Janna Ylönen

USER REQUIREMENTS ELICITATION METHOD COMPARISON FOR A SYSTEM UPGRADE

JYVÄSKYLÄN YLIOPISTO
INFORMAATIOTEKNOLOGIAN TIEDEKUNTA
2021

TIIVISTELMÄ

Ylönen, Janna
Käyttäjävaatimusten keräysmenetelmien vertailu järjestelmän jatkokehitystä varten
Jyväskylä: Jyväskylän yliopisto, 2021, 47 s.
Tietojärjestelmätiede, kandidaatintutkielma
Ohjaaja(t): Seppänen, Ville

Tässä tutkielmassa kuvataan käyttäjävaatimusten keräämismenetelmiä kiinnittäen huomiota siihen, kuinka menetelmää voidaan hyödyntää olemassa olevan tietojärjestelmän ja sitä ympäröivien prosessien kehittämisen apuna. Tutkielmassa kuvatut 16 menetelmää jaettiin kuuteen luokkaan: perinteisiin, ryhmä-, prototyyppi-, kognitiivisiin ja kontekstuaalisiin menetelmiin. Tutkielma koostuu kirjallisuuskatsauksesta ja käyttäjävaatimusten keräämismenetelmiä vertailevasta taulukosta, johon on kerätty tiivistetysti kirjallisuuskatsauksessa esille tulleita kuvattujen menetelmien vahvuuksia ja heikkouksia.

Asiasanat: Käyttäjävaatimusten kerääminen, menetelmät, vaatimusmäärittelyprosessi, menetelmän valitseminen.

ABSTRACT

Ylönen, Janna
User requirements elicitation method comparison for a system upgrade
Jyväskylä: University of Jyväskylä, 2021, 47 pp.
Information Systems Science, Bachelor's Thesis
Supervisor: Seppänen, Ville

This thesis describes and evaluates methods for user requirements elicitation, paying particular attention to how they would fit to a case of upgrading an existing information management system and its processes. The 16 different user requirement elicitation methods were divided into six categories: traditional, group, prototyping, model-driven, cognitive, and contextual methods. This thesis consists of a literature review and a requirement elicitation method comparison table that summarises the pros and cons that have emerged in the literature review.

Keywords: User requirements elicitation, Elicitation methods, Requirement process, Technique selection.

FIGURES

TABLES

CONTENTS

1 INTRODUCTION

User requirements elicitation is an essential phase in software product development. In this phase, all the stakeholders using the system and the processes involved are identified and understood. User requirement elicitation, or sometimes called requirement gathering or collecting techniques are methods used to determine the needs of users, so that the systems can be built with a high probability of satisfying their needs. In other words, user requirement elicitation ensures that the product that is being built is of the right type.

Selecting the best user requirement elicitation method for the task at hand is often challenging. One elicitation method cannot work for all situations and each elicitation method can be intended to be used in specific product development phase.

This thesis is a literature review that acts as a preliminary study to find out the user requirement methods that could be used in a system upgrade. In this study, different popular user requirement elicitation methods are listed, and the pros and cons are searched of them from literature to finally help in the requirements engineer in the selection process to find the methods that would be the most effective for the case.

1.1 Motives for the study

The success or failure of a system development effort depends heavily on the quality of the requirements (Jones, 1996). The quality of the requirements is greatly influenced by methods used in requirements elicitation because elicitation is all about learning the needs of users and communicating those needs to system builders. How we select an appropriate elicitation method out of the plethora of available methods greatly affects the success or failure of requirements elicitation. (Hickey & Davis, 2003.)

Software engineers tend to choose a method to apply on one of the following grounds: it is the only method they are acquainted with; it is their favourite

method for all situations; they are using a methodology that prescribes a particular method; or they guess that the method is effective under the current circumstances. This subjective decision can bias the elicitation results, degrade the quality of the output requirements, and ultimately, have an impact on the quality of the final software product. (Carrizo, Dieste & Juristo 2014.)

Many of requirements elicitation methods have been imported from fields like cognitive psychology, anthropology, sociology and linguistics (Nuseibeh & Easterbrook, 2000) and have been successfully used in knowledge engineering, and lately, software engineering. Most requirements engineers are nonetheless unfamiliar with this range of methods and miss the chance of optimising the requirements elicitation. (Carrizo, Dieste & Juristo, 2014.)

If we could improve the average analyst's ability to select elicitation methods, we will most likely improve our record of successful products (Hickey & Davis, 2003). This study aims to help the requirements engineer in selecting the optimal user requirement elicitation methods that best utilise existing data in the system as well as stakeholder knowledge and experience.

1.2 Research problem

The research problems in this study are the following:

- What user requirement elicitation methods are there that could be used for gathering user requirements for a system upgrade?
- What pros and cons can be found of them according to different authors?

1.3 Research method and objectives

To be able to answer to the research problems, a literature review was carried out. The target of the literature review was to search various kinds of user requirement elicitation methods that would be suitable to be utilised in course of a returns management system upgrade.

In the search for scientific articles mainly the databases ACM Digital Library and IEEE Xplore accessed through the Nelli-portal were used. Also, JYKDOK library database and Google Scholar were utilised in the search for material for the literature review. The number of references for each search result in Google Scholar were used to evaluate the research, and papers with small number of studies found of it were dropped out from this study.

In addition to the database searches, snowballing (Wohlin, 2014) was also used on the referenced studies to ensure that more of relevant literature was included in the thesis.

In the search, the elicitation methods' names and the following search phrases and their combinations were used: user requirement elicitation,

acquisition, collecting, gathering, requirements engineering, elicitation methods, techniques, method comparison.

While describing the different methods, pros and cons were listed based on how the author of each reference viewed them. The pros and cons are then summarised and compressed to a table consisting of all the methods for easier comparison.

The result of the study is the descriptions of the methods and the comparison between them. Every method has also its cons that must be acknowledged. Therefore, a list of cons are a list of things to avoid or pay attention to in the actual elicitation phase.

1.4 Case system process

The case company (Company) manufactures devices for the use of other organisations. The customer company (Customer) takes care of the installation of the devices to end-customer locations. In a situation where a device is broken, the Customer uses the Company's returns management system to handle the return of the faulty device.

A common scenario is that the person who is performing the installation or a maintenance operation on the field, notices that a device is broken. From there on forward, the process goes as follows:

1. Customer's field technician uninstalls the device and attaches a fault ticket sticker with a unique ticket number to the broken device.
2. The technician describes the fault (that is, the reason of return) to the ticket as a code or only in writing if there is no matching code.
3. The ticketed devices are transported to the Customer warehouse, where Customer's personnel report the devices to the return management system based on the information on the tickets, usually in a larger batch.
4. In the system, the devices are selected to be sent and a packing list is printed out and included in the package.
5. Customer opens a collection request form from the system.
6. Customer fills the form, saves it to the computer and sends it by email to Company.
7. Company orders the transportation and replies to the email by sending a waybill and a tracking ID for the delivery.
8. Customer prints out and attaches the tracking IDs to each package in the delivery.
9. When the devices arrive to Company's warehouse, Company personnel perform the incoming material inspection and create repair order rows for the devices to an ERP system.
10. The repair personnel repair or scrap the devices and report the actions done and the status of the devices to the returns management system where Customer can view the information.

11. The scrapped devices are replaced by new devices of which new order rows are created to the ERP system.
12. A collection request is made for the device packages and they are sent back for Customer.

During the process both the Customer and the Company can monitor the state of the devices in the system.

The information in the returns management system can be printed out as CSV format files. The Company uses the data files in creating visualisations for reports, and Customers use the files to import device data to their other systems.

1.5 Case system upgrade

The current returns management system was development in 2009, and since then the processes in the surrounding the system have improved a great deal. A lot of usability issues in the system has been patched up along the years, but the basis is still the same, and the implementation of a new system is inevitable at some point.

The system was developed at a time when there was no need to handle big masses of devices, but since then the material flows have grown notably. In the return process for tens of devices, there can be manual phases, but when reporting by the hundreds or thousands, the manual phases must be minimised, or the outcomes of human error rise to be significant.

For companies in the hardware manufacturing business, the customers' need to return a faulty product is a situation, which will come up eventually, and which the customer can view as unavoidable and a burden. A system that is easy to use and learn can ease the feeling of the "necessary evil". When the whole return process is working efficiently, the customer can rather see that they have been treated fairly and well and can possibly improve and adjust their own processes accordingly.

A system that better answers to the user needs has advantages for both customer and company. Higher usability will encourage the customer to report the fault in more detail, the company will gain more information about the causes for returns and information for product development. Finally, when the returned devices are handled quickly, time and money is saved for the company as well as the customer who will get their repaired or substitute devices back sooner.

There are also environmental and sustainability issues to be seen in the return management process. The product return system can be seen as a service product. If it is not functioning, customers may see that it is not providing them any value. Dissatisfaction to the product will lead to discontinuation to the use of the system. This can have two outcomes that can be considered from an environmental sustainability point of view. The first scenario is that none of the devices will be returned to be repaired and therefore none of the devices or parts of them will be re-used. Another poor outcome is that the customer returns all the

non-functioning devices in one large batch, the company goes through them and only then scraps the irreparable devices, when the shipping of those devices could have been avoided.

Whether the system will later be updated or if the company will choose to buy the returns management software as a ready-made package, this thesis will act as a preliminary study to gather information of stakeholder needs and requirements for the new system.

