CHALMERS

Visualization manager for production data

Master of Science Thesis in the Programme Software engineering and Technology

MARTIN IVARSSON

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Göteborg, Sweden, September 2009

Visualization manager for production data

MARTIN IVARSSON

Examiner: JAN SKANSHOLM

Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden September 2009

Abstract

This master thesis report describes the work with evaluating and designing how a visualization manager for production data should be built and also the work of implementing a prototype of the product fabricated for Binar Elektronik, Trollhättan.

The most important requirement when starting this project was to evaluate and research the product area and context where the visualization manager will work. To create a, in as high degree as possible, a web based prototype application with the technology within some framework was also stated as an important requirement.

The result is a visualization manager that can present production data or equivalent at LCD displays(regular large TV/computer displays) and LED displays(displays built up by LED diodes that can be switched on or off in a single color). Everything is administrated by a web portal, where displays, groups of displays and production areas can be administrated. All data that are presented is collected from a database where different kind of data sources(production units etc.) insert their data.

Two different subsystems have been created to serve the two displays types; LCD and LED. Code reuse and the same architectural approach have been used where possible in the development of the subsystems. Due to the fact that the LCD display shows, dynamically by the system generated, web pages and the LED displays show text strings pushed to the display and other hardware related aspects, it have been some major differences in implementation between the two systems.

Microsoft .NET was evaluated as the best development environment for this project and its concepts and tools have been used frequently during the implementation of the visualization manager.

Sammanfattning

Den här exjobbsrapporten beskriver arbetet med att evaluera och designa hur en visualiseringsmanager för produktionsdata bör byggas samt arbetet med implementeringen av en produktprototyp framtagen för Binar Elektronik, Trollhättan.

Det viktigaste kravet när arbetet med examensarbetet startade var att evaluera och undersöka produktområdet och sammanhanget som en visualiseringsmanager kommer att jobba i. Att skapa en, i så stor grad, webbaserad prototypapplikation med den senaste teknologin inom något ramverk var också viktiga krav.

Resultatet är en visualiseringsmanager som kan presentera produktionsdata eller liknande på LCD-displayer(stora TV/data-displayer) och LED-displayer(displayer uppbyggda av LED-dioder som kan sättas på eller av i en färg). Allting är administrerat av en webbportal där displayer, grupper av displayer samt produktionsområden kan administreras. All data som presenteras hämtas från en databas till vilken olika datakällor(produktionsenheter etc.) skickar in data.

Två olika delsystem har utvecklats för att kunna använda de två önskade displaytyper, LCD och LED. Återanvändning av kod och nyttjande av samma arkitektur har gjorts där det varit möjligt vid utvecklingen av de två delsystem. På grund av att LCD-displayer visar, av systemet dynamiskt genererade, webbsidor och att LED-displayer visar textsträngar som trycks till display samt andra hårdvaru relaterade aspekter finns det några betydande skillnader i implementationen av systemen.

Microsoft .NET ansågs som den bästa utvecklingsmiljön för det här projektet och dess koncept och verktyg har använts mycket under implementeringen av visualiseringsmanagern.

Preface

This is the master thesis for Martin Ivarsson, presented to Chalmers University of Technology as partial fulfillment of the requirements for obtaining the Master's Degree in Software Engineering. The thesis has been written under the supervision of Karl Rosenberg, Software engineer at Binar Elektronik in Trollhättan. Formal supervisor at Chalmers University of Technology has been Jan Skansholm.

All work has been in Binar Elektronik's facilities in Trollhättan. Writing my Master Thesis in collaboration with Binar Elektronik has been a joyful experience. I have had access to an extraordinary software environment and hardware resources to test my application.

The work experience has also given me insight into the work flow at a modern innovation driven company. I would like to thank my technical supervisor at Binar Elektronik, Karl Rosenberg, for all work he has put into mentoring and supporting me during the process. My sister, Ann Stridh, is also worth a special advertence for her proofreading of the report. Last but not least, I would like to thank Jan Skansholm for giving tips and for always being available for questions and inquiries.

Table of contents

Introduction

Background

In today's industry world the need for being efficient is vital for a company, who strives to be successful and competitive. This means that they always have to optimize all parts of the production, from construction board to the support and reclamation services. To be able to optimize an activity to its best, is probably to measure and from the measured values draw conclusions. This is one of the key values of the Lean production. To make the efficiency visible for everyone in a production setting is called visualization.

That is one of the keys for LEAN production. If a display shows that a machine and its operators produce far too little compared to desired values, a production manager, or similar, can react immediately and try to solve the underlying problems.

Binar Elektronik, the company that setup this master thesis project, has a system for handling LEAN production in a variety of industries, although the system is a bit old and needs to be renewed. Through some discussions within the steering board of the company, a vision of a new modularized system has come up. To break up today's system in modules and even add new modules to it would provide a greater flexibility and possibility to meet new customers and gain a greater market share.

Through their current system has Binar Elektronik support for so called LED displays when visualizing production data. The LED displays are although challenged by the newer LCD display technology, basically TV displays used for displaying production data. The next version of the visualization system should support LCD displays and have support to easily add other display solutions.

Another motivation for a system is that more and more customers want a web based system. The current system is a desktop application[1]. To access and configure a system through a web page means a great flexibility when it comes to choice of operating system, computer etc. A web based system also opens also up for the usage of mobile phones and other handheld tools to check current production status in general, or for a special production unit when you are far away from the factory.

Purpose

The purpose can be split into two different parts; project purpose and report purpose.

Project purpose

The purpose for the project was to develop an application that will handle visualization in different factories and production settings. The application should handle both LCD-displays and LED-displays. Since a factory can be structured in many different ways with production groups and areas, it was recommended to develop a web portal as a platform from where you can handle all displays and the hierarchies of groups and areas. In addition, the application should be able to be further improved in the future, especially with possibility to configure expression logic for what type of production data that

should be presented depending on a condition, whether it is true or false. The company, where the product was developed, wanted at least a well developed prototype for visualization, ready for demonstration to their potential customers. The company did clearly precise that it is more important for them with a well done research and a requirements specification built on that, instead of an application that is built on loosely investigated facts.

My own purpose with the project was to work at a company and follow their wishes and requirements on a product. When developing a program in school nobody will use it and the task is often too well structured. This is probably not always the case in the industry. Furthermore, it was interesting to develop a larger project and structure it in a good way. In addition, to the goals mentioned above it was interesting to develop a project at the .NET platform with the programming language C# and use Microsoft's different products and concepts for software development.

Report purpose

The purpose for this report is to describe how the project has evolved both with respect to project management and software development. Problems and delimitations of the project will of course also be covered.

Project description

The master thesis project consists of the following moments:

- Preparations.
- Prestudy and specification of the product
- Learn necessary techniques for developing the product; C#, MS SQL server, LINQ, LEAN, ADO.NET framework
- Implementation
- Document the work in the thesis report
- Present the thesis project

Limitations

I have recognized that Binar Elektronik have a lot of good ideas for the Visualization tool and more good ideas have come up during the development of this product. It has therefore been important to delimit the work with respect to time. The thesis work should be about twenty weeks of full time work, roughly fifteen of these I have aimed for development of the application and the rest, five, is reserved for writing and presentation of the report.

Due to the time frame, the following parts were prioritized in the following order:

- Research and collection of relevant data for specification and requirements of the application
- Development of administration- and configuration portal
- Functionality for LCD-viewer
- Functionality for LED-viewer

The major unsolved part and the next part to solve, is development of expression logic. Roughly this means that if some conditions hold, than a special page should be viewed, and if some other conditions are true then some other pages should be viewed. In current mode all conditions are true.

The structure of the report

The report starts with this introduction and continues with a prestudy section describing the purchasing company; Binar Elektronik, LEAN production and the industry's demands. It continues with analysis of the thesis problems before a theoretical background of programming framework etc. is given. After that part is the chapter about design and implementation of the visualization manager located. The report is ended by a discussion and conclusion in the result chapter.

Prestudy

Binar elektronik

Binar elektronik is a company founded in Trollhättan in 1984. The company has about 30 employees and their main business´ are industry automation and industrial IT. Initially the company was named Binär Elektronik, which then became Binar, which furthermore in 2007 became a mother company for several other industrial technology companies. At the same time did the operational part of Binar become Binar elektronik and Binar remained as the pure mother company. It has no industrial production – only an administrative and managing role.

Binar elektronik has three different branches of business:

- System development – electronics development and high level programming
- System integration – Industrial IT, process steering, movement steering and automation
- Lean production – Knowledge and products for resource economic and quality secure production

The customers of the company belong to a wide range of companies within many different industry segments. Most customers are located in the vehicle industry, and the majority of these customers are to be found within the truck industry.

The interest for the Lean concept at Binar elektronik has grown bigger ever since the birth of the company. The very first Lean connected product was the BiDisp; a display composed by LED-diodes, usually found in industrial environments to show the clock time, *number of* produced units etc. Since then the company has produced takt systems among other things.

