A flexible method for requirements engineering

Requirements development and management in a highly turbulent environments

Master Thesis Information Science

Radboud University Nijmegen

Author	P.B.F. Arts (0412570)
1st Reviewer	Prof. Dr. Ir. Th.P. van der Weide
2nd Reviewer	Dr. S.J.B.A. Hoppenbrouwer
Thesis number	120 IK

Research facilitated by ING

Supervisor	Christiaan Koenders
Mentor	Jan Willem van der Grijp

Preface

Dear Reader,

Hereby I would like to proudly present the results of my final research to my Master's degree Information Science at the Radboud University Nijmegen. Six months of effort have been put to it and I hope it was worth the while. I'm quite happy with the results, but of course I couldn't have managed it all by myself.

There are three persons I would like to give my special thanks. The first is Theo van der Weide, my mentor and reviewer at the University. He was of great inspiration during my research and he came up with new ideas whenever I was struggling to make progress. The second and third person are Christiaan Koenders and Jan Willem van der Grijp, both employed by ING, where I conducted my research. Christiaan, in a supervising role, learned me a lot about the differences between the academia world and the business world, and helped me with understanding the political game in such large organization. Jan Willem, in a mentoring role, helped me to get started and was an excellent sparring partner whenever there were new developments. I had an excellent time working together with them.

I hope you'll enjoy reading.

Pepijn Arts

Table of Contents

1. Introduction

The role of requirements engineering (RE) grows more and more important within Software Engineering projects. Requirements Engineering can be defined as a systematic analysis of the requirements of a particular system [3]. A requirement can be defined as a description of what a product or service should do [3]. Requirements Engineering processes makes sure that all business, customer and system requirements are defined, described and communicated. This set of requirements, called the requirements specification, specifies what the future system should do and forms the basic specification to the whole software development process.

Requirements Engineering can be divided into two sub domains, Requirements Development (RD) and Requirements Management (RM). Requirements Development is concerned with eliciting, formulating, analyzing and validating requirements [15]. Requirements Management is concerned with managing the lifecycle of the requirements and the requirements design [15]. In a turbulent business environment the requirements engineer has to cope with changing requirements as a result of environmental changes or new insights once (parts of) the requested functionality is demonstrated.

Agile system development focuses on handling these changing requirements by developing a system in an incremental way such that stakeholders can provide frequent feedback and changing requirements can be taken into account [16]. On the other hand project based working methods demand a certain amount of control on time, money and quality of the project [18].

This master thesis describes a method for Requirements Development & Management capable of handling changing requirements while maintaining control in project based working environments. The thesis is presented in the following order. In chapter 2 the problem statement is addressed more specifically and divided into sub problems. Chapter 3 describes the justification for the master thesis in terms of academic and practical relevance. In chapter 4 a theoretical framework is presented which identifies underlying theoretical concepts used throughout this thesis. Chapter 5 describes the method used to conduct the research followed by chapter 6 which presents the requirements of a requirements engineering method. Chapter 7 summarizes and discusses the suggested process model for Requirements Development & Management[1]. Chapter 8 discusses the case study in which the model was put to the test. Chapter 9 presents an instrument to measure the real-time performance of the RE process. Chapter 10 provides some discussion regarding the outcome of this research and indicates future research possibilities. In the last chapter a conclusion is given in terms of the solution to the research question.

[1] The full document can be found in appendix I

2. Problem Statement

As in many projects good requirements development and management plays an important but complex role. Miscommunication, time losses, dissatisfying results and lack of commitment to the project are the result of a lack of emphasis to a decent requirements analysis. But how do you write the business needs into a decent requirements specification when the world around you changes constantly, affecting the business needs?

One effective way to take changing requirements into account is to just go with the flow. Start working, keep changing and see where it may lead. Probably many people working in the business environments would like to see this happening. But of course changing the requirements of an information system takes time, nonetheless money. Time and money need to be controlled in order to prevent exceeding budgets and time frames. Where do you draw the line?

> *The most likely way for the world to be destroyed, most experts agree, is by accident. That's where we come in; we're computer professionals. We cause accidents. - Nathaniel Borenstein*

The key issue here is to find a balance between flexibility and control. But how to find such balance when every project is different? Some project are small and simple, others are highly complex and expensive. The right balance can only be found if project management can adapt to the specific situations the project is in.

