DEVELOPMENT OF A MOBILE SOLUTION FOR DELIVERING PRICE QUOTES IN THE CONSTRUCTION INDUSTRY

Filip Andersson and Anders Pantzar

2013-10-29
Carried out at: Byggwalle AB
Advisor at MDH: Afshin Ameri
Advisor at Byggwalle AB: Peter Vahlfridsson
Examinator: Rikard Lindell

Mälardalen University, Västerås
ABSTRACT

The rapid growth of the mobile consumer market has influenced companies to make their workplace digital and mobile, with the goal to remove ties to a physical workplace, increase productivity, and trivialize tedious tasks. The construction industry is in need of modernization; it is a competitive market and to get a contract it is vital to provide a reasonable price quote fast. This bachelor thesis was performed together with Byggwalle AB in Västerås with the objective to create a mobile solution for price quotes, based on an earlier experimental prototype.

This thesis will look at how the manual price quote creation process can be sped up and made more transparent to build up trust between the companies and the customers. It will also look at the choices made when developing a mobile application for the construction industry, how to make it flexible enough to fit different kinds of businesses, and look at how the data is administrated using a web based administration system.

The result of this thesis is a promotional prototype, which sped up the delivery of price quotes from 3 weeks to 3 hours, and will be distributed to selected companies for further testing in the near future. We hope that this product can change the way business is done in construction industry.
PREFACE

We started with Byggwalle AB's predecessor to this project, Rummenkalkylator, and got to experience delivering a product to a happy customer. In that project we were all beginners, but we learned a lot and came out much more experienced. Getting the opportunity to once again partner up with Byggwalle for an improved version of the product was something we couldn’t turn down.

Working with this, we have grown as developers and learned new techniques, but also made some mistakes. There have been the usual misinterpretations of requirements; setting the project scope has been challenging and documents have not been updated properly. These mistakes serve as great lessons, and will affect the way we work in the future.

We would like to thank Byggwalle AB for this opportunity and also for them providing us with an office of our own to work from every day, this has been of great help. Especially we would like to thank Peter Vahlfridsson for his enthusiasm, all the lunches to keep us energized and for a pleasant evening at his cottage.

We would also like to thank our supervisor Afshin Ameri for his feedback and support.

Västerås, October 2013
Filip Andersson
Anders Pantzar
<table>
<thead>
<tr>
<th><strong>Word</strong></th>
<th><strong>Meaning</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface. Specifies how software components should interact with each other.</td>
</tr>
<tr>
<td>CPU</td>
<td>Core Processing Unit. Used for general purpose processing of software.</td>
</tr>
<tr>
<td>GPU</td>
<td>Graphical Processing Unit. Used for vector-based processing for graphics or other highly parallel tasks.</td>
</tr>
<tr>
<td>IDE</td>
<td>Interactive Development Environment. A software that provides comprehensive facilities for software development. Normally consists of a source code editor, build automation tools and a debugger.</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol. Defines how data is addressed when transferred across a network.</td>
</tr>
<tr>
<td>JavaScript</td>
<td>A script that runs locally in the client's web browser.</td>
</tr>
<tr>
<td>RAM</td>
<td>Random Access Memory. Small and fast memory used to speed up computations.</td>
</tr>
</tbody>
</table>
CONTENTS

Introduction ........................................................................................................................................... 1

Background ........................................................................................................................................... 2

Problem Definition .............................................................................................................................. 12
  Problem............................................................................................................................................. 12
  Limitations ........................................................................................................................................ 12

Methods ............................................................................................................................................... 13
  Making a price quote digital................................................................................................................ 13
  Mobile Application ............................................................................................................................. 14
    Why tablet? ........................................................................................................................................ 14
    Why native?........................................................................................................................................ 14
    Why android?.................................................................................................................................. 15
    Data storage ...................................................................................................................................... 15
    Database communication .................................................................................................................. 15
    Security ............................................................................................................................................ 16
  Administration system ........................................................................................................................ 17
    Framework and CSS-Layout............................................................................................................... 17
    Data Storage and formats ................................................................................................................ 17
    Security ............................................................................................................................................ 18

Design ................................................................................................................................................. 20
  Mobile application.............................................................................................................................. 22
  Administration system ....................................................................................................................... 25

Implementation .................................................................................................................................... 30
  Development Methodology ................................................................................................................. 30
  Work flow .......................................................................................................................................... 30

Tests .................................................................................................................................................... 31

Discussion .......................................................................................................................................... 33
  Future Work ....................................................................................................................................... 34

References ........................................................................................................................................... 35

Appendix A – Quote for Harakersgatan 10 ......................................................................................... I

Appendix B – Generated price quote exampleAppendix C – Questionnaire ........................................ II

Appendix C – Questionnaire ................................................................................................................ III
FIGURES AND TABLES

Figure 1: Graph showing the number of downloaded mobile applications per year since 2010....2
Figure 2: Pie-chart depicting the market share distribution of smartphone sales in 2013 Q2.......5
Figure 3: MEPS interface..................................................................................................................8
Figure 4: Byggnyckeln interface ......................................................................................................9
Figure 5: Byggsamordnaren interface..............................................................................................9
Figure 6: Byggkalkylatorn interface................................................................................................10
Figure 7: Struqtur interface .............................................................................................................11
Figure 8: Admin view from the previous project ...........................................................................20
Figure 9: Application view from the previous project .....................................................................21
Figure 10: Early sketches ...............................................................................................................21
Figure 11: Log-in screen ....................................................................................................................22
Figure 12: View for entering information about the current price quote ......................................22
Figure 13: View for adding surfaces to the price quote .................................................................23
Figure 14: View for choosing which articles to apply to which surface(s) ....................................23
Figure 15: View showing a preview of the quote .........................................................................24
Figure 16: Old Edit view of Article tree ..........................................................................................25
Figure 17: Old View for changing account information .................................................................26
Figure 18: View of articles in a template .........................................................................................27
Figure 19: View creating a new article ............................................................................................28
Figure 20: List of registered companies ........................................................................................29
Figure 21: The work flow of the system ..........................................................................................30

Table 1: Comparison of mobile development platforms ..................................................................14
Table 2: Score from the questionnaire ..............................................................................................31
INTRODUCTION

Founded in the 1950s, Byggwalle is one of the oldest construction companies in Västerås. They are a turnkey contractor that does new housing estate, rebuilding, and renovations and reparations in connection with insurance loss [1]. In this line of work it is necessary to give potential customers an estimation of the costs of performing a certain task, this is known as a price quote.

Employees at Byggwalle spend most of their time with compiling these quotes. If this process could be sped up it would increase the efficiency of their employees. In order to achieve this goal, Byggwalle came up with the idea to have a webpage where customers could perform their own calculations and then send it to a partner company in the area. There would also be an iPad application with identical functionality. Later it was also planned to make an Android version. All available work items and prices would be managed via a web system by an administrator at Byggwalle.

With no developers working at the company, Byggwalle turned to Mälardalen University for help. Their idea was presented in the course Software Engineering 2: Project Teamwork (DVA313) as one of the projects the students could work with [2]. A team of seven students (including the authors) were assigned the task of developing the product under the name "Rummenkalkylator".