In 1992 the company released a product named Mandon. This was developed from the Andon(signal an error) concept but for material supply, therefore the name – Material Andon(Mandon). The Mandon product was developed to signal that there is a demand for more material at one station. This is Kanban according to the Lean concept. To develop Lean products with lack of theory was symptomatic for Binar elektronik in 1990's and the early 2000's. Since then the company has gradually learned more of the theory of Lean and know how to promote the Lean concepts to potential customers [2].

Lean production

The early steps toward lean production

The start of a serious hunt for productivity in industrial environments started with Frederick Winslow Taylor. He was a mechanical engineer in the early 20th century who sought to improve the efficiency at different factories during his career. One of many famous articles he wrote was – "*Principles of Scientific Management*". Through the article he tried to argue that it is better to work systematic and with

scientific methods rather than to depend on rule of thumb when solving a task. Furthermore, he argued for the importance of organized training of workforce, back in the days at that time it was still up to the worker himself to improve his skills. He was especially interested in studying a worker in detail and measure the time of different tasks, i.e. time study[3].

The company Ford took the next step in the way of producing more efficient. Henry Ford introduced the moving production line, i.e. the assembly line, instead of craft production. The innovation that made Henry Ford most famous was probably the “fordism”-concept. The core of all systems was to standardize, which meant for example standardized components, standardized manufacturing process etc. These innovations made the Ford car possible to buy even for the middle class, which generated in an early “mass consumption” of cars [4].

The innovation of Lean production

The family of Toyoda was from 1937 trying to build automotive vehicles with the brand name Toyota. In 1950 they had built 1685 cars in total, during these years. The production was done more or less by hand craft, in the mean time did Ford, in their main factory, produce 7000 cars in a single day, of course due to the assembly line[5].

Eiji Toyoda, son to the founder of the Toyota Motor Company, visited Ford’s headquarters in Detroit in 1950 to see what Ford did to produce that huge numbers of cars compared to Toyota. He saw the moving production lines and understood that it was not possible to plagiarize the concept to the Japanese culture but he took some ideas with him back to Japan. Back in Japan he started to produce the Toyota Production System, in general known as Lean production, together with a veteran in the Toyota factory, Taiichi Ohno[5].

Lean Concepts

Toyoda and Ohno developed numerous of famous concepts named after Japanese expressions. Below are some general ones and some specifically to the visualization context connected.

Kaizen

Kaizen means continuous improvement throughout the overall process and in every part in every individual process. This means that all employees of a unit production, from Chief Executive Officer to the assembly line workers, should strive to always improve the production.

Just-In-Time production

A factory should never keep more material than needed. To store material prohibits the production in the factory from being optimized since it is always a buffer, i.e. you hide the real problems within the production by using the stock and you are therefore not forced to optimize. Furthermore, it is not only expensive to buy material that is not used immediately(capital binding), but also space consuming.

Andon

The meaning of Andon is to signal that there are problems with either the quality of a product or the process of the procurement of a product.

Kanban

Is the concept for that a “special activity” is requested. For example you may signal a special Kanban signal when you are running out of material. The Kanban action authorizes a material refillment, since you are not allowed to store more in production than necessary. The last statement refers to the JIT-production.

Takt time

The takt time is the time it should take to assembly/fix a unit at a specific assembly station. For example if you have 480 minutes per day to produce and you should produce 60 units, then the takt time is 8 minutes/unit(480/60). Every month it is a review of the takt time for every unique process with possibility to tweak the takt time ten days later(after testing the new takt). The takt time is a parameter that usually is visualized by a display with a timer counting down for every unit.

Industry needs

A manufacturing industry wherever it is located is or will at least eventually be exposed to competition. With the ongoing globalization is manufacturing industry in the industrialized world since long time extremely vulnerable for low cost countries improvements of their manufacturing industry. For a Chief Financial Officer at a regular manufacturing industry it is perfectly clear that at least the only thing that matters in the long run is how well the produced products are received by the market, i.e. how well the products sell.

To be able to produce efficient it may be good to follow some kind of standardized behaviour. The Lean concept can be such standard to produce efficient. For a blue collar worker it is very important to know how much is produced and how much is expected to be produced. During the first day of this master thesis project I was guided through one of the manufacturing lines at SAAB Automobile AB, Trollhättan. The many moments that were synchronized in the assembly process of a car do require some kind of pace system to inform the workers how much they are supposed to produce/assemble.

The following pictures illustrate how important it is to visualize the production, i.e. to have visual control:

Figure 1: A comparison between not visualized work unit and visualized ditto.

The picture to the left shows two guys, they seem to work actively and seem to know what they are doing. The picture ahead of the right tells us something else. They have produced three units and they should have produced ten units. For a production manager or similar this is no good news. For a situation like the left without visual control it is probably a big risk that you miss where in the production you have problems. In the picture to the right it is visualized in real time if the factory workers get in trouble.

Production managers of a manufacturing line with some kind of visual control can in case of problems at one or several stations take measures immediately. The risk that problems in one manufacturing station propagates to the rest of the production is much smaller than if no action is taken at all because then it is hard to track from where the problem arose.

The paragraph just ahead this shows the essence of this master thesis. If you do not have visual control you do not know how efficient your business is. There may be a case where one area in the production line inhibits the rest of the production and therefore the business is put into profitability issues.

Requirements

The requirements for the project are formulated with the purpose as framework. The following requirements have been established for the project.

The most important requirement is to research the product area and investigate the needs for a visualization manager. This work is highly prioritized because the company wants a product that is well investigated with less functionality than a product with a lot of unnecessary features that are not needed or implemented in the wrong way(less researched and investigated area).

The second requirement is to implement a visualization manager with as many features as the investigation of needs gave. The product should be seen as a prototype rather than a commercial product, although it should be possible to use the prototype's source code and architecture when developing the commercial version of the product.

Finally the prototype should be developed with the latest technology within chosen development framework. One of the underlying motivations to this demand is to learn something that is attractive for the labor market.

Problem analysis

Since the project in the very end aims to develop an application, rather than investigate and draw conclusions, the problem analysis part is not as thorough as it may have been with a more theoretical and research oriented field of study. However, there are some topics that need more careful analysis. These are the following:

Configurability

One of the main goals with the application is that it should be really versatile and workable in virtually all kind of production settings. Binar Elektronik has several times developed a product with an implied intention how it should be used by the customer. When the installation is finished and the customer starts to use the product it is used in another way than the original intention. This means that configurability needs to be generous to cover as many cases as possible. To be able to cover all kind of factory environments the software has to be flexible when it comes to the following topics:

Displays

A display can be of different types. In visualization industry it is two main categories of displays; LED and LCD. In the future there may be additional display categories and flexibility to add more display types should be taken in consideration. The need for LED displays, which only show text in one single color on a black background, may also be phased out as the visibility of the LCD displays constantly are improved. The long sight visibility and the durability are the only reasons that the LED display category survives the competition versus LCD.

As much programming code as possible for presenting production on the two, current, display options should be common. Because the information that will be displayed will roughly be the same irrespective of display category, this will encourage the logic for the different displays to be as similar as possible. Some kind of template system will probably be useful to build pages, it is called page although viewed on a LED-display, that are as customized as much as possible but still provided in a systematic manner.

Templates

To make it easy for a user to develop and create customized pages it is necessary to have some templates, otherwise the application will lose a lot of its simplicity. A user will then have a few templates to choose from. Each template will have template sections as building blocks. This corresponds to the concept of page sections, which is the corresponding building block for the page. Reasonably the template section will cover the positioning of the section and the page section will provide the data that should be presented and how it is presented, i.e. layout.

Graphics

When using a template to be able to design a page or even more relevant; a page section, the user should be able to configure the appearance of the section it wants with respect to the following settings, at least:

- Type of font
- Font size
- Font color
- Background color
- Border color of the section
- Horizontal alignment of the text in the section

With parameters for deciding the design it is possible to create very versatile pages that will cover most customers´ preferences.

Development environment

Included in the task was to choose a development environment that is "future friendly" and that the company had some experience of. Because the company only has experience with two programming language platforms with sufficient web programming support, namely Delphi and .NET, I had to choose between these. To notice, before reading the motivation for the choice of development environment, is that I am most experienced with Java. The decision was done with the following motivation:

Delphi

Delphi is the primary programming language at Binar Elektronik. It is not fairly common in the programming world but it has web programming support, which has been used in a few projects within the company. The language has its roots in Pascal and it is said to be relatively easy to learn as a Java programmer.

.NET

The .NET-platform is invented by Microsoft and has a very extensive web programming support. When programming with C#, which is the biggest language in the .NET-family, it is fairly close to programming in Java. .NET is extremely common in the industry and the support at internet and through books is very extensive. In addition, a newly graduated with .NET experience is by far more interesting for employees than an equivalent Delphi programmer. The supervisor at Binar Elektronik had good experience in web programming and .NET as well.

