BACHELOR THESIS
Individual Professional Practice in the Company
Hereby I declare that this bachelor thesis was written on my own. I have quoted all references I have drawn upon.

In Ostrava 6\textsuperscript{th} of May 2011 

..........................
Abstract

This thesis focuses on two subjects. First of the two subjects was to find the right tool to create technical documentation of the wide system that consists of several different technologies. This procedure has three different stages, first being definition of the requirements, second is about testing and evaluating the tool and third is the decision making based on the requirements and the evaluation.

Second subject is about how to implement Java into the Microsoft Windows Azure Platform. Mainly this is about figuring out how to overcome the obstacles of Java being partly limited on the platform. Introduction to the Microsoft Windows Azure Platform will be given and basic understanding of how to extend Java with the external libraries to connect with the interfaces of the Azure Platform.
Keywords

Java
Microsoft Windows Azure Platform
Restlet
Visual Studio 2010 Ultimate
Symbols and abbreviations

**Asynchronous communication** – Communication that is done indirectly between the client and the server.

**CRUD** – Create, Read, Update and Delete. This is used to describe operations supported by some class or application.

**Internal endpoint** – Interface that can only be accessed from the inside of the Windows Azure Platform.

**JAR** – Java Archive, generally used to distribute Java applications or libraries.

**JRE** – Java Runtime Environment, required when executing Java applications.

**REST** – Representational state transfer

**SDK** – Software Development Kit

**Synchronous communication** – Communication that is done directly between the client and the server.

**VB** – Visual Basic

**Worker** – Term used to describe the running applications on the Windows Azure Platform. These mainly in this document are applications that handle CRUD operations of the storages and communicate with the clients requesting for info.

**XML** – eXtensible markup language
Contents

1 INTRODUCTION ................................................................. 1
   1.1 PROGRAM USABILITY IN CREATING TECHNICAL DOCUMENTATION .............. 1
   1.2 IMPLEMENTING JAVA ON TO THE MICROSOFT WINDOWS AZURE PLATFORM ..... 1

2 PROGRAM USABILITY IN CREATING TECHNICAL DOCUMENTATION ........................................... 2
   2.1 REQUIREMENTS FROM THE PROGRAM ......................................................... 2
   2.2 VISUAL STUDIO 2010 ........................................................................... 4
   2.3 CONCLUSION ....................................................................................... 7

3 IMPLEMENTING JAVA ON TO THE MICROSOFT WINDOWS AZURE PLATFORM ........................................ 8
   3.1 MICROSOFT WINDOWS AZURE PLATFORM ................................................. 8
   3.2 GETTING STARTED ................................................................................. 9
   3.3 EXTERNAL JAVA LIBRARIES ................................................................... 10
   3.4 SCENARIO 1 – ASYNCHRONOUS COMMUNICATION .................................... 12
   3.5 SCENARIO 2 – SYNCHRONOUS COMMUNICATION .................................... 13
   3.6 “POISONOUS” MESSAGES ....................................................................... 14
   3.7 CONCLUSION ....................................................................................... 16

4 REFERENCES................................................................................. 17
1 Introduction

1.1 Program usability in creating technical documentation

Technical documentation is used in order to make the work of the project team easier, showing how each part of the system works. This specific project team was looking for a tool that would create technical documentation out of already long developed system.

The main problems were the large size of the system, and the numerous amount of technologies used. On top of these problems, there were also the requirements for the tool to be user friendly and easy to learn, to avoid unnecessary use of resources to use the tool itself.

At first the requirements of the tool were listed, and the descriptions of the requirements were given. After the requirements were given, the testing of the program was done, where the program is tested against the requirements.

Conclusion was done based on how well the program fills the needs of the specific project team.

1.2 Implementing Java on to the Microsoft Windows Azure Platform

Microsoft Windows Azure is cloud computing platform, with features to help the users to manage the service to fit into their own needs. The users can specify the scalability of the service, manage their storages and run applications on the platform. The platform is designed to support .Net code, but also other technologies, like Java, are supported.