The resulting product was used and evaluated by Byggwalle and new ideas were born. After some testing it was apparent that it was more common that an on-site employee used the tablet to present a price quote to the customer instead of a customer taking the time to fill it out themselves. It was concluded that a product focused on the on-site employees would be more useful and in long term it could also be sold as a service to other companies. This new idea was presented to the development team from DVA313 and two of them agreed to work on it which resulted in this thesis.

As mentioned earlier, when an employee compiles a quote in the office, the customer might feel that he is not part of the process and worry that the price is manipulated by the company. Examples of this can be found in forum threads, such as at byggahus.se [3], discussing the subject. Delayed price quotes and customers feeling cheated is a problem for the company. This thesis will look at how the manual process can be sped up and made more transparent; it will look at the choices made when developing a mobile application for the construction industry, and how to make it flexible enough to fit different kinds of businesses; and investigate how the data is administrated using a web based administration system. A qualitative study will be performed with regards to customer reactions and usability; additionally a quantitative study will investigate the actual time gained using the system.
BACKGROUND

The rapid growth of the mobile consumer market has influenced the corporate climate. Employees expect the same kind of time- and place independency that they are used to get in their private lives [4]. This has made companies to want to modernize by making their workplace digital and mobile, with the goal to remove ties to a physical workplace, increase productivity, and trivialize tedious tasks.

This is a relatively new trend and would not be possible without the modern mobile devices. Since the launch of the 3G (Generation 3) mobile network in 2001, and with the standardization of the 4G network in 2012, the network speed of mobile devices have reached a level high enough to make it affordable to have a constant Internet connection.

The always-connected capabilities of 3G technology and the increasingly stronger processing power of the still relatively small devices removed the need for businesses to be tied to a physical location. With the right software installed on their devices, employees could do their work on site, without the need to be tied to an office.

The early (2G) devices had a limited variation of mobile applications, such as calendar, address book, calculator and alarm. The increase in processing power in mobile devices, the availability for specialized APIs and IDEs, together with the launch of the first mobile application distribution systems in 2008, lead to an increase in the variety of mobile applications. The number of applications downloaded each year has steadily doubled since 2010 (See Figure 1) [5] [6].

![Mobile application downloads per year](image)

FIGURE 1: GRAPH SHOWING THE NUMBER OF DOWNLOADED MOBILE APPLICATIONS PER YEAR SINCE 2010.

One of the industries that is in need of modernization is the construction industry, it is a competitive market and to get a contract to perform for example a renovation or a new construction project it is vital to provide a reasonable price quote fast. If a company is too slow
giving a quote there is a risk of losing the contract to a competitor, even though they may in fact be more expensive. The problem is that most software used to provide an accurate quote require an on-site employee to take measurements, return to the office and spend hours sorting through long material lists to find the exact materials they need [7].

As stated by Lundgren, customers want to be highly involved in the selection and specification of a product that they have put a high investment into [8]. Seeing as both renovation and construction is expensive (a high investment for the customer) this holds true for the construction industry. Lundgren continues to say that the customer satisfaction is positive if the outcome is as the customer perceived it would be [8]. A positive experience of the development leads to recommendations [9] and thus a growth in market share. This is due to customers relying on recommendations when making purchases more than anything else. Another study shows that 89% of Europeans trust recommendations from people they know [10]. It is therefore important to find a way to involve the customer as much as possible in the creation of the specification and price estimation/quote. Involving the customer will increase transparency and thus make it more risky and less attractive for companies to cheat [11]. This will in turn lead to positive customer experiences and improved company reputation.

Studies have been made regarding improving business by moving to a mobile work solution. These studies reveal that the technological challenge of implementing the solution might not be as difficult as the challenge of incorporating the solution in your business [4].

There is a difference between generations regarding their attitude, requirements and needs concerning technology. Companies which keep up with the technological progress and can provide technological tools and solutions which people use in private life may be more likely to attract younger individuals. At the same time less technologically adept people may need different kinds of support and challenge the process of incorporating mobile solutions. Consequently more effort may be spent on changing attitudes rather than keeping up with technological advancement [12].

Due to this, it is important to plan ahead on how to introduce the new mobile procedures to employees. Using mobile solutions enables employees to become less dependent on others to finish business processes, but that only applies if the solution is understood and used [13].

According to Basoule and Rouse, there are four phases necessary to go through in the progress toward a mobile organization [14].

1) Make resources and information accessible for employees
2) Improve the current work process
3) Assess and reformulate business strategy on a broader plane
4) Open up opportunities for creating new core competence

This means that the first step is to provide the mobile solution to the company’s employees, the second step is to streamline the work process using those solutions. These go fairly hand-in-hand and should both be achieved through the incorporation of the mobile solution. Once employees are familiar with the new work process the third step is to reformulate the overall
According to Brandes, when mobilizing a none-mobile business, it is important to work in an iterative fashion and to have a ‘trial and error’ mentality [4]. The area is unexplored and every actor needs to learn from each other’s experiences. The change to mobile need to happen gradually, the attitude and mindset need time to adapt to the new way of work, as does the general technical knowledge and implementation.

Even if there is a current software solution which performs the required task, it cannot be directly ported to a mobile device. Due to the smaller screen size of the mobile devices, compared to a personal computer, the interaction design needs to be specialized and cannot be directly copied from a personal computer version. Another important aspect to consider for a mobile solution is that it is best suited as a complement to current software. The portable properties of a mobile solution open up for a new way to work, since it can be used outside of an office environment, and can therefore be used for tasks unsuitable for stationary tools [15].

When it comes to the choice of the mobile platform to use, there are three main actors on the mobile market today; Android (Google), iOS (Apple), and Windows Phone (Microsoft). Out of these, iOS was the first to be released in June 2007. It is based on the Darwin OS and is not licensed for non-Apple hardware. Applications are only available through the Apple App Store. iOS held the early lead in regards to market share. It was not until the second half of 2010 that iOS was surpassed by Android. Android was released by Google in October 2008 and is an open source mobile operating system powered by the Linux kernel. Due to its open source nature, devices running Android are highly customizable. Applications are available through Google Play but can be installed from other sources as well. In October 2010 Windows Phone was released by Microsoft. Windows Phone is the successor of Windows Mobile and is a proprietary (closed source) mobile operating system. Applications are available through the Windows Phone Store. As of the second quarter of 2013, Android holds the majority of the world-wide market share, with iOS in second place [16].
The possibility to be constantly connected to the Internet opened up the option for mobile applications to store and share data in databases away from the physical device. Databases make storing large quantities of information easy and quick. To control the data flow a DBMS (DataBase Management System) is necessary. It is the DMBS' responsibility to keep data consistent over time. Doing so is a very intricate task and DBMS software is often considered some of the most complex software created [17].

There are many kinds of DBMS but their purpose is all the same. They manage the logical data structure, keep track of relations and allow modification of the stored data. They also have ways to restrict access to data for different users [18].

Some of the more well known DBMSs include MS Access, MS SQL, MySQL, and Oracle. In a study by Saiyad, a complete review of these DBMSs is presented [17]. The study also concludes special circumstances in which each of these DBMSs fit the best. According to Saiyad, MS Access is best suited for internal use by 20 or less concurrent users and offers graphical tools; it is also cheap [17]. MS SQL is excellent when reports need to be generated but there is no need to use forms for entering data into the database. MySQL and Oracle are both great choices for web based systems with unlimited users which should stay online 24 hours a day. MySQL is the cheaper alternative, but Oracle is better suited if handling sensitive data.

