Requirement Gathering for an IT Project

Nella Pihlajaniemi

Haaga-Helia university of applied sciences

Bachelor of Business Administration in Information Technology

Practice–based thesis

2023

Abstract

Author	
Nella Pihlajaniemi	
Degree	
Bachelor of Business Administration (IT)	
Title	
Requirement Gathering for an IT Project	
Number of pages and appendices	
32 + 5	

The thesis examined what requirements should be gathered before an IT project can be started and how the requirement gathering is conducted. The result of the thesis was a guidebook that offered guidelines for a successful requirement gathering and a requirement document template for storing the requirements. The goal of the thesis was that by following the pre-study guidebook a company should be able to gather enough requirements to make an educated decision about starting the implementation of the project. The theoretical foundation of the thesis was gathered from published literature around the topic of requirement gathering. This was a practice–based thesis, since in addition to the theory, it included an empirical study where the guidebook was assessed in a real project pre-study. The pre-study was conducted in 2023 and the thesis was also written in the same year.

In the theory chapter, the concepts of business, user and functional requirement were first defined and their importance in a pre-study was explained. Requirement gathering can be split into four stages: elicitation, analysis, documentation, and validation. The stages were described in detail and guidelines were given for conducting the requirement gathering activities in each stage. The guidelines also included a section about how to compile a stakeholder analysis. Lastly, the theory chapter specified what information should be gathered to create a comprehensive requirement document, why it should be gathered and the best ways for documenting the requirements. The guidebook was created based on the guidelines gathered in the theory section.

For conducting the empirical study, a project pre-study was carried out first according to the instructions in the guidebook. The project was about studying the possibilities and use cases of a new technology and how suitable the cases would be for the company. The empirical study then analysed how well the guidebook worked in practice, what could have been improved and did the guidebook serve its purpose of offering the company enough information to start the implementation phase. The study examined the usefulness of both the requirement gathering guidelines and the requirement document template.

A conclusion chapter followed the empirical study where it was concluded that the thesis and the guidebook met their goal. The company was able to make a decision about the start of the implementation and the project could move to the next phase. The company agreed on the use cases produced by the pre-study, however, a more concrete project plan was needed before the budget for the implementation was granted. Even though the initial version of the guidebook helped achieve the goal of the pre-study, small additions and improvements were made to the guidebook based on the findings in the empirical study.

Keywords

requirement elicitation, requirement document, business requirement, user requirement, functional requirement

Table of contents

1 Introduction

In my thesis, I study what requirements should be gathered before an IT project can be started and how the requirement gathering is conducted. The result of my thesis is a guidebook that offers guidelines for a successful requirement gathering for a pre-study. The guidebook has a template for documenting the requirements and other information gathered during the pre-study. The template will specify the minimum requirements that should be gathered. The guidebook can be found in the appendices. The thesis has been split in two parts. In the theory section, I will study the different theories for eliciting and documenting requirements. The second part is an empirical analysis where I will try the guidebook and the requirement template in practice in a real pre-study and evaluate their usefulness. In addition, I will examine what changes would be needed to make the guidebook more usable and suggest improvements. The guidebook attached to the thesis is the finished version made according to the findings during the pre-study.

The goal of my thesis is that if a company follows the pre-study guidebook and presents the findings to a steering group, the steering group will have enough information to make an educated decision about starting the implementation phase of the project. The guidebook should give directions to a successful pre-study where the business and user needs have been understood and possible risks or constraints to the project have been uncovered. Since the requirements have been elicited and validated together with the stakeholders, the filled-out pre-study template will also serve as a written proof of what was agreed on before implementation. When the solution has been implemented, the company can go back to the requirement document and verify whether the results are as specified.

This will be a practice–based thesis since I am doing it for my workplace. I am currently working in a project that is studying the feasibility and use cases of a solution made with a new technology. In the project pre-study, my colleague and I are gathering business and user needs, analysing how well the solution would answer to them and identifying what would be needed technically to implement the solution. I will assess the pre-study guidebook in this pre-study and review whether the requirements were enough to start the implementation. My department has started to invest more in product life cycle management, and they would like to create processes and guidelines for each phase of the product life cycle from pre-study to maintenance. My thesis will provide guidelines for the pre-study, and the guidebook will be taken into use in my department.

2 Theory

In the theory section of the thesis, first, business, user and functional requirements are defined. Second, the different stages of requirement gathering are described and how they should be conducted including how to compile a stakeholder analysis. Lastly, what information needs to be gathered to successfully carry out requirement gathering is specified in an example requirement document template. There are several ways of categorizing and documenting the requirements needed to start a project and implement a solution. Terminology also differs. Wiegers and Beatty (2013, Chapter 1) offer a useful diagram that depicts the different components to produce comprehensive requirements, and this is the categorization that is used throughout the thesis.

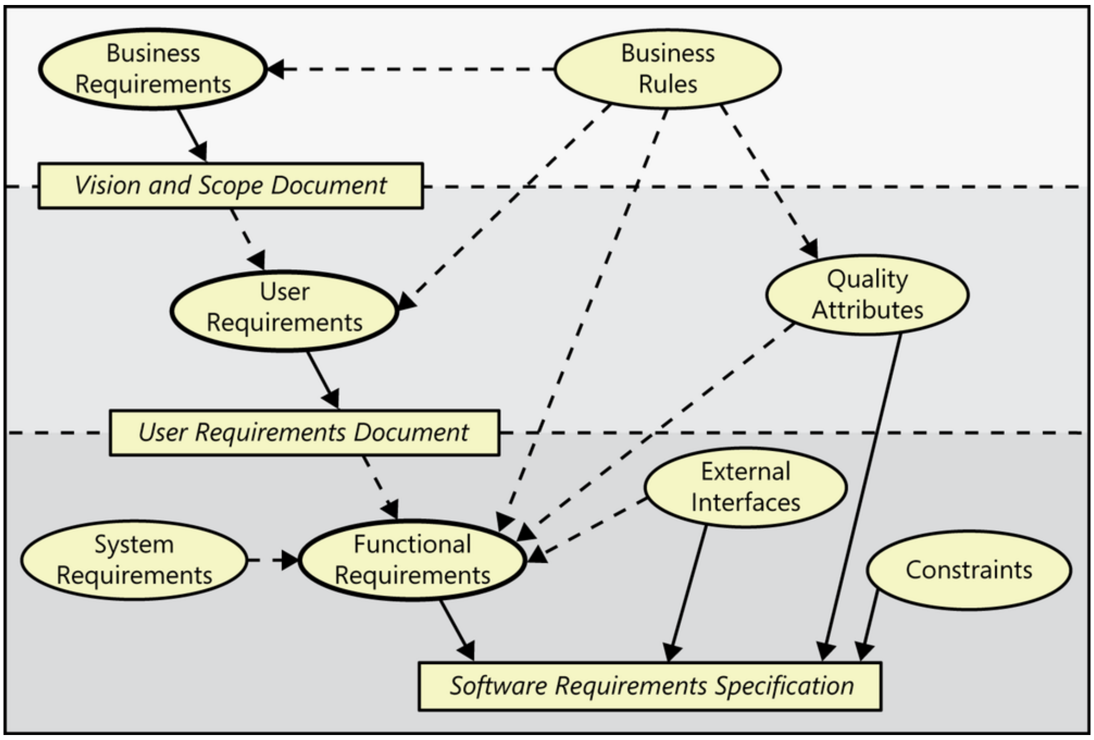

Figure 1. Relationships among several types of requirements information (Wiegers & Beatty 2013, Chapter 1)

The theory is primarily going to focus on the actual requirements and how they are elicited, analysed, documented, and validated. Business rules and constrains are touched upon. In addition, Wiegers' and Beatty's vision scope document and user requirements document will form the basis of the requirement document template. Software requirements specification is left out on purpose since it does not fit the scope of the pre-study guidebook.

2.1 Requirement types

As mentioned in the introduction of the theory section, there are different types of requirements needed to form a comprehensive understanding before the project can start. Wiegers' and Beatty's (2013, Chapter 1) categorization places the requirements into business, user, and functional requirements. In addition, Wiegers and Beatty list other categories that are not strictly speaking requirements. Business rules specify the conditions under which activities are performed. They are not requirements, but impose conditions on them, for example "Support needs to fix the critical problems within 4 hours." Quality attributes are closely linked to functional requirements since they describe how well the solution works, for example, "the solution must be user-friendly" and "photo upload must happen fast". Constraints restrict options available in the design and technical implementation. Lastly, solution ideas are ideas that the stakeholders have suggested for the final solution. They can be treated as requirements, but it will limit the ways the requirement can be implemented. However, in this chapter, the focus is on requirements and the meaning of each requirement type and what is their purpose will be explained.

2.1.1 Business requirements

Caudle discusses business requirement in his book Streamlining Business Requirements. According to him (2009, Introduction) a business requirement is "something a business requires a manual or automated system to do to satisfy a driver, such as the business's operation, vision, principle, or mission". He adds that business requirements might change over time due to, for example, changes in the business case, but they should never change to align with the solution. It might be that the proposed solution does not fit the business requirements perfectly, and then it is up to the business to fill in the gaps or accept them.

Wiegers and Beatty (2013, Chapter 5) elaborate that business requirements consist of business opportunities, business objectives, success metrics and a vision statement. Business requirements need to be gathered first before the pre-study can move to user and functional requirements since they provide a reference and lead the requirement gathering to the wanted direction. Wiegers and Beatty stress that a business should not start any project without understanding the value it will add. Because of this reason, success metrics should be defined for the business objectives to measure whether they have been met at the end of the project. The vision describes the end goal and ensures all stakeholders know where the business is hoping to go, whereas the scope defines how much of that vision will be solved in the particular project. (Winters, Johnson & Blank 2013, Chapter 2.)

