Exploring usability in the context of 1C:Accounting 8.2 for Latvia
The purpose of this Bachelor thesis is to explore usability in the context of the software 1C:Accounting 8.2 for Latvia. Recently, the usability of ERP systems and other enterprise software has become an important issue. 1C:Accounting is the most successful accounting software in the Commonwealth of Independent States and is meant for automating bookkeeping and tax accounting. However, users of this accounting software still encounter many problems during their work with the program.

The main research question of the thesis is to identify what forms usability for 1C:Accounting 8.2 for Latvia. This question was supported by two subquestions that aimed to discover the weak and strong areas in the current user experience with 1C:Accounting 8.2 for Latvia.

The empirical part of the thesis was conducted based on grounded theory method. Primary data was collected during interviews and think-aloud sessions with users of 1C:Accounting 8.2 for Latvia. The data was analyzed using coding procedures of grounded theory.

The findings of the research consist of emergent categories and substantive theory that uncovers 1C:Accounting 8.2 for Latvia role and user’s role in creating user-friendly interaction. One of the key usability components for 1C:Accounting 8.2 for Latvia would be to provide low error rate, clear system feedback and consistent user interface. Findings could be taken into consideration during future work of localization engineers and 1C:Accounting 8.2 for Latvia consultants.

Key words: usability, ERP, accounting, grounded theory
## Contents

1 INTRODUCTION 1
   1.1 Background 1
   1.2 1C:Accounting 2
   1.3 Reasons for selecting 1C:Accounting 8.2 for Latvia 3
   1.4 Research Questions, Objectives and Scope 4

2 RESEARCH FRAMEWORK 6

3 RESEARCH METHODOLOGY 8
   3.1 Research Approach 8
      3.1.1 Qualitative Research 8
      3.1.2 Inductive Research 9
      3.1.3 Case study 9
      3.1.4 Grounded Theory 10
   3.2 Data Collection 11
      3.2.1 Semi-Structured In-Depth Interviews 11
      3.2.2 Think-aloud method 12
   3.3 Data Analysis 13

4 DATA ANALYSIS 15
   4.1 Research Data 15
      4.1.1 Case company 15
      4.1.2 Employees 15
      4.1.3 Interviews 17
   4.2 Data Analysis 18
      4.2.1 Open coding and selective coding 18
      4.2.2 Theoretical coding 25
   4.3 Data Interpretation 27
      4.3.1 Categories 28
      4.3.2 Substantive theory 35

5 FINDINGS 41
   5.1 Answers to the research questions 41

6 DISCUSSION 44

REFERENCES 47

APPENDICES 51
LIST OF FIGURES

Figure 1. First version of selective codes 21
Figure 2. Final version of selective codes 22
Figure 3. Overall coding procedure 24
Figure 4. Relationship between open codes 26
Figure 5. Integrative diagram 27
Figure 6. Sequence of related codes 36
Figure 7. Spradley’s semantic relationships (Spradley 1979) 36
Figure 8. Example of relationships between categories 37
Figure 9. Compensation mechanism 37
Figure 10. Counterbalanced codes 38
Figure 11. Integrative diagram 40

LIST OF TABLES

Table 1. Example of open coding 19
Table 2. Example of selective codes 20
Table 3. Opposite categories 38

LIST OF APPENDICES

APPENDIX 1. List of initial open and selective codes
APPENDIX 2. Final categories, open and selective codes
1 INTRODUCTION

This thesis aims to study usability in context of the program 1C:Accounting 8.2 for Latvia. This chapter introduces background information on the phenomena of usability gaining more respect over the last few years in enterprise resource planning (ERP) systems field, provides information on 1C company and justifies the choice of studying 1C:Accounting 8.2 for Latvia in the current research. In the second part of the chapter the research question, the objectives and the scope of the study are introduced.

1.1 Background

In today’s competitive business environment, when technological changes are rapid and companies become more and more international, companies attempt to serve their customers faster and cheaper than their competitors. One of the most effective ways to achieve this is to have efficient and integrated information systems. Enterprise systems are one of the most complex and powerful information systems nowadays. Enterprise resource planning systems (ERP) are the largest and most complex enterprise systems. ERP system is exactly this kind of system, which integrates company’s operations and may act as a company-wide computing environment. Usage of an ERP system in a company might dramatically improve costs and operational efficiency. (Monk & Wagner 2006, 16-18.)

Some of the studies conducted during the recent years give an opportunity to assume that sometimes ERP systems are not as efficient and productivity boosting as they are supposed to be. Almost 60 percents of respondents of the IFS study are unsatisfied with the lack of intuitive usability and inflexibility of the software configuration in ERP systems or other enterprise software. It leads to decreased productivity and inability to react to the rapidly enough when new opportunities arise. (Mills & Rathmann 2011.)
Adding new features and functionality to ERP systems creates value for the end users, nonetheless, it increases complexity and becomes a source of confusion for many (McManus 2013). McManus (2013) states that ERP systems that focus on user productivity and incline to simplicity are becoming leaders of the market.

To conclude, now usability is seen as a commercially important software property that can improve competitive advantage of a specific program. Usability is especially important in a professional context and either may gain a product commercial success, or may result in productivity depreciation or even become a reason of an accident.

1.2 1C:Accounting

1C company was founded in 1991 in Moscow, Russia. It specializes in software development, distribution, publishing and support. In 1994 partner network that included dealers and 1C:Franchising teams was created. Currently 1C company is one of the biggest software developers and publishers in Russia. (1C Company 2015.)

In 1996 1C company has released the first version of 1C:Enterprise software product, which is intended for automation of enterprise routine activities: management and business accounting, CRM, HR management, etc. 1C:Enterprise consists of two parts: a platform, which is an integrated framework, and applied solutions that are created and run in this framework. The current platform version is 1C:Enterprise 8 and some of the most popular applied solutions are: 1C:Accounting, 1C:Trade Management, and 1C:Payroll&HR (1C Company 2015). At the beginning of 2015 1C franchises have had a leadership position in the Russian market by the amount of integrated ERP-solutions based on 1C:Enterprise platform (TAdviser 2015).

The most successful accounting software in the Commonwealth of Independent States is an applied solution 1C:Accounting 8, which is meant for automating bookkeeping and tax accounting in companies that
deal with any kind of trade business. It ensures compliance with all mandatory governmental reporting requirements for Russia and a list of other countries. Developer of 1C:Accounting 1C company states that this software suite solves almost every problem that arises in the accounting department. (1C:Enterprise 8 Configuration 2013, 11)

1.3 Reasons for selecting 1C:Accounting 8.2 for Latvia

Usability of 1C:Enterprise system has not been studied much outside of 1C company. The main 1C product that has received a lot of attention regarding usability issues is 1C:Bitrix, since it is a content management system and in the web development industry there already is understanding of the high importance of usability. But the trends are changing, and manufacturers are realizing the importance of usability of ERP systems and other enterprise software as well.

1C:Enterprise is proprietary software, and therefore the information concerning its development is not opened to public. It is known that 1C company pays attention to usability: performs usability testing, collects data through user observations and interviews. After that, the data is analyzed and decisions are made regarding changes in user interface, changes in interaction/user experience or other. (Bezborodov 2010; USABILITYLAB holds a workshop on usability for project-managers 2008.)

The author of the thesis did her practical training in a Latvian company that is a part of the 1C:Franchise network. During the practical training period the author has observed other employees of the company having different kind of user experience with the software: sometimes a task would be accomplished in no time and almost effortlessly, while another task would require lots of attempts and would cause errors. It was obvious that there are usability issues, but what exactly forms usability in context of the specific enterprise software was not clear.

Due to the fact that 1C:Accounting is the most widely-used applied solution and current study has time restrictions it was decided that the best
value would provide studying applied solution 1C:Accounting 8.2. This study explores 1C:Accounting 8.2 for Latvia (further in the study 1C:Accounting 8.2). It is a Latvian localization and it is different from the original Russian version, since Russian version is more up to date. Nonetheless, exploring the usability of the Latvian version brings value for the case company, since knowledge of the strengths and weaknesses of the software is created.

1.4 Research Questions, Objectives and Scope

The main objective of this study is to explore usability of 1C:Accounting 8.2 for the users of the software.