Databases work by storing data in 2-dimensional tables, referred to as relations in this context. There is however a difference between a regular data table and a database relation. A table is simply a collection of rows which are bound by order whereas a relation is limited to unique rows but their order is irrelevant. In these relations columns define what attributes are stored for each element. These relations have a set of key constraint, the primary, and only required one, is an identifier unique for each row. These identifiers can be paired in another relation to represent a connection between two elements [18].
Accessing a database is usually done from a script on the same server as the database. The most commonly used server script language today is PHP [19]. PHP supports connecting to several kinds of databases e.g. MySQL, Oracle and MS SQL. This implementation was introduced in 1996 and is used by large companies today [18].

A framework can be used to simplify the server implementation. Using a framework lets developers focus on the system’s business logic since most basic operations are already available through the framework. This includes for example database connections, language controls, sessions and internal data-passing between scripts. By using these implementations developers are encouraged to write extendable and reusable code. Frameworks also provide a standard layout and procedures to follow which makes it possible to bring in new experienced developers without much need for guidance [20].

One such framework is CodeIgniter which is based on the Model-View-Controller (MVC) development pattern. This approach separates business logic from presentation in separate files [21].

- The Model-files takes care of the business logic and database communication
- The View-files presents information to the user
- The Controller-files pass data between the Model, The View, and other resources

To present data to the user the server provides an HTML document. HTML stands for Hypertext Markup Language and it contains two things; information to the browser on how to display information, and the information extracted using PHP [18].

Defining the layout for all elements in an HTML document may result in repeated definitions. This is solved using a Cascading Style Sheet, CSS, which applies layout parameters to multiple elements. This is done by assigning ‘classes’ to elements, and defining its parameters in the CSS [22]. For example a class ‘default_background_color’ may be assigned to both the header and body of a webpage and in the CSS be defined as ‘red’. If the color later needs to be replaced only the definition in the CSS needs to be changed to apply it to all elements with that class.

When a client requests a webpage the server first scans the PHP for scripts to execute. These scripts provide a static HTML output which is then sent to the browser and displayed, showing no trace of the script which generated it [18]. Upon connecting to a server a session is created associated with the connected client. In this session small tidbits of data can be stored to enhance the user’s experience interacting with the website. A good example is shopping carts used in web stores. These carts keep track of your intended purchases as you browse the store.

There is an issue with sessions however, they do not last long. For example the default setting for an Apache server is to close connections after 15 seconds of inactivity, meaning that the associated data is no longer retrievable. Oddly enough Apache stores the session data for 300 seconds, or 5 minutes, which is rather ineffective seeing as the connection it belongs to is long since closed. Increasing the connection timeout would technically address this disparity, but it would cause higher memory usage for connections that may or may not be used and thus limit the concurrent user capacity of the server. Instead the common solution is to use cookies.
A cookie is data stored on the client’s device by their browser. It can keep track of for example information about the user, the application, and preferences. Cookies are stored on a per domain basis, e.g. google.com, and are passed to the server with every request in a connection. To solve the session/connection timeout issue a session ID is assigned during the initial connection and stored on the client’s device as a cookie. This enables the server to identify the relevant session even after the connection which initiated it has closed [23].

These cookies introduce security issues to the system. If the session information in a cookie is stolen it may be used to illicitly access user accounts. When it comes to web security there are three kinds of adversaries necessary to defend against; the active, the eavesdropping, and the interrogative. The interrogative adversary queries the site and tries to exploit security vulnerabilities to extract compromising information. The eavesdropping adversary is not only capable of querying the site; it can also listen in to all network communication between the client and server, this is generally referred to as “sniffing”. Lastly the active adversary is the most versatile of them all; it works as a middle hand in the network communication. Apart from querying and eavesdropping it can also modify each individual network package between the client and the server [24].

If a password is stolen by any of these adversaries the data should still not be able to be interpreted directly. Sensitive data should be hashed using a cryptographically secure algorithm. This means that plain text data is obfuscated in such a way that it cannot be reversed. The hashed string is unique for every possible input text. To find out if a given text matches the stored information it is necessary to hash the text and then compare the hashes, if they match the inputted text and the stored information are equal [25]. Two such algorithms are bcrypt and the newer scrypt.

The bcrypt algorithm is from 1999 and only requires 4MB of RAM to perform and with modern technology it can be performed much faster than anticipated 14 years ago. Despite this bcrypt remains secure, but other algorithms are being developed which will be able to grow with increased processing power. One such algorithm is scrypt developed in 2009.

Scrypt uses memory-hard functions which makes it harder to parallelize. It also has a parameter for defining the amount of memory it uses as opposed to a constant amount like older algorithms. This lets it scale with new hardware performance to keep its execution cost high [26].

To ensure that a service is safe and effective there are several organizations which develop standards for e.g. security and data. Two of the more prominent are NIST and ISO.

NIST (National Institute of Standards and Technology) is a part of US Commerce founded in 1901 [27]. They define crucial security standards for the web and how to authenticate users. Following these standards ensure that user’s personal information remains reasonably safe [28].

ISO (International Organization for Standardization) is an organization that since 1947 has developed voluntary standards to streamline the industry today and make collaboration easier. Conforming to ISO standards regarding data formats ensure that the information can be used
elsewhere if necessary, or that importing information can be done without converting it to some proprietary format [29]. By following these standards the administration of data is facilitated.

Even if data is in a proper format there is usually too much of it to handle manually, instead special applications are used to make the administration easier. Early administration systems were complex and cumbersome, and many still are today. They provide a full view of every detail of a system and can control everything, if the administrator has the appropriate training and advanced understanding of how the system works. Today there is a trend towards more simple administration systems where the users do not need specialized training to understand it, and where only the information and functionality that is actually needed is presented.

For example the Swedish company Tictail provides an E-commerce platform which focuses on simplicity and usability [30]. In 10 months 10'000 stores have been opened through their service demonstrating the impact of an easy to use administration system [31].

In contrast to this, the currently available systems which specialize in compiling quotes for the construction industry are bulky and old (Figure 3). The most common one is MEPS [32]. It is also the oldest system, established in 1988. It is a complete system and has become a standard in Sweden with over 700 connected companies.

![MEPS Interface](image-3)

**FIGURE 3: MEPS INTERFACE.**
Another common tool is Byggnyckeln [33]; a calculator for “simple and fast” quote compiling released in 1997 (Figure 4). It is the most spread system in Sweden with 3000 users.

FIGURE 4: BYGGNYCKLN INTERFACE.

Byggsamordnaren [34] is yet another calculation tool for the construction industry, released in 2001 (Figure 5). What separates this from the previous systems is that Byggsamordnaren has a web application for mobile devices. This app does however require the buyer to set up their own server.

FIGURE 5: BYGGSAMORDNAREN INTERFACE.
There is also an online calculator for price quotes which is based on prices from a price compilation magazine: Byggkalkylatorn [35], it does however have restrictions on the customizability of the calculations due to being based on a physical medium (Figure 6).

FIGURE 6: BYGGKALKYLATORN INTERFACE.
Recently a new tool was released: Struqtur [36]. It is a system that handles project management, quote- and contract templates, invoices and salary, and collaborations (Figure 7). Struqtur has a mobile app, but it only provides the ability to create time reports.