With the discussion above it was fairly easy to choose .NET and C# as the programming language, even if the company mostly used Delphi in their other projects. The main reason was that .NET is widely used and I reckon that it is a good experience to have been programming in .NET environment when applying for software development jobs.

Restrict dynamics of web pages

Most of Binar Elektroniks customers would like to restrict the use of active components in browsed web pages. Since part of the application is a web application, it has therefore been important to avoid AJAX and Javascript as far as possible. This have led to that functionality have been changed slightly compared to if there were no restrictions at all.

Control of page generation

Since Binar Elektronik is a commercial business, one of its main goals is to earn money on their products. In the case of the Visualization application it is therefore important to have control of the number of pages a customer generates. There are two reasons for this:

Reason number one is that you pay for the number of pages you want to display in the factory. This is of course with the intention to earn money to the company. This can be done by a special billing system, e.g. if you want to show 46 pages you may buy a license for 50 pages. When you have reached 50 pages the software restricts you from creating more pages and instead encourages you to contact Binar Elektronik for an upgrade of the license.

The second reason for controlling the number of pages is that a large amount of pages presented may affect the performance of the infrastructure in the factory, i.e. network and servers. If the performance is low it will of course affect the whole impression of the application due to misusage of the customer.

Method

Project methodology

The thesis work has been split into different phases. The reason for this is simply to be able to track the work progress and to be able to solve problems individually. With programming vocabulary the bottom up approach has been used.

The different phases during the work have been:

Spike-phase

The Spike phase was established to get to know the tools and see how they can collaborate, i.e. MS SQL server management studio and Visual Studio 2008. Furthermore, it was important to penetrate the customer's need and to learn about the task. As fairly common when elaborating with requirements a customer will learn more about one's needs and therefore reformulate the task partly, luckily the change frequency tends to slow down the longer you get into the project.

The software DIA modeling tool and the whiteboard beside my office space was used a lot in the "Spike"-phase to be able to understand the customers wishes and articulate them as clear pictures and not only as, sometimes, very vague formulations. To create a picture of an interface made it a lot easier to be able to understand what the company needed, and how they wanted it to be like. When it came to the communication between different application layers and the isolated communication between each other the modeling tools were as least as important as before to be able to understand each other's thoughts around a, partly, complex communication scheme.

The ideas that came up during these brain storming sessions were then evaluated a bit by me - "Is it possible and time efficient to implement a solution like this?" I was sometimes uncertain regarding the solutions I have come up with and called for a new briefing with the others just to be sure that they agreed with the solutions.

The output of the Spike-phase was a Requirements specification, which was sent out to key persons at the company for assessment and feedback. The requirements specification established a great knowledge platform about the product for both me as a developer and also for Binar elektronik as customer.

Develop web interface-phase

When the initial Spike-phase was finished the project went on with the web interface development. The decision to develop the web interface before other parts of the project was basically because without an interface it is hard to test the logic for people not very involved in how the application behaves. With the web application it is easier to describe a function and then demonstrate the functionality this function provides.

Mid-development-meeting

After about eight weeks of development a discussion meeting about the developed application so far was planned. The chief executive officer, marketing director, the group manager, and my supervisor at the company were all invited.

The agenda for the meeting basically consisted of three parts:

- Demo of the product so far
- Discussion concerning the upcoming development, what should be prioritized?
- Other known but not solved issues

The meeting decided that the next phase in the project should be the development of the LCD-display viewer.

Develop LCD-viewer phase I

The first viewer connected to the web interface was decided to be the LCD. The main reason for that decision was based on that it should be the easiest part to continue with. Another reason for building the LCD-viewer in this stage is that the web interface and the LCD-viewer basically are the same things, i.e. the LCD-viewer is only some extra functionally built on top of the web interface.

Develop LED-viewer phase

It felt natural to continue the development with the LED-viewer. The viewer is less connected to the web interface than the LCD-dito but a lot of the code and thinking were reused. It was put effort in making the architecture for both of the viewers to be as similar as possible and because of that, be able to not have redundant code parts.

Writing session I

A suggestion from the examiner of the thesis was to mix the implementation of the application with some thesis writing. With respect to that a writing session was planned when the main skeleton of the application was finished. The goal for this phase was no to create a fully fledged thesis report, but to think of the report disposition and look for interesting sources to the theory parts of the report. Another reason for the phase in between the two implementation parts was also to spread the, sometimes, demanding report writing over a longer time period.

Develop LCD-viewer phase II

During the writing session, the application skeleton was reviewed by the marketing director and the group manager. Since the first version of the LCD-viewer was built upon a HTML-table structure, which meant little flexibility for the end user. The structure became basically too static. To be able to meet as many customer wishes as possible they suggested a reorganization of the LCD-viewer structure with respect to underlying design, i.e. not use HTML-tables as base. This was accepted by me and my supervisor at the company since the project seemed to have used less time than expected so far.

Testing and debugging phase

As with all kind of systems you need to test it and solve errors that come up during the test sessions. Some time with code refactoring was also spent during this phase. The code redundancy was optimized.

Writing session II

The second writing session has one distinct mission – to finish the already started thesis report writing. Included in this phase is also preparation of the master thesis presentation both at Chalmers and at the company. Some minor debugging and improvements of the application may be done as well.

Theoretical framework

The .NET framework 3.5

The .NET framework is built up by several programming languages and even more different technologies and features. The reason for providing that many programming languages as C#, VB, Perl .NET among other things is that different programmers have different preferences and different tasks to solve. Not surprising is the different code from the different languages compiled down to a common code base, i.e. a common intermediate language(CIL) and type metadata. The CIL instructions are similar to Java byte code. One advantage with this common intermediate language solution is that programmers can code in different languages and then integrate their code in a project[6].

The CIL-"code" itself is compiled by something called a Just-In-Time compiler(commonly abbreviated JIT-compiler). Visual Studio has different JIT-compilers depending on the environment(with respect to machine memory, CPU-speed etc.) the code will be executed in.

Common Type System(CTS) is the .NET environment way to organize the different building blocks(classes, interfaces, structs etc.). For example the class building block has rules for inheritance, rules for methods that must be implemented, visibility of the whole class etc. The CTS language works independently, i.e. every class of a .NET-language has the same set of rules to stay "within". Each building block has its own CTS-characteristics.

When a .NET-program is executed it is done with the Common Language Runtime(CLR) which is the common execution system for all different .NET-languages. The CLR is like the Java runtime system – Java Virtual Machine(JVM) – with the exception that it is shared by many languages, not just a single one as in the Java case.

One unambiguous advantage with for example Java is the fact that it is platform independent and one clear disadvantage with the .NET-languages is that it has to be executed in Windows operative system. The last statement about .NET's platform independency is not completely true, since when C# was invented its cornerstones like syntax, semantics and the CIL among other things, were all published and ratified by the standardization organ ECMA. This opened up for projects like Mono that is fundamentally platform independent and roughly implements the standard set of libraries in the .NET environment.

ASP.NET

ASP.NET is an improvement of ASP to be able to separate HTML from more sophisticated programming language. The 1.0 version of ASP.NET was released in 2002 and the current version is 3.5. The by far most common way of creating ASP.NET home pages is to create an aspx-file for each page.

The aspx-file contains the so called server controls(button, combo boxes etc.),connected to the aspx-file. The aspx-file is normally a file with programming logic connected to it. In this project a C#-file will be, in general, connected to each aspx-file. When an aspx-file is accessed by a web browser the ASP.NET runtime system is executed. The, at least, two(aspx- and c#-file) files connected to the page is compiled

separately by the C# compiler. First the C#-file(also known as code behind) is compiled, when that is done the C# compiler will continue with the aspx-file that the ASP.NET runtime system split into two parts. The part that is compiled is inheriting the already compiled C#-code, the code mentioned as code behind.

If there is code connected in the requested file, this code will now be compiled. Finally the web application is up and running. The compilation of the aspx and c#-file is only done if the page request is done at first; otherwise the page is rending from the cach where the already compiled page's assemblies are stored. Separate files that are connected are always compiled with every request[7].

Visual Studio 2008

Visual Studio is .NET's answer to the open source world's flagship Eclipse. Both editors are sophisticated in the sense of features with massive support for IntelliSense(auto completion), integrated compilation, database integration and debugging. Visual Studio follows the .NET-environment's development with respect to newly invented features etc.

Below are some of the features that I used during the software development:

Lambda expressions

A Lambda expression is an expression that works on a collection and selects some items according to what is specified in the expression. One way to translate a Lambda expression into natural language is: "For all x of type Integer where x is equal to z".

```
List<Integer> l = {1,2,3,4,5};

Int i = l.Where(x => x.equals(3));
```

The Where()-method on the list is an example of LINQ, more about that below.