Caudle (2009, Introduction) also lists attributes that make a good business requirement and explains their meaning.

- **Verifiable** means there should be a process to easily verify that the product meets the requirements.
- **Traceability** makes it possible to trace a functionality back to a business requirement and vice versa, which in turn allows more efficient testing.
- A requirement should be **unambiguous** so its description should include, for example, any rules surrounding the process, roles who can perform the process and the conditions under which the process can be performed. The requirement can leave no room for interpretation.
- Since there is no one source of truth, evaluating the **correctness** of a requirement can be difficult. Stakeholders might not have the same view on the topic, so several people should be interviewed to find the requirements that align with the businesses' vision and goals.
- The **completeness** of business requirements can be tested to ensure that the requirements have been well investigated.
- Business requirements are not system requirements so they should be **implementation-free**. Business requirements are no way dependant on the actual solution.

2.1.2 User requirements

Wiegers and Beatty (2013, Chapter 1) write that user requirements describe what the user will be able to do with the solution that will provide value to the business. They add that a broader term "stakeholder requirements" can be used since stakeholders contribute to the requirements in addition to end users. In requirement elicitation, instead of asking what users want the system to do, what users need to accomplish should be asked. User requirements make a shift from product-centric to user-centric perspective. (Wiegers & Beatty 2013, Chapter 8.) Product-centric approach is a valid option as well, but Wiegers and Beatty state that in most cases user-centric is preferable, since it leads to functionalities that are truly needed and used. In addition, it helps with requirement prioritization. The company can prioritize user requirements based on, for example, how many users would use the functionality or how important the user class is who requested it.

According to Wiegers and Beatty (2013, Chapter 8), user requirements are commonly represented by use cases and user stories. They further explain that a use case is a sequence of interactions between a system and an external actor. Some of their examples of use cases are "create an invoice", buy an item" or "view customer information". A use case describes how the user imagines interacting with the system to achieve their goals. On the other hand, user story is a concise statement that describes a feature from the perspective of the person who needs the functionality. (Wiegers & Beatty 2013, Chapter 8.) The format is typically “As a <type of user>, I want <some goal> so that <some reason>”. A corresponding user story to one of the example use cases might be “As an invoice clerk, I want to create an invoice so that I can bill a customer”. User story provides valuable additions since it specifies the user’s class and motivation.

As can be seen in the diagram below, use cases can be translated into functional requirements and tests, whereas user stories can be used in creating acceptance tests. Since functional requirements is the topic of the next chapter, the use case path is now studied in more detail. Once the business has uncovered a use case, next step is to identify the actors. (Wiegers & Beatty 2013, Chapter 8.) According to Wiegers and Beatty this can be done by asking, for example, who or what provides information or services to the system, or who or what helps the system respond to complete a task. Use cases are seldom standalone, but they are connected to other use cases and actors which all together form a flow. Wiegers and Beatty (2013, Chapter 8) argue that often functional requirements can already appear from this analysis of the actors. As an example, they use the use case of making a request. First, the requester makes a request and then the system shall assign a unique sequence number to each request. Assigning a unique sequence number is already one functional requirement the system must do. When the flow is followed and the next use cases in it are examined, more functional requirements can be discovered.

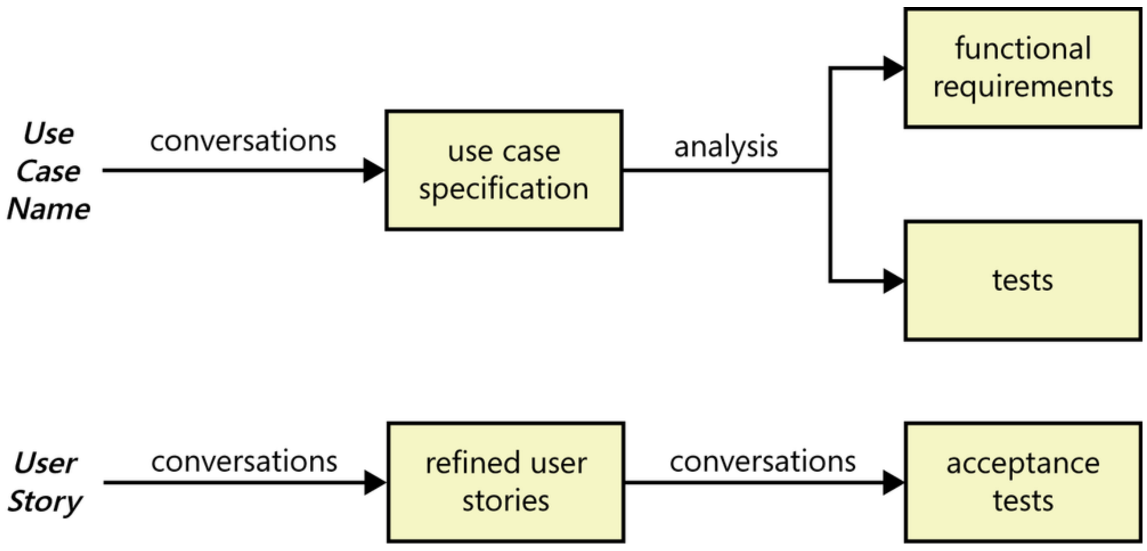

Figure 2. How user requirements lead to functional requirements and tests with the use case approach and the user story approach (Wiegers & Beatty 2013, Chapter 8)

It is difficult to discuss user requirements without mentioning personas. Users are not a homogenous mass of people with the same needs and goals; instead, they can be categorized into classes. As seen in the diagram below, Wiegers and Beatty (2013, Chapter 6) categorize users based on the tasks they perform, the features they use, the frequency of use, whether they interact with the system directly or not, and so on. Every person related to the project is a stakeholder. Then depending on how they interact with the system, the user is either favoured, disfavoured, or ignored. This kind of categorizations will also help in understanding and prioritizing user requirements. To bring the user classes to life, Wiegers and Beatty recommend creating personas for them. They write that "a persona is a description of a hypothetical, generic person who serves as a stand-in for a group of users having similar characteristics and needs" so a stereotypical representative of the user class. Persona details usually include demographic information, behaviours, needs and goals. Persona's will make it easier to understand the user classes motivation and it will also help when presenting findings to stakeholders.

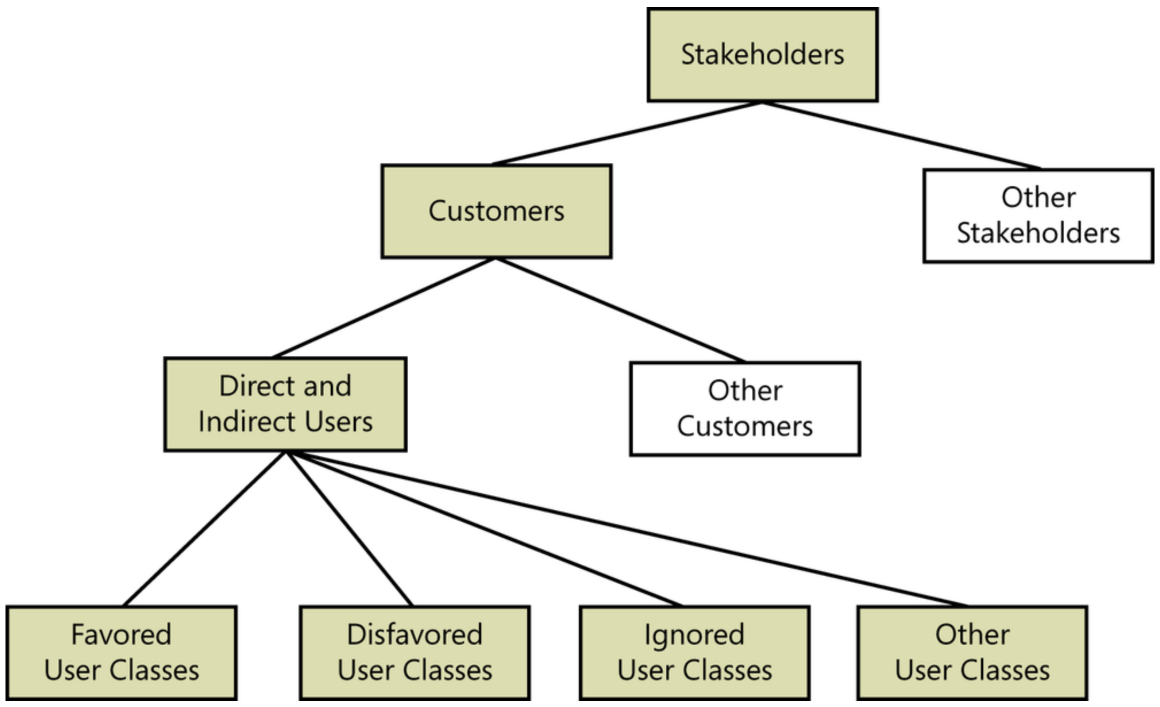

Figure 3. A hierarchy of stakeholders, customers, users, and user classes (Wiegers & Beatty 2013, Chapter 6)

Caudle's (2009, Chapter 2) models and diagrams of the relationships between business events offer more insight into how to find the events that form the basis of user requirements. He explains that a business event is a situation which the business must respond to or something the business needs to perform to reach its goals. He instructs not to begin an event with a verb since an event is not a process. Processes support events. Customer creates a support ticket is an event. How the business responds to the event is determined by the process. The next event might be a dispatcher dispatching the ticket to a technician and then the dispatcher starting the diagnosis and so on. Business rules can be used to find the events. They are guidelines that define what actions need to be taken under certain conditions. For example, a work order must be created, if a contract work is estimated to take more than 50 hours.