The goal of this project was to overcome the difficulties of using Java on the Microsoft Windows Azure Platform, figure out how to deploy the Java Worker on to the platform, how to manage the storages and more. The main problems were mostly with the outdated libraries and figuring out the correct format of the data to be sent to the interfaces of the platform.

Short description of the Microsoft Windows Azure Platform is given, with the scenarios of Asynchronous and Synchronous communication from the client to the Java Worker; also the problems and the solutions are described. Also the external libraries used are explained and small introduction is given.

Conclusion was done based on how well Java fits into the Microsoft Windows Azure Platform, and how it compares to .Net on the said platform.
2 Program usability in creating technical documentation

This chapter provides information on one way of choosing a program to create technical documentation from an already existing system. The main problem was that the system is wide and consists of many different technologies, therefore finding one program to be able to interpret all the technologies in the system was difficult.

2.1 Requirements from the program

First thing to do was to define requirements that the program must fill. This section contains the functionality that the program must meet in order for it to be valid for the job.

2.1.1 Ability to interpret needed technologies

As the system that the technical documentation is generated out of is complex, old and uses different components written in many different languages, the ability to convert as many programming languages to technical documentation as possible is required. The languages playing the biggest part in this case were C#, C++ and Visual Basic 6.

2.1.2 Traceability

Traceability means that we can trace the chain of function and method calls logically. For example we can go deeper in the call tree and see what calls are made from the deeper levels of the system hierarchy.

2.1.3 Dependency tree

Dependency tree should bring view to the user that helps to understand the dependencies between the components. From this view user should be able to easily understand what links will need rechecking after the changes are made to some part of the system. The user should also be able to easily navigate through the whole hierarchy and also split the model into smaller parts to help to understand the smaller parts of the system.
2.1.4 Sequence diagram

The sequence diagram shows function-calls made from the function that is being observed, resulting into a large view where you can see the logical flow through the specific part of the system hierarchy.

2.1.5 One click access

As the external tools should make the work of the programmer easier and faster, there should always be easy access from one place to another. This means that when, for example, the user is browsing the dependency tree, there should be an easy navigation from the view into the code; in best case it would be, for example, double clicking a node of the tree.

2.1.6 Easy to learn

When the user already has his hands full of work, he does not want to spend a lot of time learning something new. Because of this the tool must be easy to learn, maybe something that the user is already familiar with, for example a plug-in or an extension to the program(s) that he is already working with.

2.1.7 Familiarity

When the user is learning new things, it always helps to have a new tool that feels familiar from the first sight. The best case would be when the user only has to install a plug-in or an extension to the program he has already been using earlier, because in most of these cases the new tool works the same way as the main program.
2.2 Visual Studio 2010

This section contains information of the Visual Studio 2010 and its ability to create technical documentation.

2.2.1 Program description

Visual Studio is designed to be a platform which you can use to easily generate and compile code for multiple technologies, such as C++, .Net and more.

Visual Studio 2010 itself offers many different additional features which graphically help to understand wide systems. The amount of these features depends on the edition of the Visual Studio 2010. Available editions are Professional, Premium, Ultimate and Express [1].

In the case of creating the technical documentation, only Ultimate edition has the features required for the task.

2.2.2 Evaluation

This section contains the step by step evaluation of the Visual Studio 2010 Ultimate.

2.2.2.1 Ability to interpret needed technologies

Visual Studio 2010 Ultimate is able to create technical documentation out of C# code as a default.

To extend the functionality to interpret C++ code, the user must download and install Visual Studio 2010 Feature Pack [2], which is only available for subscribers.

When it comes to interpreting VB6 code, the user must first convert VB6 code into VB.Net code with Visual Studio 2008, and later convert the VB.Net code further to fit it into a Visual Studio 2010 project. Also after converting the code, there might be difficulties in compiling the code because of the technology changes.
2.2.2.2 **Traceability**

Architecture Explorer offers so called Traceability functionality in Visual Studio 2010 Ultimate. The user can advance from one level to another inside the architecture and get an idea of where the different parts of the architecture are used and where they lead to.

