Saimaa University of Applied Sciences

Faculty of Technology, Lappeenranta

Double Degree

Information Technology

Libor Gorcica
Robert Helesic

Content management applications for Juki-Lux media Pylon

Bachelor's Thesis 2012

Abstract

Libor Gorcica, Robert Helesic

Content management applications for Juki-Lux media Pylon, 44 pages

Saimaa University of Applied Sciences

Faculty of Technology, Lappeenranta

Information Technology, Double Degree

Thesis 2012

Instructors: Lecturer Pasi Juvonen, Saimaa University of Applied Sciences

Pylon project is remote controlling and management of interactive screen. The connection scheme is client → web application → pylon. The work mostly describes how a web application serves the client to create and edit different media-type presentations and to manage all his pylons from one place. There is also focus on pylon side – how to securely connect to the database, and overall discussion about used services and the way it all works together.

Keywords: Pylon, client, server, remote controlling, .NET, Silverlight, LINQ.

Table of contents

1 Overview about Pylon

Pylon project was assigned to us on demand from Juki-Lux company which is a Lappeenrantas-based company developing various advertisement and special furniture products. According to the new age of advertisement there was strong need inside of this company to be able to compete with other companies and that is the reason why Pylon project was invented.

Illustration 1: Media Pylon

In general Pylon is a 2-meter high metallic column equipped with 40” screen which is vertically built-in behind a protective plexi-glass. Inside its robust construction there is a small computer running Microsoft Windows 7 connected to the Internet either via a LAN cable, a Wi-Fi network or 3G. Further specification of used components is not reliable because the Pylon was just a prototype where all teams were trying to figure out which components are the best and it is still subject for future changes depending mostly on company's decisions.

The main idea is to create an attractive and interactive screen which is able to display all media formats. What is shown is completely in hands of the customer, that is why customer is having 24/7 connection to Pylon through the Internet. The customer is able to select which media and in what time will be shown on Pylon's screen. That is the part where the team came to development – to make connection and modification by the customer as easy as possible.

This task needed to be divided between two teams, each consisting of two people. The task of the first team was creation of web pages and the overall logics of net application, the task of the second team is application for connection to the Pylon-server and the ability to create presentations online.

Previous studies have done a good job creating “Presentation viewer” and “Presentation creator”, both of those applications will be described later in this chapter.

1.1 Existing projects

The first two parts describes the base from which the main ideas and some technologies were taken because they were already almost developed by another team. The last part was co-created by the first team and was developed with ideas from second team all together.

1.1.1 Presentation viewer

The Pylon-side application which is constantly running on background checking if there is a new packet (form of transferring presentations) to the predefined folder. If there is a new presentation, the program transfers it to own directory, extracts and starts to show it directly on fullscreen.

1.1.2 Presentation creator

This application was written also by the Finnish colleagues in C#. It is a nice desktop application which has most of the functionality as Internet based Presentation creator. There was strong need to transfer and improve the desktop application functionality to a web-oriented application.

So it is capable of collecting pictures which are meant to be shown, choosing switching methods and visualizations and time for how long each picture should be shown. It can also work with other media like video. Then after collecting all medias and setting up properties it compresses all files in one file which is ordered in proper structure for later extraction and use by Presentation viewer.

1.1.3 Web application

The main creators of web application are students from the first team with some help from second team. The idea is to control Pylons, users and presentations only via web interface. Implementation includes complex database, web engine running on .NET 3.0 and more technologies described also in further chapters.

The application is nowadays stable and fully working. They have done excellent job involving new ideas and complex design of well-working application.

2 Objectives

There were assigned several smaller projects during the implementation phase and each of them with different specifications and different difficulties. The current state of the project was shown among with objectives what would be nice to achieve in the future. However it was only background and new teams were given the opportunity to find out how to achieve these goals. Initially the objectives were to be able to control more Pylons at the same time and make Pylon more independent on the company's network. Nevertheless later there came up new ideas for improvements and most of them were also implemented. There was strong need to separate the project between two teams. This will be described bellow. They will be separated to the supervisor's objectives and new team objectives which occurred during development.

2.1 Presentation Downloader

After new team made this web application with another team for controlling Pylons that are described in the first team's work several needs appeared. One of them was to allow program for displaying presentations 1.1.1 Presentation viewer to access and display presentations from the website.

Supervisor's Objectives:

1. Find out how the communication between Pylon and server will work. There was an issue with communication in direction from server to Pylon. Pylon could be hidden behind router or IP address could change in time.
2. Download presentation from the server and copy it to the folder where the mentioned program could access and display it.
3. Start from scratch and do not interfere with the original program.

Team objectives:

1. Allow user to create a time schedule for displaying various presentations on different Pylons. The client has to look into the database and choose the assigned presentation depending on time schedule.

2.2 Presentation Creator

After new solution was made and presented for 2.1 Presentation Downloader there emerged an idea to allow the customer to create presentations on the web application without need to install any additional software.

Supervisor's objectives:

1. Make program for creating presentations on web site.
2. Use original XML structure to keep compatibility with the mentioned programs from Finnish students.
3. Simulate the selected Pylon screen (aspect ratio) to allow the customer to see how the picture or video will be displayed on the Pylon screen.

Team Objectives:

1. Use advanced technology like Silverlight or Flash to be able to create a catchy and nice design of an application.
2. Add possibility to combine videos and pictures in one presentation.
3. Allow user to use transition effects between pictures.
4. Separate administrator from normal user and allow him to edit all presentations.
5. Possibility to assign Pylons to the desired presentation.
6. Display the whole presentation as a “slideshow” so a user could see how it will look like on the Pylon.

2.3 Pylon Client

After there was presented solution of Presentation creator the supervisor and the customer were impressed by the used transition effects between pictures. Response was that the current solution of program displaying presentations by Finnish students does not support these transition effects. That was a reason why the supervisor asked to create an application for displaying presentations on Pylon that will be supporting these transition effects. The first idea was to use Silverlight like for presentation creator but after little consideration was decided to use Windows Presentation Foundation (WPF) because it fits better to desktop applications and supports the same and even more libraries than Silvelight.

Objectives:

1. Smooth transition effects between pictures.
2. Integrate 2.1 Presentation Downloader with the 2.3 Pylon Client and run it in different thread.
3. Achieve good hardware acceleration for rendering transition effects

3 Used technologies

Because teams didn't start this project on their own (were just added to running development environment) there were only limited choices to use new technologies. In the end this wasn't a big problem.

The original program was developed by using C# programming language, so it was chosen in order to be able lately to merge the application source codes. As time has shown it was actually a really good choice by the original team because of a variety of already made modules and principles for use. The main one used for the development of interactive web environment is Silverlight which is described in the next chapter.

3.1 Silverlight

“Silverlight is a powerful development platform for creating engaging, interactive user experiences for Web, Desktop and mobile applications when online or offline” [10] this is how Microsoft advertises its popular tool. The main goal of this product is to be number one in interactive web page design. The user can run this tool on various operating systems and platforms, native support is given to all newer Windows versions, to Mac OS, Symbian and Windows Mobile. Thanks to Novell company, which created the port called Moonlight, it is possible to run Silverlight applications even on Linux-based systems. No matter on which system it is running, it works like a plug-in integrated in web browsers - all main browsers are included and of course in Internet Explorer 7 and newer it is already preinstalled, for the rest of browsers there is a need to download and install.

This makes a problem for Microsoft which is trying by this product to take over the market which is now fully in the hands of Adobe Flash. Statistics from year 2011 shows those percents of plug-ins installed in browsers: Adobe Flash – 95.26%, Java – 76.51% and Silverlight – 64.16%. It obviously puts Silverlight on the third position in tools for Rich Internet Applications (RIA) development.

Since 2007 when Silverlight version 1 was released and its approach was “Interactivity was provided by Silverlight, but user input controls are HTML controls overlaid on top of Silverlight content”. [11] At that time the main focus was on streaming, which changed among the time and versions. The newest version is number 5 and its features include GPU accelerated video decoding, 3D graphics, playback speed controls, remote control and 64-bit support.

“Silverlight provides a retained mode graphics system similar to Windows Presentation Foundation (WPF), and integrates multimedia, graphics, animations and interactivity into a single run-time environment. In Silverlight applications, user interfaces are declared in Extensible Application Markup Language (XAML) and programmed using a subset of the .NET Framework. XAML can be used for marking up the vector graphics and animations. Silverlight can also be used to create Windows Sidebar gadgets for Windows Vista.” [11]

The strong side of this tool is the support for various codecs like H.264 video, Advanced Audio Coding (AAC), Audio Coding, Windows Media Video (WMV), Windows Media Audio (WMA), VC-1 video and MPEG Layer III (MP3).

As explained later in .NET section there is a various number of options how to create programming logic i.e. in any .NET language, including some derivatives of common dynamic programming languages like IronRuby and IronPython.

3.2 .NET

Microsoft started the development on the .NET Framework in the late 1990s originally under the name of Next Generation Windows Services (NGWS). By late 2000 the first beta versions of .NET 1.0 were released. Since that time occurred those versions: 1.1, 2.0 (which was breakthrough), 3.0, 3.5, 4.0 and 4.5 in 2012.

.NET is a software framework - pack of tools used mostly for the development of Web applications, Windows and also Pocket PC applications.

“The biggest power of this framework is in its variability and big number of modules. It includes a large library and provides language interchangeability (each language can use a code written in other languages) across several programming languages. Programs written for the .NET Framework execute in a software environment (as contrasted to hardware environment), known as the Common Language Runtime (CLR), an application virtual machine that provides important services such as security, memory management, and exception handling. The class library and the CLR together constitute the .NET Framework.” [6]

The project used version .NET 3.0 which is a common standard pre-installed on most OS which are nowadays running on PCs. On the one hand it is powerful enough to easily handle all the problems and on the other hand it should be available on the most existing computers, so there should not be big problems with compatibility.

Great variability is also given by the option of using many programming languages, so the developer does not have to learn a new language when in need of some specific feature of this framework. The most used languages for development of .NET applications are C#, Visual Basic .NET and Delphi. In addition to those the developer can use also Managed C++, F#, J#, IronPython and Boo.

The power of the possibility to use any input language, its further processing like compilation and the way through Common Language Interface to the final executable native code is shown in the next figure.