In Chapter 2 we will define the key concepts used in this thesis. In Chapter 3 we will describe 16 different user requirement elicitation methods, shortly the process of how the method is used, and the pros and cons found of each of them from literature. The elicitation methods are divided into 5 categories based on the type of the method. In Chapter 4, we will present the pros and cons in a comparison table. We conclude the study with the discussion in Chapter 5.

2 DEFINITIONS OF KEY CONCEPTS

In this chapter, we will describe the key terms and concepts used throughout this study.

2.1 User requirement

One typical definition for a user requirement can be found from the IEEE Standard Glossary of Software Engineering Terminology: A requirement is a condition or capability needed by a user to solve a problem or achieve an objective (IEEE, 1990). On the other hand, Robertson & Robertson (1999) state that a requirement is something the system must do or a feature that the system must have.

Kotonya and Sommerville (2002) define requirement as a definition that describes the services that the system should offer and the restrictions for the system's functionality. The requirements are collected from the stakeholders using different methods to fulfil the needs and wishes they have brought up. This definition splits the requirements into functional and non-functional requirements. The functional requirements are the system's functions, what the system must be able to do, or what the user must be able to do in the system. The non-functional requirements are the system's restrictions, for an example what quality and timing related requirements the system has.

2.2 Requirements engineering

Software development is inherently complex (Brooks, 1987) and systems requirements engineering is considered to be the most difficult step in the process (Metersky, 1993; Zmud, Anthony & Stair, 1993).

The primary measure of success of a software system is the degree to which it meets the purpose for which it was intended. Broadly speaking, software systems requirements engineering is the process of discovering that purpose, by

identifying stakeholders and their needs, and documenting these in a form that is usable for analysis, communication, and the implementation that follows. (Nuseibeh & Easterbrook, 2000.)

According to Sommerville & Sawyer (1997) requirements engineering consists of activities that cover discovering, analysing, documenting, and maintaining a set of requirements for a system. In addition, requirements engineering is a process, where visions about a system are established in a context. (Pohl, 1994, 245.)

There are several models of the requirements engineering process with similar activities. Pohl's model (Pohl, 2010) consists of elicitation, negotiation, specification, documentation, and validation/verification. Wiegers's model (Wiegers 1999) breaks down into two sub requirements engineering activities: development and requirements management, whereby the development activity is broken down into elicitation, analysis, specification, and verification.

Sommerville (2005) presents the activities of requirements engineering as a cycle, where individual activities repeat as the software requirements are derived, and the iteration continues during system implementation and operation. The activities repeat in a cycle and in addition, requirements documentation and requirements management are done always on the side as their own process.

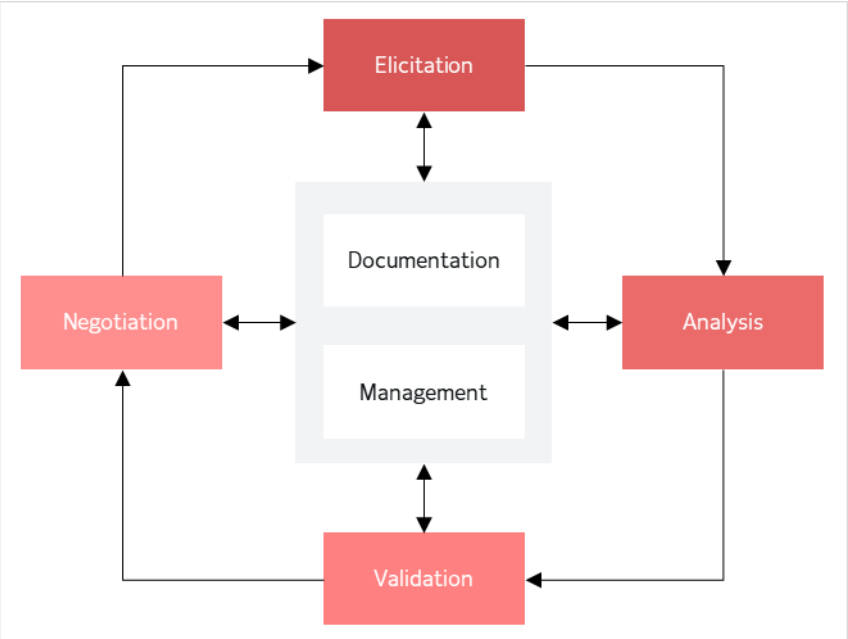

FIGURE 1 Requirements engineering process by Sommerville (2005)

The activities can be described shortly as follows. The activities include firstly requirements *elicitation*, which means identifying the sources of information about the system and discovering the requirements from these. Then the requirements' overlaps, and conflicts are *analysed*. In user requirement *validation* we go back to the system stakeholders to check that the requirements are what they really need. *Negotiation* activity aims to reconcile conflicting views and to generating a consistent set of requirements as inevitably the stakeholders' views

will differ, and the proposed requirements might conflict. In *documentation* the requirements are written down in a way that stakeholders and software developers can understand. Requirements *management* controls the requirement changes that will arise. (Sommerville, 2005.)

2.3 Requirements elicitation

Bohem (1981) argues that requirements elicitation is the first and most critical step in the requirements engineering process and doing it improperly will lead to poor quality products, late delivery dates and high costs.

Requirements elicitation is how the problems and needs of customers are determined, so that a system that resolves those problems and addresses the customers' needs can be constructed (Hickey and Davis, 2003).

A big part of requirements is not listed or documented anywhere – they only exist in the minds of the users of the system. Elicitation means "to call forth or draw out (something, such as information or a response)" as defined by the Merriam-Webster (2021) dictionary. And therefore, it is called user requirements elicitation because the developers want to draw out the requirements from the stakeholders. Eliciting user requirements is finding out the users' needs.

2.4 Requirements management

Requirements management can be seen as a parallel support process for other requirements engineering processes (Sommerville & Sawyer, 1997; Kotonya & Sommerville, 1998). It also continues after requirements specification phase because the requirements continue to change during system development and these changes must be managed (Kotonya & Sommerville, 1998). Sommerville and Sawyer (1997) define the principles concerning requirements management:

- Changes to the agreed requirements need to be managed.
- Relationships between requirements should be described and managed.
- Relations between requirements documents and other documents produced during the systems and software engineering need to be managed.

2.5 Stakeholder

A stakeholder is an individual, group of people, organisation or other entity that has a direct or indirect interest (or stake) in a system (Hull 2011). In addition, Somerville and Sawyer (1997) identify system stakeholders as people who will

be affected by the system and who have a direct or indirect influence on the system requirements.

In this case, the stakeholders are for example:

- project managers,
- project coordinators,
- financial controllers,
- device repair personnel,
- warehouse workers,
- device installation and field maintenance technicians.

2.6 Returns management

Returns management is the supply chain management process by which activities associated with return, reverse logistics, gatekeeping, and avoidance are managed within the firm and across key members of the supply chain (Rogers, Lambert, Keely and García-Dastugue, 2002).

3 USER REQUIREMENT ELICITATION METHODS

This chapter describes the different methods used for eliciting user requirements. For each method, there is a description of what the method is, a short process description of how the method is intended to be used, and then the pros and cons that were found from literature. Lastly, each method is reflected against the case company's returns management system update – how could the method be applied there.

The categorisation of methods in this chapter is done according to Nuseibeh and Easterbrook (2000). They describe the main methods and characteristics of each category as follows.

- *Traditional methods* include a broad class of generic data gathering techniques. These include the use of questionnaires and surveys, interviews, and analysis of existing documentation such as organisational charts, process models or standards, and user or other manuals of existing systems.
- *Group elicitation methods* aim to encourage stakeholder agreement and buy-in, while utilising team dynamics to elicit a richer understanding of needs. They include brainstorming and focus groups, as well as JAD workshops.
- *Prototyping* has been used for elicitation where there is a great deal of uncertainty about the requirements, or where early feedback from stakeholders is needed (Davis, 1992). Prototyping can also be readily combined with other methods, for instance by using a prototype to provoke discussion in a group elicitation method, or as the basis for a questionnaire or think-aloud protocol.
- *Model-driven methods* provide a specific model of the type of information to be gathered and use this model to drive the elicitation process. These include goal-based methods and scenario-based methods.
- *Cognitive methods* were originally developed for knowledge acquisition for knowledge-based systems (Shaw & Gaines, 1996). The methods include for example protocol analysis, laddering, card sorting and repertory grids.
- *Contextual methods* emerged in the 1990's as an alternative to both traditional and cognitive methods (Goguen & Linde, 1993). These include the

use of ethnographic methods such as participant observation. They also include ethnomethodology and conversation analysis, both of which apply fine grained analysis to identify patterns in conversation and interaction (Viller & Sommerville, 1999).

Model-driven methods consist of goal-based methods, such as KAOS and I* and scenario-based methods, such as CREWS. These are supporting methods used for representing and modelling goals or scenarios from separately gathered user requirements. Therefore, these methods are left out of the scope of this study.

3.1 Traditional methods

3.1.1 Document analysis

Document analysis is a systematic procedure for reviewing or evaluating printed and electronic documents (Bowen, 2009, p. 27). It focuses on studying existing documentation, such as business plans, technical documents, problem reports, and existing requirement documents about the system.

Document analysis, in line with any other qualitative method, requires data to be explored and interpreted by researchers for identifying meanings relevant for the study aims and, through systematic coding, developing understanding about the phenomena under interest (Bowen, 2009).