Caudle (2009, Chapter 3) continues to explain that the event process model (EPM) gives information not only about business events but the processes that support it, and the actors and objects in it. Actors are external entities which the business has no control over, such as customer or employee. Objects, on the other hand, can be a person, place, thing or a concept that participates in the business events and contain information required by the processes. According to Caudle, the EDM will help demonstrate to the stakeholders that pre-study team has understood what their systems do. The EDM consists of a data flow diagram that depicts the process around the event and its data flow, and a process description which is a description of the tasks done within the process. The EPM does not describe what systems or tools are used, but what the business must do to reach its objectives.

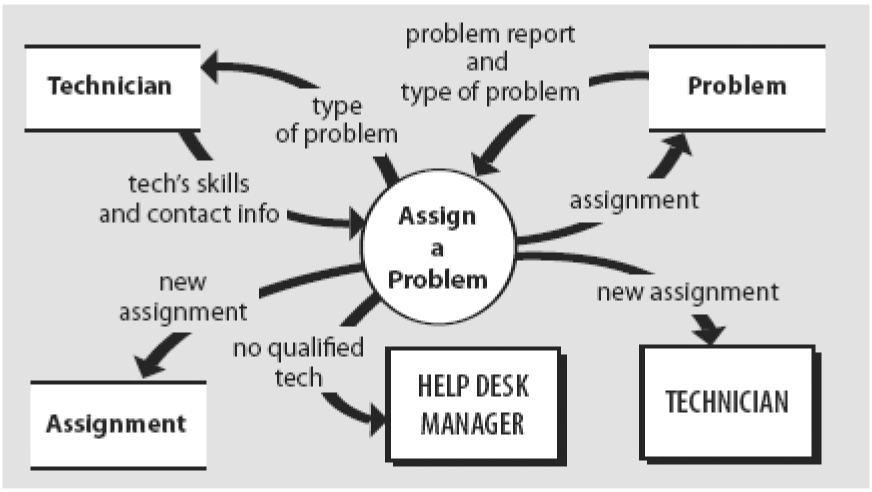

Figure 4. Data flow diagram for assigning a problem to a technician (Caudle 2009, Chapter 3)

Once the EPM has been created, the project team can start identifying missing requirements using the event entity relationship diagram (EERD). Caudle writes (2009, Chapter 4) that the diagram illustrates the relationship between the objects in an event process model. It has been used by database designers and architects. Some objects might be data attributes that describe an object, for example, price is the data attribute of product. Caudle tries to clarify the EERD using customer and customer deposit as an example. To understand their relationship, questions should be asked such as "can a customer have more than one deposit" and "can a customer not have any deposits at all". Some of these might be answered by business rules, for example, there might be a rule stating a customer cannot open an account without making a deposit.

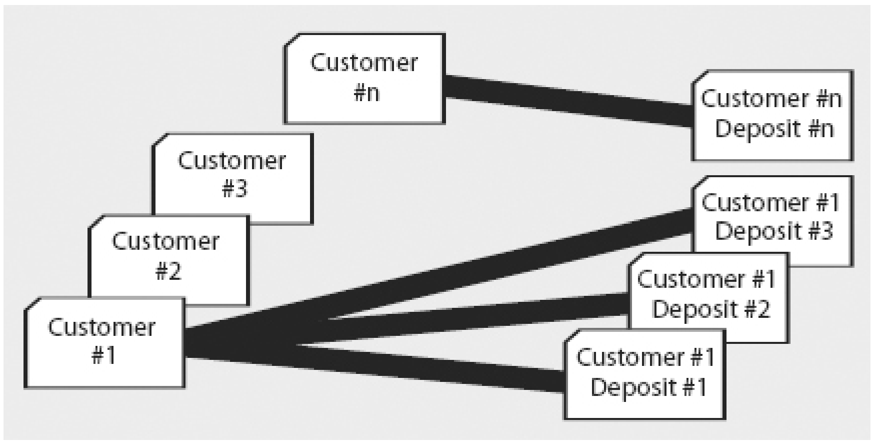

Figure 5: Relationship between customer and customer deposit objects (Caudle 2009, Chapter 4)

2.1.3 Functional requirements

Wiegers and Beatty (2013, Chapter 8) write that “software developers do not implement business requirements or user requirements”, they implement functional requirements. If business requirements describe *why* the business is implementing the solution, functional requirements describe *what* must be implemented to satisfy the business and user requirements. (Wiegers & Beatty 2013, Chapter 1.) Functional requirements describe the way the system should behave and are often written in a “shall format”, for example, “the system shall forward the feedback inputted by the user to the product support”. Functional requirements were already touched upon in the previous chapter when discussing use cases. Sometimes use cases and functional requirements are treated as the same, but Wiegers and Beatty (2013, Chapter 8) warn that the business will be in trouble if they simply pass use cases to the development team for implementation. Use cases focus on the user’s perspective, but do not touch the internal behaviour of the system. It is therefore recommended by Wiegers and Beatty that the functional requirements necessary to implement each use case are specified.

However, Wiegers and Beatty (2013, Chapter 8) make a point that the company can get some functional requirements directly from the use case, for example, “save customer information” is a use case, but the same case can be translated to a functional requirement “the solution shall save the customer information”. That is only the sunny day scenario, and the use case does not offer guidance for what will happen if the preconditions are not met, the customer information is missing or in the wrong format, or the user does not have the rights to save customer information. Separate functional requirements are needed to cover those cases. Wiegers and Beatty clarify that functional requirements can also be uncovered from postconditions, business rules and other requirements. By specifying the use case flow, different turns it can take and the rules that govern it, the requirements that the system should follow are revealed, as well. Functional requirements that come from the written-out use case flows can easily be traced back to the user requirement they came from, which helps with change management and when writing test cases.

2.2 Requirement gathering

To produce good requirements, business must have a good process for gathering them. According to Hossenlopp and Hass (2007, Chapter 1), this elicitation process is designed to give an understanding of the business environment and to gather the needs of the customers and users that the project should satisfy. They add that a good elicitation process provides the foundation to analyse, specify, document, and validate requirements. Caudle (2009, Introduction) has offered guidelines for this process of elicitation. First, the process must have structure and discipline and defined steps. These steps help the process become predictable and repeatable, since the process should always be the same regardless of the project. In addition, the process should offer a way to estimate the time it takes to complete the elicitation.

Wiegers and Beatty (2013, Chapter 1) write that requirement gathering can be split into four stages: elicitation, analysis, documentation, and validation. They also give a description of each stage. Elicitation includes activities related to discovering requirements, such as identifying stakeholders and understanding use cases and business objectives. Analysis involves reaching a deeper understanding of the requirements. Activities in this stage are distinguishing quality expectations and business rules from the elicited information, producing functional requirements, and identifying gaps in requirements. During documentation stage, the collected requirements are stored into a document in a visual and written format and in a way that they are easily understood and reviewed by the stakeholders. In the last stage, the requirements are reviewed to verify they are correct, and acceptance tests are developed for them to ensure that the requirements meet the business needs and objectives. In the next chapters, each stage is examined in more detail.

2.2.1 Elicitation

When planning for elicitation, Hossenlopp and Hass (2007, Chapter 6) recommend first identifying the stakeholders involved in the requirement elicitation and conducting a stakeholder analysis. Stakeholders can be end-users, customers, subject-matter experts, or business unit managers. Own section is dedicated for stakeholder analysis at the bottom of this chapter. Next, they advise planning elicitation activities such as workshops, interviews, prototyping, reviews of existing systems and business documents, and so forth. The deliverables should be specified, as well, so that it is clear what is expected from each elicitation session whether it is a list of stakeholders and what sessions they will participate in, interview notes, analysis of interview findings or list of requirement statements. Lastly, the activities are carried out according to the plan. Caudle (2009, Introduction) writes that participants in the elicitation need to all agree to use the process and be fully engaged. In additions, everybody’s expectations should be established before an elicitation session, and everybody should agree to follow the agenda. It helps that the sessions are led by only one facilitator. Results of elicitation should be measurable, documented in an organized manner, follow a reasonable standard of quality and be made accessible to all participants. Caudle adds that when every participant knows their role in the process and what is expected of them before each session, time is used more efficiently and everybody should feel that their time was well spend.

Caudle (2009, Introduction) lists a couple of ways how information can be gathered. The most popular one is interviewing which works well if it is clear what needs to be asked and there is only one interviewer and interviewee. As soon as there is more than one, interviewees providing conflicting answer and interviewers having different styles of questioning might become an issue. Interviews are also time consuming. Surveying is less time consuming, but producing the right questions can be challenging. Shadowing and prototyping can also be used in understanding the customer’s needs. Wiegers and Beatty (2013, Chapter 7) add user interface analysis and workshops to this list. User interface analysis will reveal user and functional requirements in existing solutions. However, the company should not assume a functionality is needed in the new solution just because it is in the existing one. According to Wiegers and Beatty, workshops might be more time consuming to organize, but they are effective in resolving disagreements and for making quick decisions. Workshops require good planning and facilitation as not to waste multiple people’s time.