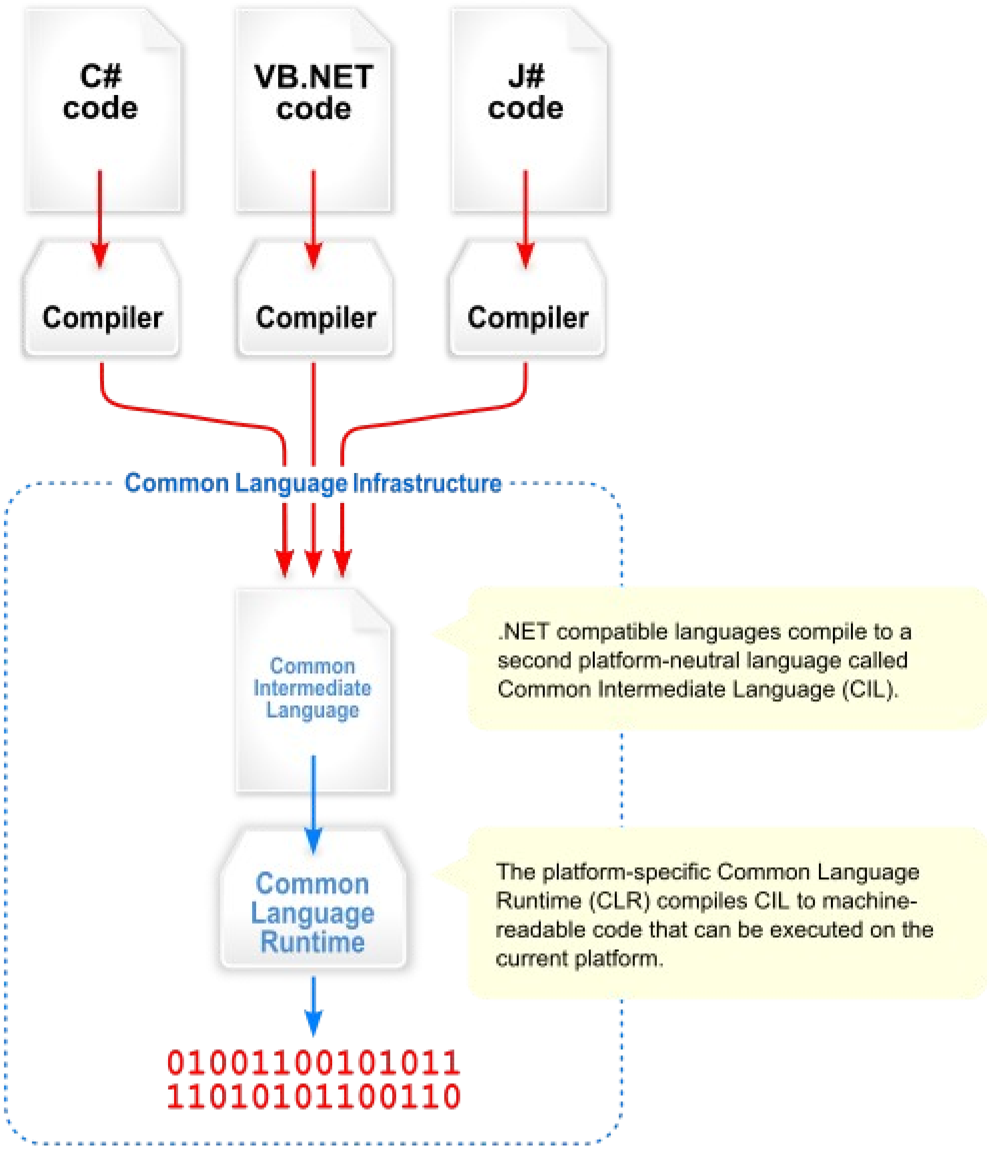

C# code
VB.NET code
J# code
Compiler
Compiler
Compiler
Common Language Infrastructure
Common Intermediate Language
.NET compatible languages compile to a second platform-neutral language called Common Intermediate Language (CIL).
Common Language Runtime
The platform-specific Common Language Runtime (CLR) compiles CIL to machine-readable code that can be executed on the current platform.
01001100101011
11010101100110

Figure 2: Processing of different language inputs in .NET CLI.

3.2.1 Design features

This whole sub-chapter is citation from [6].

- **Interoperability**

  Because computer systems commonly require interaction between newer and older applications, the .NET Framework provides means to access functionality implemented in programs that execute outside the .NET environment. Access to COM components is provided in the System.Runtime.InteropServices and System.EnterpriseServices namespaces of the framework; access to other functionality is provided using the P/Invoke feature.

- **Common Language Runtime Engine**

  The Common Language Runtime (CLR) is the execution engine of the .NET Framework. All .NET programs execute under the supervision of the CLR, guaranteeing certain properties and behaviors in the areas of memory management, security, and exception handling.

  Common Language Runtime (**CLR**) is the virtual machine component of Microsoft's .NET framework and is responsible for managing the execution of .NET programs. In a process known as just-in-time (JIT) compilation, the CLR compiles the intermediate language code known as Common Intermediate Language (CIL) into the machine instructions that in turn are executed by the computer's CPU. The CLR provides additional services including memory management, type safety and exception handling. All programs written for the .NET framework, regardless of programming language, are executed by the CLR. It provides exception handling, Garbage collection and thread management. [8]

- **Language Independence**

  The .NET Framework introduces a Common Type System, or CTS. The CTS specification defines all possible data types and programming constructs supported by the CLR and how they may or may not interact with each other conforming to the Common Language Infrastructure (CLI) specification. Because of this feature, the .NET Framework supports the exchange of types and object instances between libraries and applications written using any conforming .NET language.

- **Base Class Library**

  The Base Class Library (BCL), part of the Framework Class Library (FCL), is a library of functionality available to all languages using the .NET Framework. The BCL provides classes that encapsulate a number of common functions, including file reading and writing, graphic rendering, database interaction, XML document manipulation, and so on.

- **Simplified Deployment**

  The .NET Framework includes design features and tools which help manage the installation of computer software to ensure it does not interfere with previously installed software, and it conforms to security requirements.

- **Security**

  The design is meant to address some of the vulnerabilities, such as buffer overflows, which have been exploited by malicious software. Additionally, .NET provides a common security model for all applications.

- **Portability**

  While Microsoft has never implemented the full framework on any system except Microsoft Windows, the framework is engineered to be platform agnostic, and cross-platform implementations are available for other operating systems (see Silverlight and the Alternative implementations section below). Microsoft submitted the specifications for the Common Language Infrastructure (which includes the core class libraries, Common Type System, and the Common Intermediate Language), the C# language, and the C++/CLI language to both ECMA and the ISO, making them available as official standards. This makes it possible for third parties to create compatible implementations of the framework and its languages on other platforms.

3.2.2 Parts of .NET

One of the main power of .NET environment lays in ability to use many technologies and approaches which are well combined. Here is the list of those parts:

- **ASP.NET** – technology for development of Web applications.
- **Windows Communications Foundation (WCF)** – technology for development of Web services and communication infrastructure of applications.
- **Windows Workflow Foundation (WF)** – technology used to define heterogeneous sequence processes.
- **Windows Presentation Foundation (WPF)** – is used for creation of visually impressive graphical user interfaces for applications.
- **Windows Card Space** – implementation of Information Cards standard.
- **LINQ** – Language INtegrated Query ensures object-related access to data in databases, XML and other objects, which implements interface IEnumerable.

.NET environments:

- **Microsoft .NET Framework** is most distributed platform for PC with OS Windows. Starting from version Windows 98 where it can be installed later, then it is included directly in installation in Windows Vista and newer.
- **Microsoft .NET Compact Framework** is platform used on compact devices and mobile phones which uses OS Windows Mobile.
- **Microsoft .NET Micro Framework** is directed to use on even smaller and less performance devices than Compact Framework, such as low-end mobile phones and for experimental use.
- **Mono** is product of independent open source community implementing .NET runtime for Unix-based OS such as Linux, Mac OS X...

3.2.3 Architecture

“The whole .NET Framework is based on the general idea of CLI - Common Language Infrastructure, but Microsoft called it CLR - Common Language Runtime, which provides a language-neutral platform for application development and execution. It includes functions for garbage collection, exception handling, interoperability and security. Thanks to implementation of those core aspects within the scope of the CL it does not stick to one language, but gives possibility to use across all languages supported by framework.” [6]

Security is ensured by its own mechanism which consists of two general features: CAS - Code Access Security which is explained in the next paragraph and a well-known, used and proven mechanism of validation and verification.

“Code Access Security (CAS), in the .NET framework, is Microsoft's solution to prevent untrusted code from performing privileged actions. When the CLR loads an assembly it will obtain evidence for the assembly and use this to identify the code group that the assembly belongs to. A code group contains a permission set (one or more permissions). Code that performs a privileged action will perform a code access demand which will cause the CLR to walk up the call stack and examine the permission set granted to the assembly of each method in the call stack. The code groups and permission sets are determined by the administrator of the machine who defines the security policy.” [7]

“This framework includes set of standard class libraries, it is organized in a hierarchy of namespaces usually begins with “System.*” Those libraries implements large number of common functions, such as file reading and writing, graphic rendering, database interaction, and XML document manipulation, among others.” [6]

There exists “Base Class Library” which works as the basic API of CLR and it consists of a small subset of the entire class library.

“The next important library is Framework Class Library (FCL) which is a superset of the BCL classes and refers to the entire class library that ships with .NET Framework. It includes an expanded set of libraries, including Windows Forms, ADO.NET, ASP.NET, Language Integrated Query, Windows Presentation Foundation, Windows Communication Foundation among others. The FCL is much larger in scope than standard libraries for languages like C++, and comparable in scope to the standard libraries of Java.” [6]

Memory management is handled on framework side so the developer does not have to worry about it. It is handled in a separate thread from the main application where periodically runs the garbage collector. It is non-deterministic, compacting, mark-and-sweep designed garbage collector and runs only when certain amount of memory has been used or when there is shortage of memory in the system.