Bowen (2009) lists the advantages and limitations of document analysis:

- Document analysis is an efficient method in that sense that it is less time-consuming than other research methods as it requires data selection, instead of data collection.
- Many documents are freely available in public internet.
- Document analysis is less costly than other research methods and is often the method of choice when the collection of new data is not feasible. The data, that is the documents, have already been gathered.
- Documents are unobtrusive, non-reactive and stable – that is, they are unaffected by the research process. With for instance observation, an event may proceed differently because it is being observed. A possibility that the investigator has influence on the research or that the investigator's presence alters what is being studied is not an issue when using documents for research purposes.
- Documents are produced for some purpose other than research; they are created independent of a research agenda. Consequently, they usually do not provide sufficient detail to answer a research question. More likely, in an organisational context, the available (selected) documents are aligned with corporate policies and procedures.

The pros here clearly outweigh the cons that are more possible flaws than major disadvantages.

As document analysis focuses on existing documentation it would be useful when upgrading an existing system or in a migration project. But in this case, there is only little system documentation that is available of the system in question, and the target documents would be some problem reports and the user manual. On the other hand, use cases, reviews, and descriptions of other manufacturers' product return systems and how they have handled the processes surrounding the system could provide plenty of useful information. Studying documents that describe how others are managing their returns process could provide valuable lessons and compensate the sparsity of available case system documents. In this case, the document analysis would need other methods by its side, such as interviews and observation, that have more direct contact and communication with the stakeholders of the system. Documentary data could be then combined with the information from those methods to evaluate accuracy and establish credibility of the elicited requirements.

3.1.2 Existing data analysis

Many research questions can be answered quickly and efficiently using data or specimens that have already been collected. There are three general approaches to using these existing resources. Secondary data analysis, ancillary studies and systematic reviews. (Hulley et al., 2013.) Of these three, the ancillary studies and systematic reviews include utilising previous studies, and therefore those are dropped out of this thesis.

Due to a lack of a defined process for the secondary data analysis method, Johnston (2014) proposes it. It begins with developing research questions, then the dataset is identified, and lastly thoroughly evaluated. In most cases, a research is started with an investigation on what is already known and what there is to be learned about a topic, including related and supporting literature. But previously collected research data on the topic should be also considered, as useful data for addressing the research questions may already exist. (Johnston, 2014.)

Using existing data and specimens is a fast and efficient way for researchers with limited resources to begin to answer important research questions, gain valuable experience in a research area, and sometimes have a publishable finding in a short time frame. Studies using existing data has also some disadvantages. The selection of the population to study, which data to collect, the quality of data gathered are all predetermined. All these factors contribute to the main disadvantage of using existing data: The investigator has little or no control over what data have been collected, and how. (Hulley et al., 2013.)

In this case, the data set would be the returns management system's database rows of reported, returned and then repaired or scrapped devices accumulated over the years. The large amount of data has been entered to the system by different users in various situations, some with high detail and accuracy and some with very low detail. The poor quality of many reports in the system may prove as a disadvantage. On the other hand, that might already be a user

requirement for the new system – the users must be encouraged to report the devices into the system with more detail. Other disadvantage with the data is that the reports obviously describe the faults in the devices and not in the system. The first step in getting started with this method is to investigate whether the research question can be addressed with the existing data set.

3.1.3 Interviews

Interviews is a conversational or verbal method which is considered easy and effective for idea sharing and expressing needs between analysts and stakeholders. It includes face to face conversations with one or two people asking questions and documenting the results that finally lead to requirements. (Yousuf & Asger, 2015.)

Interviews are of 3 types: Structured, Semi-Structured, and Unstructured. The goal of first two methods is to acquire quantitative data, whereas the later method points towards understanding user expectations through open-ended debates with the stakeholders and acquiring qualitative data (Arif, Khan & Gahyyur, 2010).

Structured interviews are the most formal type with a set of predefined questions that are asked from the stakeholder. Zowghi and Coulin (2005) consider it an effective method but claim that it does not allow generation of new ideas. I would argue this to be highly dependent on the quality of the research questions, as also Hickey and Davis (2003) take the opposing stand and point out that the method is primarily used to surface new information or uncover conflicts.

Semi-structured interviews are a combination of predefined and unplanned questions. Here the interviewer may ask further defining questions and share and discuss ideas. Unstructured interviews are informal interviews containing unplanned questions. It is an open discussion between analysts and stakeholders producing qualitative data. (Yousuf & Asger, 2015.)

Yousuf and Asger (2015) note that interviews are good with complex topics that require dialogue and in eliciting detailed requirements. On the other hand, only limited number of people can be reached, and arranging the interviews can be effortful and time-consuming.

One problem with interviews according to Goguen and Linde (1992) is that people know how to do many things that they cannot describe. It is common in ethnography that people's descriptions of how they weave a basket or select a chief of write a program, bear a complex and opaque relation to how they can be seen to do these things when they are observed. This problem is so familiar that is has a nickname in social science: *the say-do problem*; also, philosophers speak of *tacit knowledge*.

End-users are usually happy to describe their work and the difficulties they face. Interviews might be a good tool to get to know and meet customers and therefore later contacting them with a lower threshold. Interviewing customer users could engage them and give them trust for future activities. In this case, the interviews is a worthy option. The processes in which the different users utilise the system vary and there are multiple types of internal users and customer users.

Interviews could give a good overview of all the different users and what they are utilising the system for.

3.1.4 Questionnaires

A questionnaire is a form of survey that involves asking the stakeholders to answer a set of questions. In many cases, questionnaires are the cheapest method for gathering quantitative data. A questionnaire can reach a lot more stakeholders in cases where face-to-face interviews or brainstorming sessions are not possible.

When designing the questions, researcher must decide whether to make the question open to which respondents answer in their own words or closed to which the respondents select the answer from a set of choices.

Both question types have their pros and cons. Open questions allow more freedom but the results are usually harder to analyse (Laplante, 2009). Open questions can add richness to survey results that is difficult, if not impossible, to achieve with closed questions, so including some (on their own or as follow-ups to closed items) can bring in significant benefits. Closed questions will generally suffer more than open questions from correct guessing. If two people differ in their interpretations of the points on a scale, they may give different responses even though they have identical underlying attitudes. Open questions might be more likely than closed questions to elicit "don't know" answers from people who know the correct answer but are not sure of it.

A successful questionnaire must have well-chosen participants (Kothari, 2004). To get the most accurate results, targeting is the key. In this case, the important thing is to find and target the customer users who have used the system. Laplante (2009) notes that questionnaires are most useful when the domain is well understood by the stakeholders and the requirements engineer.

Goguen and Linde (1993) present a few pros and cons for questionnaires:

- Questionnaires have the benefit of appearing scientific because they use statistical analysis.
- Useful with a large population.
- Easily combined with other methods. For example, start with ethnography to uncover typical patterns of work, then use questionnaires to find out the most important problems, and finally apply conversation analysis to gain deeper understanding.
- Questionnaire-based interviews are limited by their question-response type of interaction, which assumes that a given question always has the same meaning for the stakeholders.
- Moreover, this method does not have the kind of interaction where a shared meaning of the issue could be established between the stakeholder and the interviewer.

When asked to rank or select features that a system could have, users are likely select more of them that they would ever use. (Laplante, 2009.)

A questionnaire can be inflexible due to that the questions asked, as well as the method of administering it, must remain unchanged during the process of data gathering (Okwemba, 2019).

In this case, true-false questions would provide information about what the stakeholders consider important in the returns management system. Open ended questions that request free-form responses, and multiple-choice questions might be useful here as we are looking for requirements that the current system cannot answer.

3.2 Group methods

3.2.1 Brainstorming

Brainstorming is a common problem-solving method of sharing and improving ideas in a group session. The method was first introduced by Osborn (1957), who suggested that this method considerably increases the quality and quantity of ideas produced by group members. (Diehl & Stroebe, 1987.)

Brainstorming provides an open environment of discussion, where users are free to give their requirement and expectation of system. The data (ideas) collected after this process is then discussed and analysed. Brainstorming contains two phases - the generation phase where ideas are collected without criticising, and evolution phase, where collected ideas are discussed. (Tiwari et al., 2012.)

Osborn (1957) states that a brainstorming session should address a specific question, as sessions addressing multiple questions are inefficient. Osborn's rules for a brainstorming session are as follows: The more ideas the better, the wilder ideas the better, improve or combine ideas already suggested, and do not be critical. When engaging in a brainstorming session where the above rules are followed, the average person can think up twice as many ideas when working with a group than when working alone (Osborn, 1957).

Taylor, Berry, and Block (1958) were the first to test Osborn's claim in a study called "Does group participation when using brainstorming facilitate or inhibit creative thinking?". In the experiment to answer the question, they used nominal groups. (Diehl & Stroebe, 1987.)

The ideas of each nominal group were combined, eliminating redundant ideas as though the members had worked together. Contradicting Osborn's claim Taylor et al. found that nominal groups produced nearly twice as many different ideas as the real groups, and therefore concluding, that group participation when using brainstorming inhibits creative thinking.

Diehl and Stroebe (1987) searched for evidence for the superior productivity of nominal groups. They listed three major factors that account for the lower productivity of real groups. These are *production blocking*, *evaluation apprehension*, and *freeriding*.