In the traditional software development, the development will not start until all the requirements have been elicited, specified, and validated. In the most extreme cases, once the requirements have been gathered, they remain unchangeable. However, there is an alternative and more modern way of requirement gathering. Kelly (2019, Chapter 4) writes in *The Art of Agile Product Ownership* that “the basis of agile software development is the belief that it is better for a team to be flexible and responsive to changing requirements rather than highly adapted to fixed requirements”. Even if the company would be able to gather and document requirements that are 100 % accurately, the world changes. The needs that the requirements were born from can change already during the implementation. That is why Kelly recommends companies to take a more flexible approach to requirements and talk about discovery rather than requirements. The aim is to discover the user needs and their value. Some of the discovery takes places before development, but more importantly, the discovery will continue during development. This ensures that the company is always prepared for changing needs.

Important part of elicitation is stakeholder analysis. Hossenlopp and Hass (2009, Chapter 4) detail in their book how a stakeholder analysis is compiled. First, they define that stakeholders are people or organizations who are actively involved in the project and whose interests might be positively or negatively impacted by the outcome of the project, such as customers, sponsors and users. Stakeholder analysis identifies the stakeholders and provides a common understanding of their roles in the project and their level of interest and influence on it. Hossenlopp and Hass state that stakeholders should be identified because they are a potential source of requirements, and they can be used for elicitation and validation activities. Even if the stakeholder does not participate in the elicitation process the project team should still be aware of them since they still might have an interest in the progress of the project. Important to note is that stakeholder analysis is not only done once, but it needs to be revisited when new stakeholders emerge when the project proceeds.

Hossenlopp and Hass (2009, Chapter 4) offer guidelines also for how the stakeholder analysis is conducted. It starts by listing all stakeholders that are affected by the execution of the project or its outcome. This is done by the project team, and they can use business case information as guidance to find the linked organizations, groups and individuals and their representatives. Hossenlopp and Hass use the RACI (responsible, accountable, consulted, informed) model to map out the roles of the stakeholders. Responsible stakeholder participates in the project work activities, accountable stakeholder is accountable to the customer for the result of the project, consulter stakeholder is asked for comments on the objectives, constraints, etc., and informed stakeholder should be kept informed about the project status. By understanding the roles, the project team can uncover possible issues, for example, if too many stakeholders want to only be informed or too many want to be consulted on all requirements.

When the roles have been identified, the next step in the stakeholder analysis is to determine who is involved in what project activity and in what capacity. Hossenlopp and Hass (2009, Chapter 4) suggest using RACI resource assignment matrix which lists the activities and the stakeholders where the project team can mark what role each has. They remind readers to confirm the roles and activities with the stakeholders, and that they have time for the project. Next the project team needs to identify the stakeholder’s interest. When interviewing the stakeholders for the first time, the project team should make the expected business values of the project clear and to ask questions such as these to determine their level of interest: “What are your project expectations?” or “How does the organization benefit from the project implementation”. The interest can be prioritized which can help discovering interests that are out of the project scope. It is also important to map out the stakeholders’ power and influence on the project. Armed with knowledge of both interest level and influence, the project team can rank the stakeholders and allocate time and resources to keep the stakeholders informed and involved accordingly.

2.2.2 Analysis, Documentation and Validation

Analysis stage starts by selecting a small team of business and technology experts who can be the same people as in elicitation. (Hossenlopp & Hass 2007, Chapter 6.) They will be involved in analysing and elaborating the requirements that came from the elicitation. Hossenlopp and Hass write that the analysis activities include putting requirement information into categories, prioritizing requirements, studying if the requirement is legally, technically, operationally and economically feasible, assessing requirement risks and constraints, and modifying requirements to mitigate the possible risks. In addition, attention should be paid to the way the requirements are represented. Block of text can make it difficult to understand what is being said, so lists, tables, graphs and other visualizations should be utilized to clarify the requirements. Hossenlopp and Hass also recommend specifying the deliverables of each activity. After elicitation, the requirements are not yet usable since it is raw data without context or meaning. It is only after analysis that the raw data becomes information that can be further transformed into knowledge.

There are different ways of documenting business, user and functional requirements. Wiegers and Beatty (2013, Chapter 3) write that it is essential to document the requirements in a consistent, accessible and reviewable way that is understandable to the intended audience. They continue that business requirements can be recorded in a vision and scope document, user requirements are written as use cases and user stories and functional requirements can be stored in a software specification document. Winters, Johnson and Blank (2013, Chapter 2) also advocate for the vision and scope document and regard it as a frame for the solution. Caudle (2009, Chapter 5) recommends using a business requirements document. He adds that the requirements can also be documented at various levels: a high-level requirements document identifies the functionalities expected of the solution without describing the data requirements, process definitions, and business rules. However, if the document is used for development, more details are needed. Breyter (2022, Chapter 5), on the other hand, would use a business requirements document only on high-level business requirements and store the functional requirements needed to fulfil the business needs in a functional requirements document. Hossenlopp and Hass (2007, Chapter 6) add that during this stage areas are often detected that are not defined in sufficient detail. All these sources are taken into consideration during the creation of the requirement document template.

In addition to documentation, the requirements should also be validated. Wiegers and Beatty (2013, Chapter 3) state that “validation ensures that the requirements are correct, demonstrate the desired quality characteristics, and will satisfy customer needs”. For validation, they suggest gathering a small team of reviewers consisting of, for example, customers and developers, and reviewing the written requirements. Hossenlopp and Hass (2007, Chapter 6) agree with the idea of review sessions since it ensures the quality of the requirements. They also comment that after validation the project schedule, cost and scope estimates can be validated along with the business case. However, documentation does not end once the requirement document has been created and validated. Caudle (2009, Chapter 5) writes that requirements can change so freezing the requirements after creation is not always a practical approach, but changing requirements during development is always a risk that should be mitigated. Caudle recommends prioritizing the changes and assessing the risks and benefits of including them. Adding version-control to the document is advisable to keep track of the changes. Each change should also be approved and validated.

2.3 Requirement document template

The different requirement types have been discussed in previous chapters, however, what is still missing is the exact specification on what information should be gathered to create a comprehensive requirement document. In this chapter, various example requirement documents are brought together and examined to form an example requirement document template. The examples have been taken from several sources and while there are similarities, each of them has their own take on what should be included and how the information should be displayed. The sources mentioned in this chapter are Caudle's Example Appendix: Business Requirements Document (2009), Hossenlopp's and Hass' Appendix A (2007), and Wiegers' and Beatty's Appendix D. Sample requirements documents (2013) unless otherwise stated.

Version control

Caudle's document template has a page for version control that includes sections for document history and document reviewers/approvers. Document history lists the document version, date, author and the changes made, and document reviewer/approvers lists the people who have reviewed or approved the document and the date. Version control is a smart idea, but how easily it would be maintained and its usability in an agile project are open to question.

Business requirements

Hossenlopp and Hass have a high-level summary of the business solution in their document, which could be a good way to start the business requirements chapter with. They recommend using a diagram to explain the components and the actors of the solution in a visual format. In the same summary, the business need or purpose should also be described. Caudle writes that it should be stated why the project is being initiated: what is it trying to fix or improve. The purpose could be improved process, risk reduction or increased revenue. Related to the need is the business goal that describes what the project should achieve. Wiegers and Beatty add that the goal can split into smaller measurable business objectives that give concrete steps to what needs to happen for the project to reach the goal. As an example, in Wiegers' and Beatty's example document the goal is to have more efficient process for ordering cafeteria food and the objectives are to "reduce the cost of cafeteria food wastage by 40% within 6 months following initial release" and "increase average effective work time by 15 minutes per cafeteria-using employee per day within 6 months following initial release". However, goal and objectives mean nothing if there is no way to measure whether they have been reached. The success metrics can be derived straight from the objectives, so one success metric for the previous example could be "the cost of cafeteria food wastage is reduced by 40% within 6 months following initial release".

Other requirements that could be included in the document are the vison, scope, risks, assumptions, dependencies and constraints. Wiegers‘ and Beatty's example document has a vision statement since it is important to understand the future goal of the project or the bigger vision that the project might be just one part of. The scope specifies what is included in the project. Wiegers and Beatty display the scope as a list of functionalities that will be implemented within the project. Sometimes functionalities out of the scope are also worth mentioning. According to Caudle the risks should explain what happens if the solution is not implemented. Hossenlopp and Hass expand this definition and write that risks are any potential future events that can positively or negatively impact the project. Caudle states that assumptions are events that are likely to occur but are outside the control of the project. Dependencies can be, for example, other projects whose completion the project is relying on. Restrictions affect the delivery of the solution, and they can be related to schedule or money.

Stakeholders

Hossenlopp's and Hass' (2007, Appendix C) requirements document has a section for stakeholders that includes a list of stakeholders, a description of why their input is required, their degree of influence and the required project involvement. The RACI model discussed in the stakeholder analysis chapter of my thesis will be useful here. Wiegers' and Beatty's list of stakeholders is more high-level, and instead of individual people, it lists stakeholder groups such as corporate management, cafeteria staff and patrons. The value they see in the solution, their attitudes towards the solution, major interests and constraints are included in the list.

User classes

Wiegers' and Beatty's list of user classes is simple and has only the name of the user class, a brief description and a comment whether the class is favoured, disfavoured or ignored. Hossenlopp and Hass have a similar list that also describes the user classes behaviour and goals. Both ways of listing user classes are their uses; it is important to have a way of prioritizing the users, but also to understand their goals and need. User classes could be included in the stakeholders, but user class descriptions should be more detailed and should include information gathered during elicitation, so they have been kept separate in this thesis.