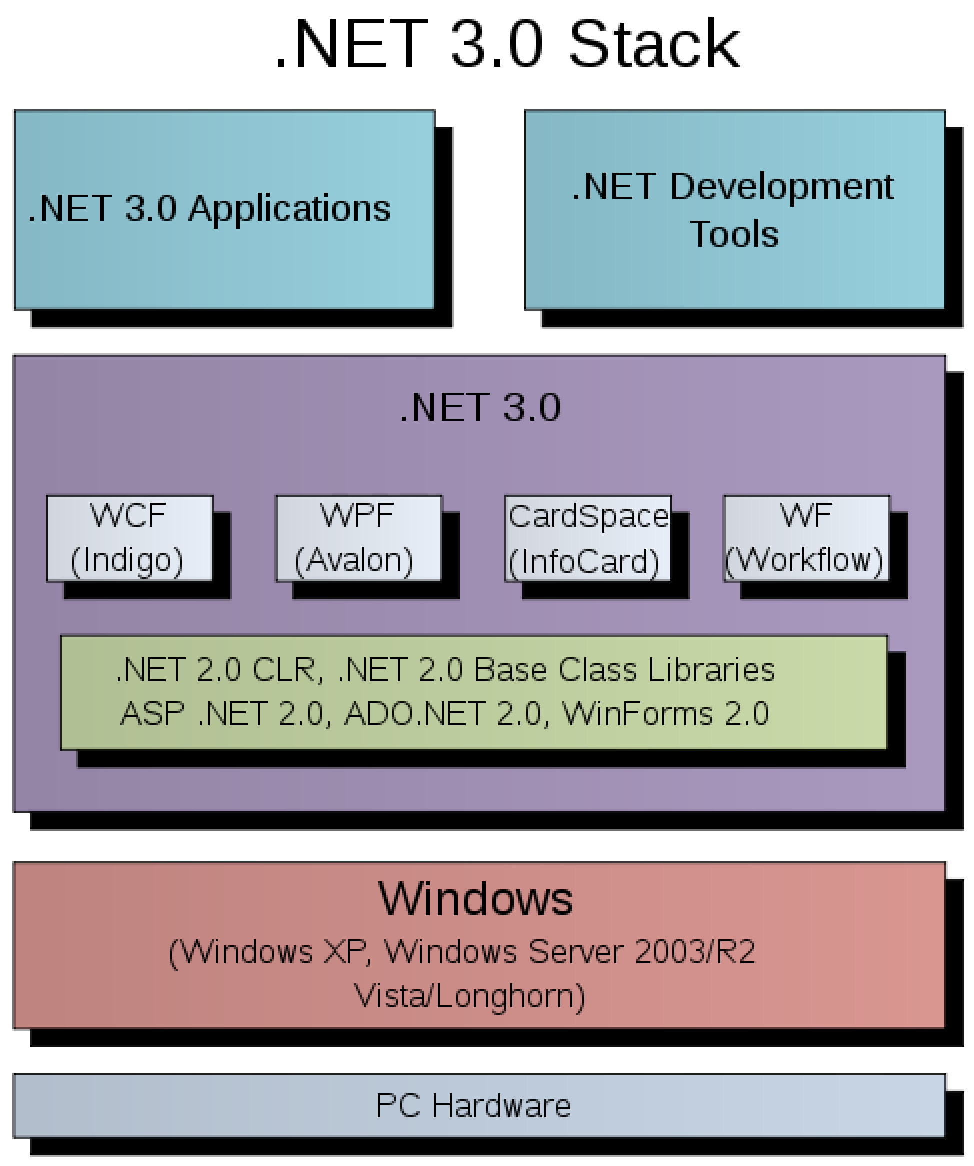


Figure 3: NET architecture overview

3.3 XML

eXtensible Markup Language is a well-known standard developed and standardized by W3C for storing hierarchical data in text format mostly used for information exchange and data transfer.

From the beginning the main aim was to create an international easy-to-use open text standard for information exchange which can be used by anyone. This aim was achieved.

Internalization was supported by native use of Unicode coding. XML documents have very high information value – especially in information given vs bytes used ratio. There is built-in automatic control of document structure which helps to avoid errors. Also there is an easy way to convert this strictly hierarchical document to a various number of other formats and thus makes it quite easy to use in hundreds of applications.

There are two main approaches to process XML document either Document Object Model (DOM) parser which creates a tree in memory from XML document or Simple API for XML (SAX) parser which is going through XML document and raises events which the programmer can process.

3.4 SQL

Structured Query Language is a standardized programming language designed for queries over relational databases. Its development is dated from the 1970's in IBM company. It is also called SQL-86 which signifies the year when SQL was standardized by ANSI, since that time many improvements were made and that is why there were new versions SQL-92 aka SQL2 and the newest one SQL-99 aka SQL3.

The structure of the queries is hierarchical and logic. Let us show an extended example on the most used command – SELECT. This command - like most of other commands – has to be used only after there is an established connection to the database

- SELECT – specifies what data (columns) should be shown e.g. name, address, ID...
- FROM – this command points to the table where the desired data is stored.
- WHERE – specifies the subjects (rows) in which we are interested.
- ORDER BY – and finally if there are more results this command is designed to sort them.

There are four main SQL queries groups:

- Commands for data manipulation (SELECT, INSERT, UPDATE, DELETE, …)
- Commands for data definition (CREATE, ALTER, DROP, …)
- Commands for access privileges (GRANT, REVOKE)

- Commands for transactions (START TRANSACTION, COMMIT, ROLLBACK)

Every relational database these days implements SQL standard. But still transferability of code from one environment to another is problematic. Problems occur during transfer when some environment-specific expressions are used. Those expressions use most of the databases nowadays. Some translators were made, but not all of them work properly and because of this it is very important to choose the database very wisely before implementation.

3.5 LINQ

Language INtegrated Query is a new approach how to access and work with different data sources using still the same code which is a great advantage for programmers – they need to learn just one approach. Most of its principles were presented among with .NET Framework 3.5 even though the basic concepts were used even before.

Data could be stored in various formats and environments like Objects, SQL, XML or DataSets. Access to them is still the same from the programmer's point of view. The only thing which is changed is the interface to each of these environments. Thanks to well-designed LINQ architecture it is possible to develop implementation for new data sources i.e. Amazon web pages. This freedom is based on Expression Trees which enables to work as well with the code as with the data, thus every LINQ query can be translated for adequate data source, no matter if the type of the data is relational, hierarchical or else.

Also working with data in code brings one very important improvement which is controlling all accesses to data during compilation and not when the program is already running. This stronger control is very useful and able to protect most of the programmer's mistakes in the beginning and thus it is a big time saver.

Along with translation rules from so-called “query expressions” to expressions using method names, lambda expressions and anonymous types. These can be used for example to project and filter data into arrays, enumerable classes, XML (LINQ to XML), relational databases, and third party data sources.

In this project were used LINQ to XML and LINQ to SQL, thus those will be described below.

3.5.1 LINQ to XML

“LINQ to XML provides an in-memory XML programming interface that leverages the .NET Language-Integrated Query (LINQ) Framework. LINQ to XML uses the latest .NET Framework language capabilities and is comparable to an updated, redesigned Document Object Model (DOM) XML programming interface. The LINQ family of technologies provides a consistent query experience for objects (LINQ to Objects), relational databases (LINQ to SQL), and XML (LINQ to XML).” [15]

3.5.2 LINQ to SQL

“In LINQ to SQL, the data model of a relational database is mapped to an object model expressed in the programming language of the developer. When the application runs, LINQ to SQL translates into SQL the language-integrated queries in the object model and sends them to the database for execution. When the database returns the results, LINQ to SQL translates them back to objects that you can work with in your own programming language. Developers using Visual Studio typically use the Object Relational Designer, which provides a user interface for implementing many of the features of LINQ to SQL.” [14]

3.6 WCF

Shortcut WCF stands for Windows Communication Foundation and was previously known as “Indigo”. Basically it is an Application Programming Interface (API) operating inside the .NET Framework mostly used for the development of Service Oriented Applications (SOA) which needs to maintain connection. Where it occurs in the hierarchy of .NET Framework can be seen in the figure in chapter 3.2.

One of the strongest parts of WCF is connection of various Microsoft technologies which are used to create distributed applications:

1. ASP.NET Web Services (interoperability)
2. Web Services Enhancements (security, reliable messaging, sending attachments)
3. .NET Remoting (performance)
4. Enterprise Services (life cycle maintenance of objects, distributed transactions)

5. Microsoft Message Queuing (reliable data delivery)

Back in the days the choice of using some concrete technology was based on the desired application behavior, not anymore. Nowadays during development of quality application there is strong need to use at least two technologies of the above list. It brings problems as well as better solutions. The main idea is to use all of those applications not even knowing that you are using them. So there is no more hard time during the development and maintenance of such applications. Thanks to connection of those technologies in WCF it is much easier to develop, implement and maintain distributed applications.

3.6.1 Services

Services are a basic structural element in WCF applications. Service is on the highest place in WCF hierarchy. It is basically a system with which we communicate by its endpoints. Every service can have one or more of them. The main idea of the endpoint is to serve for receiving messages and sending answers to those messages.

Three messaging patterns exist:

- simplex - oneway message transfer.
- duplex - asynchronous twoway message transfer.
- request-reply - synchronous twoway message transfer.

We can set up application behavior also by three ways:

- in code.
- configuration.
- combination of the previous two.

With the apperance of WCF came also new terminology which needed to be understood, so it will be explained now in details:

Terminology

Message is the basic communication entity between client and server - between particular services and their concrete endpoints. It is a bunch of data consisting of head and body. Body could be whatever is needed by concrete service. Head has to consist of details how and where to transfer this concrete message.

Service is a basic element of the whole concept. It is a system which provides one or more endpoints. Its surrounding is seen like a collection of endpoints. It consists of three basic parts:

- Class which implements the desired service.
- Host environment.
- One or more endpoints.

Client-service communication by sending messages between endpoints is clearly observable tin the following figure:

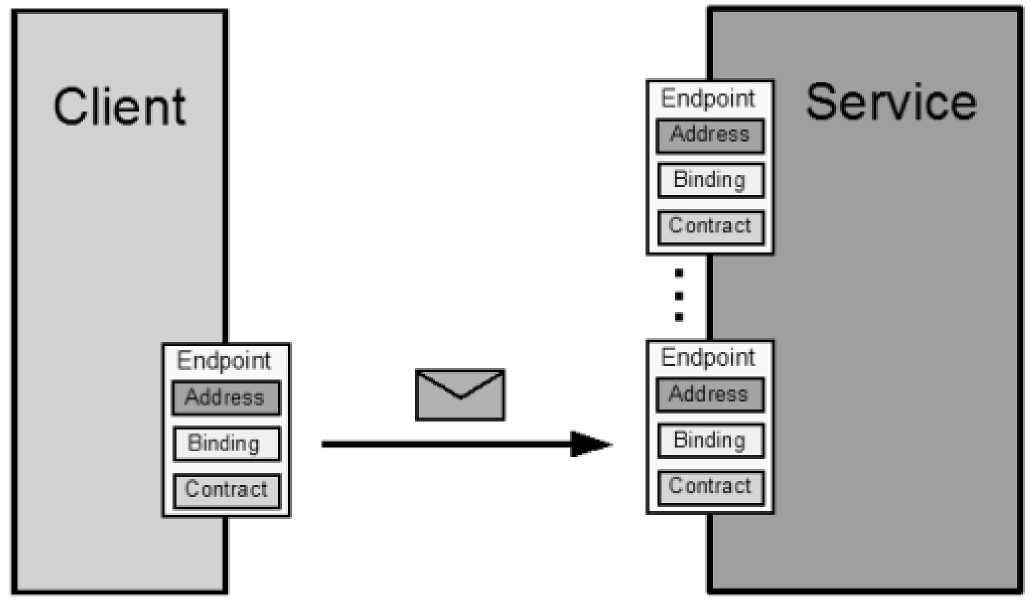


Figure 4: Client-Service communication

Endpoint is place for receiving and sending messages consisting of three main parts:

- Address
- Binding
- Contract

Address specifies where the messages are sent to.