This study aims to understand what forms usability, and a positive user experience in the case of 1C:Accounting 8.2. The target audiences of this study are the members of 1C:Franchising network in Latvia and other users of 1C:Accounting 8.2 in Latvia.

Hence the following research question and subquestions are addressed in this study:

**Research question**

- What develops usability in the context of 1C:Accounting 8.2 software?

**Subquestions**

1. What makes 1C:Accounting 8.2 easy to use for the users of the software?
2. What makes 1C:Accounting 8.2 hard to use for the users of the software?

**Limitations**

This research focuses on exploring one version of the applied solution: 1C:Accounting 8.2. Usability of other applied solutions like 1C:Trade Management is not within the scope of this study.
Due to the researcher’s time constraints, this research will only focus on one case company.

The last limitation concerns relying on self-reported data from users in this research. Self-reported data may be a potential source of bias, such as selective memory or wrong interpretation of past situations.
2 RESEARCH FRAMEWORK

In this chapter the idea of usability is explained for the purpose of understanding the concept research question refers to. There exist many definitions of usability. Some of them are presented in this chapter in order to create a comprehension of what usability means.

International Standards Organisation (ISO) provides a following formal definition of usability: “Extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use” (ISO 9241-11:1998).

Jordan (2001, 5) gives an informal definition referring to usability issues as to something concerning easiness of use, “user-friendliness” of a product.

As stated by Demir, Karakaya & Tosun (2012, 7), usability term explains interaction between humans, their activities and tools used for accomplishing these activities. The aim of usability is to increase users’ satisfaction in their usage of any tool.

Steve Krug (2014, 9) presents his definition of usability by stating that if something is usable, when a person of average abilities and experience can find a way to use and accomplish desired task without it being more trouble than it is worth.

Another definition is given by Nielsen (1993, 26), where he states that usability includes several components, which are traditionally defined as five usability attributes:

**Learnability:** The system should be easy to learn, so that a user can quickly start accomplishing tasks with the system.

**Efficiency:** The system should be efficient to use, so that once a user has understood how to use the system, he can reach a high level of productivity.
**Memorability:** The system should be easy to remember, so that an average user should be able to return to work with the system after some break, without having to learn how to use it all over again.

**Errors:** The system should have a low error rate, so that if a user makes an error, he can easily recover from it. Catastrophic errors are not tolerable.

**Satisfaction:** The system should be pleasant to use, so that users feel satisfied when using it and like it.

The definitions mentioned above demonstrate some of the different approaches of defining usability.
3 RESEARCH METHODOLOGY

There exist various research approaches and methods, each of them has their own benefits and shortcuts. This chapter introduces research methods used in the current research, identifies them as suitable in this case, and describes research process using these methods.

3.1 Research Approach

3.1.1 Qualitative Research

Qualitative research seeks to expand understanding of human behavior by investigating the underlying reasons of people’s actions and ways of thinking. It describes some phenomena within its context, deeply and with an emphasis on details. Qualitative research is used to investigate such things as attitudes, attributes, features, qualities, feelings, or the particulars of individual events and occurrences. (Patton 2002, 14; Cottrell 2014, 93.)

Qualitative research is based on field observations that are analyzed without statistics. There exist three kinds of qualitative data: interviews, observations, and documents. (Patton 2002, 4.) Qualitative research often begins with a single case and this case is often chosen because of their convenience or interest (Silverman 2014, 6).

In this study qualitative research methods are used, because this thesis focuses on human behavior, users’ feelings and actions associated with software program usage, software program’s qualities and features that raise specific emotions, e.g. satisfaction or displeasure after using 1C:Accounting 8.2 program. This study concentrates on one case company and studies it in depth.
3.1.2 Inductive Research

Inductive research starts with the researcher conducting empirical observations of the real world. After that the researcher formulates theories to explain empirical observations or theories might be developed based on observed patterns, discovered through real life observations. (Crowther & Lancaster 2009, 31.)

On the other hand, a given theory does not necessarily only summarize observed patterns or relationships. A theory might express the theorist’s guesses about unobserved relationships, and the theorist may invent new concepts and make suggestions about new links between concepts. (Dooley 2001, 64.)

Inductive research is a very flexible research approach, and this is one of the reasons why it is particularly well-suited for studying human behavior. It allows a problem to be studied in different possible ways, and it also allows alternative explanations or interpretations of the real-life observed material. (Crowther & Lancaster 2009, 31.)

In this study the inductive approach choice is grounded on the study objectives and goals. This thesis studies easiness of use of the software program, its usability, and its users’ attitudes and feelings concerning their experience with the 1C:Accounting 8.2 program and some other issues. One of the main advantages is the inductive approach’s nature that prescribes to build a theory based on field observations. The inductive approach helps to answer questions concerning human behavior by building a theory that explains the features of human-technology interaction.

3.1.3 Case study

The case study approach is appropriate when in-depth understanding of particular situations is a goal of the study. A case study helps the researcher to focus on the interrelationships between the factors, such as people and technology. (Fisher 2010, 69.)
The case study approach is an appropriate method when:

- The researcher has little or no control over events;
- The study focuses on the contemporary phenomenon within a real-life context.

The case study approach is particularly used for understanding the real-life phenomenon in depth, when the boundaries between phenomenon and context are not clearly evident. (Yin 2009, 2, 18.)

This study is done applying the case study method, because conducting a case study is the best choice for the current thesis. First of all, a case study is meant for studying interrelationships between different factors, such as people and technology, and this is an exact focus of this thesis. Second, in this study the phenomenon and context boundaries are not evident, and it could be hard to divide them from each other. This is not a problem for a case study, because it concentrates on studying a phenomenon within its context.

3.1.4 Grounded Theory

One of the creators of grounded theory, A. Strauss, describes grounded theory as a theory that was derived from systematically gathered and analyzed through the research process data (Strauss 1998, 12). In grounded theory research concepts are obtained through the analysis of collected data, not decided before the research process was started (Corbin & Strauss 2015, 7).

The grounded theory method is used to build theory from existing data. Urquhart (2013, 11) sees this as one of the grounded theory’s most innovative and exciting aspects, since it encourages scholarship and innovation in all disciplines where grounded theory is applied.

The most frequently used data collection types in grounded theory are interviews and observations; however, nearly any type of written, observed or recorded data can be used (Corbin & Strauss 2015, 7).
The grounded theory research method was chosen for this study, for that reason it builds a theory from collected information and gives the researcher an opportunity to discover something new, generate new ideas from the obtained data. The grounded theory method is the most suitable method to find answers to the research questions and accomplish the current research’s goal.

3.2 Data Collection

3.2.1 Semi-Structured In-Depth Interviews

Semi-structured in-depth interviews were selected as one of the data collection methods because they generate rich, qualitative information about individuals’ experiences, attitudes or perspectives. During an interview it is possible to immediately follow up, ask questions, and ask for additional information or clarification in order to elicit the desired information. (Cottrell 2014, 157.)

At the moment case company consists of 5 permanent workers. It was decided to conduct three interviews with employees, who are using 1C:Accounting 8.2 in their job or are currently in the process of learning how to work with it.

Perspective interviewees were invited for an interview by email. They have received a letter with an explanation of the goals of the current research, insight of the impact they would make by participating in it, and a rough description of the topics that would be discussed. Due to that, interviewees could think about their experience related to the mentioned topics in advance.

Interviews were conducted face-to-face in the case company’s office in Riga, Latvia. Interviews were conducted in Russian, since it is the native language of interviewees and the researcher. Interviews were taped using “Easy Voice Recorder” Android application for later data analysis.
Interview questions were based on usability definitions from the “Research framework” chapter and have covered the following areas:

- Personal information, such as age, previous work experience, accounting programs he or she has used in the past, amount of time user has been working with 1C:Accounting 8.2;
- Starting point of using 1C:Accounting 8.2 for a daily working tasks: amount of time it took to learn how to use it, problems that arose at the beginning, ways these problems were solved, biggest challenges in acquiring the new technology and reasons for these challenges, areas that were easy to learn and reasons for that, skills and previous experience that have helped during the education period and, from today’s perspective, the ways for education period could have been easier;
- Current point in working with the software: evaluation of easiness/uneasiness in the daily work, key factors for understanding the software better, current difficulties and proposals on the program's change for becoming easier, emotions that program causes, issues of memorability.

