Code. *PHPAzure: Windows Azure SDK for PHP* [online]. 2012 [cit. 2012-05-07]. http://phpazure.codeplex.com/SourceControl/changeset/view/67037#1056744
10. CENTRE, R. M. C. Diagnostics. *PHPAzure: Windows Azure SDK for PHP* [online]. 2012 [cit. 2012-05-07]. http://phpazure.codeplex.com/wikipage?title=Diagnostics&referringTitle=Documentation
11. MICROSOFT. Introducing Geo-replication for Windows Azure Storage. *MSDN Blogs* [online]. 2011 [cit. 2012-05-07]. http://blogs.msdn.com/b/windowsazurestorage/archive/2011/09/15/introducing-geo-replication-for-windows-azure-storage.aspx
12. MICROSOFT. Service Level Agreements. *Windows Azure* [online]. 2012 [cit. 2012-05-07]. http://www.windowsazure.com/en-us/support/legal/sla/
13. MICROSOFT. SQL Azure. *Windows Azure* [online]. 2012 [cit. 2012-05-07]. http://www.windowsazure.com/en-us/home/features/sql-azure/

14. MICROSOFT. Virtual Machine Role in Windows Azure. *Windows Azure* [online]. 2012 [cit. 2012-05-07]. http://www.windowsazure.com/en-us/home/features/virtual-machines/
15. MICROSOFT. Compare SQL Server with SQL Azure. *Microsoft TechNet* [online]. 2011 [cit. 2012-05-07]. http://social.technet.microsoft.com/wiki/contents/articles/996.compare-sql-server-with-sql-azure.aspx#Similarities_and_Differences