Debugging

The debugger in Visual Studio is probably one of its advantages compared to Eclipse's debugger. One of its most useful features is the" inspect variable value" in runtime, you simply put the mouse pointer over an executed program line, after which you are able to inspect what values the variables in that line will adopt.

Implicitly typed variables

With the key word *var* you are able to assign any kind of value to for example a temp-variable, provided that the temp-variable is typed with *var*.

```
var i = 5;
```

The compilator translates the variable into correct type and the IntelliSense functionality can be used.

ADO .NET Entity Framework(EF)

ADO.NET Entity framework is an improvement of Linq to SQL which was Microsofts first ORM-tool. An ORM-tool is something that makes the interface between the database and the programming code easier to handle. ORM is an abbreviation for Object Relational Mapper. It provides a very handy way to communicate with the database instead of writing sometimes very complex SQL-queries. The combination of Entity framework and pure LINQ has facilitated the work a lot since virtually no SQL-queries have been written at all during the implementation of this software.

The choice between ADO.NET Entity Framework(LINQ to Entity) and LINQ to SQL was fairly easy since one of the goals with the application was to implement it with the latest technology and the following statements made the choice even easier:

- LINQ to SQL is only for SQL – EF works with XML, SQL etc
- Entity framework is much more loosely coupled between the layers and therefore more flexible
- Entity framework should work with other databases than those provided by Microsoft. Linq to SQL does only support Microsoft databases.

Another option would have been to use Hibernate's .NET-version – Nhibernate. NHibernate is a, completely free, sibling to the famous open source Hibernate ORM-tool for the Java environment. The decision to choose Entity framework instead of NHibernate was based on recommendations from more experienced .NET-programmers and because of the great integration in Visual Studio.

In Visual Studio you get a model of the entities from the database and there different couplings to each other. You are able to add attributes and change attributes of the domain, called Entity Domain Model (EDM), in Visual Studio however the changes will not affect the database. Basically it is therefore only meaningful to change attribute names. If you would like to add an attribute or delete an entity you have to do this directly in the database and then update the EDM. Fortunately the next generation of Visual Studio will support generation and altering of database entities/attributes directly from the EDM in Visual Studio. This will make it possible to design the database model straight from Visual Studio and from the model generate the database. When that is done you just go on with the regular process(i.e. coding) and alter the database from Visual Studio when necessary.

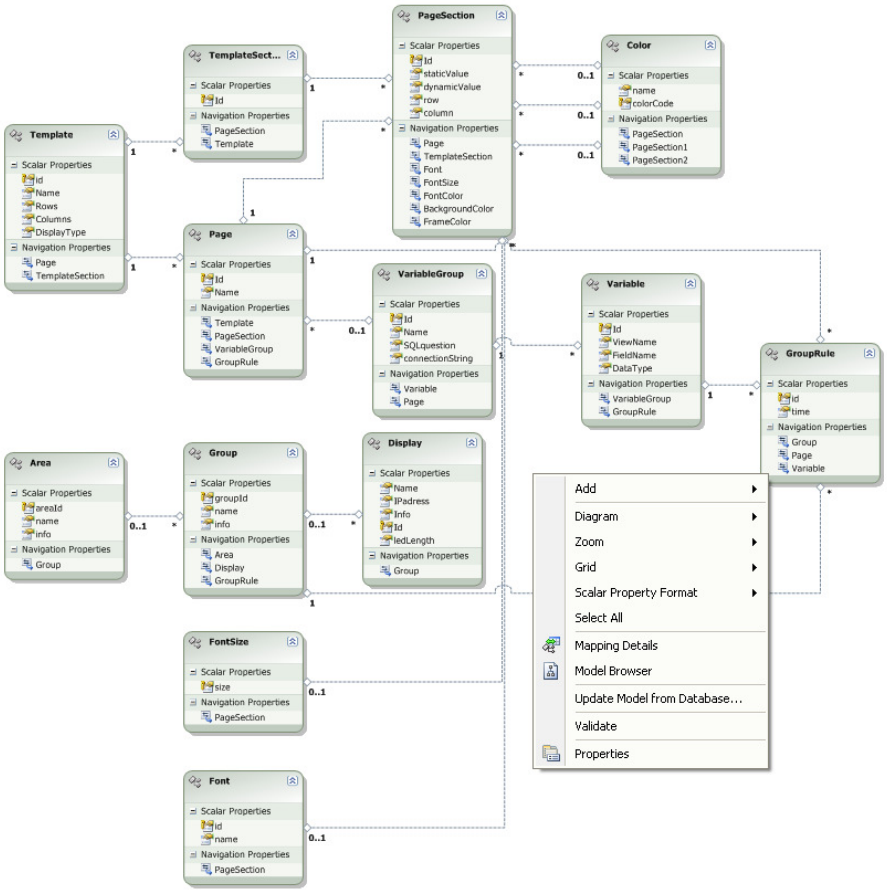

Figure 2: An overview of the entity data model in Visual studio. Basically all entities have attributes and foreign relations. Each relation is marked with a dotted line.

Entity framework in its current version only works with so called lazy loading. In this context it means that you explicitly have to “load” relationships to other entities. If the loading not is done explicitly the related entity will remain null and a NullPointerException be thrown when a related entity is accessed. In the next version of Entity framework it will be possible to choose if you would like greedy loading (“everything” is loaded) or the default lazy loading. To activate greedy loading you just change a Boolean value. The reason to that is that you have to manually activate greedy loading and that is of course a matter of memory resources. If all related entities are loaded into the memory it will allocate a great deal of memory for no reason. In some cases this will be handy but for sure not in all cases.

LINQ

LINQ is a newcomer in the .NET-environment since .NET 3.0 and stands for Language Integrated Query and is very handy tool for picking what you want from fundamentally all data structures, e.g. all collections in .NET, databases, XML-files etc. The only requirement for these datastructures is that they implement IEnumerable/IEnumerator[8]. LINQ is not a new standalone .NET-language but it is integrated in C# and VB. The structure of writing questions to a data structure is fairly similar to SQL, the language is definitely inspired by the SQL-syntax. LINQ facilitates for the programmer since it makes it possible to avoid integration with other query languages such as XML. This makes the code cleaner and easier to build robust software around without demanding explicit integration with external data sources.

The picture[9] below shows how LINQ works as a layer between the programming languages and the data sources:

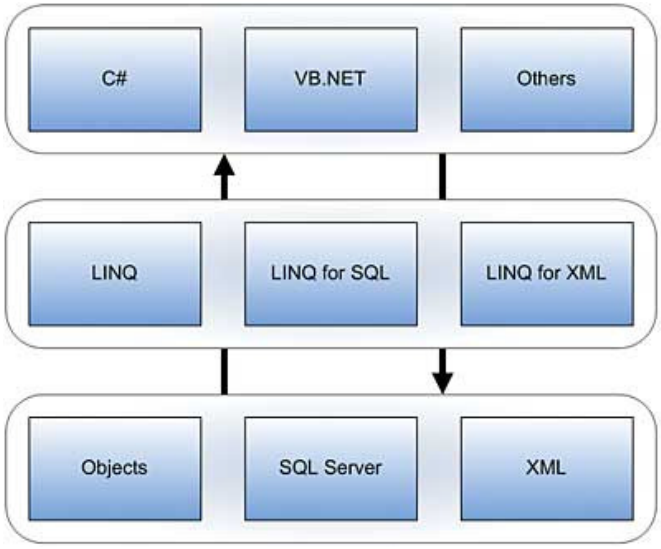

Figure 3: The layers that LINQ interact with.

Design and implementation

This chapter aims to describe how the application is designed and implemented. Despite that, the chapter will start with a summary on how the production data is collected from the production units in the industry, and then transformed and stored in the database.

Production data gathering

Since all production data that should be visualized is collected from one single database named central database, it is a great source to gather the data and insert it into this database. The data flows into the central database in three different ways(see the number in the illustration):

1. From an external application, named APP(1) in the picture below, in the factory's network, the application communicates with one or several PLC; interface to a machine's I/O-controllers, through an OPC; an interface between a PC and a PLC. When the application sends a signal to the PLC, it is also being transferred and collected by the central database.

2. The second way is the LPS, Lean Production System – Binar Elektronik's own production system, which has the same basic communication way as the external application. But LPS has a lot more features, among other things a great support for generation of visualization pages.

3. The data can also be collected from another database. This data must be transferred through some logic(probably a PC) to fit into the central database interface.