Use cases and user requirements

Wiegers and Beatty have a template for describing the use cases of the solution. It includes information about the actors in the use case, what triggered the use case, what conditions need to be met before the use case can happen, exceptions to the use case, and then a detailed flow of the use case that lists all the actors involved in the use case and what they do. If the use case is “order a meal”, the flow might look like this.

- Patron asks to view menu for a specific date.
- Patron selects one or more food items from menu.
- Patron indicates that meal order is complete.
- …

This way of describing use cases is thorough, but it would be an effective way to show the stakeholders that the requirements have been understood. Caudle also describes use cases in his example requirements document. He has a sizable portion of the document dedicated to business events, which is explored in the user requirements chapter. He uses diagrams to describe the use case flows and the actors in it. Creating detailed diagrams seems time confusing, but there’s value in offering the stakeholders a visual representation of the use cases.

Functional requirements

For displaying functional requirements, Wiegers and Beatty use a diagram that illustrates the external entities and system interfaces. It has actors in it, but it is system centric and explains on a high level what the system is expected to do. In addition to the diagram, Wiegers and Beatty also have a list of the use cases and under each is described the system flow, for example, if the use case is “placing a meal order”, the system shall first confirm the patron is registered for payroll deduction. If yes, the system shall prompt the patron for the meal date, and so forth. The flow looks similar to Wiegers’ and Beatty’s use case flow, but this time the focus is on the system. A mix of visualizations and lists can be a practical way of showing functional requirements.

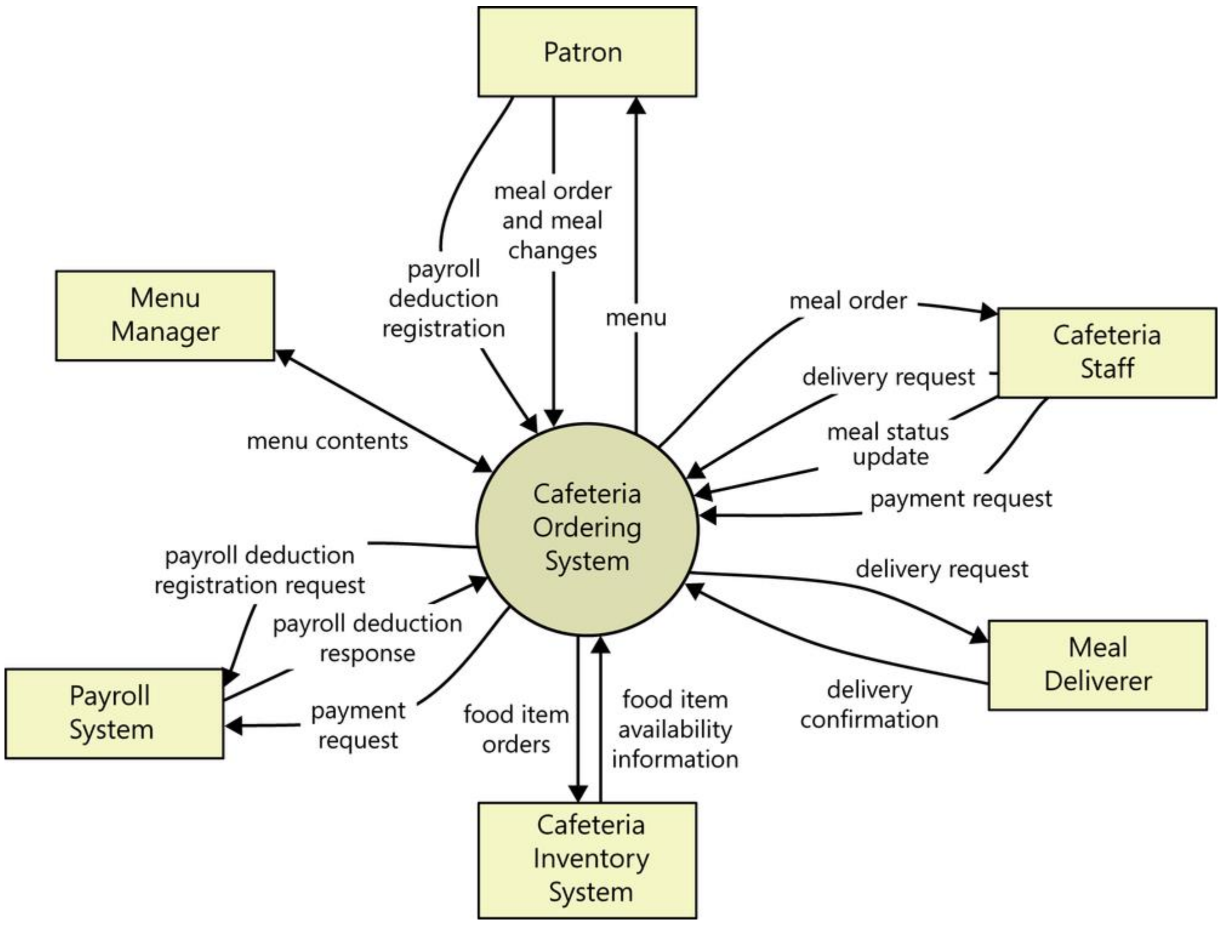
Patron
Menu Manager
Cafeteria Staff
Cafeteria Ordering System
Payroll System
Meal Deliverer
Cafeteria Inventory System
meal order and meal changes
payroll deduction registration
menu
meal order
delivery request
meal status update
payment request
menu contents
payroll deduction registration request
payroll deduction response
payment request
food item orders
food item availability information
delivery request
delivery confirmation

Figure 6. Context diagram for release 1.0 of the Cafeteria Ordering System (Wiegers & Beatty 2013, Appendix D. Sample requirements documents)

Implementation requirements

Some of the example requirement documents include requirements that would be used during implementation such as performance, availability and safety requirement requirements, business rules and UI wireframes. The practicality of gathering such information already during a pre-study is doubtful. However, if hard limits regarding, for example, availability or performance come up during the pre-study, it would be beneficial to include the information since they can shape the rest of the pre-study.

3 Empirical study

As stated before, I am involved in a pre-study project at my workplace where I have used the guidebook created according to the theory gathered in the theory section. In the empirical study, I will go through the guidebook step by step and analyse how well the guidelines work for my purposes, what could be changed and did the guidelines serve their goal of offering the steering group enough information to start the implementation phase. The empirical study is split into two parts: first part is about the usefulness of the requirement gathering guidelines and if I encountered any good practices while doing the gathering and the second part is gathering the requirements into a requirement document and how well the template fit the purpose. A separate conclusion chapter will follow the empirical study where I will analyse whether the guidelines helped us achieve the purpose of the pre-study. The project I am using in the empirical study is about studying the possibilities and use cases of a new technology and how suitable the cases would be for the company. The final version of the guidebook can be found in the appendices of the thesis.

3.1 Requirement gathering

3.1.1 Elicitation

The first stage of requirement gathering was elicitation. In the guidebook, I recommend first identifying the stakeholders and prioritizing their interests using the RACI model. It was a natural start for our pre-study since we needed to know who to invite to the review sessions and who to interview. We did not do any official planning for the elicitation activities, but our process did follow a format. First, we gathered existing documentation and familiarized ourselves with that. Then we scheduled the elicitation sessions and had the more general interviews first followed by technical interviews. I did not have thorough instructions for elicitation in the guidebook and I do not think it is needed, since I want to keep the process flexible. I did not include deliverables either in the guidebook. I see their usefulness, but I was sure people were not going to specify the deliverables, if we also thought their benefit was not great enough to warrant the time used specifying them. Because we wanted to keep the pre-study agile, we did not even try to gather all the requirements needed to start the implementation straight away. Instead, we gathered just enough that the project could move forwards, but more requirements would need to be gathered for the implementation phase.

Something that did not come up in my theory, but what I found highly recommendable, is involving UX from the start if the company has a UX team. UX already has a good knowledge of elicitation practices so there is no reason not to use that knowledge. Caudle (2009, Introduction) recommended the elicitation to have structure and defined steps so make the process predictable and repeatable. UX already had their preferred way of executing elicitation and it gave the process much needed structure. UX helped us draft the interview questions and made sure they were as non-leading as possible. The interview script was the same for every interviewee starting from introduction to guarantee that the interviewees were given the same information. UX also carried out the interviews which further helped with keeping the process predictable and repeatable. They had recommendations for how many people minimum should be interviewed for the study, and they would tell when the answers started to repeat themselves. I can imagine it can be easy to interview too few people or interview too many just in case. Projects have limited resources and carrying out interviews takes time, so it is important to know when there is enough information for an accurate analysis.

In the theory, I listed several ways to elicit requirements such as interviews, surveying and user interface analysis. The right way for the right purpose should be used. In our case, UX recommended interviews. I did not include much information about the ways to decide which method works the best in which situation, and that is why UX's knowledge was essential here, as well. Since this was a pre-study project, we wanted to understand the current processes and what would be the benefits of the new technology. We did not have a ready solution in mind when we started the study. Shadowing could have also been a good option in understanding people's work and the problem areas better. Even though we did get good improvement ideas from the interviews, our understanding of the issues stayed at high-level. Sadly, because of time and budget constraints, we could not consider shadowing. Since the topic was complex, surveying would have given even more vague answers. In the end, interviewing was the best option. Nevertheless, I think it is important to keep the other options in mind and consider mixing the different methods.