Binding specifies a communication mechanism and describes how messages should be sent. The basic settings are method of transfer (HTTP,FTP,...), encoding (text, binary), security, sessions and support of transactions.

Contract chooses which type of messages can and can not be sent and received.

Hosting is an application which controls the life cycle of a service.

Channel is a message transfer environment. It is necessary to set up channel between two endpoints before any message will be transfered.

Metadata describes a service. This description can be used by external systems to communicate with a specified service.

3.7 WPF

WPF stands for Windows Presentation Foundation also known as Avalon is a part of .NET Framework since version 3.0. WPF implements XAML, a derivative of XML, to define and link various UI elements and to develop Rich User Interface (RUI). Technology WPF is being built into Microsoft OS since Windows Vista and Windows Server 2008, it is also downloadable for Windows XP SP2 and Windows Server 2003. XAML works like divider of functionality and appearance of application.

The aim of WPF is to unify interesting user interface – vector and bitmap, 2D and 3D graphics, animations, audio and video. Let us describe some basic parts.

3.7.1 Graphics

All parts of graphics system use Direct3D libraries which enable hardware acceleration by graphics card and advanced graphics abilities. Also supports 3D rendering. Vector graphics is innovated in a way of describing objects by mathematic expressions which allow zoom-in without blurring.

3.7.2 Supporting older libraries

Thanks to cooperation with WinAPI it is possible to host there WPF code and vice versa. Also possibility to cooperate with WinForms is available by using classes of libraries WindowsFromsHost and ElementHost.

3.7.3 Data binding

Allows work with data from a variety of data sources – databases, XML files etc. There are three basic ways of bindings:

1. “one time” just download data and ignores further actualizations.
2. “one way” as name presumes the connection is one-way but persistent with read-only access to the source.
3. “two way” standard way to communicate – actualizations are being synchronized on both parts.

3.7.4 Animations

Playground for animations is called storyboard – there is rendering of all animations being processed in specified time intervals or on demand by some event like mouse click, start of application or finish of some specified task. This part of WPF is delivered with few predefined effects of how every picture should be switched to the next one i.e. effect fade out.

4 Implementation

This chapter will describe how the objectives specified in 2 Objectives chapter were met and we will also go through the implementation part. Objectives are separated in three different projects so we will describe these one by one in order how we worked on them.

4.1 Database

At the beginning the database structure should be explained because it makes an important part of the project and actually each of the programs used it. Although each program uses different part of the database altogether we used all tables. This chapter will explain this database in general. There will be more detail information on parts that were actually used with our programs and we will refer to them in the text later.

4.1.1 General overview

As you can see in figure Figure 5: Database Structure 8 tables were used in the database structure. Our database structure is holding valuable data that is used by our programs and also by the mentioned 1.1.3 Web application. There will be a short description below about the purpose of each table.

1. **User:** This table contains useful information about user. We also used password salt and password hash for storing this information safely.
2. **User in role and Role:** These two tables are used for storing information about which role is belonging to the user. Roles define access rights for the user. in section 4.3 Presentation Creator will be described how this information is used.
3. **Owner:** This table is a joining table between User and Pylon. It allows to hold information that one user can manage many Pylons and one Pylon can be managed by many users. This relationship is called many to many (M:N) and in this case additional table like Owner is needed.
4. **Pylon:** This table contains information related to Pylon.
5. **Upload:** Upload table is a joining table between Pylon and Content. It allows to have

more Contents on the same Pylon and more Pylons displaying the same content. We came out with the idea to have time schedules for presentations on the same Pylon and that is the reason why there is need to have more contents on the same Pylon. Section 4.2 Presentation Downloader will describe this solution deeper.

6. **Content:** This table contains information about Contents.

7. **User Profile:** This table contains more detailed information about User.

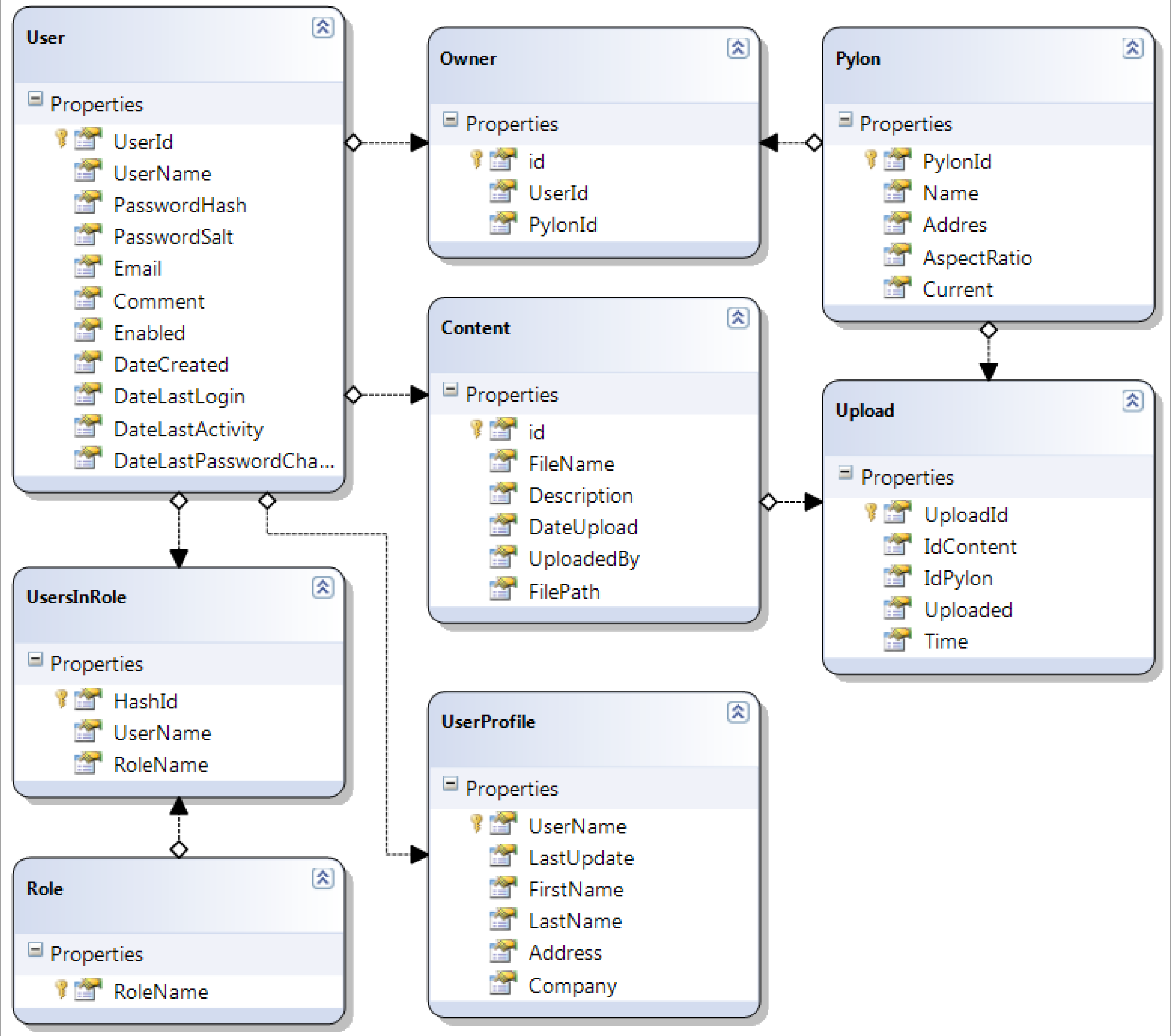


Figure 5: Database Structure

4.1.2 Database implementation

At the beginning we decided to use the standard way how to work with the database. We wanted to make Object-related mapping as a whole by ourselves though we were aware that it is quite a time consuming work. However, we considered that there could be an easier way how to do it and so we decided to use some more sophisticated technology that will do all this stereotypical work. At the end we used 3.5 LINQ instead of coding everything by ourselves. We were amazed how easy it is to use this technology. Firstly we generated all objects from tables by pressing one button. All generated Entities were generated into C# Objects to DataClassesDataContext class. We could spend couple of hours to implement all of this code by ourselves. However querying into database was a bit more complicated. We had to learn about lambda expressions that are used for writing a query into the database.

```
public IEnumerable<Content> GetContent(string userName)
{
    using (var dc = CreateDataClasses())
    {
        User user = (from p in dc.GetTable<User>()
                            where p.UserName == userName
                            select p).SingleOrDefault();

        int userID = user.UserId;
        return dc.Contents.Where<Content>(p => p.UploadedBy ==
userID).ToList<Content>();
    }
}
```

Code 1: LINQ example

As you can see on Code 1: LINQ example syntax of LINQ and lambda expressions are a bit different than the standard code. Let us explain this example code line by line. At the first line we specify the function in a classic way. This is a public function that is returning collection of Contents belonging to specified user. The next line is used for calling the constructor of Class containing all generated objects and all the code needed to connect to the database and work with it. Statement Using ensures disconnection from the database. In this Using block actually a more interesting part of querying to the database is beginning. We select User with the specified name by parameter userName and store him in the variable user. Object User is essentially converted table User from Figure 5: Database Structure to the Object class User with the same parameters. Because we got all information about user we can use his userID as a foreign key to the table Content. This allows to get a collection of Contents that belongs to the user. As it can be seen on the last line lambda expression was used there

`p => p.UploadedBy == userID`. Operator Where expects a delegate function that accepts single a object as an input and returns boolean value. Because lambda expressions are substitutions for a delegate we can use them for this purpose. This lambda expression returns true when foreign key UploadedBy in the table Content is equal to the obtained UserID. This instance of Object Content is then added to the collection and the whole collection is then returned by this function.