3.2.2 Think-aloud method

Think-aloud techniques in their current form have emerged from cognitive psychology (Charters 2003, 69). Currently think-aloud data collection methods are widely used in usability testing, psychology, and in a host of other social sciences (van Somersen, Barnard & Sandberg 1994, 32).

Think-aloud is the method to explore what person thinks and how his or her thinking process proceeds while solving a problem. To achieve that, person is asked to talk aloud everything that comes into his or her mind during the problem-solving process. (Beaton, Nicholson, Halliday & Thomas 2015.)

Think-aloud method collects data from different sources. First of all, the researcher is able to observe how and what user is doing. Second, user is uttering his or her cognitive process, which generates a material for later analysis. (Beaton, Nicholson, Halliday & Thomas 2015.)
Nielsen (2012) appreciates think-aloud method and describes it as ‘#1 usability tool’. He has discovered the following thinking-aloud method’s advantages: it is cheap, flexible, robust, convincing and easy to learn. On the other hand, think-aloud techniques put users in unnatural situations, do not lead to detailed statistics and the researcher could unintentionally influence user’s monologue by asking questions or prompting user. (Nielsen 2012.)

Think-aloud method may be used in two distinct scenarios:

1. The researcher gives user a specific task,
2. User is free to choose task to be undertaken. (Beaton, Nicholson, Halliday & Thomas 2015.)

In the current research the latter scenario was applied. It was decided to let users choose their tasks by themselves, because users who have participated in the study had different previous experience and education, as well as different level of proficiency in working with 1C:Accounting 8.2.

Overall, think-aloud method was selected as one of the data collection methods in this research because it shows different ways of thinking and problem solving that users employ when dealing with 1C:Accounting 8.2 program on a daily basis. In addition, think-aloud method generates data that can be analysed using grounded theory method, which is very important in the framework of this research.

3.3 Data Analysis

In the Grounded Theory Method coding procedures are used for data interpretation and analysis. As Urquhart (2013, 35) defines it: “coding is the term used for attaching conceptual labels to data.” The process of data coding has two approaches: Glaserian and Straussian. Urquhart (2013, 23) sees Glaserian standard of coding procedure as a simpler one. Glaserian version of grounded theory method consists of three stages: open coding, selective coding and theoretical coding (Urquhart 2013, 24).
Open coding — is an initial step in coding and is deliberately ‘open’ in order to see what data might be telling. It is recommended to do open coding line by line, as it helps to free the researcher of his/her preconceptions. (Urquhart 2013, 24.)

Selective coding — selective coding starts, when definite themes are emerging and new open codes are not found anymore. In selective coding the open codes are organised into selective codes that will contribute to the core categories of research. (Urquhart 2013, 49.)

Theoretical coding — is a phase when relationships between codes are established. It is a very important stage, since every theory consists not only of constructs, but also of relationships between those constructs. (Urquhart 2013, 50.)
4  DATA ANALYSIS

4.1  Research Data

4.1.1  Case company

The case company is situated in one of the Baltic countries, Latvia. The company was founded in 2013 and in 2014 turnover of the company was 130 000 EUR.

The main scope of activity of the company is distribution of localized versions of 1C software products in Latvia. That means that above mentioned 1C programs are sold and services including installation, basic and advanced education, private consulting, and additional programming for the program to meet specific customer’s needs are offered.

4.1.2  Employees

The company has five permanent employees, four outsourced positions and one intern position. All employees have very different background and previous work experience.

Owner of the company and managing director is about 40 years old, and has about 10 years of experience with 1C programs. His main company has been using 1C:Accounting and 1C:Trade Management for several years before he made a decision to found a new company and cut expenses on service and additional programming for 1C programs used in his main business. Currently he governs the company, manages it, and does decision-making based on reports.

Project manager is about 40 years old. Before starting her work in the case company, she has been working with Latvian accounting program Tilde for about 10 years, and has about 7 years of experience in a project manager’s position. She got to know 1C:Accounting program before starting the current position and has about 3 years of experience with this
program. She understands what program can do, what it cannot do, in which cases it is possible to adjust program to clients’ needs better by using additional programming. She is coordinating every customer, starting with program purchase and then providing necessary education and maintenance.

One of the customer service consultants’ is a recently graduated person, who has obtained Bachelor of Science in Telecommunications degree. She has a previous work experience working in a shop as a cashier. Currently she has about 2 years of experience with 1C programs and executes a wide range of tasks, from users’ education to writing product requirements documents.

Another customer service consultant is in her early 30’s. She has Economics degree and was working as a warehouse manager in the past. She also had her very own company, where she has been doing bookkeeping all by herself. This valuable experience has helped her in the process of learning how to use 1C:Accounting 8.2 software. She has about 1 year of practice working with 1C programs and is educating beginner users and resolving standard questions.

1C programmer is in his mid-thirties, and has obtained Bachelor of Science in Information Technology in the past. He has spent the most part of his working life working with 1C programs and writing handlings/localizing configurations on 1C programming language. He is able to do nearly anything with 1C programs.

Intern position is taken by a person in the early 30s, who has an urge of career change. Her first profession is a bartender, nevertheless she has done an internship as an accountant’s assistant in the past. After that she has taken a break and now has returned to accounting field. She comes to the office 1-3 times per week and studies 1C programs by following manuals for the beginners and performing relatively simple tasks to solidify skills and knowledge. She has been doing so for about 2 months.
Outsourced positions include: accountant, translator, graphical designer for website and social media, technician.

For this study three persons with very different background and amount of experience were chosen: owner of the company with his profound acquaintance with 1C programs, customer service consultant, who has 2 years of experience with the programs, and intern, as she has the less experience with the programs and her first career is non-academic.

4.1.3 Interviews

Three interviews and Think-Aloud sessions have happened in consecutive days. Respondents were asked similar questions based on the list of topics. Follow-up questions were asked if an interviewee has mentioned something that has a hint of being valuable information.

Respondents have chosen meeting times that would suit their schedules. Yet some of them were open to the dialogue, while others gave more restrained answers.

Numerical data considering length of the interviews and Think-Aloud sessions is as follows:

- Business owner: interview length — 12,5 minutes, Think-Aloud session length — 8 minutes. It has produced 4 pages of interview transcript and 1 page of Think-Aloud session transcript.
- Customer Service Consultant: interview length — 26 minutes, Think-Aloud session length — 5,5 minutes. It has produced 5 pages of interview transcript and 1 page of Think-Aloud session transcript.
- Intern: interview length — 11 minutes, Think-Aloud session length — 6 minutes. It has produced 3,5 pages of interview transcript and 0,5 pages of Think-Aloud session transcript.

The amount of asked questions was very similar, so the difference in interview lengths was created by difference in answers of the respondents. As it could be seen from numerical data, Customer Service Consultant has had lots to share, her own experience with the 1C programs and her experience educating accountants from other companies gave us a fruitful
field for a discussion. On the contrary, intern has not shared much, mostly for the reason of lack of experience with the program in hand.

4.2 Data Analysis

4.2.1 Open coding and selective coding

As was already mentioned in the previous chapter, in this study grounded theory method is used for inductive data analysis and substantive theory generation. From the literature on grounded theory method it becomes evident that grounded theory is one of that kind of research approaches, where different stages are interconnected. For example, data collection happens at the same time as data analysis, open and selective coding stages are not linear, as some discoveries made during selective coding stage might suggest researcher to return to open coding stage, and theoretical memos that lead to emergence of theoretical code and are often used in a final report are mostly written during open and selective coding stages (Urquhart 2013, 8; Charmaz 2006, 58; Glaser 2004).

As for this study, interconnection has showed up in the situation that while open coding was still in progress, some categories started to appear, which means that selective coding was going simultaneously with open coding. When open codes were transferred from paper to electronic form, most of them were already forming some categories.

Nonetheless, procedure track was followed and after the data was collected and recorded through interviews and Think-Aloud sessions these interviews and Think-Aloud sessions were transcribed to a written form. After that, the data was printed and line-by-line open coding was applied.

Open coding procedure was based on the recommendations from Urquhart (2013) book, where she went through all the coding procedure using Glaser’s grounded theory method version. The majority of the first open codes were descriptive and more often reflected the data than analysed it. For future theory building it is more beneficial to have
analytical codes rather than descriptive ones, but at the same time often descriptive codes are necessary as a first step towards analytical codes (Urquhart 2013, 37).