3.1.2 Analysis

In the analysis stage, me and my colleague sat down and went through all the material we had gathered from interviews and benchmarking. UX would transcribe the interviews so that it would be easy for us to go through the material. I would not have trusted only notes. We categorized and prioritized the information and made conclusions from it. We did not have much structure in the way we did analysis, but I think most important is to be methodical and consistent. Analysis should not be left to the end of the study, but it should be done throughout the requirement gathering. We started the analysis part already after a couple of interviews and when we carried out more interviews and combined the new information with the existing, our understanding grew, and the analysis got better.

Something I would consider adding to my initial guidelines is that the person doing the pre-study should have somebody they can regular brainstorm with, preferably somebody who is involved in the project. I understood the value of this only later. I am currently also doing a study, but my team is smaller and there is nobody to immediately share my ideas with after the interviews. Not only does it generate more ideas when there are two people, I will also get instant feedback and validation from the other person. The core ideas stayed the same throughout the pre-study, but otherwise our initial analysis went through quite the transformation. If it had been only me analysing the material, I wonder if the result of the analysis would have been as refined.

3.1.3 Documentation

We gathered a huge amount of information and when documenting the requirements, we had to come back to the original research questions and consider what information the steering group needed to decide on the start of the implementation. Hossenlopp and Hass (2007, Chapter 1) put it succinctly that “requirements elicitation activities are designed to give the project team an understanding of the business environment and to gather the customer and user needs that the project outcome is expected to satisfy”. For us, the main question was about feasibility, but the requirements gathered will also serve as a starting point for the implementation. I would not put a minimum or maximum word limit to a requirement document, but it is wise to be realistic about the readers’ attention span. If the pre-study team puts everything they have gathered to a document that spans two hundred pages, it is likely not going to be read. In addition to being concise and remembering who the readers are, categorizing the material well is essential. This is the reason our department wanted to create a pre-study guidebook with an example template: to help people put pre-study findings in an easily readable format.

Because we technically had two audiences, the steering group and the project group, we needed to create two documents. We had a Word document that had more information and a slide presentation that was more geared towards the steering group and had only the most essential information. We presented our findings to the steering group in a meeting and a slideshow was more suitable for that purpose. We wanted the presentation to be easy to read and that the steering group could quickly grasp the information, therefore, the presentation was not too text heavy and there were more images, tables and graphs. The steering group was more interested in the business case and the possible risks, limitations and showstoppers. For example, in the slide about security and legal issues, we focused more on the implications on the feasibility and left the longer technical descriptions to the Word document. The audience for the Word document was the future project group so there we had information that was not necessary in decision-making but would be helpful when the implementation started.

I did not include much about documentation in the pre-study guidebook apart from what information should be included. Whether to use Word documents or slideshows, how detailed the document should be or how many diagrams and images to include, depends on the project. I wanted the guidebook to have the minimum requirements that a pre-study document should have to make it usable. By giving too strict guidelines for documentation could in the worst-case scenario mean that the guidebook is not used at all. Especially in an agile company, it is preferred to gather just enough requirements to get the project started.

3.1.4 Validation

We did validation together with my colleague throughout the requirement gathering. Every week we would sit down to go through the information we gathered so far and check that it corresponded with the initial research question. It was useful to do this in the middle of the elicitation, since had we found our interviews were not giving us the results we wanted, we would have still had time to change our interview questions. It was helpful that we had previous knowledge of the topic so we would have noticed obvious inaccuracies. Before the final presentation, we also had review sessions with a couple of topic experts we had interviewed before. No matter how well the research team thinks they have gathered information, there are always misunderstandings, or the understanding is too shallow. Having that review meeting with people who really understand the topic can provide the team with insight and confirm they are on the right track. For us, it gave confidence for the final presentation because we knew our findings had already been validated. I did not have anything about validation in my initial guidebook, but I added it later. Even then I did not have long instructions for it. It was mentioned to remind my department that validation should be done, but the style it is conducted in was left open.

During the final presentation we got validation also from the steering group. Majority of the meeting was about us presenting our findings, but we left time for discussion. I think that is important because even if the information the pre-study team has gathered is 100 % correct, the requirements and the recommendations might not align with the steering group's vision. Steering group has the people we are trying to convince since they have the power and influence to start projects. During the final review, we also gauged the people's reactions to our findings: what requirements they saw the most benefit in, what they thought was easy to implement, what could require resources that are scarce, etc. In elicitation, as well, some stakeholders' opinions hold more weight than others. Fortunately, in our case the steering group agreed with our findings and the final meeting was more about discussing the next steps than correcting our findings.

In the theory, I talked about acceptance tests and how Wiegers and Beatty (2013, Chapter 1) recommend writing them for requirements. I agree that acceptance tests should be written, but for our case they were not functional. I already discussed before how in agile companies it is not advisable to plan the project from start to finish before it has even started. The same applies here; our project was more of a feasibility study, so writing detailed acceptance tests would have been misuse of time. Even if the project had funding and would have been greenlit to start, it is still recommended to plan only a couple of sprints ahead. There should be a happy medium of providing enough information for the development team that they can start immediately and not spending too much time on requirement gathering. Nevertheless, I left acceptance tests out from the guidebook.

In my opinion, where we fell short in validation was pairing requirements with the business case. What I mean by this is that our requirements were not measurable enough. When we discussed requirements in the pre-study, we did mention risks and how the implementation could be done, but there was scarcely anything about the concrete benefits. The business case does describe the benefits the whole solution should bring, but that should be brought to the use case and requirement level, as well, to verify that does every step I take lead me to the wanted result. Benefits can be hard to put into figures since it is not always so straightforward. There might be requirements that enable another requirement that bring savings, for example. We still should have tried to describe the benefit each requirement brings and how it supports the business case even if we could not produce exact figures. In the original guidebook, I only mentioned the overall benefit the solution would bring, but I added a remark that each use case should be examined separately, as well.

3.2 Requirement document

3.2.1 Version control

I do understand the purpose of version control and it is recommended by Caudle (2009, Chapter 5), but I question whether it would be left unused in my department. I have it in the guidebook template as a recommendation, but we did not use it for the pre-study during requirement gathering. In an agile project, requirements are ever changing so I would not see it as wise to use time on version controlling a pre-study. Version control does have its uses, but I wanted to create guidelines that would be used. People would forget to mark their changes and it would mean that somebody would need to monitor that the version control is filled in. This becomes especially difficult since user and functional requirements can be clarified and specified by anybody in the project team.

In a scrum project, user stories live in the backlog and companies use tools, such as Jira, to oversee ticket management and version control. In my department, I would let Jira handle version control for the rest of the requirements, but with business requirements version control might be a good idea. Business requirements should be relative unchanged and there are only a couple of people who edit them. After the business requirements have been agreed on with the stakeholders, it would be detrimental to the project that changes are made to them without validation or the rest of the team knowing who did the changes.

3.2.2 Business requirements

In the sources I read for the theory, there were several different opinions on what should be included in the business requirements, and, surprisingly, choosing the most useful business requirements to gather and document was not self-explanatory. In a project study, it is easy to focus on the solution itself and the user and functional requirements, but business requirements are the backbone of the study. They give the study direction and specify the wanted end-result. Business requirements are consulted throughout the study and the implementation to ensure that every step is leading towards the vision described in the business requirements. As Caudle (2009, Introduction) stated business requirements can change, but they should never change to align with the solution. For the initial requirement document template, I chose description of the solution, business goal, business objectives, scope and success metrics. Those business requirements were a good start, but after the pre-study was finished, I felt they were not enough, so I added vision, business risks, assumptions, dependencies and constraints.

In my opinion, business goal, objectives and success metrics were the meat of the business requirements. Business goal captures what business wants the project to achieve. It can be money and money saved or customer satisfaction increased. In our case, it was money saved and money coming from new business opportunities. The goal can be more high-level, because the company has the business objectives that provide the more concrete steps to reach the goal, such as a certain process is done more efficiently with the new technology or new business models are developed with the new technology. Success metrics complement these by providing ways to measure whether the goals and objectives were fulfilled. Success metrics need tangible numbers. In most case it is impossible to get 100 % accurate numbers since we are talking about hypothetical situations, but for example, if a certain process takes X number of hours and with the new solutions it takes half as much, the costs associated with the process can be halved. Success metrics are not only good for measuring success, but also for showing the stakeholders what success looks like. “Increase in customer satisfaction” means nothing, but “increase in customer satisfaction that will lead to 10 % increase in sold services” gives a clear idea what the stakeholder can expect from the project if they invest in it.

Vision was not in the original guidebook template, but afterwards I realized it is essential to have it. Wiegers and Beatty (2013, Chapter 6) wrote that the vision describes the ultimate product and ensures all stakeholders know where the business is hoping to go. It gives a deeper understanding of the wanted solution when the research team knows what the long-term goal of the stakeholders is. The project can be simply one part of a bigger vision. Just like the business objectives must support the business goal, the goal must support the vision. After I understood the vision, it was later easier to have discussions with the stakeholders about their needs. This was out of scope, but we thought vision was so important to have before any further discussions about the project, we made a separate vision document for gathering the vision, description of the solution, business goals and business objectives. This document was made into a template to be filled together with the stakeholders before going to any pre-studies. We realized if the stakeholders did not have answers to what is their vision or business objectives, the pre-study could not be started.