FIGURE 7: STRUQTUR INTERFACE.
PROBLEM DEFINITION

PROBLEM
This thesis is focused on solving three key problems in the construction business. In this section each of these will be presented together with a more detailed explanation of their meaning.

1: How can a price quote be delivered faster?

Today the company employee has to return to the office to compile a price quote. During this time another company might present a quote faster and get the job instead.

2: How can a price quote be made more transparent?

Since a price quote is made from what the company employee and the customer decide together there is always a risk for misinterpretation. And since the price quote is compiled away from the customer, he might feel left out of the process and lose trust in the company.

3: How can the system be designed to be usable for different kinds of construction businesses?

Different kinds of construction businesses do not offer the same kind of products or services, and they might not categorize them in the same way. For example a renovation company might divide its services based on what kind of room it concerns while a painter divides it into priming, painting, etc.

LIMITATIONS
Only a promotional version of the system will be created due to the short time available for development. This means that not all functionalities will be implemented, but the user interface will be complete.

The insight into the construction business is based solely on one single business contact and his associates’ personal experiences. With regards to this, the final product can be considered as a tailored product for Byggwalle and might require some changes in order to be used in a more general way.
METHODS

MAKING A PRICE QUOTE DIGITAL

Before designing software to handle price quotes it is necessary to decide how to represent them digitally. Appendix A shows what a price quote looks like today and represents a guideline for what this software should produce as output. There are a few key elements to take note of: the customer and quote information, the specification, and the deductions. Other areas such as the logotype and company information are more straightforward.

At the top of the page there is an indication of what type of ‘Object’ the quote is about. The object defines what is to be worked on, e.g. a kitchen or a bathroom. At the bottom of the page there is a specification of costs divided by what kind of entrepreneur performs the work. This is how it is usually done in the construction business since they have partnerships with other companies such as painting or heating and sanitation [7]. Their customers often want to know what the different parts of a project will cost which is why it is displayed in this fashion.

For Swedish companies there is also a deduction section based on the ROT-program. This program is meant to allow a private person to get a reduction on construction work costs. It is a 50% deduction applied to only the cost of the work, not the material, and for a maximum of 50’000 SEK deduction each year, or 100’000 SEK if there are two eligible adults in the household [37].

To be able to discern which entrepreneur will perform a certain work on each work item (article) must be assigned a field. These fields include HVAC\(^1\), Construction, Painting, etc. Apart from defining a field, each article needs a descriptive name such as ‘demolition’, and an account for the base, material, and work cost. Without the different types of cost it would not be possible to calculate the ROT.

A price quote is usually given for a specific or a group of Objects. Each of these has different available work to be done, for example demolition may be done in both a living room and a bathroom, but only the bathroom needs plumbing. This means that articles can belong to one or several templates. Each template represents an object such as kitchen or bathroom, and a price quote can consist of one or several templates.

Articles require measurements to calculate their final cost. For example the cost of putting in a new carpet depends on the area of the floor. To accommodate this, every article has a specific calculation type. The available types are: Amount, Area (Without doors and windows), Circumference, Door area and Window area. These articles can then be assigned to areas defined in a calculation. An area can for example be 3x10m with a 2x1m door, if the cost of painting it is 50SEK/m\(^2\) then the final cost of painting would be 1400SEK.

To administrate the information, a web application will be developed. The information will then be used in a native Android tablet application to compile price quotes.

\(^1\) Heating, ventilation, and air conditioning
MOBILE APPLICATION

To meet the requirements of a fast price quote delivery and transparency, an android tablet application was developed. The purpose of the application is to involve the customer in the process and to give them an understanding of how changes to the proposed work would affect the price. The application would eliminate the need for the company employee to return to the office to compile the quote.

WHY TABLET?

Tablets were chosen due to their larger screen sizes, compared to phones. A large screen is needed to give a good overview of all the information the application needs to convey. A large screen is also preferable when working together with a customer as it provides an easy way for both parties to see how the price quote is compiled.

WHY NATIVE?

There are two types of applications: native and web. A native application runs directly on the device, gaining access to hardware features such as GPS, accelerometer, camera etc. A native application also does not require a constant Internet connection and can thus be programmed to run offline. Due to the ability to access hardware, the application must be developed for a specific platform. Each platform uses different languages and IDEs [38] (Table 1).

<table>
<thead>
<tr>
<th>Platform</th>
<th>Language(s)</th>
<th>IDE</th>
<th>OS for IDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>iPhone</td>
<td>Objective-C, C</td>
<td>Xcode</td>
<td>OS X</td>
</tr>
<tr>
<td>Android</td>
<td>Java, C, C++</td>
<td>Eclipse/Netbeans</td>
<td>Windows, Linux, OS X</td>
</tr>
<tr>
<td>Blackberry</td>
<td>Java</td>
<td>Eclipse</td>
<td>Windows</td>
</tr>
<tr>
<td>Symbian</td>
<td>C, C++, QT, Java</td>
<td>Any text editor</td>
<td>Windows, Linux, OS X</td>
</tr>
<tr>
<td>Windows Phone</td>
<td>C#, VB.NET</td>
<td>Visual Studio</td>
<td>Windows</td>
</tr>
<tr>
<td>webOS</td>
<td>HTML, Javascript, CSS</td>
<td>Any text editor</td>
<td>Windows, Linux, OS X</td>
</tr>
</tbody>
</table>

TABLE 1: COMPARISON OF MOBILE DEVELOPMENT PLATFORMS

Even platforms with the same language and IDE (i.e. Android and Blackberry) cannot share the same application since they use their own APIs.

A way to get around this platform dependency is to develop a web application. A web application is basically a web page running on the native web browser. These applications require a constant Internet connection and they cannot access the hardware features of the device. Web applications face the same problems as a normal web page does. Depending on the browser, JavaScript runs at different speeds causing inconsistency in performance. The same could however be argued for different mobile devices and is not something that should influence the decision. Also, CSS can be interpreted differently in other browsers. This is a more serious concern since it will require more testing to ensure that the application is working as intended in all browsers. If it does not, then the usability of the application might differ, and even small differences can render the entire application useless.

There are frameworks that can work around the hardware access problem. Three of the most common ones are PhoneGap [39], Sencha Touch 2 [40], and Appcelerator Titanium [41]. They provide an API that work as an abstraction from the native mobile platform allowing the same code to be used on multiple platforms. The frameworks provide access to most hardware features, but they do not solve CSS-differences [38].
It was decided to develop a native application. The hardware access for valuable user data collection and the offline capabilities were more valuable to the project than portability.

**WHY ANDROID?**

When developing a tablet application there are three main platforms to choose between: iOS, Android, and Windows Phone. Due to the low budget of the prototype build, the price had to be taken into account. Both iOS and Windows Phone require an annual payment to allow developers to distribute their applications [42] [43]. While Android gives the developers the option to pay a onetime fee to get access to Google Play, but developers are allowed to distribute their Android applications in other ways and thus payment is not mandatory for a prototype project [44].

Even though iOS currently holds the largest market share for tablets, Android is gaining more and more ground each year [45]. Because of iOS’ popularity there are plans to develop an iOS version of the application, but that is not within the scope of this thesis.