4.1.3 XML Database

To preserve compatibility between our applications and applications created by Finnish students we had to use their XML database model. They used an XML file named Diaconfigure.xml to store all valuable information about media items that belong to the presentation. The structure of this file is presented on 4.1.3 Code 2: Diaconfigure.xml example. There is one picture with the name Desert.jpg and other needed information.

```xml
<?xml version="1.0" encoding="utf-16" standalone="yes"?>
<Lista>
  <Rivit>
    <Id>0</Id>
    <Polku>Diat\Desert.jpg</Polku>
    <Alkupolku>C:\Desert.jpg</Alkupolku>
    <Nimi>Desert.jpg</Nimi>
    <Formaatti>Kuva</Formaatti>
    <Koko>845941</Koko>
    <Efekti>0</Efekti>
    <Esitysaika>5</Esitysaika>
  </Rivit>
</Lista>
```

Code 2: Diaconfigure.xml example

4.2 *Presentation Downloader*

Presentation Downloader was the first project and this section describes how the objectives were met from 2.1 Presentation Downloader.

4.2.1 Connection with server

The first objective was to find out how to solve this communication issue. We came out with two variants. First of them was to open an VPN tunnel between Pylon and server from the Pylon side and the second one was to check every few minutes if there is something new on the server for displaying. Below there will be a description of those two variants in detail.

4.2.1.1 VPN tunnel

A virtual private network (VPN) is a secure network that uses primarily public telecommunication infrastructure. One of the strongest pros of this solution is that after we open the VPN tunnel between server and Pylon then the communication could work in both ways. This means that a presentation can change on the Pylon immediately when the user will change it. Another advantage is also that VPN ensures a very secure connection. However we finally decided to take the second variant because it was a much easier way and it did not matter that much if the presentation was displayed immediately or after at most one minute. We also did not know how many VPN connections the server could hold at the same time.

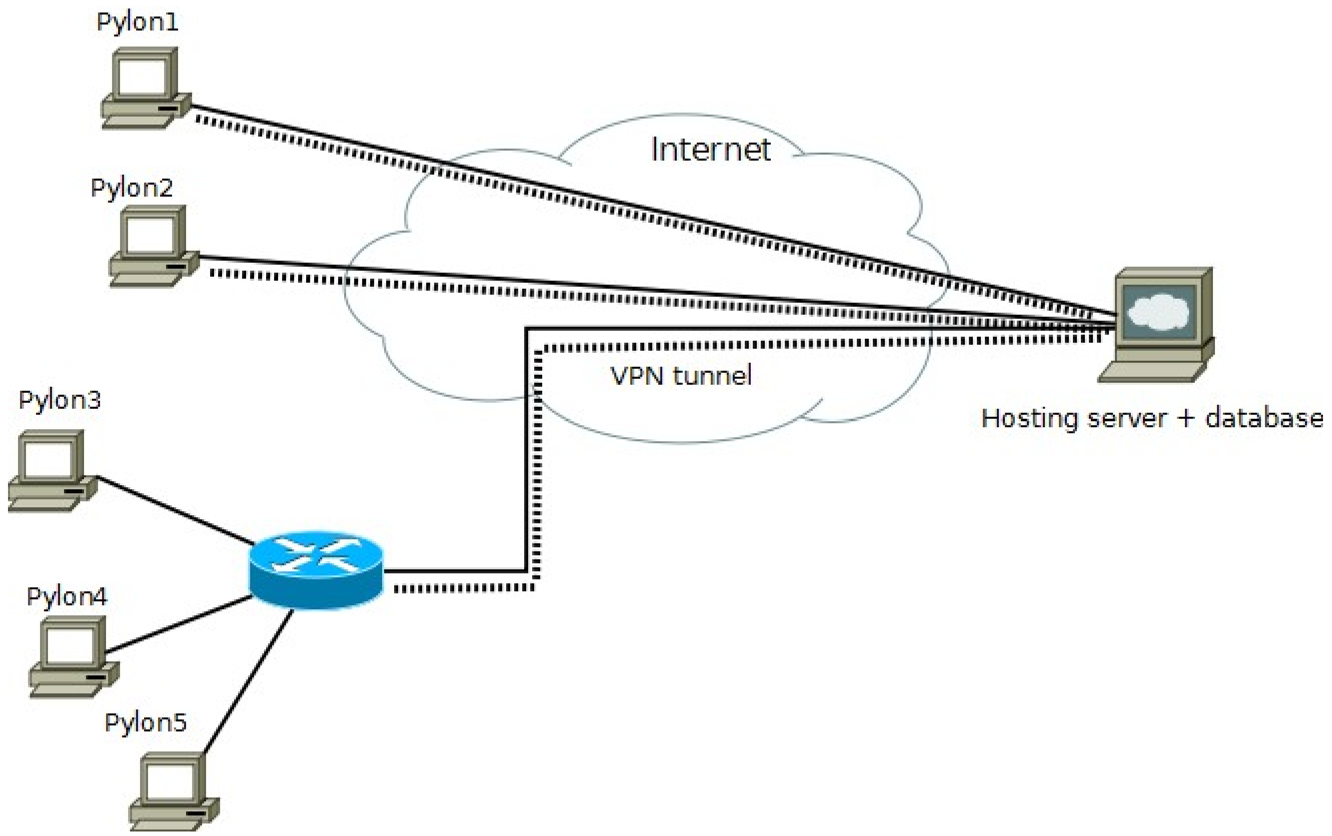


Figure 6: VPN tunnel example (VPN tunnel represented with dotted line)

4.2.1.2 Pylon – server communication

This solution is based on one side communication from Pylon side to server. Pylons connects to the server each minute and checks if there is something new in the database to display. You can see that this solution is quite easy and solves perfectly an issue specified in 2.1 Presentation Downloader. The above mentioned disadvantage is that a presentation will not be displayed immediately, but at most after one minute plus download time. After consideration we finally decided to use this approach to achieve the objective.

4.2.2 Integration

The second objective was to consider how to integrate the program with the existing program for displaying presentations 1.1.1 Presentation viewer. There were two solutions how to do it. First of them was to run Presentation Downloader in separated a thread and the second one to run Presentation Downloader as a separated process. Because Finnish students were still working on their program it would not be a good idea to dig into their current version and make a fork of their program with our integrated Presentation Downloder. As you can see in the objectives there was no required communication between these two programs. We just had to download a new presentation from the server and then provide it to their program by copying to the predefined folder. Because of these reasons we decided to make our program as a separated process. We of course used the same programming language C#. These programs thus could be easily merged together after the final version of 1.1.1 Presentation viewer will be released.

4.2.3 Implementation

After we considered 4.2.1 Connection with server and 4.2.2 Integration we started working on the implementation. According to the above requirements for program functionality it was not too complicated so even our program is not so complicated. We will show the whole functionality by describing Figure 7: Presentation Downloader Flow Chart.

1. After our program is running we need to find out what is the PylonID of this current Pylon. There were many ways how to do it. We were thinking for example about storing some unique characteristic like mac address at the server side. Pylon would connect to the database and send this characteristic to the server and receive his PylonID as a result form the server. Finally we decided to just store this PylonID in a file and so our program. Then it reads its own PylonID from this file.
2. In the second phase there is "endless" loop until the program will be terminated. The first step in the loop is to check if there is something new in the database. Our program uses its PylonID that it gets from the encrypted file and uses it for getting all presentations that belong to this Pylon. That means a user made a link between a created presentation and Pylon with this PylonID.
3. When we get all the belonging presentations we have to decide which one should be currently presented. We came out with an idea that there will be a possibility to create time schedules for presentations. Each presentation has a specified time when it should be presented. This is done by comparing differences between Pylon's and presentation time. For example if right now it is 12:00 and we have two presentations Presentation1 with time 11:00 and Presentation2 with time 12:30 our program will

choose Presentation1. Our program will provide Presentation2 after it will check the database and current time will be later than 12:30.

4. After we decide which presentation should be displayed on Pylon we can proceed and look if this presentation is already downloaded or if we have to download it.
5. When our presentation is downloaded on a hard disk we should according to 2.1 Presentation Downloader objectives provide an archived file with the exact name to the predefined folder where the program for displaying presentations 1.1.1 Presentation viewer is expecting it. For archiving files we used DotNetZip library. After all these steps are done program can sleep for 1 minute and then jump to point 2 again.

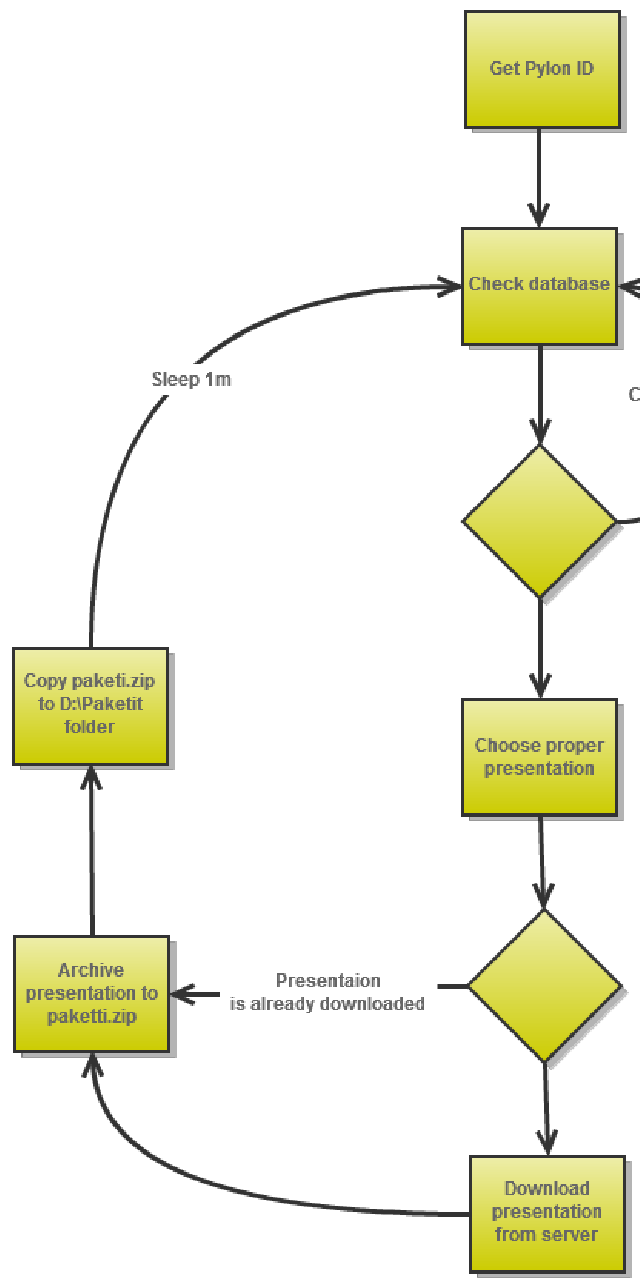

Get Pylon ID
Check database
Connection Failed
Sleep 1m
Sleep 1m
Copy paketi.zip to D:\Paketit folder
Choose proper presentation
Archive presentation to paketti.zip
Presentaion is already downloaded
Download presentation from server

Figure 7: Presentation Downloader Flow Chart

4.2.4 Testing

This application was the first project and we used it later during the whole implementation phase. Due to this reason the application is well tested and flawless. As you can read in section 4.4 Pylon Client this program is also a part of the last project Pylon Client. It means that if Pylon Client is tested then this program is tested also.