In the theory, I only mentioned constraints, but I should have stressed them more. The reader is mostly likely not interested in every little constraint, but if there are deadlines, limitations or hard limits that will drastically affect the schedule or feasibility of the project, they should be made known. In our case, the constraints were related to standards, cyber security and legal aspects. We both interviewed experts and did our own research on these matters. Since the technology and the way we want to use it are new in my company, it was important to investigate whether there were legal obstacles or security risks that might stop the development. We were not interested in minor hindrances but wanted to find showstoppers. Even if the issue is not a showstopper, it is essential for the stakeholders to know if the implementation could be more expensive or take more time than what they had reserved. Investigating standards was not necessary, but finding information about the standards that should be used with the technology will lay a good foundation for the implementation phase. As I mentioned a couple of paragraphs back, the aim was not to define every requirement, but the pre-study would work as a nice starting point for the implementation, so the project team would not need to start from nothing.

As mentioned, I added other some business requirements in the template. I would not say they are mandatory, but I added them because in some cases, they can be important to have. Scope is good to have if the pre-study also serves as a statement of work or project plan. The project team and stakeholders must agree on what the project scope is so that when the business is doing a pre-study, they we will focus on the right stakeholders and requirements. In our pre-study, we described risks in the requirement level, but risks can be associated with the whole project. According to Caudle (2009, Appendix: Example Business Requirements Document), and Hossenlopp and Hass (2007, Appendix A) risks explain what happens if the solution is not implemented or what could go wrong during the implementation. The use of new technology can bring information security and legal risks, but we mitigated that possibility by researching ahead. In our case, we did not find any major risks, but if we had, those would have been included in business requirements. Many projects face constraints in time and money or dependencies which should be mentioned. Lastly, if the research team made the pre-study based on assumptions, those should be listed. Our pre-study was slightly different from typical since it was more abstract, and it was not immediately followed by an implementation. If the project is already confirmed, every risk, constraint and dependency should be mentioned in the pre-study.

3.2.3 Stakeholders

I cannot stress the importance of stakeholder analysis enough. Instructions for stakeholder analysis were included in the guidebook template, but since we already knew the people who had commissioned the project, we assumed we know who should be involved in the project. We did know who should be involved in our steering group meetings, but we did not define these people any further. If I could go back in time, I would use the RACI model. It is easy to assume who the stakeholders are and what is their role, for example, we had our project team who were responsible for the project, the steering group was accountable and informed, and the people who we interviewed were consulted. This was an oversimplification since among the steering group the stakeholders were in various positions: some simply wanted to hear about the results of the study, but a few had power to decide whether the implementation would be started.

Hossenlopp and Hass (2007, Chapter 4) were correct when they suggested mapping out the stakeholder interests, power and influence. Had we followed their instructions, we would have put more focus on the stakeholders with power and tried to understand their interests better. In addition, we only did research on the people to be interviewed but did not try to find the people who could have the power and the budget to start the project. The steering group could be enough, but it is never useless trying to understand more about the landscape and to network. It has happened to me before that I am presenting a business case and persuading the person I assume to be the budget owner to invest in the project only to find out later the person is not the one making the final decision. It shows how important it is to be thorough when mapping the stakeholders related to the project.

3.2.4 User classes / Personas

Since we had clearly defined user classes, we decided to create personas to represent them. I did not have personas in the initial guidebook but added the recommendation for them after the pre-study. Personas might not work in every project, but their use should be considered. UX was helpful here since they had been creating personas in previous studies and could recommend what should be included. In our persona’s we had the persona’s job title, goals, needs, pain points and benefits of the new technology. The personas were not only a way to condense and visualize data for presentation purpose, but it was also useful for us while doing our analysis. Instead of having to read through the interviews, we could skim the personas and quickly refresh our memory on what were the user groups and their needs.

Wiegers and Beatty (2013, Chapter 6) categorized the users into favoured, disfavoured and ignored class. We did not use such categorization since we only focused on the main user classes. However, I do see the relevance of trying to prioritize the users. Especially if the solution has multiple user classes with diverse needs, the company cannot listen to them all. In another project study, we first categorized users by country and then examined which group brings in the most revenue. It was a straightforward way of determining which was our favoured class. The categorization could also be done by user roles or demographic information, such as age or gender. The prioritization should be done already during stakeholder analysis, so that before the company starts the study, they understand who the stakeholders are they should listen to.

3.2.5 Use cases and user requirements

Already during the analysis phase, we started trying to form use cases out of the gathered user requirements. There were three clear ways how the new technology could be used and those became our three main categories where all the use cases were put under. At this stage, we did not try to fit the requirement into proper use case or user story format, but the focus was nevertheless on the user. We took ideas from Caudle's (2009, Chapter 2) event process model and tried to write down the events and actors that make up the use case. Later, the system was added to the flow. Following Wiegers' and Beatty's (2013, D. Sample requirements documents) example we also made alternative flows for sunny and rainy-day scenarios. We knew the ideas would get clarified even further on, but writing these flows as early as possible in the requirement gathering really helped us understand the big picture.

In the guidebook, I did recommend visualizing or listing the user requirements, but eventually, we only included the three high-level use case categories in our pre-study document. I still did not take that away from the guidebook since user-centric approach is recommendable, but it did not work in our case. Even though the use case flows we wrote during the analysis were useful in understanding the user and functional requirements, they were too detailed for the stakeholders and likely subject to change when implementation started. We also did not follow the recommendations in the theory to put user requirements into a user story format. User stories are more useful when the implementation has started so we did not find them an effective use of time.

3.2.6 Functional requirements

When we started the study, I was sure we would have spent more time on functional requirements and specifying what the system must do to fulfil the user requirements. In the end, we only had high-level system requirements such as "system shall visualize and contextualize data" or "system shall update unit settings remotely", since we did not think they needed to be more detailed. What we did was to put the high-level use cases into a table where we shortly described the functional requirement for the solution, what is needed technically to implement the solution, risks and issues of the implementation and a work estimate. To give an example, here is one functional requirement under one of the use cases.

Requirement	Description	Changes needed	Risks/issues	Work esti-mate
System shall visualize and contextualize existing data	There is useful data in the cloud, but it is not utilized well enough User friendliness would make the data more easily under-stood without extensive previ-ous knowledge	Better understanding of the data existing data and its use cases Analytics Tool to view the data	Difficulty in obtain-ing resources to study the existing data More data should be collected	**Medium** **(**6-12 months**)**

This way of listing functional requirements will not work in every project, but if the requirements are still vague and the topic is both complex and new, it will give a useful summary. It was also useful in prioritizing the requirements; if the work estimate was big and there were many risks and issues, it probably was not the first possible requirement to be implemented.

I agree with Kelly (2019, Chapter 4) that the aim should be to discover user needs, but not to define all the requirements before implementation can be started. The company can keep up better with changing requirements if they do not keep the requirements static. Especially in our case this was true since the pre-study unearthed several use cases that each would need their own project. If we had written exhaustive list of functional requirements for all of them, it would have taken a good amount of time, and the requirements could have become outdated if the project did not start immediately. We needed to provide just enough information to the steering group for them to decide if any of the project ideas should be started at all.

3.2.7 Project plan and opportunities

At the beginning of the project, we were asked to provide short project plans for the use cases, since a list of requirements alone would not give the stakeholders enough information to decide about the feasibility. The project plans included a brief description of the solution, needed resources, scope, deliverables, timeline and budget, and especially the functional requirements were useful in defining what should be included in each project. In the theory, none of my references used these kind of project plans in requirement gathering, and I did not add this in the guidebook either. Having short project plans is more agile, since it gives the stakeholders a rough estimate of what to expect with a certain budget and timeline. The pre-study was a unique example since several projects could be born out of it, so we could not spend time in defining all of them from start to finish. If the project was focusing on one clearly defined idea, a separate short project plan would not be needed.

In addition to requirements, my colleague and I also listed potential new business opportunities that could be realized if the use cases were implemented. I did not cover this in my theory, but I think it was a beneficial addition to the business case since they did help prove the importance of the use cases and why they needed to be implemented. Pre-studies need to be based on facts, but it does not harm to excite the readers' imagination about all the possibilities implementing the use cases could open. Solutions are seldom truly finished after they have been implemented, since there is always something to improve or new features to develop to keep up with the changing needs. Ideally, our description of the opportunities would have had more numbers about what impact the opportunities could have on the business. However, they were not our focus, so we kept them simple.

4 Conclusion

The culmination of pre-study was a final review meeting where we presented our findings and proposed the next steps. Since the topic of the study was vast, the idea was never to draft detailed project plans, but to find the direction the company should go with the technology we were researching and convince the steering group it was a direction worth taking. After the presentation, the steering group was in agreement that the technology was useful and should be utilized in our company. However, it was harder trying to plan for the concrete next steps and get budget for the implementation phase. Since the timeline for the use cases spans many years and the use cases would be implemented in several different projects, I understand the steering group's difficulty in deciding what is the budget they should give to the project. It is easy to say yes to an idea, but when budget is mentioned, people want to know what exactly they are getting for their money.

Since we knew we were heading to the right direction, after the final review meeting, we spent time drafting a more concrete project plan. We took the use case that was the first on the proposed roadmap and split it into smaller parts and focused on one of the parts. Splitting use cases into smaller parts could have been done already during the pre-study, but as I said, if the use cases will be implemented throughout a couple of years, it is more agile to go deeper into a use case only when its implementation becomes relevant. After creating the project plan with realistic work and budget estimates, and deliverables, we presented it to the steering group and were granted the budget to continue to the implementation phase. What we could have done differently from the beginning would have been to have more discussions with the stakeholders during the pre-study and established what would be the first small step to implement. In that case we could have created and included the project plan already to the final presentation. However, since the topic is vague, I see it beneficial to have the final steering group meeting before making any decisions on what should be implemented.