The open coding procedure was when applied for the second time, giving names to the chunks of data. Owing to the first stage with descriptive codes, it was easier to look deeper and give more analytical names at the second time. The examples of the first and second open code versions can be seen from the Table 1. In the Table 1 fragments from interview transcript, initial open codes and final open codes are listed. As it can be seen, final open code often included less information than initial one, since context of the issue on numerous occasions became a reason for subcategory grouping. Subcategories are examined in more details in selective coding stage description below.

<table>
<thead>
<tr>
<th>Fragment from interview transcript</th>
<th>Initial descriptive open code</th>
<th>Final open code (“name” for a fragment)</th>
</tr>
</thead>
<tbody>
<tr>
<td>‘User interface looks different indeed and operating procedures, accounting transactions had to be filled in there, every time one needed to fill in accounting transactions, but here there is a standard operation and it is easier.’</td>
<td>In “1C:Accounting 8” there are “typical operations” there user doesn’t need to indicate accounting transactions’ numbers</td>
<td>informationSearchSkills</td>
</tr>
<tr>
<td>‘When I actually had had to fix some mistakes, I started to search in the Internet and ask colleagues and together we</td>
<td>Information search skills</td>
<td>informationSearchSkills</td>
</tr>
</tbody>
</table>
could find where it is possible to fix it, double-check that it went right and find out what was the reason for a mistake.'

My previous experience [using another accounting software] has helped me a lot.’

‘Usually people have one main learning difficulty: they don’t want to study.’

‘To run one operation in the beginning in an empty [information] base, very much [information] should be filled in at that exact moment.’

Possibility to ask someone
Experience with other accounting software
Lack of motivation during education
Before active work starts many things need to be filled in

Table 1. Example of open coding

The full table of initial open and selective codes can be found from Appendix 1.

The second step was applying selective coding, which means categorizing open codes into bigger groups by having something in common or describing the same subject from different perspectives.

<table>
<thead>
<tr>
<th>Selective codes</th>
<th>Open codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software based sources of</td>
<td>possibilityToCheck Earlier Examples</td>
</tr>
<tr>
<td></td>
<td>possibilityToAlter Earlier Documents</td>
</tr>
<tr>
<td>facilitation</td>
<td>checkingAccountingTransactionsViaReport</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>Errors</td>
<td>userCaused</td>
</tr>
<tr>
<td></td>
<td>userForgotToFillInFields</td>
</tr>
<tr>
<td></td>
<td>fromAdditionalDevelopment-very rare</td>
</tr>
<tr>
<td>Learning problems</td>
<td>accountingTransactionsInvisibility</td>
</tr>
<tr>
<td></td>
<td>atFirstNotIntuitivelyUnderstandable</td>
</tr>
<tr>
<td>Software based sources of problems</td>
<td>changeOfStandardProcedureTrack</td>
</tr>
<tr>
<td></td>
<td>notAppropriateProcedureTrack</td>
</tr>
<tr>
<td></td>
<td>tooManyDocumentsForOneOperation</td>
</tr>
<tr>
<td>Change</td>
<td>forDevelopers-similarInterfaceForDocuments</td>
</tr>
<tr>
<td></td>
<td>forBeginners-followInstructions</td>
</tr>
<tr>
<td></td>
<td>forAdvanced-doNotBeAfraidToTry</td>
</tr>
</tbody>
</table>

Table 2. Example of selective codes

Operations that are hard to learn in 1C:Accounting 8

Barriers

- Errors

Operations that are easy to learn in 1C:Accounting 8

Facilitation

- Experience
  - Experience with accounting programs
- Programming skills

Advices on how to make some things easier

Figure 1. First version of selective codes
Figure 2. Final version of selective codes

Selective code names were given based on common context of open codes that describes specific features of the software. For example, interviewees have mentioned that previous experience with accounting software, either with previous version of the software under study or another accounting software, made the learning process easier. Other
previous experience, knowledge and skills were mentioned in context of facilitating work with the program. All of this has formed “Knowledge and experience” subcategory, which later appeared to be a part of the larger “Sources of facilitation” category. The list of the initial selective codes can be seen in the Figure 1 (page 21).

At first some general categories have emerged, such as “Barriers”, “Facilitation”, “Operations that are hard to learn in “1C:Accounting 8”, “Advices on how to make some things easier” and “Operations that are easy to learn in “1C:Accounting 8” as well as some categories that later appeared to be properties of these general categories: “Experience”, “Errors”, “Programming skills”, “Experience with accounting programs”.

After revisiting open coding stage, selective codes needed to be revised as well. As a result some new selective codes and even categories have emerged, and other changes have happened, see Figure 2 (page 22).
An overview of the whole open and selective coding processes is illustrated in Figure 3 (page 24), starting with chunks of data from interviews or Think-Aloud sessions and ending with categories.

Figure 3. Overall coding representation

<table>
<thead>
<tr>
<th>Interview fragment</th>
<th>Open code</th>
<th>Subcategory</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>But knowing &quot;hot keys&quot; you can copy elements much faster taking already existing</td>
<td>hotKeysForQuickerInput</td>
<td>Functions and features</td>
<td>User-friendly</td>
</tr>
<tr>
<td>elements as a basis, copying them and modifying it with something new you are</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>just inputting.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>It would be much harder to do the same amount of work without this software.</td>
<td>makesAccountantWorkEasier</td>
<td>Software in general</td>
<td></td>
</tr>
<tr>
<td>If I judge by time, now inputting the same amount of documents as in the beginning</td>
<td>timeSaving</td>
<td>Knowledge and experience</td>
<td>Sources of facilitation</td>
</tr>
<tr>
<td>of my work takes 3 times less time, because now I know for sure what is necessary</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>to fill in and what fields could be left blank.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Well, this concept of bookkeeping this is what previous experience gave me. There</td>
<td>anotherAccountingSoftware</td>
<td></td>
<td></td>
</tr>
<tr>
<td>is no horror, like here are documents, they have to be set into the program and</td>
<td>knowledge-accounting</td>
<td></td>
<td></td>
</tr>
<tr>
<td>then registered.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4.2.2 Theoretical coding

One of the first noticeable relationships between categories comes from open coding stage. Due to the follow-up questions in the interviews, interviewees have often described their user experience in full: have mentioned problem or positive moment in user experience, what caused this problem or positive experience and have proposed a solution for that or described the benefits in case of a positive feature of software.

In Figure 4 some open codes that form this kind of relationship are represented. In the first arrow there is an open code “payrollAccounting”, which belongs to the subcategory “Specific operations” of “Problems” category. This operation was proven to be uneasy to handle by all three participants, even though they had different level of experience with the program. Next three arrows tell possible reasons why this operation is hard for users: one of the reasons is that in the previous version of the program, 1C:Accounting 7.7, procedure track was different, it was simpler than in the current version, and due to that users have to spend additional effort re-learning procedure track and adapting themselves to a more complex version. The reason why in the current version procedure track for payroll accounting is complicated is that this procedure track is adjusted for big companies with a large amount of employees. But since there is just a handful of big companies in Latvia and not all of them are using 1C:Accounting 8.2 for their accounting needs, the greater part of 1C:Accounting 8.2 users in Latvia are small and medium-size companies. This makes current procedure track for payroll accounting inappropriate for Latvian localization of the program. It is inappropriate since four documents have to be created instead of one to accomplish payroll accounting for one employee. Possible solution for this would be keeping all related operations in one document, as it would save users time and effort on producing these documents.

Thinking about the codes like this has helped the abstraction process. Although open codes lie in different categories, they are still related to each other.
This was the first emergence of theoretical code. It comes from the data, has ready substantive meaning and is explicit, as most of the first substantively emergent theoretical codes (Glaser 2005, 31).

![Diagram showing relationships between open codes](image)

**Figure 4. Relationship between open codes**

It is evident that the data is richer than just causes, problems or positive experiences, and sequences. Categories, their properties and interconnections suggest that there is a room for another theoretical code. In order to make it explicit following steps were applied.