Due to functional requirements the lowest supported android version is 4.0. This means only 63% of android devices are supported [46].

**DATA STORAGE**

To allow continuity between uses, and to provide an offline mode, the applications need to store data on the device. There are two common ways to do this: In XML files or in an SQLite database. XML (Extensible Markup Language) is a markup language that is readable for both humans and machines. It focuses on simplicity, generality, and usability over the Internet. XML is often used to represent data structures. SQLite databases are self-contained and serverless. It is popular in applications and embedded systems due to its small size [47].

It was decided to store data in XML files due to its simplicity. The fact that Android already uses the XML format to represent the visual layout made the learning curve very low. An SQLite database is however considered for a future version of the application.

To prevent long loading times when reading data from the files relevant data will be read to memory and read/write changes will be performed on the memory instead of the files. The files are only accessed when major changes happen, e.g. on startup, the save button is pressed, or another calculation is loaded.

**DATABASE COMMUNICATION**

Accessing a web database directly from a mobile device running Android is not possible since the operating system does not contain a native database connection service [48]. Even if it was available, as it is in iOS, it would be ill advised to use it for security reasons. The database credentials would have to be shipped with every instance of the application and the connection could possibly be sniffed to steal those credentials [49]. In order to safely access a database from a mobile device there needs to be scripts on the server hosting the database to provide the functionality, that way the database connection is only ever made internally on the server.
Another security issue when using databases is SQL injections. An SQL injection is when a user inputs data into the system which is formatted in a way that hijacks a database command, allowing them to access otherwise protected data. To prevent this, user input should be sanitized from malicious formatting. This is done using the built-in database connection functions in CodeIgniter, which automatically sanitize all input parameters.

Before the tablet application can communicate with the database it will need to set up a valid session using the tablet login credentials and deviceID. This is to ensure that no unauthorized parties can access the database by sending requests of their own.

If a session is set up, the tablet application can send requests to a set of scripts on the server. Each script corresponds to an SQL query and will perform the query based on the parameters in the request. The result of the query will be printed to an XML-file and passed back to the device, allowing the tablet application to read the information and parse it to understandable data [48]. If a session is not setup, the query will not be performed and an error code will be sent back to the tablet application.

**Security**

**Authentication**

To use the application, the user needs to authenticate itself. The application sends the login credentials together with the device’s ID to the server and waits for a response. The login credentials are used to validate the user and to ensure that the user is still active. While the deviceID is used to make sure that the user is not registered with another device. This is a business strategy to discourage users from sharing their accounts with non-registered users, and to protect business secrets.

After the first successful login the application stores the user credentials in the Internal Storage, and not in the XML files. This is to guarantee the integrity of the login credentials since the Internal Storage cannot be read by anyone else than the application itself and will be removed if the application is uninstalled [50]. By saving the login credentials the need for the user to provide the credentials every time they open the application is removed.

**Business secrets**

Since the articles and prices could be considered business secrets it is important to protect the data from unlawful access. In the case that a person with ill intent manages to gain access to a user's login credentials they will not be able to log in to the account from any other device than the one the user is registered for. This is because the login process also compares the registered deviceID with the id of the device sending the login request.

Since the business secrets are stored in XML files on the device to allow for an offline mode, they can be accessed by rooting the device. A person with ill intent could do this to steal the information, but would need to have access to the physical device to do so. The data could be encrypted to prevent this, but for the sake of simplicity and performance this has been omitted. To prevent the information from being stolen the device itself should be handled with care just like physical documents containing sensitive information.
ADMINISTRATION SYSTEM
It is necessary to have a way to administrate the data within the system. This will be done through a web-based application. The reason for making it web-based is because it makes it easy to connect to a database, and it is easy to distribute to users, no installation necessary. Also by hosting the service it will be easier to charge companies for using it.

The alternative would be to distribute it as a native application. While this would make it easier to create a dynamic and responsive tool, it has several downsides. First and foremost it would require several different implementations for use on different operating systems. It would be possible to use one implementation by using a cross-platform framework such as Mono, but this would require users to have Mono’s runtime environment installed [51]. Another downside would be the increased difficulty in maintenance if the application needs to be updated.

Via the administration system users should be able to handle their company data and manage all of their prices and templates. It will also be necessary to be able to download finished price quotes in order to print, email, or archive them.

FRAMEWORK AND CSS-LAYOUT
To speed up the development process it was decided to use the PHP-framework CodeIgniter when implementing the administrative system. The reason for choosing CodeIgniter is that it is lightweight (does not impact performance of the service) and well-documented, which makes it easy to learn [52].

Sometimes a framework does not contain a specific feature or it works slightly differently from what is required. To solve this CodeIgniter offers the possibility to write helper functions. These can be loaded and used as a complement or as a replacement to a function in the framework [53]. This has been utilized in order to make a feature for remembering language selection using a cookie and for handling passwords.

To ensure that the user interface of the administration system can easily be changed at a later time it has been built with a modular CSS layout using SMACSS (Scalable and Modular Architecture for CSS). SMACSS is a style guide on how to structure the CSS and its core principle is to categorize the CSS into five rules: Base, layout, module, state, and theme. The base rule defines the defaults, e.g. what a normal button should look like, how links and paragraphs are displayed. The layout rules define the layout of the page and holds modules together. Modules are the reusable and modular parts, e.g. sidebar sections, product lists, tables. State rules specify how something will look in a certain state, e.g. when it is active or inactive, hidden or visible. Theme rules allow for customization of color themes [22].

DATA STORAGE AND FORMATS
A few different DBMSs have been considered for storing the application data, each have their own circumstances where they fit best. For this thesis there are a few requirements and limitations regarding the data storage; the goal for the system is to be online 24 hours a day for a high amount of people, and due to the fact that the budget is small for the project the solution cannot be too expensive.
For this software it is important that a lot of users can access the system, this means that MS Access cannot be used as it only allows a maximum of 20 simultaneous connections. It is also supposed to be constantly online which points towards either Oracle or MySQL which are suitable for this. The deciding factor between these is the price; MySQL is much cheaper than Oracle and was therefore chosen to be used in this project [17].

To ensure consistency, a number of ISO-standards govern over the format of data. This also allows for easy importing of such data from other sources. Prices can be defined using different currencies. These currencies follow the ISO 4217 standard regarding how they are represented. Each currency has a three letter representation e.g. SEK or USD [54], and a definition of its prefix or suffix e.g. 10kr or $10. Another standard being followed is ISO 3166-1 regarding country codes. More specifically it is the 2-alpha version being used, meaning that each country is represented by two letters. This standard defines 676 distinct countries which was deemed to be enough granularity for this software [55].

In the future it is planned to be able to get audits from the database to specify for example the average price of renovating a kitchen in a specific city. For this it is necessary to store historical data in the database. A common solution for this is to never actually delete entries in a database. Instead there is an attribute called ‘isDeleted’ which is set to ‘true’. This kind of soft deletion is easy to implement, but in a large system this can lead to unnecessarily large data tables. In general this does not affect the speed of retrieving data much since most DBMSs are designed to handle this kind of flagging, but it may impact writing speeds when storing new data [56]. An alternative to soft deletion would be to use archive tables where deleted data is moved to reduce the clutter in the regular tables. Neither of these has been implemented though since it needs further investigation to decide which solution is the most suitable.