2.2.2.3 **Dependency tree**

Visual Studio 2010 Ultimate offers a feature to create a dependency tree from an existing code. However this feature is quite limited as adding new nodes to the tree is only possible through the XML file that the dependency tree is based on.

Copying and splitting the tree into smaller pieces is also possible. On top of this, the user can also specify how many nodes from the current node are shown.
2.2.2.4  *Sequence Diagram*

In Visual Studio 2010 Ultimate you can create a sequence diagram easily. The user only has to right click on the function name and the user is automatically offered to create a sequence diagram out of the function calls.

![Automated creation of sequence diagram](image1)

After you have created the sequence diagram from the existing code, you can view the diagram and easily see which functions are called and at what point. Also to mention, you can navigate straight into the code from the sequence diagram by right-clicking the function name and selecting the “Go To Definition”.

![Sequence diagram](image2)  ![Going to the source code from the sequence diagram](image3)
2.2.2.5  **One click access**

In most of the cases, Visual Studio 2010 offers one click access. However, in some rare cases the user might think that for example double clicking would lead into a completely different place than expected, but after learning these little things, using the features offered is really effortless.

2.2.2.6  **Easy to learn**

Using Visual Studio 2010 is pretty much straight forward, and in the cases when the user does not know how to proceed from current situation, there is wide documentation available for the program. Also the user base is broad and in almost all cases you can find the solution to your problem with only a little effort.

2.2.2.7  **Familiarity**

Users in this project have so far been using Visual Studio 2010 Professional, so upgrading to Ultimate would be an easy step for everyone. Also the functionality of the features Ultimate edition brings, are similar to what Professional edition already has, so the learning curve in this case is low.

2.3  **Conclusion**

When it comes to creating technical documentation with Visual Studio 2010 Ultimate, for this specific project team it would serve its purpose quite well.

If we do not take into account a few faults, for example not able to interpret VB 6 code, the program fills the needs of the programmers. Familiarity, One click access, Dependencies and so on, they are all there and easily accessible. The tools itself are really easy to use and fast to learn, so getting started should not be a problem.
3 Implementing Java on the Microsoft Windows Azure Platform

There are several different approaches to get started with implementing Java on the Microsoft Windows Azure Platform. This section provides information of one way of implementation of Java on the Azure Platform.

3.1 Microsoft Windows Azure Platform

In this chapter a short introduction to the Microsoft Windows Azure Platform will be covered.

3.1.1 Basic idea and usage of Microsoft Windows Azure Platform

Windows Azure is a cloud computing platform. This platform provides the user the operating system, storage, and scaling service that help the user only pay for what they use. This differs from the so-called normal server, where the user pays for the whole server, instead of just what he uses.

3.1.2 Microsoft Windows Azure account

For this project, a free trial account from Microsoft Windows Azure service was requested [5]. This kind of account has limitations on the computing time and database size, but to get to know the Platform and to see the basic idea and usability of the service it is more than enough.

Microsoft also offers virtual machines to emulate the Windows Azure Platform to avoid unnecessary transactions to the Platform itself, avoiding the use of resources.
3.2 Getting started

The first thing when starting on Windows Azure Platform with Java is some kind of declaration of the functionalities wanted should be done. Afterwards the libraries used during the project are chosen according to the functionality of the service to be created. Also there are slight differences and advantages in different deployment styles depending on the system.

For the Java Worker deployment, deploying executable JAR and JRE within the Visual Studio 2010 project was chosen.

The project requirements were to create Java Worker with an internal endpoint and basic Queue and Table CRUD operations. Based on that Restlet was chosen to handle the internal endpoint and for Queue and Table CRUD operations Windows Azure SKD imported to handle the main functionality.

Later on Restlet was also used to perform the Queue read operations.

3.2.1 Deploying Java Worker on to the Azure Platform

As the project is integrated with another project that consists of working with Windows Azure Platform with Visual Studio 2010, JAR deployment was done with Visual Studio 2010 that is extended with Microsoft Windows Azure SDK.