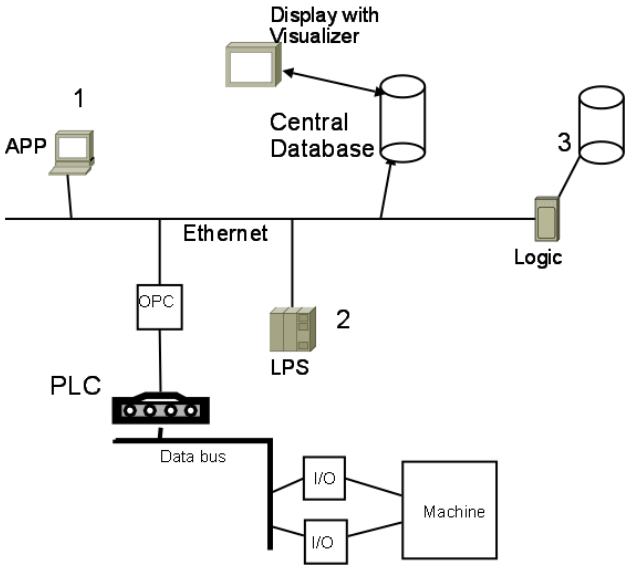

Figure 4: The picture illustrates how the production data is gathered and inserted into the central database. There are three different ways from an external application(1), LPS(2) and from an external database(3).

The reason why Binar Elektronik does not want to use the existing LPS for visualization tasks in the future, is that it is not general enough, at the moment it does not have support for collecting production data from an external application(1) nor an external database(3). It is therefore desirable to separate the visualization part as an external module.

Database

The company uses Microsoft SQL server a lot both in their Delphi applications, but also in other applications. It was therefore straight forward to use their existing database server with Microsoft SQL server 2005 as the central database. The fact that ADO.NET Entity Framework works absolutely best, so far, with Microsoft's products in general and especially Microsoft SQL server, made the choice even easier.

Database structure

To be able to handle all different kind of data stored in the database it was important to find an optimized general database solution. There is one table that is essential for creating a general solution. It has the following structure:

VariableGroup

Id	Name	SQL-question	Connection String
1	productionTime	Select top 1 noOfproducedUnits, numberOfDirectOk from statBase orderBy setValue	*ConnectionString to the database with the table statBase*

"Statbase"

Id	noOfProducedUnits	StopTime	numberOfDirectOk	setValue
3	40	33	37	37
4	38	42	33	45

When choosing a variable group you will get, as the name suggests, a group of variables. In the example above the group "productionTime" will return the values of noOfProducedUnits and numberOfDirectOk for the row with the lowest setValue.

The concept of storing a SQL-question and a connection string for each variable group makes the solution very flexible; you are not even tied to one database of raw data. You can collect data from as many data bases as you have variable groups.

When you create a page you simply choose which variable group you would like to use and then you are able to use the variables connected to this group when configuring the page. Instead of picking just one variable from a very long dropdown list, the variable group concept work as a filter and the number of variables connected to a variable group is easier to get an overview of. Furthermore, the data is collected in chunks, which make the frequency of the database requests optimized in comparison to as if it would have been with separate database requests for each variable.

General application design

The application is divided in a number of parts which are more or less integrated, they are the following:

- User interface
- LED-viewer, server
- LCD-viewer, web client
- LCD-viewer, server

One of the main goals with the application is to present the same production data on both LCD displays, with possibility for graphics etc., and LED displays. The former display type uses regular web pages to present the production data, and the latter display type uses an interface, where text strings are sent to the display's IP-address. The application has a hierarchy of different elements, namely displays, groups and areas. Each one of the element needs a more accurate description:

Display

A display is an element in the application that displays the product data or what else(can be important messages etc.) the application administrator configures a page to show. The LCD displays just show a web page as any arbitrary computer display. The LED display type is a bit different, because it has some restrictions, that is to say; very limited graphics and space.

A LED display is built on a lot of LED diodes, i.e. small lamps, which is either switched on or off. Normally the diodes form a pattern and show either a letter or a number. Usually all diodes are on a LED display of the color, commonly red or yellow/orange. For natural reasons this do restrict the graphics of page made for LED page to very elementary usage.

The other disadvantage with the LED concept is the limited space. A normal display cannot show more than twenty signs, in very extreme cases close to thirty signs. This means that you need to be very careful when designing a page for a LED display. Usually a "multi-row" display is just a broken LED display, illustrated by the following:

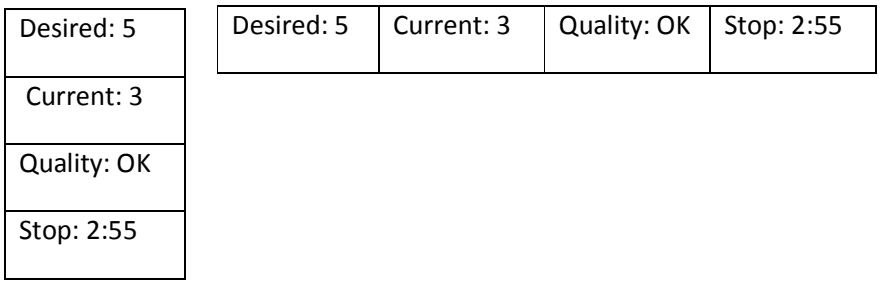

Figure 5: The display to the left is the same as the right display with the only difference that the left one is broken horizontally.

There are two main reasons for still using LED displays after the invention of LCD displays; namely visibility and durability. The visibility for a LED display is about 80 meters when it is roughly the half for the LCD ditto. When it comes to durability you can run a LED display thirty years constantly whereas the LCD only works for about five years of constant use [2].

Group

A group constitutes of one or more displays. A display can belong to only one group. A group can have different rules for how it should work in different situations. For example if there is a gas leakage at a machine with a display connected to the group then a warning sign will be shown on all displays within the group.

Area

An area is just a hierarchy definition to be able to keep track of groups; it can be many groups in a production setting that makes it too complex to overview. You are only able to connect a group to one area.

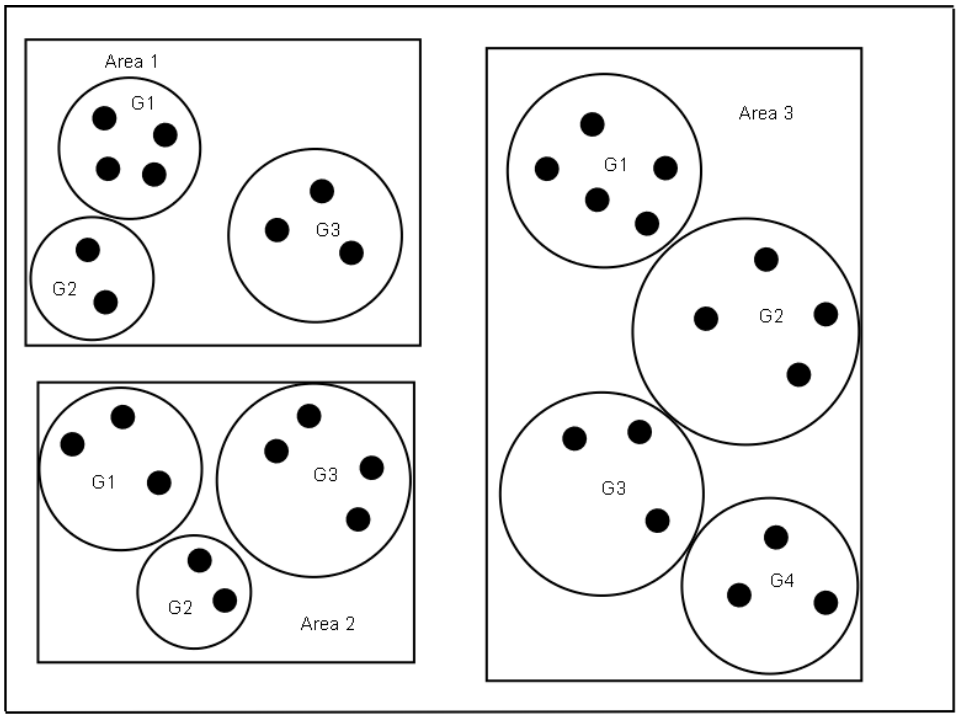

Figure 6: The illustration tries to visualize an arbitrary production setting with respect to the hierarchy in the visualization application; the black dots are the displays, the circles illustrate the groups and the rectangles show the areas.

Template

As mentioned earlier in the report all data are presented as pages, irrespective to whether if it is a LED or a LCD display. The pages are based on templates. These provide the possibility to give the customers flexibility but also reusability. A wide variety of pages can be created from one single template; it is just a matter of preferences by the user. Of course a page can originate from only one template. In the LCD display case all pages are pure HTML pages, possible to show on any PC display / TV. The HTML pages are generated by an ASP.NET class – ViewPage.aspx. In the LED display case a page is only a text string.

In addition, a template is divided into sections called template sections. The template sections correspond to the parts that the pages are built upon; page sections. A template section, in the LCD display case, has information about where on the page each page section should be located, done by pixel information, and if it has a corresponding page section or not. Some template sections do only act as borders when the real pages are generated and each template section has therefore an isFrame field in the database.