One bug appeared when Presentation Downloader was running on the Pylon's hardware. Our program ended with an exception because 3G connection that is used on the Pylon seems to be a bit unstable. This issue was immediately resolved.

4.2.5 Conclusion

In these steps we went through the program functionality so it should be clear now that we met all the requirements specified in 2.1 Presentation Downloader. We also presented our program to the customer at the meeting and everything worked perfectly. The customer, the supervisor and also the Finnish students were satisfied with the solution so we released this project in this state. It should not be a problem at all to merge our program with 1.1.1 Presentation viewer and run it in a separated thread because it is written in C#. We discussed this possibility to run our program in a separated thread as a same process in section 4.2.2 Integration. Our program is as mentioned above using database so there is needed to merge DataClasses.dbml as well. This part is in detail explained in part 4.1.2 Database implementation.

4.3 Presentation Creator

Second project was to create a program for creating presentations directly on the website 1.1.3 . Firstly we were thinking about creating something simple just in asp.net like the pages but then we decided to use some more advanced technology to be able to give a nice user interface. We used Silverlight technology instead of flash because everything was written in C# yet and we wanted to take advantage of that. We were also able to use Microsoft development tools like Expression Blend because our school in Ostrava allow us to download some of them for free. We used many technologies from Microsoft in this project and most of them are described in section 3 Used technologies. We will refer to them them specifically in the text below. As you can see in the objectives in section 2.2 Presentation Creator we had many objectives related to this project and we also made our own objectives. As you can guess from our objectives this project becomes quite difficult and thus we were of course struggling with many things. There still also remains some stuff that needs to be done.

4.3.1 Design

This section focuses on the design. Functionality and implementation will be described in next section. The design was influenced by the design of currently developed pages 1.1.3 because it will be hosted on them. We wanted to keep the same design and colors as on this web page. In Silverlight we are using User Controls for displaying the desired items in the Silverlight program window. User Control can have as a child object for example grid, stack panel, canvas and so on. We can add to this objects our desired items that will be displayed like images, buttons, ListBox, etc. We used two User Controls in our program. The first for choosing which presentation should be edited 4.3.1 Figure 8: User Control 1 and the second for editing currently selected presentation Figure 9: User Control 2. As you can see on second User Contro we used customized buttons with presentation name and picture belonging to presentation I. The first button is for creating a new presentation. For customizing buttons we used Expression Blend because there are many tools for doing such things like customizing controls elements. We used this tool for customizing all used elements like Combo Box, TextBox, Numeric Up Down, etc.

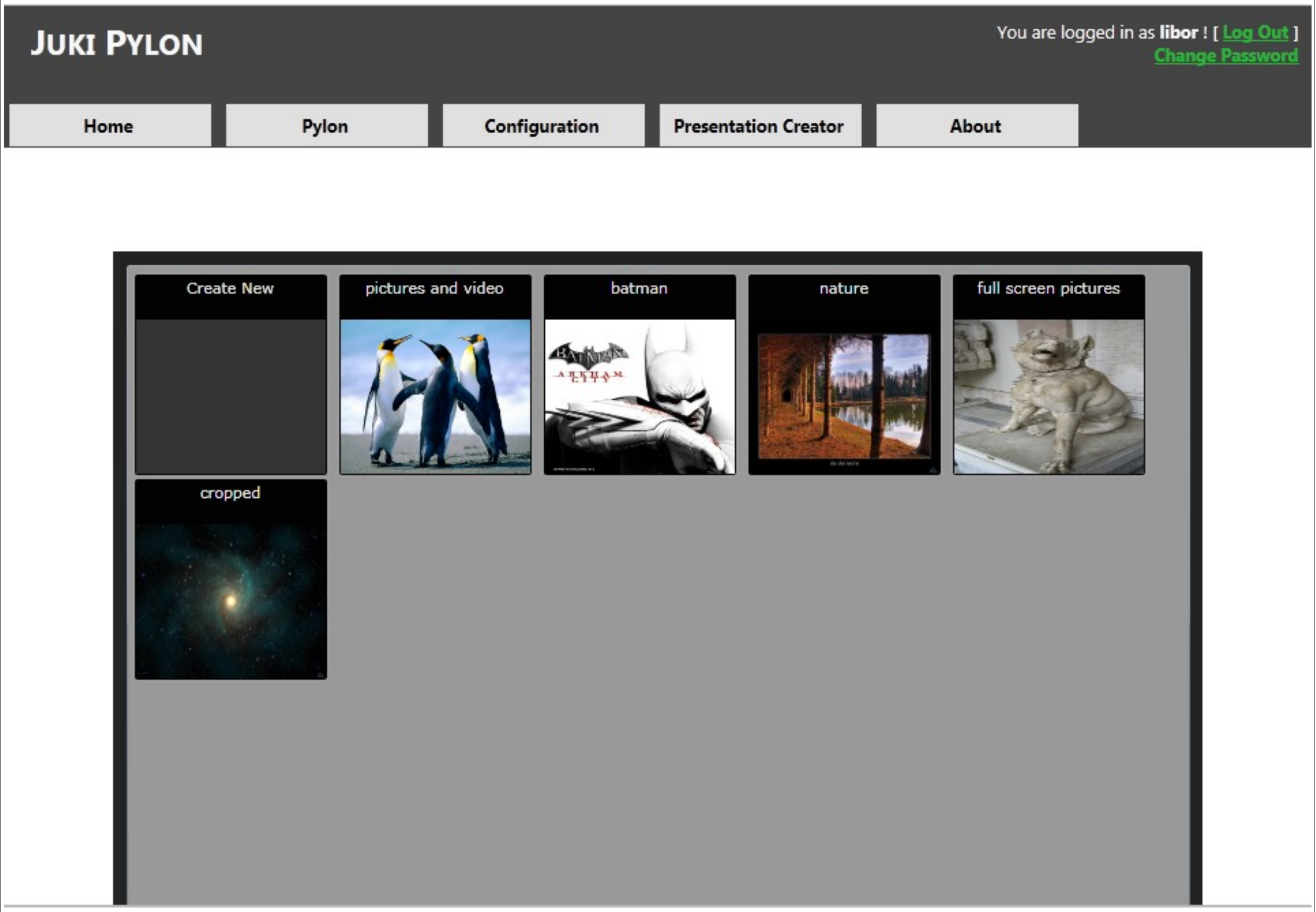
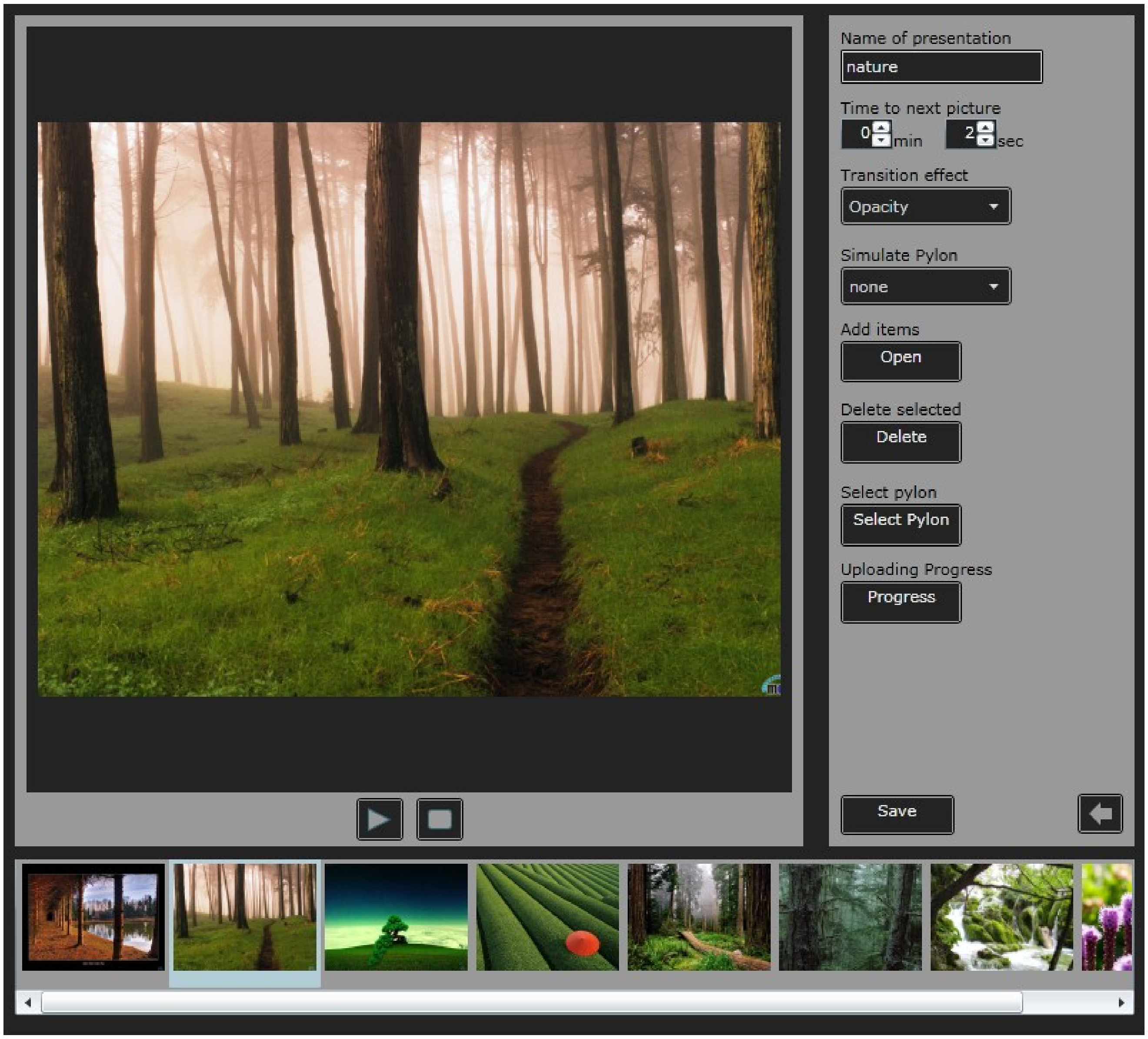


Figure 8: User Control 1

The second User Control Figure 9: User Control 2 is used for editing or creating presentations. All desired functionality from 1.1.2 Presentation creator is accessible here. As you can see we decided to divide this User Control to 3 parts. The first part on the left is used for displaying media elements like videos and images. At the bottom of this part two buttons are placed. They are for controlling the slide show presentation. Media elements are displayed according to the order of elements in the ListBox. This list box is positioned on the bottom of the page. The user can change the order of items by dragging one item over the others and placing it on a desired position. We also added one extra animation for this created in Expression Blend. For generating thumbnails of video elements we used program ffmpeg. This program could also be used for other things but it will be discussed in later sections. The main controlling panel is located on the right. The purpose of each control element will be shortly described.