Overall, the study was successful since it gave the steering group enough information to concur that the new technology would be feasible to implemented. We did not get the budget confirmation until we split the use cases and created a project plan, but that was to be expected. In our study, we could have focused more on the user and functional requirements, but if the goal is to study whether the project should be started at all, business requirements are most important. If the pre-study scope had been smaller, I believe the same pre-study guidebook could be used also to gather the necessary requirements for the implementation phase. I am looking forward to seeing what results the guidebook will bring in my department. I am also certain that the guidebook will go through changes in the future when more people start using it and the guidelines are applied to different projects. Nevertheless, the guidebook in its current form already offers a comprehensive base for requirement gathering.

5 Analysis

Even though the goal of my thesis was met and following the guidebook we were able to gather enough requirements to start the project, I did not feel the pre-study guidebook itself was the biggest achievement. I was glad the guidebook fit the intended purpose, but for me the most important gain was the research I did for the theory section. Since I tend to rely on my own experiences and intuition when working, this was a refreshing way of working because I had strong theoretical foundation before starting the pre-study. Whenever questions arose about why or how some part of the pre-study should be done, I could reference actual published sources and find the answer. Before my answer would have been “I think” or “I feel”. I am not saying you cannot rely on our own experiences, but it gave me confidence to have theory to back me up. I have already used what I learnt during the thesis research in other projects, for example, I have become the biggest advocate of the stakeholder analysis and the RACI model.

If I could go back in time, I would have started writing the empirical study while working on the pre-study. I wrote most of the theory section before the pre-study started, which was helpful, since it forced me to gather my thoughts and think what information would be useful. The empirical study would have benefited from the same, since if I had started writing it when the pre-study was still ongoing, it would have forced me to analyze our requirement gathering methods and I would have had time to make changes if needed. I mentioned in the conclusion that the guidebook will continue to be polished when it is used in more projects. A pre-study project started at my workplace almost immediately after the first pre-study was finished. Another of my regrets is that I did not have time to use the guidebook in that pre-study, and add the findings to my thesis. It would have added more depth to my empirical study.

During the thesis writing process I did learn to make compromises. After researching the theory and living in the ideal world of requirement gathering, it came as a shock to be returned to the real world when I started the pre-study. I did have a draft of the guidebook and many ideas for the best ways to conduct a pre-study, but I quickly realized real life does not work this way. Already during business requirement gathering I noticed my perfect guidelines were not usable without adjustments. If the research team does not see a benefit in gathering a specific requirement and it does not fit the scope of the pre-study, it is not worth gathering. Many of the guidelines in my initial requirement document template went through the same thought process and were discarded or made less strict. When finishing the guidebook, I was constantly fighting between making the guidebook as theory based and thorough as possible and at the same time making it easy to understand and follow. If the guidebook was cumbersome to even read let alone follow the directions, it would not have been of use. At the end, I decided that long as the reader understands the main concepts of requirement gathering, why it is being done and what requirements would be beneficial to gather, it is already an improvement from having no guidebook at all.

References

Breyter. M. 2022. Agile Product and Project Management: A Step-by-Step Guide to Building the Right Products Right. Apress. New York City. E-book. Read: 27.9.2023.

Caudle, G. 2009. Streamlining Business Requirements. Berrett-Koehler Publishers. Oakland. E-book. Read: 11.9.2023.

Hossenlopp, R. & Hass K. 2007. Unearthing Business Requirements. Berrett-Koehler Publishers. Oakland. E-book. Read: 12.9.2023.

Kelly. A. 2019. The Art of Agile Product Ownership: A Guide for Product Managers, Business Analysts, and Entrepreneurs. Apress. New York City. E-book. Read: 21.9.2023.

Wiegers K. & Beatty, J. 2013. Software Requirements, 3rd Edition. Microsoft Press. Redmond. E-book. Read: 12.9.2023.

Winters. N., Johnson N. & Blank, N. 2013. Microsoft Exchange Server 2013: Design, Deploy and Deliver an Enterprise Messaging Solution. Sybex. Indianapolis. E-book. Read: 25.9.2023.

Appendices

Appendix 1. Pre-study guidebook

Pre-study guidelines

Introduction

The document gives general guidelines for gathering and documenting requirements for a project pre-study. Based on the pre-study, the steering group and customers should be able to make an educated decision on the feasibility of the project. The pre-study also works as a basis for the implementation phase.

These guidelines are suited for any <department> project. The project might have requirements already gathered, but the document will offer a template for how to document that information. If there's no information gathered yet, the document will also offer guidelines for requirement elicitation.

It is good to note that while business requirements should stay relatively unchanged during the development, user and functional requirements will change. The gathering of the user and functional requirements will continue during development, so they don't need to be set in stone during the pre-study. There should be just enough requirements gathered to start the development.

4 stages of requirement gathering

Elicitation

Elicitation includes activities related to discovering requirements, such as identifying stakeholders and understanding use cases and business objectives. Results of elicitation should be measurable, documented in an organized manner, follow a reasonable standard of quality and be made accessible to all participants.

The first step in the elicitation is identifying the stakeholders using, for example, the RACI model and prioritizing the stakeholders' interests.

- **R**esponsible stakeholders participate in the project work activities.
- **A**ccountable stakeholders are accountable to the customer for the result of the project.
- **C**onsulted stakeholders are asked for comments on the objectives, constraints, etc.

- Informed stakeholders should be kept informed about the project status.

You will find the people you need for requirement gathering among the stakeholders. During this stage, decide also who belongs to the steering group.

For the actual gathering of the requirements, we recommend consulting UX. The most popular way to elicit requirements is interviewing which works well if it is clear what needs to be asked. Surveying is less time consuming, but coming up with the right questions can be challenging. Shadowing and prototyping can also be used in understanding the customer’s needs. Consider user interface analysis if you have an existing solution you want to improve.

Analysis

Analysis involves reaching a deeper understanding of the requirements. Activities in this stage include putting requirement information into categories, prioritizing requirements, studying if the requirement is legally, technically, operationally and economically feasible, assessing requirement risks and constraints, and modifying requirements to mitigate the possible risks.

Documentation

During specification stage, the collected requirements are stored into a document in a visual and written format and in a way that they are easily understood and reviewable by the stakeholders. This document offers a template for storing requirements.

Validation

Validation includes verifying that the requirements are correct and that they meet the business needs and objectives. In practice, this would mean asking stakeholders to review the requirements or organizing steering group meetings and presenting your findings there.

Pre-study template

Version control

If there are several people editing the requirements, version control is recommended.

Version	Date	Author	Comments

Business requirements

Business requirement is something that the business requires a solution to do to satisfy a driver, such as the business' vision, principle or mission. Business requirements might change over time, but they should never change to align with the solution. Business requirements need to be gathered first before the pre-study can move to user and functional requirements since they provide a reference and lead the requirement gathering to the wanted direction.

High-level summary

The summary describes the project's background, the possible solution, where the idea originated from and why the project should be started.

Business goal

The goal describes that the project should achieve, such as a more efficient process, risk reduction or increased revenue.

Business objectives

Business objectives describe the concrete steps that need to be taken to reach the goal, such as "increase average effective work time by 15 minutes".

Success metrics

Success metrics describe the way to measure whether the goal and objectives have been reached. The success metrics can be derived straight from the objectives, for example, if the average effective work time has increased by 15 minutes in a set amount of time, the objective has been met.

Vision

The vision describes the end goal and ensures all stakeholders know where the business is hoping to go.

Scope

The scope describes how much of the vision will be solved in the project and what features are left out.

Risks

Risks describe any potential future events that can positively or negatively impact the project, or they explain what happens if the solution is not implemented.

Assumptions and dependencies

Assumptions and dependencies describe the events that might happen, but are out of the project team’s control, and any related projects or systems that might hinder or help the project.

Constraints

Constraints affect the delivery of the project, and they can be deadlines, or limited budget or resources.

Stakeholder analysis

The stakeholder analysis should produce a list of stakeholders and their roles (responsible, accountable, consulted, informed). Analysis can also include a prioritized list of the stakeholders’ interests and possible conflicts in interests.

Personas

When the users of the solution have been identified, they should be categorized into classes based on needs, goals and demographic information. Persona is a typical, hypothetical representative of one class, and it helps in bringing the user class to life, but also when prioritizing user requirements.

User requirements

User requirements describe what the user will be able to do with the solution that will provide value to the business. The approach is user-centric so instead of asking what users want the system to do, the users should be asked what they need to accomplish. User requirements are commonly represented by use cases and user stories.

- **Use case** is a sequence of interactions between a system and an external actor, for example, “create an invoice”.

- **User story** is a concise statement that describes a feature from the perspective of the person who needs the functionality, for example, “As an invoice clerk, I want to create an invoice so that I can bill a customer”.

This section can include a diagram of the process the solution will be used in. Consider also defining the benefit each use case/user story brings and how it supports the project goal.

Functional requirements

If user requirements tell what the user wants to do with the solution, functional requirements describe what the system needs to do to fulfil the user requirements. Functional requirements describe the way the system should behave and are often written in a “shall format”, for example, “the system shall forward the feedback inputted by the user to the product support”. Functional requirements can be derived from use cases, but they are separate and it’s risky to pass use cases to the development team for implementation.

Functional requirements can also include a diagram to visualize the system flow.