**Security**

**Password Storage**

People are lazy and forgetful when it comes to remembering passwords. This results in them often using the same password for multiple accounts. Due to this fact, it is important to store users’ passwords securely in case the database is breached [24].

NIST standards require passwords to be stored after being passed through a hashing algorithm together with a pseudo-random string, called a Salt. The hashing algorithm has to be expensive enough (time consuming) to discourage the creation of rainbow tables [25].

Rainbow tables are collections of pre-hashed passwords which can be referenced with a stolen hash to find the corresponding password. However this is made ineffective by joining all passwords with a unique Salt. The rainbow table then has to be created with the Salt in mind. Using a slow hash algorithm slows down the creation of rainbow tables. Consider a hashing algorithm which takes 100ms to perform. A user entering the correct password will hardly notice it, but performing the algorithm with millions of possible passwords would be time consuming and require a lot of processing power.

NIST suggests the hashing algorithm PBKDF2, but other algorithms exist which are also approved [25]. For this software it was decided to use bcrypt. It works the same way as PBKDF2 but has the slight advantage that it cannot be sped up by implementing it on a GPU, which
otherwise can increase the speed by 10-20 times for other algorithms. This is due to the fact that
the bcrypt algorithm relies on a table access which is fast on a CPU, but much slower on a GPU
[57].

**Secondary Authentication/Forgotten Password**

As mentioned in the previous section people are forgetful when it comes to passwords. Usually
web pages offer a way to retrieve or change forgotten passwords by authenticating you in an
alternative way. The issue with this is that no web page is more secure than its least secure
component, meaning the secondary authentication has to be as safe as the main one, which is
unfortunately often not the case [57].

A common solution for secondary authentication is to use personal knowledge questions such as
your mother’s maiden name or the name of your favorite teacher. In modern society with social
media the answer to this kind of questions is easily found out. Also it is common that the
answers are not stored securely in the database and a breach may leak information for
secondary authentication for all users [58].

This kind of functionality with secondary authentication has been omitted from the first
promotional version of this software due to it needing more research to be implemented
securely. Later there will be a way to recover a forgotten password.

**Sessions/Remember me**

Once a user has been authenticated the web page needs to keep track of the user so that it knows
not to authenticate them again. There are possible security issues with this if done incorrectly. In
this solution CodeIgniter’s built-in session handler is used to ensure that a secure and tested
scheme is used. This works by saving session data in a database table and a session ID on the
user’s computer. The ID is randomly generated and renewed every 5 minutes to prevent the
cookie from being stolen and used elsewhere [59].

Another similar feature is the commonly found “Remember me” checkbox, which if ticked when
logging in, ensures that the user stays logged in on that particular computer. This stores a
persistent cookie on the user’s computer which stays there for a long period of time, e.g. a week,
and if it is present when returning to the web page the user is automatically logged in.

In systems which store sensitive data these kinds of cookies represent a big security issue since
they can be stolen and used elsewhere within their valid time [60]. Due to this, such systems
usually omit the functionality, like PayPal which handles money and credit card information
[61]. The system described in this thesis handles data which might be considered sensitive, such
as customer contact details, and will therefore omit a remember me functionality.
DESIGN

The design of this system has its roots in the predecessor to this project. It contained a similar administrative part and a client. The client was however aimed towards private persons as opposed to the current system where it is aimed towards the contractor. For the old system the client-side design was worked out as a joint effort between the CEO of Byggwalle and the development team, the administrative-side was designed by the development team alone. Functional design was left mostly to the CEO of Byggwalle although he consulted with the development team to ensure that his ideas could be implemented in the given timeframe.

In figure 8 the administrative interface for the old system is seen, more specifically it is the view for managing articles and sorting them into categories. The thought behind this design was to give quick access to creating categories and articles on the same page. This idea was passed on to the new project and can be seen later in figure 16. There are some features in the old system which were reworked for the new project. The old system had a defined set of room types which a calculation could be made of; Bedroom, Kitchen, Bathroom, Living room, and Hallway. For the new system this was deemed to be limiting and was replaced with the ability to create custom templates to allow for creating calculations for any type of work.

The application for the previous project was made for iOS as opposed to Android. It was simpler than the new application since it was limited to making one calculation at a time, had a limited set of rooms/templates it could use, and did only use one set of measurements (Figure 9). The basic functionality is however identical with the current application where the measurements of a room are linked to a set of articles and a price quote is calculated from them. In the new version there is a possibility to work with several templates in a quote calculation and to have several surfaces to define a room and divide it into sections.
When designing the new system four people have been involved; the development team, the CEO of Byggwalle and an external designer specializing in mobile user interfaces. In the early stages of the project design sketches were made by the development team and Byggwalle’s CEO and later sent to the external designer to be improved and finalized with a style guide for each page. An early sketch can be seen in figure 10. All functional designs have been the work of the CEO of Byggwalle and the development team.
MOBILE APPLICATION

The new application design does not look much like the one from the previous project, this is because the external designer was involved from the beginning and he had not seen the old version. He received sketches from the CEO of Byggwalle which described what the application should be able to do, and refined them into the design which is seen in this section.

The first view a user encounters is the login screen. This will only happen if the application is started without a user having logged in earlier (Figure 11). It will not be displayed again until the user logs out.

When a user is logged in they are presented with the option of creating a new quote. In this view they can name the quote, add a customer to it, select templates (here: modules) to work from, and whether or not to use ROT reduction (Figure 12).
When the basic settings of the quote are done the next thing for the user to do is to define what surfaces are to be used for the calculation (Figure 13). Each surface consists of 2 or 3 measurements, depending on if it is symmetrical or not, and can contain a number of windows or doors. A surface can have one of the types: Roof, Floor, or Wall. This, together with naming of surfaces, helps the user to know which surfaces to add articles to later. It is also possible to make several identical surfaces by selecting how many instances there should be of it.

In the next view all articles for the selected template are displayed and sorted into categories to make it easier to find a specific item. Examples of categories would be painting, construction and electricity (Figure 14). Articles are assigned to one or several surfaces and can easily be toggled on and off with a checkbox. Some articles can be given an amount, for example if there should be 1 or 2 lamps. For each change the display in the upper right corner is updated with the total price making it easy for the user and the customer to ensure that they stay within their budget.
After all of the surfaces have had their articles added to them the user can show the customer a summary of the current template where the individual price per category is shown as well as any applicable ROT reduction and the total cost (Figure 15). The user can add a supplement if they think the price for a certain category is insufficient.
ADMINISTRATION SYSTEM
The administration system has been made in two versions; the first was designed by Byggwalle’s CEO, and the second by the same designer who did the android application design. When the CEO of Byggwalle designed the first version inspiration was taken from the previous project.

Add content

The view for editing templates in particular drew inspiration from the earlier project (Figure 16). In order to show everything on one page the templates were represented as a tree with the field in one layer, the category in the next and the articles in the last. To edit either of these it was selected and the relevant editing surface would appear to the right. Adding or removing articles from the template was done using the green and red buttons next to their entry in the tree. Just as in the previous project this was thought to give an overview of the template and allow for quick adding. That was however not the case and users found it confusing.
FIGURE 17: OLD VIEW FOR CHANGING ACCOUNT INFORMATION

Another issue with the first design was that it did not match the design used for the application; it did not feel like it was the same system. The application had a generally flat and simplistic design while the administrative system had more gradients and rounded corners. There was also confusion regarding the overuse of save buttons (Figure 17). To modify the saved information about a company it was necessary to save each section separately, which resulted in lost data due to users not saving it correctly. Examples of sections would include name and address, password, and logotype.