1. TextBox for placing the name of presentation.
2. Numeric Up and Down elements for specifying time. Time means how long the current media element should be displayed.
3. In this combo box you can specify which transition effect will be used for the next picture after reaching this mentioned time.
4. This combo box serves for the simulation aspect ratio of selected Pylon's display. There is a menu of your assigned Pylons and if you select one them the image displaying element will change its size. Displayed media elements will be then looking exactly like on the Pylon screen. It enables you to see if your image fits to the Pylon screen well. If not then you should crop or change the size of your image.
5. The next button allows you to add media elements from your hard disk.
6. Delete button serves for deleting selected elements from ListBox with added elements. You can select one by left click, few by left click with ctrl or all with ctrl + a shortcut.
7. Select Pylon serves for selecting Pylons that will be displaying current presentation. You can not specify the exact time here but the current time will be selected autocratically after pressing save button. You can specify the exact time on the web page 1.1.3 Web application.
8. Progress button is used for displaying progress bars with the current uploading state of items and their names.
9. Your presentation will be saved on the server by pressing save button.
10. You can go back without saving by pressing this button with arrow.


Name of presentation: nature

Time to next picture: 0 min 2 sec

Transition effect: Opacity

Simulate Pylon: none

Figure 9: User Control 2

4.3.2 Implementation

We will focus there on the key parts of the project and describe all the issues that we had during the implementation and also on our implemented solutions. We want to make the text more clear so we will describe our procedure of implementation one by one in the order it was proceeded. We used WCF services 3.6 WCF for querying to the database.

1. First of all the application needs to know which user is logged in. We were struggling a

bit with these issues in the beginning. There are two ways how to do it and each has its own disadvantages. The first option is to give this user name by initial parameter to the application. This option was quite easy to do and worked perfectly but there are some security issues. The second way how to do it is by using WCF services. This way seems to be safer than the first one and it was successfully implemented. Everything worked fine on the local server but when we deployed our application to the internet hosting then a problem appeared. We were unable to start the application from the variety of browsers like Chrome and Firefox. The only browser that was working with this implementation was InternetExplorer from Microsoft. We could not find out where the problem was so we decided to use temporarily the first variant to allow users with different browsers to use our application. We hope that this issue will be resolved in the future by other development team. This code is located in App.xaml file and the second variant is commented.

2. The next step was to consider how to change the current storage method that was used on the pages 1.1.3 Web application. The old way was to just store presentation in zip files created by the program from Finnish students 1.1.2 Presentation creator and then easily download them by the already implemented 4.2 Presentation Downloader. This way of storage was very inappropriate for our Silverlight application because if we would use it then we would have to unzip files all the time to some tmp folders when somebody wants to work with them. Thus we decided to create some folder structure to keep our presentations unarchived. There was no reason to keep them archived. Even the storage space is almost the same because media elements are already compressed by sophisticated algorithms like jpeg, png or wmv. There could be a question how we will download files belonging to presentations by our 4.2 Presentation Downloader then. There is Diaconfigure.xml in each presentation folder that contains all valuable information about files belonging to the presentation. So because of these reasons we created a folder structure that contains unarchived pictures that can be used for direct displaying by our application. The folder structure is very simple. As a root folder there is uploaded_files and it contains folders with user names that used our application. Each folder with user name contains users owned presentations and one temporary folder for a new presentation. So if more than one user creates a

presentation at the same time than there will not be any collision because each one will use his own temporary folder. For creating, deleting and copying these folders we used WCF services.

3. When we successfully implemented steps one and two then we could start by creating our User Control 1 Figure 8: User Control 1. Grid view is used as a layout root and list box that contains customized buttons with the presentation name as title and one picture from the presentation as preview. For displaying these properties we need to make query to the database and get a collection of presentations that belongs to the currently logged user. We got this user name in step 1. This needed information is stored in the database table named Content as you can see in the 4.1.1Figure 5: Database Structure.
4. The user has two possibilities now. The first is to click on already created presentation and edit it and the second is to create a new one. The next user control will be the same for both options and user will have also same possibilities what he can do with presentation. Only one difference is that new created presentation will use temporary folder as we mentioned in step 2.
5. So when you choose one option from 4 then you will be redirected to second User Control. We implemented nice transition effect between user controls based on [Robust UserControl Navigation in Silverlight]. After smooth opacity animation the desired User Control for editing presentation appears.
   a) List Box: If a user clicked on some existed presentation then images can be immediately downloaded and displayed in the list box without querying to the database because we already have all presentations mapped to the Objects in our collection how we described in step 3. You can select an item from the presentation by clicking on the image in the list box on the bottom of the user control. If the item is image then it will be immediately displayed.
   b) Video: If the item is a video then the quality of streaming strongly depends on the data transfer speed between server and client and also on the quality of streamed video. Silverlight surprisingly supports a small amount of video formats. We were able to play only wmv with VC-1 encoding. That is an issue that needs to be solved

somehow because when the user uploads a video in a different format then the video will not be displayed. We will focus on this issue more in section 4.3.5 Conclusion.

c) Transition Effect: Classic Storyboard animations were used for transition effects between pictures. It allows the developer to easily create new effects without any problem. Our transition effects are just an example how it can look. We hope that somebody will create more and better animations. We created only two simple animations. One is named Opacity and the other Scale. Opacity works when you click on another image Opacity down an animation will be triggered on the current image. Opacity down will smoothly decrease the opacity of the picture from 1 to 0 in 0.5 seconds. After this animation is completed Opacity up animation is triggered on the selected picture. Opacity does the same as Opacity down but from 0 to 1. The Scale effect works very similar but with the scale transformation.

d) Select Pylon: For implementing this functionality we used a collection of check boxes that contains Pylons that belong to the user. This information can be found in table Owner. You can the check database structure on picture 4.1.14.1.14.1.1Figure 5: Database Structure. Then we had to find out which Pylons have presentation assigned to them. This information is contained in table Upload. If Pylon has this presentation assigned then its check box will be in checked state.

e) Uploading: It is very surprising but there is no easy way to upload files to the server. We had to use a very low level way to do it by creating a stream between client and server and send byte by byte manually in the while cycle. There is a method OpenWriteAsync in the WebClient that allows us to do that. We had to create Generic Handler that was receiving data from the stream on the server side. We also wanted to display a progress bar that allows the user to see how many percent of the file size was already uploaded. That functionality made our implementation even harder because we now had to write files to the stream by chunks. After each chunk is uploaded to the server the progress bar state is changed depending on how many chunks were already uploaded.

f) Save: Save button serves for saving the whole presentation with all needed information into the database. That means creating Diaconfigure.xml with information about all media items used in current presentation. If we created a new presentation or changed the name of an edited presentation then a new row should be added or current updated. We need to save information about assigned Pylons to Upload table if something changed. After these steps are done the user is warned for some problems and if not then application is redirected to User Control 1.

4.3.3 Testing

Due to the reasons described in 4.3.5 Conclusion we even did not have enough time for testing. Testing was conducted mostly during the implementation phase. We are aware about the importance of testing phase but there simply was not enough time to do it properly. Because of these reasons some bugs could still remain there. We will describe all remaining issues that we found out during the testing phase in section 4.3.4 Ideas for improvement.

4.3.4 Ideas for improvement

This section will discuss the issues that should be resolved in the future.

- The first thing that needs to be done is to consider which video formats should be supported by the application. There are a couple of possibilities. As mentioned in 4.3.2 Implementation section there is only one format that is supported by default. We should mention here that we also created a WPF application for displaying presentation 4.4 Pylon Client that supports more video formats. Thus one possibility is to support formats that are supported by Pylon Client and in the Presentaiton Creator application display just an icon of video with no support for playing. Another solution is to implement this support for another formats. The last possibility is to use ffmpeg for converting videos from unsupported formats to supported. The disadvantage of this solution is that this operation takes quite a lot of time, so the user should be warned about this, because a presentation could not be displayed immediately on the Pylon in this case.
- We discussed in 4.3.2 Implementation section an issue about getting logged user name from the server. We have shown that our solution with passing it as an initial

parameter could be hacked and the correct solution with WCF services worked only in Microsoft Internet Explorer. We presume that the solution with WCF services should work in other browsers as well.

- Support for cropping pictures. It is not comfortable for the user to re-size or crop pictures to make them fit to the desired Pylon screen. We think that there could be support for this kind of operations.
- Add more transition effects. We created just an example how these effects could look. Finnish students were not able to make these transition effects working smoothly on the Pylon hardware so we didn't know if these effects will even be supported. We will talk about the results of the work with supporting transition effects on Pylon in section 4.4 Pylon Client.

4.3.5 Conclusion

It was not easy to create this application. One thing that was different for us as programmers was that in Silverlight everything is done asynchronously. That forced us to think differently during the implementation phase of the project. It was a very good opportunity to learn how to code highly a threaded application. We regret that we did not resolve all the issues that we specified in the text and 4.3.4 Ideas for improvement and that we discussed during describing the implementation part. This is partly due to the late involvement to the Pylon project and even later assignment of this particular project. If we had enough time for the project like it was supposed to we would surely have resolved all the issues and deployed flawless application. However, our customer was really satisfied with our solution when we presented our application at the meeting. We implemented all our defined objectives that we delineated in 2.2 Presentation Creator chapter.