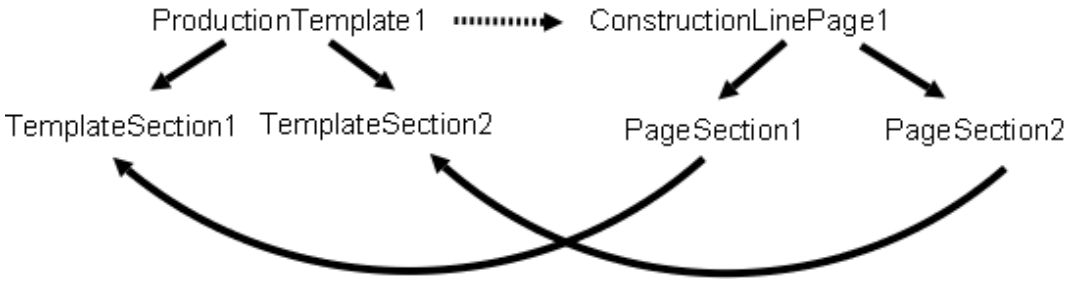

Figure 7: A picture that shows that a page(ConstructionLinePage1) has a template(ProductionTemplate1). The page has two page sections which correspond to two different template sections.

Web client

Even if the web client, for requesting pages to be displayed at LCD displays, and server are built in the same project, they could with equally the same work effort have been developed totally split. The client requests new web pages in time intervals defined by so called group rules. The request is done by the refresh mechanism in the web browser, which is manipulated; the refresh time of the page is set to the rule's viewing time.

The web client is very "thin" with because of the program logic. This derives from the fact that most customers do not want advanced web browser applications to be installed on their computers, e.g. AJAX-applications, JAVA-applets etc. It is of course a security issue.

LCD

The pages displayed on LCD displays are basic HTML pages. A Windows CE module will probably be connected to the LCD display. This module will than serve as a computer, with an installed web browser, that can request and handle web pages. The computer can be controlled by a remote through a network connection.

LCD server

The implementation of the server part for the LCD displays was done in two phases. From the beginning it was planned to be done in only one phase. The first version was yet too inflexible, and had to be redesigned. The LCD server has one mission; it is to generate a page to the requesting client.

LCD server phase I

The pages that the LCD server initially generated were designed on an HTML table. The number of rows and columns in this table were defined in the page's corresponding template. This meant that the pages were created by a nested loop that went through all rows and cols and put information in each cell and set the preconfigured layout. The layout became very static and all pages looked fairly similar.

A main disadvantage with this structure is also that you cannot have an/many element(s) inside a frame("a cell in cell"). The marketing director and the group manager asked me and my supervisor to look for a better structure that provided more flexibility.

LCD server phase II

Phase II aimed to make the pages more flexible and also to be able to create page sections inside borders/frames. We solved that by using the template sections a lot more than in the first phase, in the first phase the template sections were basically used as layer with the intention to be used in the future in some way if necessary. Position- and size data for CSS div tags, which is a container for HTML-components, were connected to each template section. Each template section has information about the corresponding page section:

- How many pixels it has to the upper border of the web page
- How many pixels it has to the left border of the web page
- How wide the container should be in pixels
- How high the container should be in pixels
- If it should contain a/many page section(s) or not

To be able to configure the pages with help of these five statements above, made the pages a lot more flexible. The last statement, whether a template section should contain a page section or not, made it possible to let some template sections only be frames around an arbitrary number of page sections.

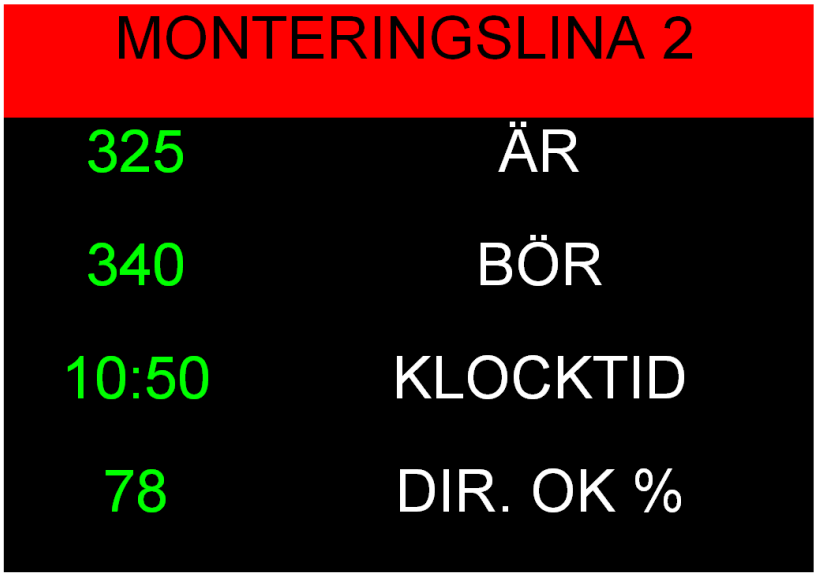

Figure 8: A typical page, in an industry production, that the LCD server can generate.

LCD server communication

The communication between server and client was fairly similar between the two phases, some minor change in how the pages were presented in the web browser was the only difference(described above).

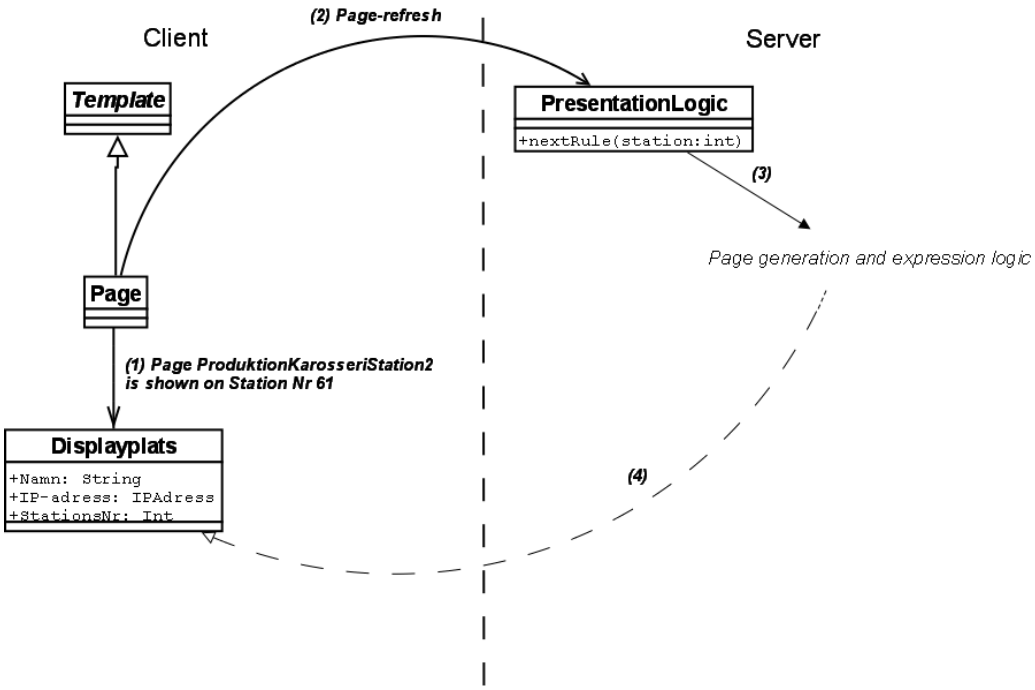

Figure 9: The illustration shows how a page is rendered. The process starts by a page refresh, followed by some logic for picking the rule. From that rule is the next page generated.

The communication is built on the client's requests of a page. The request is done because of a page refresh by the client, the page refresh time(viewing time) depends on the rule used. A method nextRule is called when the page refreshes itself. nextRule picks the next rule in queue, if there are many rules waiting to be displayed, a queue is made and the elements are prioritized on database id.

Then, from the picked rule, the page is extracted and the frames are made, it is at least one frame made; the frame around the whole page. After this stage the page sections are generated. Each page section picks the right text value from the database depending on how the page section is configured. The programming code for how the text in the page section is generated looks as follows:

```
if (pageS.watchTime > 0)
        {
            label.Text = CommonCode.getCurrentClockTime();
        }
else
        {
            if (pageS.staticValue == null && pageS.dynamicValue == null)
                label.Text = "";

            else if (pageS.staticValue != null)
                label.Text = pageS.staticValue;

            else
                label.Text = getVariableValue(pageS, v);
        }
```

Figure 10: The code snippet is an excerpt from fixLabelText method.

pageS is a page section object generated from ADO.NET Entity framework. label is the label that each page section has. Its text is manipulated both with respective to content(the example above) and with respect to graphical look. A page section has watchtime, staticValue, dynamicValue among other fields in the database. As the code snippet shows, it has got watch time, current time, highest priority followed by static value(value set by the user) and lastly has dynamic value(value collected from another database/machine) lowest priority.