Production blocking means that subjects in a brainstorming session are prohibited from verbalising their ideas as they occur. Lamm and Trommsdorff (1973) argued that the problem with real, especially large groups is that only one person can speak at a time, and in the meantime, others may forget their ideas or suppress them as less relevant. Also, listening to the ideas of others can act as a distraction or alter the original idea.

Evaluation apprehension was studied by Collaros and Anderson (1969) who manipulated the perceived expertise of group members in brainstorming groups. The study proved that in case group members think the other members have had previous experiences of working in such groups, the perceived expertise lowered the productivity of the brainstorming session.

Freeriding on the efforts of others may seem tempting when the group members know that their ideas will be pooled and analysed on the group level only. On the other hand, individually working subjects see no possibility to evade the task. (Diehl & Stroebe, 1987)

By working with these rules, the result could be a large amount of material with most part having no greater value from the system development point of view. As Nijstad and De Dreu have stated, producing a large number of ideas is never the ultimate goal of a brainstorming session. Instead, what the developers are after is a limited number of good ideas to select for further development.

Laplante (2009) notes that most group-oriented elicitation methods embody some form of brainstorming in them. Therefore, different versions and modifications of brainstorming are readily available. Dennis and Valacich (1993) introduce a method called electronic brainstorming where the brainstorming session is done on computers. The participants supply the ideas anonymously and cannot be blocked from contributing ideas.

Brainstorming can be useful for general objective setting, such as target or vision statement generation (Laplante, 2009).

Osbourne came up with the method while he was the CEO of an advertising agency. Brainstorming could be useful for searching for ideas for an example for marketing, as the method emphasizes on creativity and new ideas. In the case of software development and upgrading an existing system, numerous new ideas and going through them might even slow down the project.

3.2.2 Focus groups

Focus groups is a qualitative research method that represents a type of in-depth interviews where the interview is between the researcher (called moderator) and a small group of people with relevant characteristics to the studied case. As a result, in-depth information is collected from a group of people representing the interested population in the field of study. (Simon, 1999.)

Focus groups is one of the most widely used research tools in social sciences. The lineage of focus groups can be traced to the 20th century studies of persuasive communications and the effects of mass media that were conducted in the early 1940s. The later on, some 30 years ago, focus groups were largely the domain of marketing researchers. The following decades saw an explosion of the use of the

method, and it broadened to different science and professional disciplines. More recently, changes in focus group design and in use of technology has appeared, such as virtual focus groups bringing people together over the internet. (Stewart & Shamdasani, 2015.)

Farinha and Mira da Silva (2009) promote the focus groups method because it involves all stakeholders as well as system analysts to teams that allow more natural interactions between people than for example questionnaire or even open-ended interviews and promote cooperation, understanding, and teamwork among the users, developers, and customers.

Nuymba et al. (2017) describe the steps for the focus groups elicitation method as follows:

1. *Research design*. The process begins with identifying the main aim and defining the key research objectives of the study. List of questions and a schedule are defined. Participants, including a facilitator and an assistant are identified and recruited and a location for the sessions is agreed on.
2. *Data collection*. The session is prepared and held. The facilitator introduces the ground rules and observes the discussion and tracks the questions for completion and follow up on themes of discussion. An assistant observes non-verbal interactions and the impact of the group dynamics, documents the content of the discussion, supplementing the data. The duration of the meeting should be 1–2 hours, depending on the complexity of the topic and the number or questions and participants. The meeting is recorded.
3. *Analysis*. For this phase there are different options on how to analyse the collected data. The options include, listing/ranking, coding (key ideas, themes), content, discourse or conversation analysis.
4. *Results and reporting*. The report is tailored to meet the needs of the target audience. The findings are shared with the participants of the study.

Focus group sessions can look at current work practices and new ideas or perspectives from different individuals with distinct profiles. Focus groups help to understand both system scope and actual needs better than using traditional methods based on requirements specification, and the discussions can gather perspectives from different stakeholders that will ultimately help in the acceptance and usefulness of the information system. (Farinha & Mira da Silva, 2009.)

Nyumba et al. (2017) note that the focus group discussion created a forum to discover the "unexpected" as it allowed for negotiation and evaluation of research problems and findings between different stakeholders. Furthermore, focus groups is relatively inexpensive and easy to conduct. Most people love to be asked their opinion and are generally not shy about voicing it. (Simon, 1999.)

Krueger and Casey (2000) point out that participant recruitment can be expensive and difficult. But they refer to situations where the suitable participants are searched for example with telephone or even door-to-door canvassing. In this

case, the participants would be firstly the existing system users and then other stakeholders, that we already know the majority of. Therefore, in this case, that will be dismissed from the cons listing.

Focus groups can also be held as an online meeting or in a chat room. Nyumba et al. (2017) list as a problem that these discussion platforms are only accessible to participants with access to the internet and are prone to technical problems such as poor or loss of connectivity and failure to capture nonverbal data. Asking all participants to use a webcam in the meeting could allow for an environment closer to in-person connections. And how much nonverbal data can be really captured even in a face-to-face situation from participants that we do not know very well, is another question. In this case, all the participants' working environments require for them to have an internet connection, although connection problems can always occur.

As a summary, not many issues can be listed to the comparison table's cons side of focus groups. In this case, there all two specific types of users of the system – the customer users and the case company users, of which both use different parts of the system. One focus group session could be possible held with each user group, so that the length of the session would not get too long and to allow the company users to speak more freely.

3.2.3 Joint Application Design (JAD)

The main motivation for the development of Joint Application Design (JAD) by IBM in the late 1970s was to set up structured meetings, known as JAD sessions, to improve communication between user representatives and expedite decision making with all present (Wood & Silver, 1989).

JAD was introduced to alleviate the problem of poor requirements engineering. It is a facilitated group method that puts the emphasis on the human factors of systems development and confronts the communication issues (Wood & Silver, 1995). JAD brings together managers, information systems users, and developers from different sectors of an organisation in a face-to-face workshop to specify requirements and specify functional alternatives for the system under development (Kettelhut, 1993).

Wood and Silver (1995) describe the JAD process. It consists of 5 stages:

- *Project definition*: Determine system purpose, scope and objectives. Identify JAD team members. Establish project schedules.
- *Background research*: Gather background details about the user requirements. Explore the technical, social, political implications. Consider general system issues, agree what needs to be decided in the session.
- *Pre workshop preparation:* Prepare for the session. Finalise logistics for the meeting. Gather visual aids, working documents, and other meeting equipment. Train the scribe(s).
- *The workshop:* Pool the information and knowledge of JAD team members in the analysis of potential solution. Generate solutions (systems

requirements) during the three- to five-day session. Finalise and document meeting decisions.
- *Final documentation:* Prepare the final document that captures decisions and agreements made in the workshop.

JAD is commonly used in systems planning and systems analysis activities to obtain group consensus on problems, objectives, and requirements. Specifically, the requirements engineer can use JAD sessions for example for defining the concept of operation and system goals, requirements elicitation, requirements analysis or requirements document review. (Laplante, 2009.)

Duggan and Thachenkary (2003) list the reasons why JAD became so popular at IBM:

- It was seen useful in reducing the cycle time for systems development.
- Early adopters were fascinated by JAD's capabilities in team rapport building.
- JAD is compatible with several development methods such as rapid application development (RAD), prototyping, and information engineering.

While JAD has been well received and has contributed to improved systems requirements, its group technique uses free interacting, where spontaneous communication occurs without effective control. This makes JAD dependent on good facilitation to keep the dysfunctional behaviours in control in the meetings. (Duggan & Thachenkary, 2003.)

JAD workshops are typically supported by a variety of visual aids including flip charts and post-it notes as JAD was originally designed for face-to-face meetings. In this case, the stakeholders are in many different cities, so it would be mandatory to have the meetings virtually. There are limited number of research or guidebooks on how to host the JAD workshops online. But as JAD looks promising for this case, we would have to look into that further.

3.3 Prototyping methods

3.3.1 Evolutionary prototyping

A prototype of a system is an initial, ideally quick to make and inexpensive version of the system, which is available in an early phase of the development process (Kotonya & Sommerville, 1998).

Prototyping is a popular requirements elicitation method because it enables users to develop a concrete sense about software systems that have not yet been implemented. By visualising the software systems to be built, users can identify the true requirements that may otherwise be impossible. (Fu, Bastani & Yen, 2008.)

Prototyping methods mainly fall into one of the two main categories: throwaway (or rapid) prototyping and evolutionary prototyping. The main difference

of these two is that in throwaway prototyping a disposable prototype is created to help elicit and analyse system requirements, which are hard to understand for the customer. In evolutionary prototyping a workable system prototype with limited functionality is made available to the users early in the development process and modified and extended later to produce the final system. (Kotonya & Sommerville, 1998.) Evolutionary prototyping focuses on well understood requirements (Fu, Bastani and Yen, 2008). The prototypes permit early evaluation since they can be tested in various ways, including traditional usability studies and informal user feedback, throughout the design process (Sears and Jacko, 2009).

Fu, Bastani and Yen (2008) list pros and cons for evolutionary prototyping:

- Prototyping is especially useful when there is a great deal of uncertainty.
- Reduced time and cost. Problems can be detected in the early stages, reducing the overall cost of the product.
- Concretely presents the system operations and highlight design decisions.
- Stakeholders from all parties can actively get involved in the development process.
- Too high expectations that the final deliverable will come quickly based on how refined and ready the prototype seems.
- Poor quality code from the prototype may remain in the final system due to the tendency of reusing previously written code fragments.
- Lack of mechanisms for requirements traceability.
- In some cases, prototypes are not very quick to develop due to system complexity and technical limitations.