To work around this issue it was decided to involve the designer of the application also for the administration system. This resulted in a major overhaul of the system.
The new design for the administration system is more in line with the one for the application. It has the same color scheme, and has a more flat and simplistic layout. The way templates are presented has been completely reworked. Now instead of presenting the templates as trees they are simply lists of articles, where each article is assigned a field which is used for grouping in the application list and in the final quote (Figure 18). To make it easier to find a specific item a search box has been put above every list.

FIGURE 18: VIEW OF ARTICLES IN A TEMPLATE
Adding and editing article information has been moved to a separate interface which gives a cleaner look (Figure 19). The increased size of controls also allow for easier handling on smaller screens such as phones or tablets.
To enable collaboration between contractors and subcontractors it is possible to connect to another company and share templates (Figure 20). This makes it possible for a contractor to leave price estimates on all parts of a quote without waiting for the subcontractor to provide their current prices. When the subcontractor updates the prices in his template it is automatically updated for every company it is shared with.
IMPLEMENTATION

DEVELOPMENT METHODOLOGY
When developing software it is preferred to work according to a pre-defined method. This ensures consistency in the project as well as simplifying collaboration and increasing overall efficiency.

Due to the limited timeframe and small project group for this thesis it was decided to look into agile development methods, the approach utilized was Prototype implementation and evaluation. No specific agile technique has been followed but the process has been based on the agile manifesto [62] and influenced by SCRUM [63].

The development team has been working from a backlog from which they daily picked tasks to perform. These tasks were broken down into testable subtasks, for example the ‘Handle User’ task was broken down into the ‘Add User’, ‘Edit User’ and ‘Delete User’ tasks. Each of these were first implemented and tested individually and then as a whole.

All important decisions within the team have been made during face-to-face meetings. This also holds true for all collaboration with the client, who has a steady stream of input about improvements. The team evaluates each change request and informs the client how it affects the system, who then decides whether it should be implemented or not.

WORK FLOW

Using the system requires registering an account; this will only be open to selected companies for the time being. Initial set up includes company information, currency and unit choices, and setting up users. Before the system can be used it needs at least one template. These can be added either by downloading a standard template or by creating custom articles and put them into the company's own templates. These custom templates can be shared among partners to enable quicker collaboration with subcontractors. Once the templates are set up the application is ready to be used to compile price quotes (Figure 21).

FIGURE 21: THE WORK FLOW OF THE SYSTEM
TESTS
For this thesis both a qualitative and, in a smaller part, a quantitative study has been performed. The main point of inquiry was the customer reaction, usability and perceived increase in efficiency, secondary was the actual decrease in lead time.

To investigate the first point, interviews were performed with the employees of Byggwalle where they commented on their experiences when trying the system. This is mostly related to the usability of the system. Also, customers have been asked about their opinion regarding the system. This was to investigate whether or not it affects how the customer perceives the company and if it may influence them in their decision of who to hire for future projects.

The secondary point has been investigated by comparing the estimated lead time for delivering a finished price quote using the system with the current average of 3 weeks when not using it.

Usability
Within Byggwalle there are two persons who will be using the administration system, they were asked to fill in the questionnaire seen in Appendix C, both after using the old admin system and the new one. The questionnaire was divided into one section where they scored if they agree with a statement, and the second allowed for elaborating on their opinion. In table 2 their combined average score in the first section is presented. A score of 1 means that they do not agree at all and 5 means they fully agree.

<table>
<thead>
<tr>
<th>Question</th>
<th>New system</th>
<th>Old system</th>
</tr>
</thead>
<tbody>
<tr>
<td>It is easy to register a company.</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>It is easy to create and manage templates.</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>It is easy to manage users and company settings.</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>It is easy to create a quote in the application on site.</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>The customer has a positive attitude towards the system.</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

TABLE 2: SCORE FROM THE QUESTIONNAIRE.

The score for the new system is overall higher than for the old. The procedure of registering a company has become easier to understand, the same goes for creating and managing templates. Managing users and settings worked well in the old system but have been improved in the new version. The two final questions both refer to the mobile application which was the same for both administration systems; hence the score is the same. According to the score for these there is room for improvement in its usability, but the customer’s experience is still positive.

The main complaint regarding the old system was in regards to how templates were managed using a tree view (Figure 16). When filled with a lot of articles it became hard to overview and to find a specific article to edit. This was addressed in the new system (Figure 18).
Customer reaction
The system is not yet used in production but it has been presented to 10 potential customers who have given their opinion on it. From these discussions it was found that they find the system to be confidence-inspiring and have said that if the company can stick to their quote they would be more inclined to hire them for their next project. They have experienced the system to be innovative, transparent and honest. A common complaint is that the customers never receive an actual quote after their meeting with a company representative. The swift delivery this system provides is appreciated and the customers feel they can easier tailor the project to their budget.

For larger projects the customers say they would be inclined to wait a few weeks for other companies’ quotes, but for smaller ones they would be more inclined to accept on site, or later the same day, when presented with a quote using this system.

Lead time
According to Vahlfridsson, the regular procedure of compiling a price quote takes between 3-5 weeks before a conclusion is reached [7]. It consists of the following:

1. On site inspection (1,5h)
2. Draw blueprints and compile a work specification (1h)
3. Send specification to subcontractors
4. Calculate the company’s part of the work (2h)
5. Wait for price from the subcontractors (1-3 weeks)
6. Compile finished quote and send to customer (1h)

On average it takes 3 weeks to deliver a price quote, and another 1-2 weeks for the customer to respond.

When using the system presented in this thesis (already set-up with prices from the company and its subcontractors) this time can be greatly reduced. Assuming that instead of drawing blueprints and compiling a work specification the employee now enter the same information in the tablet to receive an estimated price, which already include the subcontractor’s prices. This is estimated to take approximately as long as step 1 and 2 combined, which thus cuts down the delivery time from 3 weeks to less than 3 hours.
DISCUSSION

This thesis had three main problems to solve regarding how to streamline the price quotes creation process in the construction industry. This section will discuss if these problems have been solved. Below the problems are reiterated and their solution discussed.

1: How can a price quote be delivered faster?
By implementing a mobile system price quotes can be compiled on site together with the customer. This minimizes travel time since the employee does not have to return to the office. Also the time it takes to find a specific article in a list is reduced with the customizable templates. When renovating a kitchen only kitchen related articles are listed in the application. The final delivery is also sped up since the customer sees the quote while it is compiled, and once finished it can be instantly emailed to the customer. On average the delivery used to take 3 weeks, with this system it is estimated to be reduced to 3 hours. This is mainly due to not having to wait for price estimates from subcontractors since their prices can be pre-loaded into the tablet application.

2: How can a price quote be made more transparent?
The customer can see the quote being compiled with the application. This way it becomes evident how changes affect the price and if the employee is adding any supplement charges. By being involved in every step of the creation of the quote the customer feels more confident that the price is not overestimated. Customers who have seen this system have experienced it as innovative, transparent and honest. They also think it will be easier to stay within their budget, which they find confidence-inspiring.