All elements on the pages displayed in an industry environment are generated on the "fly". In the aspx-file, which corresponds to a "standard" HTML-file with the exception that it communicates with .cs-files, it is only a single panel declaration from the beginning. To this panel all dynamically generated items are put.

Page

When creating a page, you first have to choose a suitable template. When that is done you decide the name of the page and which variable group it should belong to. The choice of variable group will decide the different variables you will have available when configuring the page.

When hitting the button “Create page” you reach a skeleton of your page, based on the chosen template. The skeleton has an edit button for each configurable page section. It is also possible to update the page title.

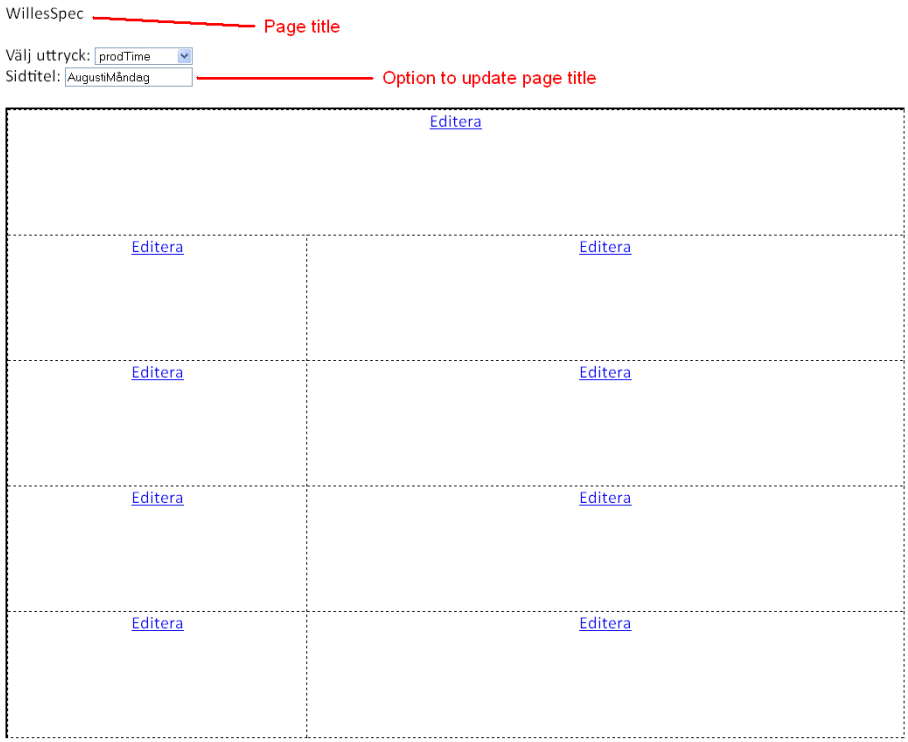

Figure 11: The picture shows what a newly created page can look like. The page has nine page sections, all editable, and a title, of course with opportunities to update.

Page section

As mentioned earlier each page is built upon page sections. The page is only an overall structure for the information; the essential building blocks are the page sections. There are two ways to configure a page section; either when the page is newly created and then you just click the edit buttons of the page or when a page is “a bit” older and then you are able to click the update buttons. Both options will lead to a page where you can configure the chosen page section.

This page has an authentic preview panel for the chosen page section and a number of options for configuration of the page section’s different options. The data base has a row where each page section’s configured parameters are saved.

The text in the page section can be set in three different ways:

- Check the clock time check box, this will show the current watch time.
- Write some text in the static value text box.
- Pick a “variable” from the database. The variable’s value will be the text.

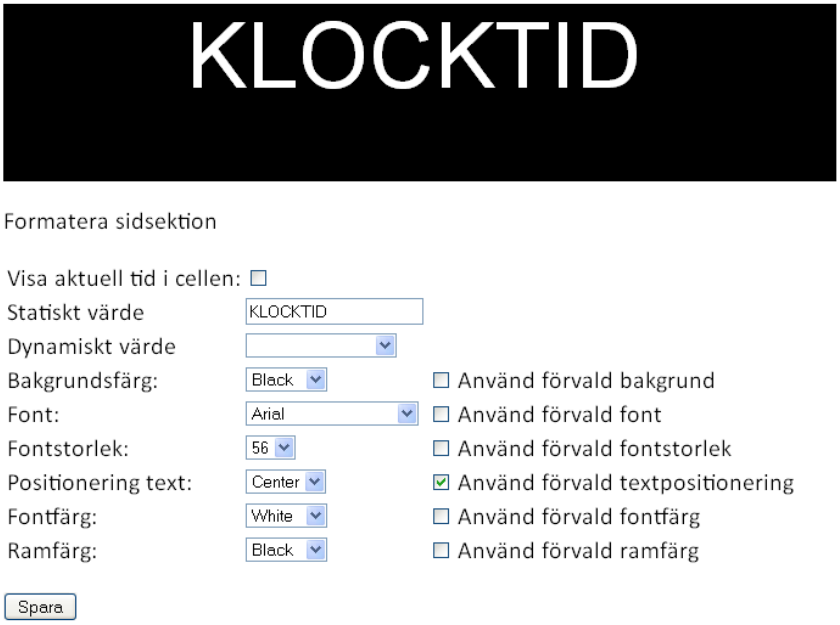

Figure 12: An illustration of how the configuration looks like for a page section. Notice the preview panel on top. The layout corresponds exactly to the values that the settings have been set to.

To the right of each graphic´s selection option, at the configuration part for the page section, is a check box where you select if the default value should be used for that parameter or not. If it is unchecked the selected value will be used.

Page rules

A group, of displays, has rules for what should be shown. For example it should be possible to display a special page if a condition is true. These conditions are called rules because they include the following: a page, an expression and the number of seconds the page should be shown until switching over to the next page.

Figure 13: The screen dump shows the rules connected to Grupp5. It is also possible to create another rule for the group by choosing an expression, a page and set the number of seconds.

There are no expression logic implemented at the moment, therefore all expressions are true right now. This means that all pages connected to a group will be shown. It is however seen as the next major feature to implement support for expression logic.

Expression logic

Even if no expression logic is implemented, it is taken into great consideration that expression logic should be available in a commercial product. To be able to provide this functionality, a special application layer is made for the expressions. At the moment do the variables just pass through the expression layer and every single variable is encapsulated to an expression.

In the future it is desirable to be able to combine variables to expressions as simple logic expressions. This will probably be made with a very limited, but still powerful, number of logical operators like AND and OR. The configuration of the expressions has to be done through some kind of interface with forms or similar.

Another feature for the future related to expression logic is the ability to change color of a page section depending on an expression value. For example, extremely low production results may be emphasized with a red background of the corresponding page section, instead of a green background when the results are normal.

Figure 14: A photo that shows how it can look like when displaying production data on a LED display

LED

Since the LED server's functionality was implemented after the LCD server the code from that subproject was reused when it was possible. The LED displays that were tested and used in this project are all provided by Binar Elektronik and are named BiDisp. Mutual for all tested BiDisps' are that they have red LED light diodes.

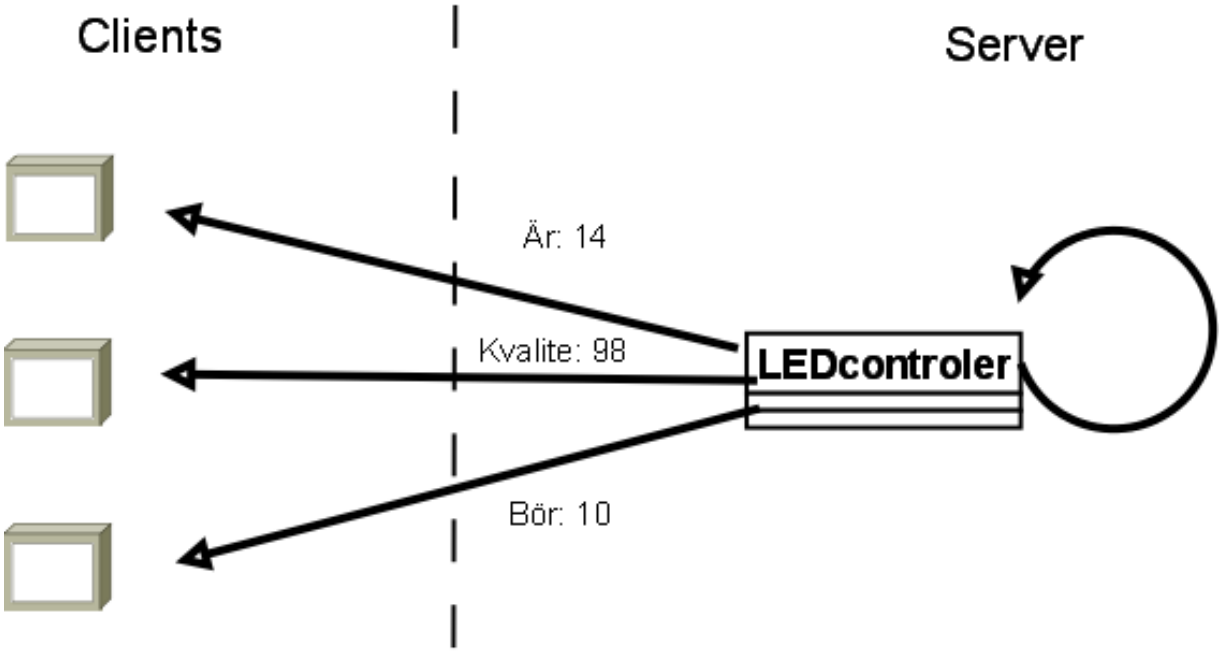

Figure 15: An illustration of how the LED server pushes data to its clients, i.e. the LED displays. The server has an "infinitive" loop that updates the clients with data.