In this case, prototyping could be easily combined with other elicitation methods to test the elicited requirements. Problem could be in finding the developer to build the prototype.

3.3.2 Rapid prototyping

The goal of rapid prototyping is to develop prototypes very quicky, in a fraction of time it would take to develop a working system. With a short prototype-evaluation cycle, the design team can evaluate more alternatives and iterate the design several times, improving the likelihood of finding a solution that successfully meets the user's needs. Rapid prototypes also serve to cut off unpromising design directions, saving time and money. It is far easier to reject an idea based on a rapid prototype than a more fully developed one, and one reduces the chance of spending a great deal of time and effort on a design that ultimately does not work. (Sears & Jacko, 2009.) Rapid prototyping focuses on unclear requirements (Fu, Bastani & Yen, 2008) which can be easily tried out and experimented with as the prototypes can be discarded with a low threshold.

There are various ways to do rapid prototyping. How rapid is rapid, depends on the project and the selected method. *Sketches* can be created in a few

minutes. With only paper and a pencil and by playing the roles of both the user and the system, designers can get a quick idea of a wide variety of different layout and interaction alternatives.

A mock-up is a 3D illustration used by architects in buildings design, but they can also be useful for interactive system designers. Typically made from cardboard, paper or foam, mock-ups help the designer imagine how an interactive system will be incorporated into a physical space. (Sears and Jacko, 2009.) For an example, a mock-up can be used in representing how a user would interact with a mobile device in the actual environment where the system would be used.

A noninteractive simulation is a computer-generated animation that represents what a person would see of the system when watching over the user's shoulder. (Sears and Jacko, 2009.) In an *interactive simulation* the user can write text to fields and click buttons to move between different pages or windows without any real functionality behind. A common tool for creating simulations is, for an example, Adobe After Effects but Microsoft Excel or PowerPoint can work just as well, depending on the system's planned user interface. Excel could be used to prototype form-based interfaces, where multiple pages can be presented with its tabbed interface. PowerPoint is already a popular tool for application, wire frame and website design prototype creation.

A precise and interactive prototype unsurprisingly takes more time to build than an imprecise or a noninteractive one. A prototype produced in a week may still be considered rapid if the final system is expected to take months or years to build.

The involvement of all stakeholders in the development of a system is at a higher level when reviewing early prototypes. Prototyping encourages communication between everyone concerned with the effort. (Jones & Richey, 2000.)

Sometimes the primary rationale for using the rapid prototyping method is that it is presumed to reduce the time needed to complete a design and a development project. However, Jones and Richey (2000) found no evidence of cycle-time reduction. That can be presupposed because prototype evaluation ensures that revisions to the product or process occur early in the project. This is linked to the misconception which prototyping creates described by Fu et al. (2008), that the project or product must be nearly finished based on how refined and ready the prototype seems.

It is not unusual to build and use multiple prototype formats in one project. Often the initial prototype emphasises only the visual aspects of the final product because these are less costly and less demanding to build. After more decisions of the product have been made, an executable prototype may be constructed to determine the product's usability. (Jones & Richey, 2000.) Prototyping can also be easily combined with other elicitation methods, such as by using a prototype to provoke discussion in a group elicitation method or as the basis for a questionnaire (Nuseibeh & Easterbrook, 2000).

Jones and Richey (2000) tested the use of rapid prototyping methods in two projects conducted in a natural work setting. The studied projects showed that the created products

- were usable for a conveniently long period of time without revision,
- were delivered faster than would have been expected using traditional methods, and
- were received by satisfied customers who had been involved throughout the development.

In this case, the stakeholders already have ideas on how the system could be evolved, made more usable and time saving. If we would first gather these ideas (for instance with some form of the interviews method), then either of the prototyping methods could be a useful to simulate proposed updates to the system and the process. In rapid prototyping the prototypes can be made with any visualisation tool, so there is almost no restrictions on who can make them.

3.4 Cognitive methods

3.4.1 Card sorting

Card sorting is a popular user-centered design method that can be used in designing an effective information architecture of a website or an application. Information architecture is the practice of effectively organising, structuring, and labelling content. It is used to create a navigation structure, including categories of content and what they are named. (Righi et al., 2013.) In card sorting, stakeholders are asked to sort cards in categories, each of which has name of some domain entity (Nuseibeh & Easterbrook, 2000).

Before the session, the stakeholders should already be familiar with the categories (Maiden, 2009). Entities are sorted into categories of the respondent's choice and the groupings can then be compared across respondents. The entities might be pictures (picture sorts), physical items (item sorts), or names of entities, descriptions of situations, and so forth, on cards.

There are several versions of card sorts. Upchurch et al. (2001) prefer a version that involves asking the stakeholders to sort all the cards repeatedly, using a different basis for the sorting each time. This way, the range of sorting criteria that the respondents used can be identified and the groups into which the cards are sorted within each criterion compared.

Card sorting can be open or closed. In an open sorting, the stakeholders decide the category names, when closed card sorting uses predetermined category names. Card sorting can also be flat or hierarchical. In flat card sorting, the stakeholders sort all the cards using one criterion. In a hierarchical sort, the cards sorted into categories are further sorted into subcategories until no further sorting is possible, resulting in hierarchies of categories and criteria. (Maiden, 2009.)

Card sorting encourages stakeholders to verbalise information about categories of items and criteria that differentiates them. The analyst can use card sorting to acquire information about which items are similar, which are different, and

in what ways. Such information can be particularly useful for an example, when eliciting requirements for grouping items and making decisions between available solutions. (Maiden, 2009.) Card sorting deals primarily with knowledge at a shallow level, as the method has limited efficiency in eliciting hierarchies but it can be used for differentiating objects by eliciting details about their attributes (Corbridge et al., 1994).

Maiden (2009) points out that card sorting is simple to understand and simple to do, and therefore usable with most stakeholders if they are sufficiently familiar with the system items to be able to sort them. It is low-tech, and the cards can be taken almost anywhere. The results are reported in a standardised format, which minimises the coding effort needed for using the results. Card sorting is also pliable to automated tool support but it requires that the system has suitable and sufficient semantic spread across the domain. (Maiden, 2009.)

Card sorting performs well in relation to other methods in terms of speed and quantity of elicited information but on the other hand, the data acquired through card sorting cannot be easily transferred for statistical analysis (Upchurch, Rugg & Kitchenham, 2001).

In this case, card sorting could be used in designing the information architecture of the return system web page, and in defining the process and the way different pages are linked when a user is performing a return or searching for information.

3.4.2 Laddering

Laddering is a structured questioning method derived from the repertory grids method, aiming to establish a hierarchy of concepts. As repertory grids involves comparing three elements (concepts) from a domain in order to elicit constructs (attributes) of a system, laddering was developed to clarify the relations between the constructs and organising them into hierarchical relations. (Corbridge et al., 1994.) Laddering has been used for example as a tool in marketing for obtaining insight into consumers' buying motives (Grunert, 1995). According to Nuseibeh & Easterbrook (2000), laddering uses probes to elicit structure and content of stakeholder knowledge.

The first step in laddering is to ask the respondent to mention the main product attributes that are used to build a value hierarchy. Corbridge et al (1994) describe the initial questioning process:

1. The knowledge engineer performing the laddering will ask the stakeholder to generate a starting point (seed item). If we use the returns management system as an example, this could be a general term “fault”.
2. Then we move down the hierarchy by a question (probe) designed to elicit examples of different types of fault. The expert may then reply, "water damage, dropping damage, installation fault".
3. The hierarchy is expanded by asking for examples of "installation fault" and so on. In this way moving down the hierarchy, the question "Can you provide *examples* of this *class*?" is asked.

4. Similar questions are then made to move across the hierarchy: "What alternative examples of this *class* are there to *Item 10,* or up the hierarchy: "What are *Item 1* and *Item 2* examples of?
5. Additional questions of the type "What is the key difference between *Item 1* and *Item 2*?" are used to elicit attributes for concepts in the hierarchy.

In the next step, the interviewer uses repetitive questions to dig deeper into the discussion about attributes to uncover personal values, such as "why is this important to you," "what does it mean to you," and "what the meaning for you is that this product has (or not) this attribute" (Veludo-de-Oliveira et al., 2006).

The aim of laddering is to produce detailed hierarchies, and from the hierarchies the requirements engineer can conclude what are the more important and less important requirements of the system for the interviewed stakeholder (Corbridge et al., 1994).

Laddering is useful in eliciting explanations of technical or subjective terms (Upchurch, Rugg & Kitchenham, 2001).

It depends on the nature of the domain when laddering is a suitable method. In classificatory domains, laddering can be used to its full extent. In diagnostic and fault-finding systems, it is common for hypotheses to be structured in hierarchies. An expert system solving a diagnostic problem might work with such a hierarchy and make decisions based on the presence or absence of attributes at each level. If the attributes were poorly defined in the actual case, then the system might use the attributes to suggest further tests to refine the hypothesis. Laddering has also been successfully used in areas where the solution to a problem is reached by a generate and test mechanism. An example of this is solving an office allocation problem, where laddering is used for both the structure of the building and the details of the staff. (Corbridge et al., 1994.)