3: How can the system be designed to be usable for different kinds of construction businesses?
With the abstraction of articles which are divided in categories companies become flexible to organize their templates how they see fit. By working with surfaces instead of pre-defined room shapes it is possible to give an accurate quote for any environment. The system is to be seen as a tool to be used as see fit, not regarded as a strict process to follow.

During the course of this thesis Byggwalle has constantly hatched new ideas for improvements and new features. Seeing as this is an unexplored area for the construction business this is to be expected. It is often not until after something is seen implemented that flaws start to show. Also with the introduction of external parties such as a designer even more improvements have been suggested. This is good for the future development of this system.

What separates the product covered in this thesis from the existing solutions presented in the Background is the simplicity of the administration system; it does not require an IT-team to setup and use, and it has the ability to create quotes on site. All other solutions (even the ones with mobile applications) require the employees to return to the office to create their quotes.

Seeing as there could be no testing together with external companies it is hard to predict the future of this system. The initial response has however been good and there are several companies lined up for when testing starts. If the response from testing is positive and the product is maintained the future looks bright for the system. The move to mobile is long overdue for the construction industry and this system might just get it started.
**Future Work**

The Android application is designed only for tablet use; it cannot be used practically with a mobile phone, this need to be addressed by creating a layout for mobile phones. Also since the largest market share for tablets is held by Apple’s iPad it will be necessary to port the application to iOS.

For the web there is a need to add functionality for auditing the system in order to extract statistics from the construction industry. Before the system is publicly released it will be necessary to implement payment solutions in order to be able to make a profit of the system.

There are a few unexplored areas left for this system which would require further analyzing before it can be implemented. These are persistent login cookies and auditing solutions. The system also needs to be evaluated with proper end-to-end tests to discover possible bottlenecks. Regarding the cookies it is necessary to analyze if the possibility to remember login credentials is worth the risk it entails of handling persistent cookies. This would depend on how secret the information within the system would be considered. For the auditing it would be necessary to analyze what information is useful for statistical purposes, and how it would be extracted and stored efficiently.
REFERENCES


[15] J. Axelsson and J. Cederberg, "From webb to app - A study in how interaction design and graphic design is affected in the developing process from website to mobile
application?,” Södertörn University, Södertörn, 2011.


[38] E. Jansson and N. Jonasson, "Speedflirt for Android, a study of porting a mobile application and its effects on user experience," Umeå University, Umeå, 2011.


[58] A. Rabkin, "Personal knowledge questions for fallback authentication: Security questions in the era of Facebook," Symposium on Usable Privacy and Security (SOUPS), Pittsburgh,
PA, 2008.


# Appendix A – Quote for Harakersgatan 10

## Specification According to Annex 1

<table>
<thead>
<tr>
<th>Service</th>
<th>Unit</th>
<th>A-pris</th>
<th>Belopp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work</td>
<td></td>
<td>80 000</td>
<td></td>
</tr>
<tr>
<td>Material incl. parkett i köket</td>
<td></td>
<td>18 000</td>
<td></td>
</tr>
</tbody>
</table>

### Underentreprenörer (Budget Price)

<table>
<thead>
<tr>
<th>Service</th>
<th>A-pris</th>
</tr>
</thead>
<tbody>
<tr>
<td>VVS</td>
<td>10 000</td>
</tr>
<tr>
<td>EL</td>
<td>10 000</td>
</tr>
<tr>
<td>Vent</td>
<td>6 000</td>
</tr>
</tbody>
</table>

**Summa** 124 000

<table>
<thead>
<tr>
<th>Service</th>
<th>A-pris</th>
<th>Belopp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Moms</td>
<td>25 %</td>
<td>124 000</td>
</tr>
</tbody>
</table>

**TOTALT** 155 000

## ROT - Avdrag

<table>
<thead>
<tr>
<th>Service</th>
<th>A-pris</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bygg</td>
<td>76 000</td>
</tr>
<tr>
<td>VVS</td>
<td>8 000</td>
</tr>
<tr>
<td>EL</td>
<td>6 000</td>
</tr>
<tr>
<td>Vent</td>
<td>5 000</td>
</tr>
<tr>
<td>Moms</td>
<td>25 %</td>
</tr>
</tbody>
</table>

**TOTALT ROT-Avdrag** 50 % 118 750 -59 375

**Administrationsavgift ROT** 1 300 300

**SUMMA ATT BETALA** 95 925
## APPENDIX B – GENERATED PRICE QUOTE EXAMPLE

### OFFERT

**FRÅN**
Byggwalle AB
Filip Andersson

**TILL**
Anders Pantzar

**VÅR REFERENS**
Filip Andersson

**ER REFERENS**
Anders Pantzar

**OBJEKT**
Kök

**DATUM**
2013-10-18
Giltig 30 dagar

### Beräkning

<table>
<thead>
<tr>
<th>Kostnader</th>
<th>Summa</th>
</tr>
</thead>
<tbody>
<tr>
<td>Platsräkning</td>
<td>224 894</td>
</tr>
<tr>
<td>Golv</td>
<td>200</td>
</tr>
<tr>
<td>Övrigt</td>
<td>23 877</td>
</tr>
<tr>
<td>Måling</td>
<td>0</td>
</tr>
<tr>
<td>Bygg</td>
<td>0</td>
</tr>
<tr>
<td>Moms (25%)</td>
<td>62 318</td>
</tr>
<tr>
<td><strong>Summa</strong></td>
<td>249 271</td>
</tr>
</tbody>
</table>

### RÖT-AVDRAG

<table>
<thead>
<tr>
<th>Kostnader</th>
<th>Summa</th>
</tr>
</thead>
<tbody>
<tr>
<td>Platsräkning</td>
<td>28 249</td>
</tr>
<tr>
<td>Golv</td>
<td>0</td>
</tr>
<tr>
<td>Övrigt</td>
<td>6 192</td>
</tr>
<tr>
<td>Måling</td>
<td>0</td>
</tr>
<tr>
<td>Bygg</td>
<td>0</td>
</tr>
<tr>
<td>Moms (25%)</td>
<td>9 310</td>
</tr>
<tr>
<td><strong>Summa</strong></td>
<td>37 242</td>
</tr>
</tbody>
</table>

**Sum inkl.moms & RÖT-avdrag**
265 037 kr

### RESERVATIONER
Questionnaire

Take a stance for the following statements and respond with if you agree or not. A score of 1 means that you do not agree at all and 5 means you fully agree.

1. **It is easy to register a company.**
   
   Do not agree at all
   1 2 3 4 Fully agree 5

2. **It is easy to create and manage templates.**
   
   Do not agree at all
   1 2 3 4 Fully agree 5

3. **It is easy to manage users and company settings.**
   
   Do not agree at all
   1 2 3 4 Fully agree 5

4. **It is easy to create a quote in the application on site.**
   
   Do not agree at all
   1 2 3 4 Fully agree 5

5. **The customer has a positive attitude towards the system.**
   
   Do not agree at all
   1 2 3 4 Fully agree 5

Below you can elaborate on your experiences with the system.

6. **How much time do you estimate that you save per quote?**

   _____________________________

7. **Is something unclear in the system?**

   _____________________________

8. **Is there any feature that you miss in the system?**

   _____________________________