4.4 Pylon Client

Pylon client was the last project. As you can see 1.1.1 Presentation viewer similar software was already created by Finnish students. However, because our transition effects in 4.3 Presentation Creator were liked by our customer we were asked by our supervisor to create an application for displaying a presentation with these effects from scratch. Although

Supervisor's idea was to use Silverlight like we used in Presentation Creator we decided to take advantage of WPF 3.7 WPF technology. WPF supports the same technologies and even more than Silverlight. WPF is also more suitable to desktop applications. In WPF it is possible to use Storyboard for creating animations like in Silverlight so that allowed to keep consistence for creating effects between both applications. This application was not so difficult to implement like Presentation Creator but our challenge was to display these transition effects smoothly on the Pylon's hardware. The problem is that Pylon's screen could be very big and the performance of hardware is not so good.

4.4.1 Implementation

Implementation of this application was not so difficult because we had already done our Presentation Downloader 4.2 Presentation Downloader. This application is an example of how easily the Downloader can be integrated into the application from Finnish students 1.1.1 Presentation viewer. Implementation of this integration is already described in Presentation Downloader chapter. The desired effects were also already created in our second application Presentation Creator 4.3 Presentation Creator.

4.4.2 Testing

This application was tested mainly on the fly during the implementation phase. However this application is not so big and complicated and some parts like Presentation Downloader was already well tested so there should not be any severe bugs. We also tried to run our application on the Pylon for a couple of hours, fortunately no bugs appeared but still longer and proper testing should be done.

4.4.3 Conclusion

Implementation of this application was not so difficult but our main objective was to achieve smooth transition effects between pictures. The crucial element to achieve this objective was in choosing proper technology that will support hardware acceleration for rendering these effects. At the meeting it was proven that WPF is a good choice for this kind of application. In spite of weak hardware that is used on the Pylon a satisfying result was achieved. Our customer and supervisor were satisfied with result so we consider that we successfully achieved our objectives specified in section 1.1.1 Presentation viewer.

5 Summary

By given objectives we were supposed to create a complex, independent software which is easy to use, operates over the Internet and gives users a easy way to configure their own Pylons. Taking into consideration all those criteria, we are proudly announcing, that all the objectives were met. The application is fully working, stable and running right now on test Pylon. On our way to meet those criteria we have learned a lot of new things.

Communication with an actual customer was a very big experience and influence on our future lives. Thanks to this possibility we gained some experiences on how to communicate in working environment. We also observed different approaches to problems by different people. By different is meant work occupation, experience, age, nationality and also position from which those people spoke to us. We were able to create a mind picture and also some stereotypes in our minds which could be helpful in the future. It was also a great experience that all this communication was in English language which is our second language. This and writing this thesis was definitely one of the reasons why our English level was improved - despite of all the mistakes in this text.

From a technical point of view it was interesting to get to know new technologies for web application development. We knew and also used before things like XML, SQL and C#, thus we got good background to broaden our knowledge and skills. One of the most exciting and useful things was .NET Framework and Silverlight. We discovered that the way to make proper web applications can be much simpler than we thought before. Thanks to it we also lost some old approaches to solve problems. In a nutshell it is more about to know which tools to use and have a clue of theirs parts than to be a really good programmer oriented only to one side. To support this assertion let us take into consideration the LINQ. On the one hand we knew how to manually create a database and then work with its data by using pure SQL. On the other hand the discovery and use of LINQ saved us so many troubles and improved our total work.

This Bachelor thesis was a big experience for both of us from many different points of view.

5.1 Teamwork

Robert Helesic

- ***Implementation and testing***
  - 4.2 Presentation Downloader
  - Integration of Presentation Downloader into 2.3 Pylon Client
- ***Text chapters***
  - 1 Overview about Pylon
  - 3 Used technologies
  - 5 Summary

Libor Gorcica

- ***Implementation and testing***
  - 4.3 Presentation Creator
  - 2.3 Pylon Client
- ***Text chapters***
  - 2 Objectives
  - 4 Implementation
  - 5 Summary

5.2 Development process

Because only one developer worked on each project we could afford to choose simple development process.

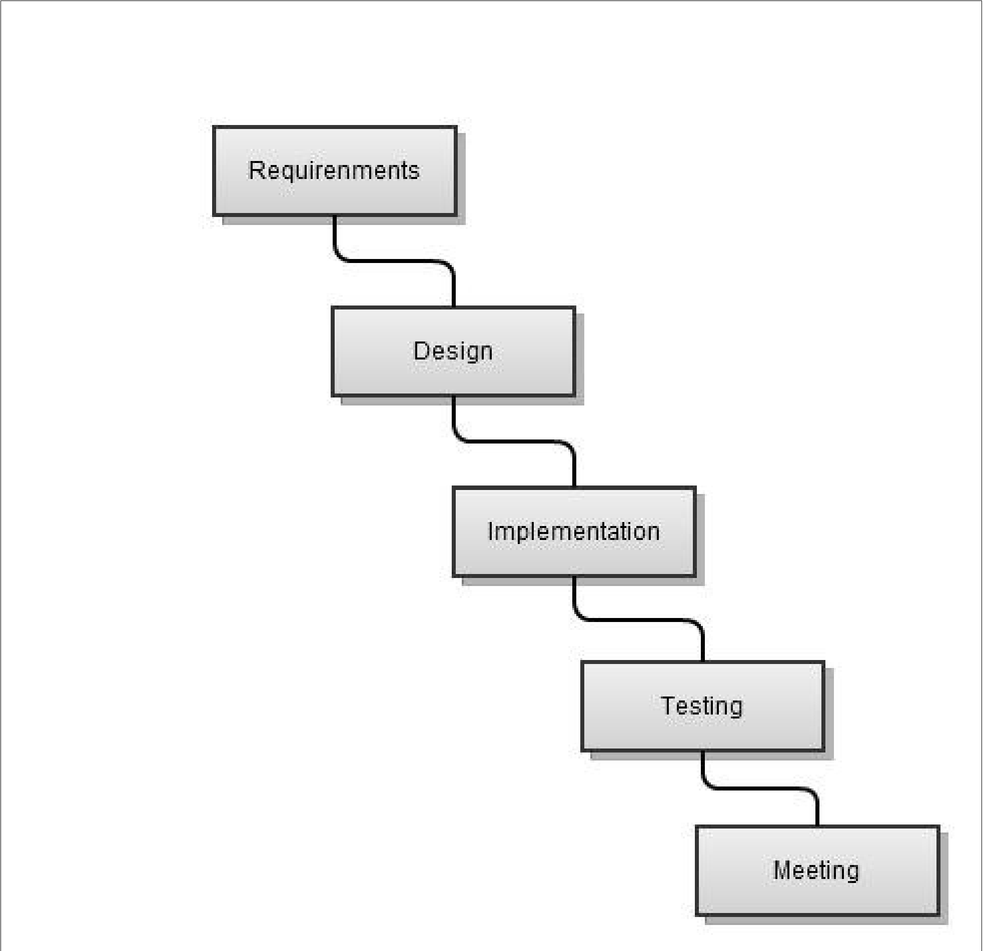


Figure 10: Development process

As you can see on the picture Figure 10: Development process we used basic waterfall model.

1. In the first phase we collected all requirements from the customer and the supervisor. We were aware of the importance of this step so we focused on understanding all needs from the customer properly. It was our first experience with collecting demands from a real customer so now we have a better picture about how it goes in reality. It's different from school tasks because they are usually more specifically described. The customer sometimes would not understand IT at all so we have to recognize what is important and find out how to realize the desired solution.
2. After we were sure we know everything what is needed for the application we proceeded with the design. We had to decide which technologies would be suitable for

this solution and think out some basic functionalities.

3. In the next step we proceeded with implementation. Implementation could be quite a time consuming part and it is almost impossible to create a flawless application without proper testing. The most obvious bugs were revealed and repaired already during the implementation phase. One of the most important phase that comes after implementation phase is testing. In the testing phase we should get rid of the remaining mistakes.
4. The last part of the process was always the meeting with the customer. We presented there our implemented solution and discussed with the customer if it meets all the requirements. This step will show if we understood all requirements in the first step correctly. Luckily the customer was always satisfied with our solutions so we presume that we collected all the requirements in the first step correctly.

6. References

1. What is Windows Communication Foundation?
http://msdn.microsoft.com/en-us/library/ms731082(v=vs.90).aspx
Accesed on 19 May 2012
2. Windows Communication Foundation, Wikipedia
http://en.wikipedia.org/wiki/Windows_Communication_Foundation
Accesed on 19 May 2012
3. Zaciname s WCF | Vyvojar.cz
http://www.vyvojar.cz/Articles/452-zaciname-s-wcf.aspx
Accesed on 19 May 2012
4. WCF – Zakladne pojmy | Vyvojar.cz
http://www.vyvojar.cz/Articles/454-wcf-zakladne-pojmy.aspx
Accesed on 19 May 2012
5. .NET, Wikipedia
http://cs.wikipedia.org/wiki/.NET
Accesed on 19 May 2012
6. .NET Framework, Wikipedia
http://en.wikipedia.org/wiki/.NET_Framework
Accesed on 19 May 2012
7. Code Access Security, Wikipedia
http://en.wikipedia.org/wiki/Code_Access_Security
Accesed on 19 May 2012
8. Common Language Runtime, Wikipedia
http://en.wikipedia.org/wiki/Common_Language_Runtime
Accesed on 19 May 2012
9. An Extensive Examination of LINQ
http://www.4guysfromrolla.com/articles/031109-1.aspx
Accesed on 19 May 2012
10. Home: Silverlight.NET
http://www.silverlight.net/
Accesed on 19 May 2012
11. Microsoft Silverlight, Wikipedia
http://en.wikipedia.org/wiki/Microsoft_Silverlight
Accesed on 19 May 2012
12. Uvod do LINQ | Vyvojar.cz
http://www.vyvojar.cz/Articles/563-uvod-do-linq.aspx
Accesed on 19 May 2012
13. Language Integrated Query, Wikipedia
http://en.wikipedia.org/wiki/Language_Integrated_Query
Accesed on 19 May 2012
14. LINQ to SQL
http://msdn.microsoft.com/en-us/library/bb386976.aspx
Accesed on 19 May 2012

15. LINQ to XML
http://msdn.microsoft.com/en-us/library/bb387098.aspx
Accesed on 19 May 2012
16. Extensible Markup Language, Wikipedia
http://cs.wikipedia.org/wiki/Extensible_Markup_Language
Accesed on 19 May 2012
17. XML, Wikipedia
http://en.wikipedia.org/wiki/Xml
Accesed on 19 May 2012
18. SQL, Wikipedia
http://en.wikipedia.org/wiki/Sql
Accesed on 19 May 2012
19. SQL, Wikipedia
http://cs.wikipedia.org/wiki/SQL
Accesed on 19 May 2012