Veludo-de-Oliveira et al. (2006) list some criticism towards the laddering's data collection practices:

- Repetitive questions can make the interview become exhausting or boring. It will help if the peculiarities of the method are discussed with the respondent before the interview.
- When asking the "why" questions, the interview may get to a too abstract level as the respondent tries to answer in a rational way, trying to find arguments to justify their behaviour.
- In this kind of interviewing method, the questions may become too personal and cause the stakeholder to get uncomfortable.

In this case, returning devices can be a long chain of tasks involving several stakeholders and systems. The laddering method could be used in eliciting requirements related to the order and importance of actions in the process.

3.4.3 Protocol analysis

The protocol analysis method consists of asking the users to perform task and *think aloud* as they work. The process of verbalisation is claimed to reveal assumptions, inferences, misconceptions and difficulties that the users are facing when solving problems and performing tasks. (Benbunan-Fich, 2001.) Furthermore, the user thinking aloud provides the observer with insights into the cognitive processes used to perform the task (Nuseibeh & Easterbrook, 2000).

According to Benbunan-Fich (2001), protocol analysis is an appropriate method for the research of process tracing, knowledge acquisition, model formulation, and decision-making behaviour, and also has been used widely in information systems literature for systems development tasks, model formulation for decision support systems, and for usability studies of computer-based systems or interfaces.

Protocol analysis is based on the direct observation of a real interaction between the user and the system. The users are instructed to give a running commentary on what they are doing, what problems they are facing and other task-related thoughts. The user's speech, keystrokes and actions on screen are recorded. The recordings are then transcribed into text and divided into segments, which are then further assigned into different categories (Cabello & Hora, 2002).

Protocol analysis of users interacting with a web site can answer the following research questions: how easy the system is to learn and to use, how flexible is it to accommodate different types of interaction styles or users, and why users form specific opinions or attitudes towards a system (Benbunan-Fich, 2001).

Benbunan-Fich (2001) speaks for the use of protocol analysis and sees protocol analysis as one of the best methods to examine user's interaction with a system from the perspective of objective usability, direct experience and perceived ease of use. Users' verbal descriptions of their actions can reveal specific usability problems and features that elicit negative opinions or customer dissatisfaction. Protocol analysis offers a wealth of information that is generally not available through other methods and due to the richness of the data collected, the sample size does not need to be large. (Benbunan-Fich, 2001.) Nielsen (2000) agrees to the previous statement that typically only a small group of users is required for protocol analysis to yield important results. A sample of five users can uncover 80% of the site-level usability problems (Nielsen, 2000).

There is a lack of common procedure for protocol analysis, with different researchers using procedures that differ, such as the presence or absence of the researcher during the session, the use of *thinking-aloud* practice exercises, the type of instructions provided, and the measures for maintaining experimental control (Cabello & Hora, 2002).

When a user is asked how they did the task after an experiment, they sometimes describe their actions and experiences in a way that that is inconsistent with their recorded behaviour (Ericsson & Simon, 1993).

Cabello and Hora (2002) note as a problem that protocol analysis requires significant time and effort due to resampling and transcribing the many hours of tapes recorded of the sessions. In this case, if we disregard the recording of

keystrokes, protocol analysis is possible to implement using the recordings of basic video conference tools, for an example, Microsoft Teams. And as only a small group of stakeholders would be needed as the users, the requirements elicitation using the method could be done in a rather short period of time.

3.4.4 Repertory grids

Repertory grids is about constructing an attribute matrix or a table for entities, by asking stakeholders for attributes applicable to entities and to rate each attribute with a value (Nuseibeh & Easterbrook, 2000). Repertory grids are mainly used in cases where the stakeholders are domain experts (Laplante, 2009).

The grids are a feature or a quality matrix in which the rows have the system entities and desirable qualities, and columns represent ranking values given by each of the stakeholders. While the grids can incorporate both qualities and features, it is usually better to have an all features or all qualities matrix to establish consistency of analysis and dispute resolution. (Laplante, 2009.) An example of a system quality entity on a matrix row could be "safety", which the stakeholders would then give a value from 1 to 5 based on the quality attribute's importance for them.

The values are either numeric or specific types of nominal values (yes, no, or not applicable). According to Upchurch, Rugg & Kitchenham (2001), this can lead to difficulties in representing domains consisting mainly of nominal categories—discrete categories such as operating system are difficult to represent in the matrix.

The strength of repertory grids is that grids composed of numeric values can be analysed using various statistical approaches, including correlations and principal component analysis. In addition, software for statistical analysis using the repertory grids methods are readily available. (Upchurch, Rugg & Kitchenham, 2001.)

Repertory grids can be very helpful in solving disputes and identifying agreements or disagreements between the stakeholder groups in an early phase of software development (Laplante, 2009).

There were no sources to be found where the repertory grids method would have been used with requirements elicitation as a goal in system development context. But as Laplante (2009) states, as it could be used in identifying the different viewpoints of different stakeholder groups, repertory grids could be useful in weighing between different solution options when upgrading the system and between opposing views of some functionality.

3.5 Contextual methods

3.5.1 Conversation analysis

Conversation analysis is a qualitative research method where data is collected by recording naturally occurring interactions. Audio and video material are then transcribed, and an archive is created of them. (Rapley, 2007.)

Such as the ethnographic observation method, conversations analysis aims to improve the understanding of human thought and logic behind actions through interpretation of human actions in context (Sommerville et al. 1993).

Conversation and interaction analysis are only applicable to situations with social interaction as these methods rely on verbal data. But the most important limitation of these methods is that they are very labour intensive. It can take a very long time to produce a transcript from a videotape of live interaction. Another limitation is that these methods cannot be directly applied to the study of systems that have not yet been built. However, they can be used to obtain tacit knowledge, because they bypass the unreliable explanations of users, and instead tell what the users really do. (Goguen & Linde, 1993.)

As conversation analysis requires recording naturally occurring interaction and for the analyst to be present at the customer's location, this method is not further investigated due to travelling and time limitations.

3.5.2 Ethnographic observation

Ethnographic observation or just ethnography is a process which was originally developed by anthropologists to understand social mechanisms in “primitive” societies. Sommerville et al. (1993) describe it involving an anthropologist spending an extended period of time (sometimes several years) living in a society and making detailed observations of its practices. Subsequent analysis of these observations revealed information about the structure, organisation, and practices of these societies. Key characteristics of ethnography are that there are no presuppositions about the society being studied, there is no list of questions to be answered and (in principle at least), the ethnographer does not try and impose his or her value judgements on the practices which are observed. (Sommerville et al. 1993.)

Lucy Suchman (1983) first identified the potential value of ethnography in deriving computer system requirements. In software development context, ethnography involves the analyst actively or passively participating in the normal activities of the users over an extended period of time whilst collecting information on the operations being performed (Aurum & Wohlin, 2006).

The goal of ethnographic research is to improve our understanding of human thought and action through interpretation of human actions in context. Ethnography differs from most other methods as it focuses more on interaction between participant and the system rather than the data. (Sommerville et al. 1993.)

Ethnography is especially useful when addressing contextual factors such as usability and when investigating collaborative work settings where the understanding of interactions between different users with the system is paramount. In practice, ethnography is particularly effective when the need for a new system is a result of existing problems with processes and procedures, and in identifying social patterns and complex relationships between human stakeholders. (Aurum & Wohlin, 2006.)

Viller & Sommerville (1999) point out that ethnography has a lot to offer as a method, but also list a few cons it has:

- *Time*. Ethnography can be a very lengthy process, lasting months or even years when doing social research. Much shorter timescales are required for the requirements engineering process.
- *Results*. Ethnography tends to produce a great deal of detailed, textual description as a result. Effectively communicating these findings to requirements engineers is not straightforward.
- *Culture*. There are significant differences in language and culture between sociologists and software developers. This can lead to problems of communication between the two groups.
- *Abstraction*. It is difficult to draw design principles and other abstract lessons from a method that is concerned with the detail of a particular situation.
- *Skill*. The lack of a systematic approach to conducting ethnography makes the method dependent on the individual ethnographer's skill.

Viller and Sommerville talk about sociologists as the ethnographers and that there may be obstacles in communicating the findings as elicited requirements for software developers. In many cases the line between the roles can be thin, a sociologist might very well understand the studied area from a developer point of view and on the other hand, the developer could be the one performing the study. The communication issues between different roles should be however acknowledged.

It seems ethnography could be used to improve and learn from existing practises. In this case, when there is an existing system, the setting would naturally involve the analyst spending time with the customer users and the case company users and observing them do their work. In the current situation, it is unlikely that this could be arranged at the location of the users. If the observation could be held online, with the use of screen sharing in a meeting, this could be arranged, but even then, the observation time might be an issue. An observation that is done in a few hours can hardly be called ethnography.

3.5.3 Interaction analysis

Interaction analysis can be used to discover details of non-verbal interaction in real work environments (Goguen & Linde, 1993). Its roots are in ethnography,

especially participant observation and in conversation analysis, among other methods.

Interaction analysis, as well as conversation analysis, is built on two assumptions. The first one is that knowledge and action are fundamentally social. The second is that experimental setups cannot capture the stakeholder's naturally occurring activities the way that recording the users work in their usual tasks on video. (Jordan & Henderson, 1995.)