First, diagram representing categories and their possible relations to each other was created. Figure 5 shows it below.
Figure 5. Integrative diagram

The integrative diagram includes all emergent categories, however not all subcategories are included. Only the subcategories that seemed critical for the research objectives are included. Relationship “is a part of” is based on selective codes, grouping of subcategories into larger categories. Relationships “is a reason for”, “is a cause of” and “is a way to” are based on cases similar to the one mentioned before, where this relationship model is explicit through an example with open codes.

After the integrative diagram was created and some memos about relationships between the categories were written down, one of the memos’ has shown an idea concerning the relationship between all selective codes, what do they represent altogether and what does it all mean.

4.3 Data Interpretation

In order to describe findings made during this study, findings from open and selective codes, and substantive theory are presented in this subchapter.
4.3.1 Categories

“User-friendly” includes open codes that describe positive user experience and features of the program that make work with the program easy for users. Category “User-friendly” consists of “Specific operations”, “Functions and features” and “Software in general”.

Selective code “Specific operations” contains open codes describing specific operations in the program that are easy to use. In particular, there are: operationsWithAccountsReceivable, operationsWithAccountsPayable, sellingGoods.

Equally or even more important is selective code “Functions and features” that includes characteristics of the software that create a positive user experience. As two interviewees have mentioned in similar context, the program is easy to start working with. One of the interviewees’ says: ‘I have received explanation, was shown how to do it and I could already perform basic operations’ (user 3, page 2 of interview transcript). This has resulted in the open code quickStart.

One of the most important challenges after learning how to work with the program is to improve work efficiency. There are three different open codes that show an opportunity for a faster data input: copyPasteForQuickInput saves time for inputting similar documents, hotKeysForQuickerInput help to navigate inside a document faster than using a mouse and, finally, companyInformationFromLursoft helps to fill in other companies information in the database quicker and prevents mistypes.

There are two open codes concerning features that have a relation to memorability: fieldValidationHelpsToRememberToFillInNecessaryFields and accountingTransactionsAreDefinedByDefaultInStandardDocuments, the latter means that user does not have to remember to fill them in every time he or she is creating a new document.
Open codes in selective code “Software in general” are more about users’ attitude towards the whole system, not some separate features: thisSoftwareInComparisonWithAnotherIsEasier, thisSoftwareMakesAccountantWorkEasier and softwareHasAllRequiredFunctionsForAccountant.

“Sources of facilitation”

As the title of this category states, here are grouped the codes that successively lead to ease of use, feeling of the program being user-friendly. There are three selective codes in this category: “Software based sources of facilitation”, “Knowledge and experience” and “Sources of learning facilitation”. Below these selective codes are examined in a more detailed manner.

Selective code “Software based sources of facilitation” includes possibilityToCheckEarlierExamples that improves memorability of the program, possibilityToAlterEarlierDocuments is a very important feature, as it plays the role of a “safety bag” for users in case they input incorrect data or just need to change earlier documents for any reasons. checkingAccountingTransactionsViaReport helps user to understand if the document was correctly registered in the system.

“Knowledge and experience” is a large selective code that unites types of knowledge, skills and experience that help users to work with the program more efficiently. There are listed some skills and knowledge areas that are obviously necessary to work with accounting program: skills-computer and knowledge-accounting. Some other skills and knowledge are facilitating work with the program: knowledge-economics, skills-informationSearch and skills-programming. Certainly, how skills-programming positively affects user experience seems to be an interesting topic, for this reason interviewee was asked to tell more upon the subject. It was gathered that skills-programming serve for adjustingConfigurationSettings, findingHiddenFunctionalityInUserInterface, and it even increaseSoftwareLayoutUnderstanding. Skills-programming is not a
necessary skill for mastering the program from user’s perspective, nonetheless it is beneficial and improves understanding of the program.

Experience concerning knowledge of principles by which software or accounting work also plays an important role in improving usability from a user’s point of view. understandingSoftwareInnerLayout will help to solve possible problems with the software and understandingBasicAlgorithmOfAllProcedureTracks is a crucial knowledge for every accountant. It states: ‘First income, then expenditure’.

It was noticed by one of the interviewees that to learn all the basics of bookkeeping in the program usually takes about

\textit{timeFrameFromBeginnerToExperienced}–3 months of user experience withThisSoftware. Users have compared amount of time it took to accomplish a certain task in the beginning and after gaining a certain experience with the program. Logically, after gaining experience it was much less then before, so experience leads to \textit{timeSaving}. During the first few months users learn frequentlyRepeatedProcedureTracks and then, as was mentioned by all participants of the study, learnedProcedureTracks become easy to accomplish. In addition, after first few months users knowHowToSolveProblems and knowHowToAvoidMistakes, as well as learnedToUnderstandWhatErrorMessagesMean, if error message does not obviously state reason of problem. Experience with previousSoftwareVersion as well as with anotherAccountingSoftware is valuable for working with the program, but even more valuable it is in the period of learning how to work with the software.

As a result, open codes \textit{previousSoftwareVersion} and anotherAccountingSoftware are found in the selective code “Sources of learning facilitation” too. Other open codes that belong to “Sources of learning facilitation” include externalMotivation by someone, who teaches how to use the new program, and this person also does explanationOfAccountingTransactionsMovements, explanationOfSoftwareInnerLayout, explanationOfProcedureTracks, and, finally, and most obviously, this person provides
personToPersonInteraction, which means that user can ask any questions or address any struggles to the educator.

What can be said for sure about open codes in “Problems”, is that one of them was mentioned 4 times during the interviews and is certainly considered as a problem by users of the program. It is operation payrollAccounting. In addition to the interview mentions, there was one Think-Aloud session where user was making payroll accounting and encountered an error, which was not easy to resolve.

In order to understand “Problems”, open and selective codes that form it should be presented. “Problems” include “Errors”, “Specific operations”, “Software in general” and “Learning problems”. The first selective code to present is “Errors”.

It is interesting to notice about selective code “Error”, that a third part of all open codes in this selective code includes users as a reason of error appearance. To be more precise, there are: userCaused, userForgotToFillInFields, and userCaused-very frequent.

The reason for it is that situation when user forgets to fill in fields while inputting a new document, or inputs wrong type of data is a trigger for an error message. And because this is a very frequent beginner situation, it can be said that “user has caused an error”.

Even though during the interviews there was gathered positive information about errors, e.g. Occurance-rare, fromAdditionalDevelopment-very rare, and systemErrors-extremely rare, 2 out of 3 Think-Aloud sessions have encountered mistakes during the sessions. These mistakes were resolved during the sessions, but the users who have encountered them were experienced with the program and errors’ causes were not easy to find and resolve.

The second selective code that is part of “Problems” is “Specific operations”. It includes just two open codes, where one is already
mentioned \textit{payrollAccounting} and the other one is \textit{performingStockTakeOfPropertyAndPlantAndEquipment}.

The third selective code “Learning problems” lists problems that are more crucial for beginner users. These are \textit{accountingTransactionsInvisibility} and \textit{atFirstNotIntuitivelyUnderstandable}. In both cases users have stated that gaining experience in working with the program has helped them to overcome these problems.

The last selective code “Software in general” is interesting, since it touches upon diverse features of the software. There is a mention of problem with an empty database, when a new user has to input quite a few data before active work starts — \textit{muchToFillInTheBeginning}, there is also \textit{specificLanguage} issue, when software sometimes is using a language not used by real users, then there is \textit{lackOfUnderstanding:WhatTypeOfInputFieldsRequire} which is self-explanatory, and, finally, a situation that was not yet encountered by the user in interaction with the program under study, but there is a firm belief that it would be an obstacle for working with the program: \textit{forcingCompletelyNewUserInterface}.

\textbf{“Sources of problems”} is a category that gives a more abstract image of what mechanisms lead to low usability in 1C:Accounting 8.2 software. “Sources of problems” includes three selective codes that roughly describe origins of users’ bad experiences. These origins are: “Lack of knowledge”, “Software based sources of problems” and “Sources of learning problems”. Now these selective codes are presented in turn.