Java is deployed on to the Windows Azure Platform by including the executable JAR file into the Visual Studio 2010 project. Also when the user deploys the JAR on to the Azure Platform, the JRE also has to be included into project in order to make the JAR file executable on the Windows Azure Platform.

The executable JAR itself was created from the Eclipse project.

3.2.2 Entry point

Entry point is one of the functionalities that cannot be done on the Java side. For this you need to create a C# Entry point from where you run the executable JAR files. Once the JAR is executed, Worker is started and ran until it crashes or is stopped. If the Worker for some reason crashes, it is automatically restarted.
3.3 External Java libraries

Java does not have any of the libraries to work with Microsoft Windows Azure Platform by default. External libraries are used in order to make the communication with Azure Platform interfaces easier. This section provides an overview of the external libraries used.

3.3.1 Soyatec Windows Azure SDK for Java

Soyatec together with Microsoft offers Windows Azure SDK for Java [3]. The SDK contains basic classes for Windows Azure, which offers CRUD functionality on the Windows Azure Blobs, Queues and Tables.

These functionalities are basically the base of the Windows Azure Platform, as Workers can use them to store the data gathered from clients or other operations, and later send them to any clients requesting for the specific info.

3.3.1.1 Blob storage

Blob storage offers unstructured file-based, binary formatted data storage mainly for large items. This storage can scale the usage of resources based on the user requests.

3.3.1.2 Queue storage

Queue storage works on first in first out principle, where data stays until it is deleted. Once data is read it is moved into the end of the Queue. The data row has “dequeue count”, which contains the information on how many times the data has been read.

3.3.1.3 Table storage

Table storage can be seen as a normal storage, where you can determine the table layout and later add data into it.
3.3.2 Restlet

Restlet is RESTful web framework for Java and supports various web standards [4]. In this case Restlet was used in order to create internal endpoint on Java Worker for HTTP Get requests.

Also in the later stages REST was used in HTTP Get requests to the pre-existing interface of Azure Platform that replies with XML formatted data. This was mainly done to fill some missing information that the normal reading operation from Queue storage did not provide.
3.4 Scenario 1 – Asynchronous communication

When you are trying things for the first time, easy scenarios are the best for learning basic things, in this case, such as CRUD operations for Queues and Tables.

1. In the Asynchronous communication, we insert messages into the Queue storage with the client.

2. Worker reads the messages from Queue storage with set interval.

3. When the Worker has successfully read a message from the Queue storage, it inserts the data from the message into the Table storage.

4. After successful insertion of data into the table, Worker deletes the message from the Queue storage.

5. Client requests data from the Table storage with set interval and updates the view based on the data received.

![Sequence diagram – Asynchronous communication](image)
3.5 **Scenario 2 – Synchronous communication**

For synchronous communication, scenario where client requests all the table data from the Worker and Worker replies with XML structured data was created.

1. Synchronous communication starts with the HTTP Get request from the client passed onto the Worker.
2. Worker gets data from Table storage and organizes it into XML format.
3. XML formatted data is sent back to the client.

![Sequence diagram – Synchronous communication](image)

3.5.1 **Internal endpoint**

To let the client communicate straight to Java Worker, an internal endpoint must be created. This was done with the Restlet, which allows the creation of the HTTP request services.

1. Setting up the interface for the HTTP Get method (file Main.java lines 119-161)

   ```java
   @Get
   public String toString()
   ```

2. Starting the service (file Main.java lines 50-55)

   ```java
   new Server(Protocol.HTTP, 8182, Main.class).start();
   ```
3.6 “Poisonous” messages

A poisonous message is defined as a message that is being read numerous times, but is never removed from the Queue storage. This can happen when the message in the Queue storage is in some way faulty and might crash the Worker.

To avoid possible continuous crashes of the Workers and to stop the message from being uncontrollably managed, we need to define that the message is poisonous after a defined amount of reads and because of that it will be removed from the Queue storage after the specified amount is filled.