For the same reasons as stated for the conversation analysis method, the interaction analysis method is left out of the scope of this thesis.

4 REQUIREMENT ELICITATION METHOD COMPARISON

In this chapter the user requirement elicitation methods described in Chapter 3 are evaluated based on how the method would provide user requirements assessing the identified problems with the current system.

4.1 Categorisation

To summarise the method categorisation described in Chapter 3, the same categories are used in the following elicitation method comparison table. The category numbers shown in the first column are as follows:

1. Traditional methods
2. Group methods
3. Prototyping methods
4. Cognitive methods
5. Contextual methods

4.2 Comparison

The following table lists the different kinds of user requirement elicitation methods described in Chapter 3. The pros and cons that were found for each method from different sources are listed in the corresponding columns.

The pros or cons that do not have a marked reference are reflections of the method by the author of this thesis.

TABLE 1 User requirement elicitation method comparison.

Cat.	Method name	Pros	Cons
1	Document analysis	Efficient due to that is requires data selection instead of collection. Inexpensive when documents are already available. Documents are unobtrusive, non-reactive and stable. (Bowen, 2009.)	The documents are not produced for research purposes and usually do not provide sufficient detail for it. (Bowen, 2009.) Difficult to implement in requirements engineering when there is little documentation.
1	Existing data analysis	Can provide quick views of the research area. A fast and efficient way to begin to answer important research questions, gain valuable experience in a research area, and sometimes have a publishable finding in a short time frame. Inexpensive. (Hulley et al., 2013.)	Lacks a defined process (Johnston, 2014). No control over what data have been collected, and how (Hulley et al., 2013).
1	Interviews	More in-depth information. Effective in various situations. (Yousuf & Asger, 2015.) Could engage users in the development of the system.	Structured interviews do not allow generation of new ideas (Zowghi & Coulin, 2005). Reach only few users (Yousuf & Asger, 2015). Say-do problem (Goguen & Linde, 1992).
1	Questionnaires	Appears scientific because they use statistical analysis. Useful with a large population. Easily combined with other methods. (Goguen & Linde, 1993.) Useful when the domain is well understood by the stakeholders and the requirements engineer (Laplante, 2009).	Users are prone to state that they will require more features that they will actually use (Laplante, 2009). Questions or the administrative method cannot be changed during the process (Okwemba, 2019). A given question has a different meaning for different stakeholders. Shared meaning cannot be established between the stakeholder and the

Cat.	Method name	Pros	Cons
			interviewer. (Goguen & Linde, 1993.)
2	Brainstorming	Produces a lot of new ideas and perspectives. Open environment for discussion. (Tiwari et al., 2012.) Easily modifiable, and modifications available (Laplante, 2009).	Large amount of material with no value (Nijstad & De Dreu, 2002) Productivity loss in large groups (Lamm & Trommsdorff, 1973). Evaluation apprehension (Collaros & Anderson, 1969). Freeriding on the efforts of others (Diehl & Stroebe, 1987).
2	Focus groups	Inexpensive Easy to conduct as users usually like to share their opinion (Simon, 1999). Promotes cooperation, understanding, and teamwork among different stakeholders. Generates complete and valid results to improve work practices. Helps in stakeholders' acceptance and later usefulness of the system. (Farinha & Mira da Silva, 2009.)	Participant recruitment can be expensive and difficult (Krueger & Casey, 2000). Requires an experienced moderator (Farinha & Mira da Silva, 2009).
2	Joint Application Design (JAD)	Improves communication (Wood & Silver, 1989). Brings together stakeholders in different positions (Kettelhut, 1993). Reduces cycle time for systems development. Develops team rapport. Compatible with several development methods. (Duggan & Thachenkary, 2003.)	Dependent on excellent facilitation and control (Duggan & Thachenkary, 2003.) JAD sessions are long, time consuming (Wood & Silver, 1989).
3	Prototyping	Available in an early phase of development process. (Kotonya & Sommerville, 1998.) Permit early evaluation since the prototypes can be tested in	Too high expectations for how ready the system is. Poor quality code from the prototype may remain in the final system.

Cat.	Method name	Pros	Cons
		various ways (Sears and Jacko, 2009). Users can develop a concrete sense of the system before implementation. Allows identification of requirements that may otherwise be impossible. Concretely presents system operations and highlights design decisions. Stakeholders from all parties can get involved. Reduced system development time and cost. (Fu, Bastani & Yen, 2008.)	Lack of mechanisms for requirements traceability. Not very quick to develop for complex systems. (Fu, Bastani and Yen, 2008.) Requires a developer to build.
3	Rapid prototyping	Prototypes are quick to make and inexpensive. Different design options can be evaluated and iterated quickly. Improves the likelihood of finding a solution to meet user's needs. Unpromising design directions can be cut off to save time and money. (Sears and Jacko, 2009.) Different alternatives available with varying amount of complexity, which can be used in same project. Helps in creating more usable products. Speeds up the development. Increases customer satisfaction. (Jones & Richey, 2000.) Almost no restrictions on who is able to make the prototypes.	False implication that the product development is also going to be rapid (Sears and Jacko, 2009). Too high expectations for how ready the system is. Lack of mechanisms for requirements traceability. (Fu, Bastani and Yen, 2008.)
4	Card sorting	Simple to understand and do, and therefore usable with most stakeholders. Low-tech, the cards can be taken and used almost anywhere.	Not easily amenable to statistical analysis. (Upchurch, Rugg & Kitchenham, 2001.)

Cat.	Method name	Pros	Cons
		Amenable to automated tool support. (Maiden, 2009.) Performs well in terms of speed and quantity of elicited information (Upchurch, Rugg & Kitchenham, 2001).	Stakeholders must be sufficiently familiar with the system items. The system must have suitable and sufficient semantic spread across the domain. (Maiden, 2009.)
4	Laddering	Useful when identifying stakeholders' motives and knowledge (Corbridge et al., 1994). Useful in eliciting explanations of technical or subjective terms (Upchurch, Rugg & Kitchenham, 2001).	Repetitive questions can make the interview become exhausting. "Why" questions may get the interview to a too abstract level. Questions may become too personal. (Veludo-de-Oliveira et al., 2006.)
4	Protocol analysis	One of the best methods to examine the interaction among usability, user experience and ease of use. User's verbal descriptions of actions can reveal usability problems and features that bring customer dissatisfaction. Does not need a large sample size due to the richness of the collected data. (Benbunan-Fich, 2001.)	Users may describe their actions inconsistently to their recorded behaviour. Lack of common procedures, such as presence or absence of the researcher in the session, the type of instructions provided and the measures for maintaining experimental control (Cabello & Hora, 2002).
4	Repertory grids	Numeric values can be analysed using various statistical approaches. Software for further statistical analysis are readily available. (Upchurch, Rugg & Kitchenham, 2001.) Useful in solving disputes and identifying agreements or disagreements between the stakeholder groups in an early phase of software development (Laplante, 2009).	Complex systems are difficult to represent in a nominal matrix. (Upchurch, Rugg & Kitchenham, 2001.)
5	Conversation analysis	Records of naturally occurring interactions improve understanding of human thought	Very labour intensive. Can only be applied on late phase of software

Cat.	Method name	Pros	Cons
		and action (Sommerville et al. 1993.)	development. (Goguen & Linde, 1993.)
5	Ethnographic observation	Useful when addressing contextual factors such as usability. Useful in finding out interactions between users. Effective when the need for a new system is in correcting existing problems and improving processes. Effective in identifying social patterns and relationships between users. (Aurum & Wohlin, 2006.)	Takes place in user's location (Sommerville et al. 1993). A lengthy process. Produces a large amount of data. Communicating the results and abstracting details of one situation into a design principle is not straightforward. Requires a skilled ethnographer. (Viller & Sommerville, 1999.)
5	Interaction analysis	Useful in discovering details of non-verbal interaction in real work environments (Goguen & Linde, 1993).	Very labour intensive. Can only be applied on late phase of software development. (Goguen & Linde, 1993.)

5 DISCUSSION

In this study, we have answered to the research problems as a literature review. The first research problem was to search for different user requirement elicitation methods that could be used for gathering user requirements for a system upgrade. The second problem was to find pros and cons for the selected methods. The found elicitation methods were described, and pros and cons of them were summarised to a table for easier comparison.

All methods described in this thesis can be used for system development, and every one of them has its pros and cons. As expected, pros of the methods were easier to be found, as can be seen from the comparison table in Chapter 4. It was mostly the method comparing articles that had the cons side, and cases that the described method would not fit, also included.

If asked to select one method that appears the most in the referenced documents that compare different requirements elicitation methods, interviews takes the lead, and for a good reason. The different types of interview options and the variability of it makes it suitable for most contexts. Although not most effective when used on their own, other maybe less popular methods combined with interviews can enhance interviews effectively when specific types of information are required.

One challenge when comparing the methods is that they are on different levels. Some can be used to improve a small detail of a system (for an example, card sorting) and some of them can be used in all phases of system development (like questionnaires). On the other hand, for example in the prototyping method, prototypes are created and tested iteratively and then other methods such as interviews or protocol analysis are used to test the prototypes. Prototypes used on their own would provide much less information, and therefore comparing it as is would be ineffectual.