“Lack of knowledge” is probably the easiest selective code to describe and prove to be a source of various problems in context of working with the program. ‘The problem is with chart of accounts, I can not always select the right one’ (user 3, page 2 of interview transcript) says one of the users and it becomes evident that in order to have an efficient interaction with the accounting program, \textit{lackOfKnowledge:Accounting} has to be diminished. This is the most important vacancy in user’s knowledge that
has to eliminated to work successfully with the program. Other vacancies in users’ knowledge that has risen problems in the work are: lackOfKnowledge:SoftwareFeatures and lackOfKnowledge:HowToCheckOperationValidity. The two last areas of knowledge have a direct relationship with experience with working with the program and can not be obtained prior to a work start. It could be considered to pay special attention to these areas in the beginning of working with the program.

“Software based sources of problems”: in this selective code is gathered users’ frustration about working with the program. Here are listed the origins for bad user experience, and what is really important, is that some of the open codes are already quite abstractive and give reader a hint about how the similar situation can become a problem when talking about another program.

In selective code “Software based sources of problems” there are mentioned: changeOfStandardProcedureTrack, since it is not easy to adapt to a new sequence actions, especially if a former seemed to be more convenient; notAppropriateProcedureTrack was mentioned in a context where a procedure track was not properly localized for smaller companies in Latvia in comparison with companies in Russia; tooManyDocumentsForOneOperation is a problem, as it requires additional time and effort to produce these documents, in addition it increases a risk of user mistake to be accidentally made by mistyping, etc.; memorability-rememberingProcedureTracks this was especially noted by a beginner user, for whom procedure tracks are not automatic yet.

The last selective code that is part of “Sources of problems” is “Sources of learning problems”. Here are gathered open codes concerning obstacles that mainly new users encounter. As two interviewees say, lackOfClearness leads to disappointment while learning to use the program, as there are no clear signs that previous operation went right and how it has affected the whole system. Another open code, mentioned two times, is memorability-rememberingProcedureTracks. At first, when
everything is new for a user and many different documents have to be filled in for the first time, it is a problem to remember sequence of actions to accomplish different tasks. The last open code in this grouping is lackOfMotivationToLearn. Motivation to learn can be missing, because in companies managerial group is making a decision to change from old ways of bookkeeping to automatic ones like with the program “1C:Accounting”. If the employees do not see the benefits from learning how to work with the program, this situation can result in a work sabotage. Therefore, even though it is one of the sources of learning problems, it is also an organizational problem and should be resolved in both levels.

Category “Change” includes users suggestions to developers and localization engineers of the 1C:Accounting 8.2 program and a couple of valuable advices for other users of the software in order to enhance their experience.

Developers are the ones who work on “real life” implementation of various proposals from stakeholders and requests for additional programming, and fix problems found by accountants. Their work is not always done perfectly and this is the reason for the following open codes to appear: forDevelopers-keepRelatedOperationsInOneDocument and forDevelopers-similarInterfaceForDocuments. The first open code is related to many times mentioned in this study operation of payroll accounting, and the second one is a request for more organized development. One of the interviewees has noticed that in different documents same features or options can be presented differently in the user interface and it is confusing for a user. This request for change asks for a successive, logical interface.

Other two open codes consist of advices: forBeginners-followInstructions and forAdvanced-doNotBeAfraidToTry. The context of these advices is that when a user is still on a beginner’s level it is very useful to follow instructions or tutorials and learn basic operations, documents, and other software’s features. Then progress is made and instructions can not teach
anything new, the next step is to try out something new and feel safe about it, because whatever is done in the program it can be changed later.

4.3.2 Substantive theory

In this study two biggest categories form a dichotomy. On the whole what was studied was user experience, and it has divided into user-friendly and user-hostile experiences. Based on definitions of usability, mentioned in “Research framework” chapter, user-friendly interaction is the same as usability. Then user-hostile experience is something opposite to usability.

As far as it can be seen from this study, these two categories form a dichotomy, because what causes positive experience does not cause negative and vice versa. For example, previous experience with this or another accounting software is something what causes better user experience, never on contrary. At the same time, lack of clearness or lack of understanding never causes positive user experience.

To provide proofs of these statements, first open codes forming cause-and-effect relationship are discussed. Figure 6 is a modified example of the code sequence from the “Data Analysis” chapter that forms cause-and-effect relationship. This was a first step in theory building, as relationships between open codes reflect relationships between categories they belong to.
To describe relationships between codes, and, hence, between categories, Spradley’s (1979) simple and useful table of semantic relationships was used as a reference.

**Universal Semantic Relationships**
1. Strict inclusion: X is a kind of Y.
2. Spatial: X is a place in Y, X is a part of Y.
3. Cause-effect: X is a result of Y, X is a cause of Y.
4. Rationale: X is a reason for doing Y.
5. Location for action: X is a place for doing Y.
6. Function: X is used for Y.
7. Means-end: X is a way to do Y.
8. Sequence: X is a step/stage in Y.

Based on the Figures 6 and 7, Figure 8 shows an exemplary model of relationships between categories.
Another type of relationship between open codes has emerged during data analysis, showing that some codes have appeared in order to compensate negative effects of other codes. The Figure 9 shows one of the cases of such relationship. In this case users have noticed that while learning how to use 1C:Accounting 8.2 they have encountered lack of cleanness about accounting transactions movements, due to accounting transaction invisibility. To diminish negative effect raised by this issue, users have found a way to check accounting transactions movements using report “Turnover Balance Sheet”. By checking “Turnover Balance Sheet” users raise awareness of what happens in the program depending on the operations they have completed and documents they have created. This raises level of understanding of the software’s inner layout.

Figure 9. Compensation mechanism
Figure 10 shows two open codes that make it very obvious that they are opposite, since one of the open codes is \textit{lackOfMotivation} and another one is \textit{externalMotivation}. Hence these open codes belong to the opposed categories: “Sources of learning problems” is contrary to “Sources of learning facilitation”.

![Figure 10. Counterbalanced codes](image)

Table of open codes that belong to opposite categories “User-friendly” and “Problems” is provided below (Table 3).

<table>
<thead>
<tr>
<th>User-friendly</th>
<th>Problems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specific operations</td>
<td>operationsWithAccounts Receivable; operationsWithAccounts Payable; sellingGoods</td>
</tr>
<tr>
<td></td>
<td>payrollAccounting; performingStockTakeOfPropertyAndPlantAndEquipment</td>
</tr>
<tr>
<td>Functions and features</td>
<td>quickStart; fieldValidationHelpsToRememberToFillInNecessaryFields; companyInformationFromLursoft; accountingTransactions AreDefinedByDefaultInStandardDocuments; copyPasteForQuickInput</td>
</tr>
<tr>
<td>Errors</td>
<td>Occurance – rare; userCaused; systemError; userForgotToFillInFields; userCaused-very frequent; fromAdditionalDevelopment-very rare;</td>
</tr>
<tr>
<td>Software in general</td>
<td>hotKeysForQuickerInput</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Software in general</td>
<td>thisSoftwareInComparisonWithAnotherIsEasier; softwareHasAllRequiredFunctionsForAccountant; goodMemoability; satisfaction; thisSoftwareMakesAccountantWorkEasier</td>
</tr>
<tr>
<td>Learning problems</td>
<td></td>
</tr>
</tbody>
</table>

Table 3. Opposite categories

Everything that was presented since the beginning of this chapter: categories, sequences of codes and relationships between categories, is reflected in the following integrative diagram (Figure 11, page 40). It shows how different categories are interrelated. Based on this integrative diagram theory about user-friendly and user-unfriendly user experiences forming a dichotomy was created.
Figure 11. Integrative diagram
5 FINDINGS

In this chapter answers to the research question and subquestions are given. First subquestions are answered, because they provide a detailed explanation of “easy” and “hard” parts of user experiences working with the program. After that the main research question is answered with more general and abstract ideas.

5.1 Answers to the research questions

- What makes 1C:Accounting 8.2 easy to use?

To answer this question, right part of the integrative diagram (Figure 11, page 40) should be examined. Right part of the diagram presents categories that form user-friendly experience. Category “User-friendly” partly answers this question: there are specific operations that are easy to use, functions and techniques that ease working with the program and general characteristics of the program that raise positive emotions in the end users. All of this is described in more details in the first part of this chapter, where “User-friendly” category is presented and open codes that form it are examined.

Nonetheless, this is only the first half of success, the other half lies in “Sources of facilitation” category. “Sources of facilitation” shows that the most essential part of facilitation is user’s own knowledge concerning accounting and related subjects, experience with accounting programs and programming skills.