For these kinds of operations, messages in Queue storage have a “Dequeue count” value, which is incremented by one with each read.

Normally, in C#.Net you can easily request this information by just reading the message and calling the method to get the Dequeue count of the message, but in Java that does not exist. For this, a different approach was taken into use.

3.6.1 Dequeue count with Java

As the requesting of the Dequeue count was not supported in the Windows Azure SDK, the approach to using Restlet to request such data from the HTTP interface of the Windows Azure server was used.

There are few steps in making the HTTP request to match the format that verifies that the request is coming from someone that is allowed such information [7].

1. Define the URL to send the request to (file Main.java line 174)

   ```java
   HttpGet get = new HttpGet("http://" + ACCOUNT_NAME + ".queue.core.windows.net/msgjavaqueue/messages");
   ```

2. Set the request headers for the HTTP request (file Main.java lines 245-248)

   ```java
   request.setHeader("x-ms-version", "2009-09-19");
   ```

3. Create the string to be encoded as a part of the verification (file Main.java lines 250-268)

   ```java
   sb.append("x-ms-version:" +
              request.getFirstHeader("x-ms-version").getValue() + 
`\n`);
   ```
4. Encode the string created from the header values and include it as a part of the header
(file Main.java lines 270-274)

```
Mac mac = Mac.getInstance("HmacSHA256");
mac.init(new SecretKeySpec(Base64.decode(key), "HmacSHA256"));

request.setHeader("Authorization", "SharedKey " + account + ":" +
        new String(Base64.encode(mac.doFinal(
            sb.toString().getBytes("UTF-8")))));
```

After the steps mentioned above, assuming header information is correct, the request can be sent
and in exchange message data from Queue storage is retrieved. In this case reading of the
response was done with BufferedReader and handling the XML formatted data was done with
NodeList type (file Main.java lines 179-182).

```
BufferedReader rd = new BufferedReader(
        new InputStreamReader(httpclient.execute(get)
                .getEntity().getContent(), "UTF-8"));

 NodeList messages = DocumentBuilderFactory.newInstance().
        newDocumentBuilder()
        .parse(new org.xml.sax.InputSource(rd))
        .getElementsByTagName("QueueMessage");
```

More information about the Get method and the header format can be found on the Microsoft
Developer Network [6].

3.6.2 Deleting messages from the Queue storage

When reading the Dequeue count from the message data received from the Queue storage, you
can also retrieve information like MessageId, PopReceipt and MessageText to mention the few
most important ones.

There are two ways of deleting the messages from the Queue storage with Java, first one being
the Http Delete method. This is mostly the same as the Get method, but differs slightly on the
URL part.

```
HttpClient httpclient = new DefaultHttpClient();
HttpDelete delete = new HttpDelete(
        "http://" + ACCOUNT_NAME +
        ".queue.core.windows.net/msgjavaqueue/messages/
            + msgID + "?popreceipt=" + popReceipt);
```
For deleting the message from the Queue storage without Http requests, MessageId and PopReceipt must be included in the instance of a class that implements IMessage interface.

1. MyMessage class (file MyMessage.java)

   ```java
   public class MyMessage implements IMessage
   ```

2. Deleting the message (file Main.java lines 332-336)

   ```java
   public static void DeleteMessageFromQueue(
       QueueStorageClient qsc, MyMessage msg){
       qsc.getQueue(QUEUE_NAME).deleteMessage(msg);
   }
   ```

3.7 Conclusion

Almost everything that can be done with C# on Windows Azure Platform, you can also do with Java. However, there is a lack several functionalities that you must write by yourself in Java, that are already implemented on the C# side.

Even though some things like Entry point for the whole system can only be created with C#, Java Worker is able to run on the Windows Azure Platform almost as efficiently as the C# Worker can.
4 References

http://www.microsoft.com/visualstudio/en-gb


http://www.windowsazure4j.org/

http://www.restlet.org/

http://www.microsoft.com/windowsazure/