LED server

Due to the fact that a BiDisp is completely passive, they do not request information as the case with LCD displays and there Windows CE modules. This means that a server needs to push out the information to the connected displays. The fact that the displays are passive has of course affected the development; it has been hard/impossible to use the same code as the LCD server.

The passive mode of the LED displays have been solved by a loop in the LED server code that every second checks if a display should change page and if so will do that. When the LED server is creating the new page, i.e. preparing page sections etc, it has to check that the page section information fits in the physical rows of a LED display. This check is important because a page can be prepared with the intention to only be used on LCD displays, with lots of space in comparison, but then used on LED displays. The method that checks if a page section fits into a LED display section prints a message, in the command line window, when a page does not fit into the desired LED display.

A BiDisp is connected to a network by a simple Ethernet port. To communicate and show information on the display you establish a socket connection and send UDP packets, to the desired display's IP-address, with the information as a string. To initiate the connection you send a so called Start Of Text(STX) message, which is a single ASCII sign.

```
private void pushToDisplay(string ip)
        {
             Socket s = new Socket(AddressFamily.InterNetwork,
             SocketType.Dgram, ProtocolType.Udp);

            IPAddress broadcast = IPAddress.Parse(ip);

            char cStart = '\x002';

             byte[] sendbuf = Encoding.ASCII.GetEncoding(865).GetBytes(cStart
             + completeString);
            IPEndPoint ep = new IPEndPoint(broadcast, 1001);
            s.SendTo(sendbuf, ep);
        }
```

Figure 16: The method pushToDisplay that sends a string to a display. The string is converted to an array of bytes with a STX sign in the beginning to start up the communication to the display.

User interface

The core of the application is a web interface built with ASP.NET Web forms. The interface aims to hold the application's parts together. It has one CRUD for each element in the application hierarchy(display, group and area). The page elements use basically the same concept but with some minor variations.

CRUD means Create, Delete and Update. So for each hierarchy element it is a listing function with an update and a delete function for each list row. The create function is placed on a separate page.

Lista displayer

Namn	IP-adress	Led-tecken	Info	
BiDispMartin	192.168.1.155	14	Fungerande	Delete
EntreDisplay	192.168.1.153	15		Delete
MarkusDisplayBruten	192.168.1.138	10		Delete

Figure 17: The picture above shows a typical listing with CRUD functionality. The Update is done by clicking the display name, the Delete by clicking 'Delete' and Create is done on a separate page.

The listing function of the different elements is built with ASP.NET listing support. This gives very nice and customizable tables with useful functions, for example assortment of a list by clicking the column header. The list above is for example sorted by name, if you would like to sort it on IP-address; you only click that header.

Result

The visualization manager is built up by two different viewers; one for LED displays and one for LCD displays. These two products are connected to one another and administrated by a web interface. In addition, a special requirements specification report was created.

Discussion

In the beginning of the master thesis project a number of project purposes were established. The most important of them was to investigate and research for how a visualization manager could be developed. The development of a prototype of the visualization manager for LCD and LED displays was ranked as purpose number two. Other intentions were to implement the application with the latest technology within some framework/platform and also to get experience of how it is to develop a product at a commercial company.

The purposes were then rewritten into requirements during the prestudy. A specification requirements report was developed as platform for how to build the application.

This project was by far the biggest project I have been involved in and I decided to do it alone. I would say that discipline and planning ability is equally important, if not more important, than programming and architectural skills, to be able to finish such project. Even if it sometimes has been less interesting and not so challenging tasks to solve I have always pushed the project further. I have had good help from the employee´s of the company where I did the work and especially my supervisor at the company.

The supervisor had a fairly good overview of what the product should look like. It helped me a lot to discuss with him different solutions and topics that I did not fully understand straight away. The experience of designing large software applications that the supervisor had was also important to avoid different issues connected to a project of this dignity. A usual manner when doing something new is the “trial and error” approach. In this project the redo activity sessions were very few.

The first requirement was to evaluate the different needs a commercial product would need to handle. That was done by interviewing several key persons at the company and also to visit an industry environment where LEAN production in general and visualization especially were used. Those activities helped me a lot in getting a good overview of how the system should operate individually and cooperate with its different stakeholders. A requirements specification was assembled to meet the requirements in an industry environment. It helped a lot to draw sketches when discussing different requirements with people at Binar Elektronik.

The second requirement was to implement a prototype of a LED and LCD visualization manager with as many features useful in a commercial product as possible. With one exception, the layout of the web interface, I am more than satisfied with this requirement. Even if the interface is not the best-looking with respect to graphics it is definitely useful. The layout is easily changed because it is separated in style sheets(CSS). In the beginning I did believe that it should be much harder to implement the

visualization manager than it actually was. The key for handling this challenge was to split and isolate different tasks and integrate them, one by one, as soon as they were implemented.

The last two major requirements were to get experience by working with a software project at a commercial company and to learn the latest software technology within the chosen programming language. To work at a company has only been stimulating. The employee's at the company have lots of experience, which has been great to take part of. The company's atmosphere is really good and no questions have been too silly to ask.

When it comes to developing with the latest technology within the chosen language it was extremely interesting since .NET was chosen as the development platform. To use ADO.NET Entity Framework as bridge between the database and the programming code was really useful. LINQ and Lambda expressions did also help to emphasize the good experience of programming within Visual Studio and .NET.

ASP.NET MVC was released at the same time as the decision of development framework was taken. My knowledge about ASP.NET MVC was very restricted at that time and the situation was the same for my supervisor at the company. If I would choose development framework today I would evaluate ASP.NET MVC and, of course, compare it to ASP.NET Web Forms to see which solution that would suit the project best.

Conclusions

When ending a project the questions; "Did the result of the project meet your expectations of the end product, when starting the project?" and "Was it worth the work effort?" are important to ask.

On the latter question I can definitely answer "Yes!". I have found the development of the application very interesting and satisfying. When starting the project my experience with the .NET environment was negligible and the task seemed hard to grip. To break down the project into subtasks and solve one and each of them has therefore strengthened my self confidence a lot with respect to computer programming and project management. This is of course a great endpoint for a student at an information technology related department.

The former question, if the result meets my own expectations is also fulfilled. In the beginning of the project I was uncertain if I would manage to implement a solution for both LED and LCD displays. This was when looking in the rear-view mirror only under estimations of my own performance. When thinking about the capacity of the product the idea is simple but still great. I have a couple of times during the implementation reflected about how profit generating a slight half years work can, I'm not sure it will, be.

The work with the product will continue by my supervisor and his colleagues at Binar Elektronik. Some changes will probably be made with the layout and some more features will probably be added before the product is commercialized.

Recommendations for further work

It is definitely worth trying to implement support for expressions and to combine variables, not single variables as it is now. This functionality can be used both by the logic for which page that should be shown in a special situation but also to format page sections in special ways depending on the production status of a production unit or similar.

In the future it may also be interesting to try and develop the product in Silverlight to give the product a “cooler” look.

Bibliography

[1] Interview with Kenneth Ferm, group manager 24/8 2009

[2] Interview with Anders Wilhelmsson, marketing director Binar elektronik, 5th of June 14.30

[3] "Taylor, Frederick W"-article accessed on Encyclopædia Britannica 8th of June 09.00.

[4] Tremblay ,G.(1995). The Information Society: From Fordism to Gatesism: The 1995 Southam Lecture, Canadian Journal of Communication, Vol 20, No 4

[5]Womack, J.P., Jones, D.T & Roos, D. (2007). The machine that changed the world – How lean production revolutionized the global car wars. Simon & Schuster.

[6] Troelsen, A. (2007). ASP Pro C# 2008 and the .NET 3.5 Platform, Fourth Edition. Apress.

[7] Boehm, A. & Murach, J. (2008). ASP.NET 3.5 web programming with C# 2008. Mike Murach & Associates, Inc.

[8]Marguerie, F., Eichert, S., & Wooley, J. (2008). LINQ in Action. Greenwich: Manning Publication Co.

[9] http://www.kirupa.com/net/next_gen_data_access_linq_pg1.htm