When deciding on the elicitation methods for this case, one criterion is that the method does not require the analyst to be physically located in the same space with the stakeholder during the research. Therefore, for a method to be selected for further study, it must be possible to have the sessions facilitated online. As

mentioned in the summary of the previous chapter, it was hard to find how some methods could be adapted to a system development point of view. Furthermore, if the referenced literature is a bit older, the method may have not seen the update to the use of more modern tools. For some methods, the way how they could be used online is obvious, but for some there seems to be some reinventing of the method to be done.

Based on this study, we will select two or three of the methods to be used in upgrading the case system. As multiple sources suggest, different methods can support each other by bringing up different kinds of user requirements. We can also make a hypothesis that different methods are more suitable for each stakeholder who are their own personalities, one may be very open in interviews, and others point out possible requirements more freely while using the system.

We hope that this study brings more tools to the toolbox for the novice requirements engineer and helps in deciding which to use for the challenge ahead.

REFERENCES

Arif, S., Khan, Q., & Gahyyur, S. A. K. (2010). Requirement Engineering Processes, Tools/Technologies, & Methodologies. *International Journal of Reviews in Computing*, ISSN: 2076–3328, Vol.2.

Bohem, B. R. W. (1981). *Software engineering economics*. New Jersey: Prentice Hall.

Bowen, G. (2009). Document Analysis as a Qualitative Research Method. *Qualitative Research Journal, 9*(2), 27–40.

Brooks F.P. Jr. (1987). No silver bullet: Essence and accidents of software engineering. *IEEE Computer, 20*(4), 10–19.

Carrizo, D., Dieste, O., & Juristo, N. (2014). Systematizing requirements elicitation technique selection. *Information and Software Technology, 56.* 644–669.

Corbridge, B., Rugg, G., Major, N. P., Shadbolt, N. R. , & Burton, A. M. (1994). Laddering – technique and tool use in knowledge acquisition. *Knowledge Acquisition, 6*(3), 315–341.

Davis, A. (1992). Operational Prototyping: A New Development Approach. *Software, 9*(5), 70–78.

Dennis, A. & Valacich, J. (1993). Computer brainstorms: More heads are better than one. *Journal of Applied Psychology, 78*(4), 531–537.

Diehl, M. & Stroebe, W. (1987). Productivity loss in brainstorming groups: Toward the solution of a riddle. *Journal of Personality and Social Psychology*, *53*(3), 497–509.

Duggan, E. W. & Thachenkary C. S. (2003). Higher Quality Requirements: Supporting Joint Application Development with the Nominal Group Technique. *Information Technology and Management, 4*(4), pp. 391–408.

Ericsson, K. A. & Simon, H. A. (1993). *Protocol Analysis: Verbal Reports As Data*. London: The MIT Press.

Farinha, C. & Mira da Silva, M. (2009). Focus Groups For Eliciting Requirements In Information Systems Development. *UK Academy for Information Systems Conference Proceedings 2009*, Oxford, UK.

Goguen, J. A. & Linde C. (1993). Techniques for requirements elicitation. *Proceedings of the IEEE international symposium on requirements engineering,* 152–164.

Grunert, K. G., & Grunert, S. C. (1995). Measuring subjective meaning structures by the laddering method: Theoretical considerations and methodological problems. *International journal of research in marketing, 12*(3), 209–225.

Hickey, A., Davis, A. M. (2003). Elicitation technique selection: how do experts do it? In: *Proceedings of the 11th IEEE international requirements engineering conference (RE'03)*, Monterey, California.

IEEE Standard 610.12-1990. (1990). *IEEE Standard Glossary of Software Engineering Terminology*.

Johnston, M. P. (2017). Secondary Data Analysis: A Method of which the Time Has Come. *Qualitative and Quantitative Methods in Libraries, 3*(3), 619-626.

Jones, C. (1996). *Patterns of software systems failure and success*. London: International Thompson Computer Press.

Jones, T. & Richey, R. (2000). Rapid prototyping methodology in action: A developmental study. *Educational Technology Research and Development*, *2*(48), 63–80.

Jordan, B., & Henderson, A. (1995). Interaction analysis: Foundations and practice. *The Journal of the Learning Sciences, 4*(1), 39–103.

Kothari, C. R. (2004). *Research Methodology Methods and Techniques*. New Delhi: New Age International.

Kotonya, G. & Sommerville, I. (1998). *Requirements Engineering Processes and Techniques.* New York: John Wiley and Sons.

Lamm, H., & Trommsdorff, G. (1973). Group versus individual performance on tasks requiring ideational proficiency (brainstorming). *European Journal of Social Psychology, Vol. 3*, pp. 361–387.

Laplante, P. A. (2009). *Requirements engineering for software and systems*. Boca Raton: CRC Press.

Maiden, N. (2009). Card sorts to acquire requirements. *IEEE Software, 26*(3), 85–86.

Merriam-Webster. (2021). Elicit. In *Merriam-Webster.com dictionary.* https://www.merriam-webster.com/dictionary/elicit.

Metersky M.L. (1993). A decision-oriented approach to system design and development. *IEEE Transactions on Systems, Man, and Cybernetics 23*(4), 1024–1037.

Nielsen, J. (2000). *Designing Web Usability: The Practice of Simplicity*. Indianapolis: New Riders Publishing.

Nijstad, B. A. & De Dreu, C. K. W. (2002). Creativity and group innovation. *Applied Psychology: An International Review, 51*(3), 400–406.

Nuseibeh, B. & Eastbrook, S. (2000). Requirements Engineering: A Roadmap. *Proceedings of the Conference on the Future of Software Engineering*, pp. 35–46.

Okwemba, R. K. (2019). *Requirement elicitation framework for re-engineering diagnostic health care information systems in Kenya.* Kolkata: Exceller Books Global Press.

Osborn, A. F. (1957). *Applied imagination: principles and procedures of creative problem-solving.* New York: Scribner.

Pohl, K. (1994). The Three Dimensions of Requirements Engineering: A Framework and its Applications. *Information Systems, Vol. 19*, pp. 243–258.

Pohl, K. (2010*). Requirements engineering: Fundamentals, principles, and techniques.* Berlin: Springer.

Rapley, T. (2018). *Doing conversation, discourse and document analysis.* London: SAGE Publications Ltd.

Righi, C., James, J., Beasley, M., Day, D.L., Fox, J. E., Gieber, J., Howe, C. & Ruby, L. (2013). Card Sort Analysis Best Practices. *Journal of Usability Studies, 8*(3), 69–89.

Rogers D. S., Lambert D. M., Croxton K. L. & García-Dastugue S. J. (2002). The Returns management process. *The International Journal of Logistics Management, 13*(2), 1–18.

Sears, A. & Jacko, J. (2009). *Human-Computer Interaction: Development Process*. Boca Raton: CRC Press.

Shaw, M. & Gaines, B. (1996). Requirements Acquisition. *Software Engineering Journal, 11*(3), 149–165.

Simon, J. (1999). How To Conduct A Focus Group. In J. Simon, *The Wilder Nonprofit Field Guide to Conducting Successful Focus Groups*, pp. 9–34, Saint Paul, Minn: Amherst H. Wilder Foundation.

Sommerville, I., Rodden, T., Sawyer, P., Bentley, R. & Twidale, M. (1993). *Integrating ethnography into the requirements engineering process.* Proceedings of the IEEE International Symposium on Requirements Engineering, pp. 165–173. IEEE.

Sommerville, I., Sawyer, P. (1997). *Requirements engineering – A good practise guide.* West Sussex: John Wiley & Sons Ltd.

Stewart, D. W., Shamdasani, P. N. (2015). *Focus Groups: Theory and Practise*. Los Angeles: SAGE Publications Inc.

Suchman, L. (1983). Office procedure as practical action: models of work and system design. *ACM Transactions on Information Systems, 1*(4), 320–328.

Taylor, D. W., Berry, P. C. & Block, C. H. (1958). Does group participation when using brainstorming facilitate or inhibit creative thinking? *Administrative Science Quarterly, 3*(1), 23–47.

Upchurch, L., Rugg, G. & Kitchenham, B. (2001). Using Card Sorts to Elicit Web Page Quality Attributes. *IEEE Software, 18*(4), 84.

Yousuf, M. & Asger, M. (2015). Comparison of Various Requirements Elicitation Techniques. *International Journal of Computer Applications, 116*(4), 8–15.

Veludo-de-Oliveira, T. M., Ikeda, A. A., & Campomar, M. C. (2006). Discussing laddering application by the means-end chain theory. *The Qualitative Report, 11*(4), 626–642.

Viller, S. & Sommerville, I. (1999). Social Analysis in the Requirements Engineering Process: from ethnography to method. In *Proceedings IEEE International Symposium on Requirements Engineering,* pp. 6–13. IEEE.

Wiegers, K. E. (1999). *Software requirements*. Washington: Microsoft Press.

Wohlin, C. (2014). Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, pp. 1–10.

Wood J. & Silver D. (1989). *Joint application design: how to design quality systems in 40% less time*. New York: John Wiley & Sons Inc.

Wood J. & Silver D. (1995). *Joint Application Development*. New York: John Wiley & Sons Inc.

Zmud R. W., Anthony W. P. & Stair R. M. Jr. (1993). The use of mental imagery to facilitate information identification in requirements analysis. *Journal of Management Information Systems, 9*(4), 175–191.

Zowghi, D. & Coulin, C. (2005). Requirements Elicitation: A survey of Techniques, Approaches and Tools. *Engineering and Managing Software Requirements,* pp. 19–46. Berlin: Springer.