As can be seen, easiness of use is a property of interaction between program and user, hence it depends not only on software’s qualities and characteristics, but on a user’s abilities and knowledge as well.

- What makes 1C:Accounting 8.2 hard to use?

Left part of the integrative diagram (Figure 11, page 40) holds the answer to this question. Left part of the diagram contains categories that do not form user-friendly experience. It includes “Problems” and “Sources of
problems” categories. “Problems” specify operations that are hard for users to perform, general characteristics of the software that raise uneasiness, and, of course, errors. The more detailed description of “Problems” category is given in subchapter 4.3.1.

Category “Sources of problems” makes two main points about origins of problems after education period is over. The first one is lack of user’s knowledge in accounting and lack of experience with the program under study. Other mentioned origins of problems are more general, such as change in standard procedure track, supposedly, would not be good for user of any program, as well as putting too much effort to perform relatively simple task could not be considered beneficial in any life situation.

To some point usability of 1C:Accounting 8.2 depends on functions, properties and features of the software itself, but user’s own knowledge and skills can raise usability level and improve user experience dramatically.

- What develops usability in context of 1C:Accounting 8.2 software?

The whole integrative diagram (Figure 11, page 40) shows how this study answers this question. Right part of the diagram contains explicit answers, while left part of the diagram and “Change” category contain implicit answers.

To sum everything up on a more abstract level, following list of components that develop usability in 1C:Accounting 8.2 is introduced:

- Software features that take away from user part of responsibility and necessity to remember complicated data, such as: accounting transaction numbers, what fields are necessary to fill in, typing in correctly other company’s information.
- Software functionality that allows to correct already created documents and to receive hints about how it was done before for complicated operations.
- External education, which implies preson to person interaction, external motivation to learn how to use the program and explanations of processes.
- Rare and understandable errors, comprehensive error messages are essential for the users.
- Operations as easy as possible, so that user does not has to spend extra effort on them. Consistent user interface.
- Visibility of processes that take place in the program, clearness of what user actions cause specific software response.
- Appropriate user's knowledge and skills. In this case appropriate and highly demanded knowledge is accounting knowledge and previous experience with accounting software.
6 DISCUSSION

The purpose of this study was to explore the components of usability of a specific accounting program. Some of the finding can be applied to the usability of software in general and a big part of the findings, especially in “Sources of problems” and “Sources of facilitation” categories, may be applied to other software as well.

The reliability of this study is supported by the use of two different data collection methods. In-depth interviews have brought rich data about the respondents’ user experiences, but think-aloud sessions provided with a possibility to test this data.

The important issue that was taken care of in order to improve the reliability of this study was interview and think-aloud sessions recoding. Due to that it was possible to return to what was said or what happened during think-aloud session numerous amount of times, which was useful for data analysis.

In addition to the factors supporting the reliability of this study, validity is supported by the clear and consistent data analysis method.

It is acknowledged that human factor might have affected this study: the researcher could have unintentionally influenced the respondents during the interviews and think-aloud sessions. The mere fact of a third person being present and the necessity to talk during the working process can affect the usual thinking process and working procedures.

Other research in the field of usability provides usability components that this study can relate to. For example, Jordan (2001, 8-11) lists the following user characteristics that affect usability: experience with the product, domain knowledge, cultural background, age and gender. In the present study the following user characteristics were discovered to affect usability: experience with this or another accounting software, and domain knowledge — in this case it is accounting knowledge.
The present study does not contradict with Nielsen’s (1993, 26) 5 usability principles, mentioned in the chapter “Research Framework”, even thought this study is more focused on a detailed examination of user experience.

In addition, learnability could provide additional value in working-life situations. The ideas listed below may appear useful and important for 1C:Accounting 8.2 educators and consultants.

- What categories of users learn how to use 1C:Accounting 8.2 faster than the others?

To answer this question it is useful to study the integrative diagram (Figure 11, page 40). It gives broad hints where to find information about learnability issues, and it turns out that selective codes “Sources of learning problems” and “Sources of learning facilitation” contain part of the answer. Other selective codes that contain part of the answer are “Experience and knowledge” and “Lack of knowledge”.

“Sources of learning problems” is opposite to “Sources of learning facilitation”. By inversion of open codes that belong to “Sources of learning problems” it is possible to obtain open codes that could belong to “Sources of learning facilitation”. This results in following conclusions: person with motivation to study and good memory in general would study more efficiently than elsewise. “Sources of learning facilitation” suggests numerous ways to ease educational process for users by introducing external help: external motivation, explanations of software layout, procedure tracks and accounting transactions movements by more experienced person. In addition, a person with previous experience working with accounting software has a big advantage in comparison with a person with no previous experience with accounting software.

Even though the titles of the selective codes “Experience and knowledge” and “Lack of knowledge” do not imply relation to learnability, there were several moments during the interviews where users stated that the lack of accounting knowledge has extended the education period, or, on the
contrary, accounting knowledge was a noticable support during an education period.

To sum up, users with one or more of the following characteristics form a category of users who are likely to learn how to use “1C:Accounting 8.2” faster than the others:

- User has motivation to learn how to use the program;
- User has a good memory in general;
- User with previous experience working with accounting software;
- User, who has received help and explanations from more experienced users;
- User with a solid accounting knowledge.
REFERENCES

Published references


Electronic sources


http://search.proquest.com.aineistot.lamk.fi/docview/1463286503?accountid=16407

http://search.proquest.com/docview/863787692?accountid=16407


Spradley, J. 1979. Chapter Nine of The Etnographic Interview. [referenced 10.10.15]. Available in:

Tadviser. 2015. Рынок систем управления предприятием [ERP market]. [referenced 22.10.2015]. Available in:
http://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B_%D1%83%D0%BF%D1%80%D0%B0%D2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F_%D0%BF%D1%80%D0%B8%D0%BD%D0%BE%D0%BA_%D0%A0%D0%BEE%D1%81%D0%B8%D0%B8

USABILITYLAB проводит семинары по юзабилити для project-менеджеров [USABILITYLAB holds a workshop on usability for project-managers]. 2008. [referenced 23.10.2015]. Available in:

1C Company. 2015. [referenced 21.10.2015]. Available in:
http://www.1c.ru/eng/
APPENDICES

APPENDIX 1. Initial open and selective codes

<table>
<thead>
<tr>
<th>Possible selective codes</th>
<th>Open codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experience with accounting programs</td>
<td>Experience with 1C:Accounting 7.7, Experience with other accounting software</td>
</tr>
<tr>
<td>Operations, what are hard to learn in 1C:Accounting 8</td>
<td>Calculating salary, Stock-taking of property, plant and equipment.</td>
</tr>
<tr>
<td>Factors increasing hardness</td>
<td>Change of the standard operating procedure, Operating procedure appropriate for big companies only (Latvian distributor didn't adapt this feature properly)(x2), Ambiguity of credit-debit accounts' operations, Lack of motivation during education, Program's way of naming operations and elements is not easy for users (x2), e.g. “Contragent” is used by program, but not in real life, Necessity to create various documents when it could be avoided (4 instead of 1), Too much effort to perform a relatively simple task, Not having enough knowledge about accounting (x3), Not having enough computer skills, Lack of clarity “what goes where” after element or document was created and saved,</td>
</tr>
</tbody>
</table>
Before active work starts many things need to be filled in,  

Lack of knowledge of program's parts at the beginning,  

Disorganization of developers: similar functions look differently in the user interface,  

Sometimes errors appear when they are least expected and users can't figure out what was wrong, and system message isn't helping either,  

If “Form validation” function does not accept what user has input, error message is not clear enough for user to understand what went wrong,  

Standard operating procedure, what buttons to push and which fields are required for specific action raise uneasiness in the beginning,  

Until standard operating procedure to perform certain action is not automatic, it can be forgotten,  

Sometimes it is not obvious what kind of information is needed to be input in field.  

<table>
<thead>
<tr>
<th>What operations were easy to learn</th>
<th>Operations with accounts receivable and accounts payable, Selling products</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factors that increase easiness</td>
<td>Understanding where it is possible to check credit/debit accounts’ operations, Understanding basics of all operating procedures in accounting software: first income, when expenditure, Clarification of operating procedure to reach a specific goal,</td>
</tr>
<tr>
<td>Good memorability,</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td></td>
</tr>
<tr>
<td>There is satisfaction after using a program,</td>
<td></td>
</tr>
<tr>
<td>Program makes user’s work easier,</td>
<td></td>
</tr>
<tr>
<td>With practice, user knows what is necessary to fill in and what could be skipped, this knowledge makes work proceed faster,</td>
<td></td>
</tr>
<tr>
<td>Experienced user knows what mistakes could happen and how to avoid them,</td>
<td></td>
</tr>
<tr>
<td>Copy-paste methods are making document entry faster,</td>
<td></td>
</tr>
<tr>
<td>Basic knowledge of economics and accounting is essential (x3),</td>
<td></td>
</tr>
<tr>
<td>Good computer skills (x2),</td>
<td></td>
</tr>
<tr>
<td>Good information search skills,</td>
<td></td>
</tr>
<tr>
<td>Programming skills, which are more important for making actual changes in the program,</td>
<td></td>
</tr>
<tr>
<td>Programming as a way for experienced user to understand how things work,</td>
<td></td>
</tr>
<tr>
<td>Working with manuals/instructions really helps at the beginning,</td>
<td></td>
</tr>
<tr>
<td>Errors are not fatal, they can be recovered from,</td>
<td></td>
</tr>
<tr>
<td>Knowledge of “hot keys” increases speed dramatically,</td>
<td></td>
</tr>
<tr>
<td>Overall good memorability, but new things could be forgotten if they are not used,</td>
<td></td>
</tr>
<tr>
<td>If something is forgotten it is always possible to find a ready example and see how it was done,</td>
<td></td>
</tr>
</tbody>
</table>
“Form validation” is very handy for users to remember what information is crucial to be filled in,

Program allows to cancel/edit operation what was done,

Program has all the necessary functionality to satisfy bookkeeper’s needs,

Is easy to manage for experienced users,

Makes bookkeeper’s life easier,

In “1C:Accounting 8” there are “typical operations” there user doesn’t need to indicate accounting transaction’s number,

Quick start of the work: user is shown how to do a certain operation and then he/she can perform it by himself/herself,

When some operation has been performed multiple times, user can perform it automatically,

Education goes more efficiently if there is someone user can ask help from,

When where are doubts, it is always worth seeing how the same operation was done before,

User with experience in working with another accounting program says this program is easier than the other one,

Easier document input and more information can be filled in,

There is satisfaction after working with program (because have learnt/understood something new),

Connecting to Lursoft and getting company information
<table>
<thead>
<tr>
<th><strong>Advices on how to make some things easier</strong></th>
<th>from there reduces the time user would need to input and check this information.</th>
</tr>
</thead>
<tbody>
<tr>
<td>What can be done with one document/operation, should be done with one document, e.g. salary calculation,</td>
<td></td>
</tr>
<tr>
<td>Settings and properties should be in one place for similar elements,</td>
<td></td>
</tr>
<tr>
<td>Similarly functioning elements should look similar,</td>
<td></td>
</tr>
<tr>
<td>Do not change documents or operating procedures too harshly, it would make old users struggle.</td>
<td></td>
</tr>
<tr>
<td><strong>Errors</strong></td>
<td>After years of practice hardly any mistakes happen,</td>
</tr>
<tr>
<td></td>
<td>Even experienced users receive mistake notifications sometimes, but know what actions to undertake to solve a problem,</td>
</tr>
<tr>
<td></td>
<td>Some errors happen because of the program itself (for example, changing operation’s happening time could solve a problem),</td>
</tr>
<tr>
<td></td>
<td>There are 2 types of user created errors: mistaken operation or user forgot to input something,</td>
</tr>
<tr>
<td></td>
<td>It is usually obvious what has caused an error,</td>
</tr>
<tr>
<td></td>
<td>There are 3 sources of errors: user, additional programming or system errors.</td>
</tr>
<tr>
<td><strong>Experience</strong></td>
<td>Timeframe: 1-1,5 years to become advanced user, 1 year to become very experienced user, 2-3 months to fully understand a program for a standard user,</td>
</tr>
<tr>
<td></td>
<td>The more experienced user gets, the more intuitive he feels the program,</td>
</tr>
</tbody>
</table>
| **Programming skills** | Knowledge about what fields are really necessary to fill in and what could be left empty increases effectiveness,  
Experienced user knows ways to fix an error and fixes it,  
Experienced user knows where and what to check in order to be sure that operation went right,  
Previous work experience with other accounting program make user self-confident about input of documents and accounting operations,  
Experienced user feels much more easier when working with 1C program,  
Experienced user knows operating procedures to reach specific goal very well and that’s why work advances faster,  
Experience in working with previous 1C:Accounting versions makes education period for “1C:Accounting 8” very short,  
With practice amount of effort per operation reduces(x2),  
With practice user learns how to deal with errors and mistakes,  
Are helpful,  
Are essential to find an answer in the code: why something works the way it works,  
Sometimes are helpful even for existing functionality: to find in the code of the function if element has various options not shown in the user interface. |
APPENDIX 2. Final categories, open and selective codes

Problems:

Errors:Occurance – rare; userCaused; systemError;
userForgotToFillInFields; userCaused-very frequent;
fromAdditionalDevelopment-very rare; systemErrors-extremely rare;
payrollAccounting-noEvidentReasons;
errorMessageIsIncomprehensible

Specific operations: payrollAccounting (x4);
performingStockTakeOfPropertyAndPlantAndEquipment;

Learning problems:accountingTransactionsInvisibility;
atFirstNotIntuitivelyUnderstandable

Software in general: muchToFillInAtTheBeginning;
forcingCompletelyNewUserInterface;
lackOfUnderstanding:WhatTypeOfInputFieldsRequire;
specificLanguage

Sources of problems:

Lack of knowledge: lackOfKnowledge: Accounting (x2);
lackOfKnowledge:softwareFeatures;
lackOfKnowledge:HowToCheckOperationValidity;

Software based sources of
problems:changeOfStandardProcedureTrack;
notAppropriateProcedureTrack (x2);
tooManyDocumentsForOneOperation;
Sources of learning problems: lackOfClearness(x2); lackOfMotivationToLearn; memorability-rememberingProcedureTracks(x2);

Change: forDevelopers-keepRelatedOperationsInOneDocument; forDevelopers-similarInterfaceForDocuments; forBeginners-followInstructions; forAdvanced-doNotBeAfraidToTry;

User-friendly:

Specific operations: operationsWithAccountsReceivable; operationsWithAccountsPayable; sellingGoods;

Functions and features: quickStart (x2); fieldValidationHelpsToRememberToFillInNecessaryFields; companyInformationFromLursoft; accountingTransactionsAreDefinedByDefaultInStandardDocuments; copyPasteForQuickInput (x2); hotKeysForQuickerInput;

Software in general:
thisSoftwareInComparisonWithAnotherIsEasier; softwareHasAllRequiredFunctionsForAccountant; goodMemoability; satisfaction (x2); thisSoftwareMakesAccountantWorkEasier;

Sources of facilitation:

Software based sources of facilitation: possibilityToCheckEarlierExamples(x2); possibilityToAlterEarlierDocuments; checkingAccountingTransactionsViaReport (x2);

Experience and knowledge:
timeFrameFromBeginnerToExperienced – 3 months; timeSaving (x2); knowHowToSolveProblems; knowHowToAvoidMistakes; learnedToUnderstandWhatErrorMessageesMean; withThisSoftware (x2); frequentlyRepeatedProcedureTracks;
learnedProcedureTracks (mentioned 3 times); previousSoftwareVersion; anotherAccountingSoftware (x2);

Knowledge-Accounting (x3); Knowledge-Economics; Skills-programming (x2); skills-computer; skills – informationSearch (x2); programmingSkills: adjustingConfigurationSettings; increaseSoftwareLayoutUnderstanding; findingHiddenFunctionalityInUserInterface; understandingBasicAlgorithmOfAllProcedureTracks; understandingSoftwareInnerLayout;

Sources of learning facilitation: externalMotivation; explanationOfAccountingTransactionsMovements; explanationOfSoftwareInnerLayout; explanationOfProcedureTracks; personToPersonInteraction (x2).