In basic terms requirements engineering is all about making business' needs and wishes explicit in such way that a software engineer can build the system accordingly. So the result of requirements engineering processes should be a common understanding of how the proposed system should work. In other words the quality of the requirements specification should high enough to lead to a common perception of the proposed system.

Given the above information we can formulate the following problem statement:

"What should the process model of a method for requirements development and management, capable of handling changing requirements while maintaining project control, look like."

This problem statement can be split up into three sub problems. These sub problems form the main challenges the method has to overcome. The method's specific abilities to overcome these challenges are what separates this method from many other requirements engineering methods.

The first sub problem is the applicability to both small and big projects. The applicability of the method has to be flexible enough to use it for all types of project. This means simplifying the method for small projects by skipping optional steps without decreasing the quality of the end result.

The second sub problem is to maximize the quality and completeness of the requirements specification. The method should not only be about the process, but also lead to quality output.

The third sub problem is to create the ability to handle changing requirements due to a turbulent business environment and new insights; Finding a balance between flexibility and control.

3. Justification

3.1. Relevance

The relevance of this research can be found in both practical and academic areas. The practical relevance of this research can be found in the flexible application of the method. It guarantees a minimal level of quality without resulting in overkill on documentation and bureaucracy, by performing only the obligatory steps. This research provides a method which can cope with changing business environments and evolving insights and requirements when the system design gets clearer over time.

The academic relevance of this research can be found in the balance between flexibility and project control of the requirements design and management process. These two visions within requirements engineering seem to contradict and it is a big challenge to combine the best of two worlds in a practical solution. A specific process model based on this combination is an enrichment of the requirements engineering research field. More information of the two streams within requirements engineering can be found in the next chapter, *Theoretical Framework*.

3.2. Scope and limitations

The scope of this research is limited to the requirements engineering phase within a single project. Requirement management in specific is also limited to managing requirements within a project. Although a very interesting topic, managing requirements across different project is outside the scope of this research. Based on this limitation project management, design and construction, all very important aspects of the software engineering domain, are also outside the scope of this paper.

The method uses ideas and aspects from both agile software development and PRINCE2 project management. These ideas are used in order to provide extra dimensions to the requirements engineering model. Therefore the suggested model is only applicable to requirements engineering issues and it does not provide much value for either project management or software engineering issues.

Please note that this research is conducted within a data warehouse environment. The above sub problems exist within the particular business environment and are therefore specifically addressed in this paper. Also some metarequirements (see chapter 6) were discovered in this context. General metarequirements identified in other research as well as overall requirements engineering guidelines are taken into account as well, making the result of this research more generally applicable.

4. Theoretical Framework

In this chapter the research' underlying theoretical disciplines are discussed. All the relevant information used to conduct the research and construct the requirements development & management method is explained below. Roughly said, three major disciplines are presented. These disciplines are *Requirements Engineering, Software Engineering* and *Project Management Engineering*.

4.1. Requirements Engineering

4.1.1. What is requirements engineering?

> *Those who cannot remember the past are condemned to repeat it.*
> *- George Santayana*

As stated in the introduction requirements engineering is all about defining, describing and communicating requirements. The success of the requirements engineering process depends on a couple of aspects and activities. As Cheng and Atlee describe [1] *"Successful Requirements Engineering involves understanding the needs of users, customers, and other stakeholders; understanding the contexts in which the to-be-developed software will be used; modeling, analyzing, negotiating, and documenting the stakeholders' requirements; validating that the documented requirements match the negotiated requirements; and managing requirements evolution"*. Looking at this definition of the success of requirements engineering processes, we notice that the concept *understanding* is the single most important aspect of requirements engineering. Central question here: *"Does the requirements engineer understand what a stakeholder needs?"*. It's this understanding what makes or breaks the success of requirements engineering processes. Requirements engineering is difficult [1]. In general the type of challenges concerning requirements engineering are slightly different from those concerning software engineering. Requirements engineering challenges reside mainly in the problem space, where software engineering challenges reside in the solution space. This leads us to another requirements engineering definition [15]: *"Requirements engineering is about defining precisely the problem that the software is to solve"*.

4.1.2. Why requirements engineering?

A picture says more than a thousand words…

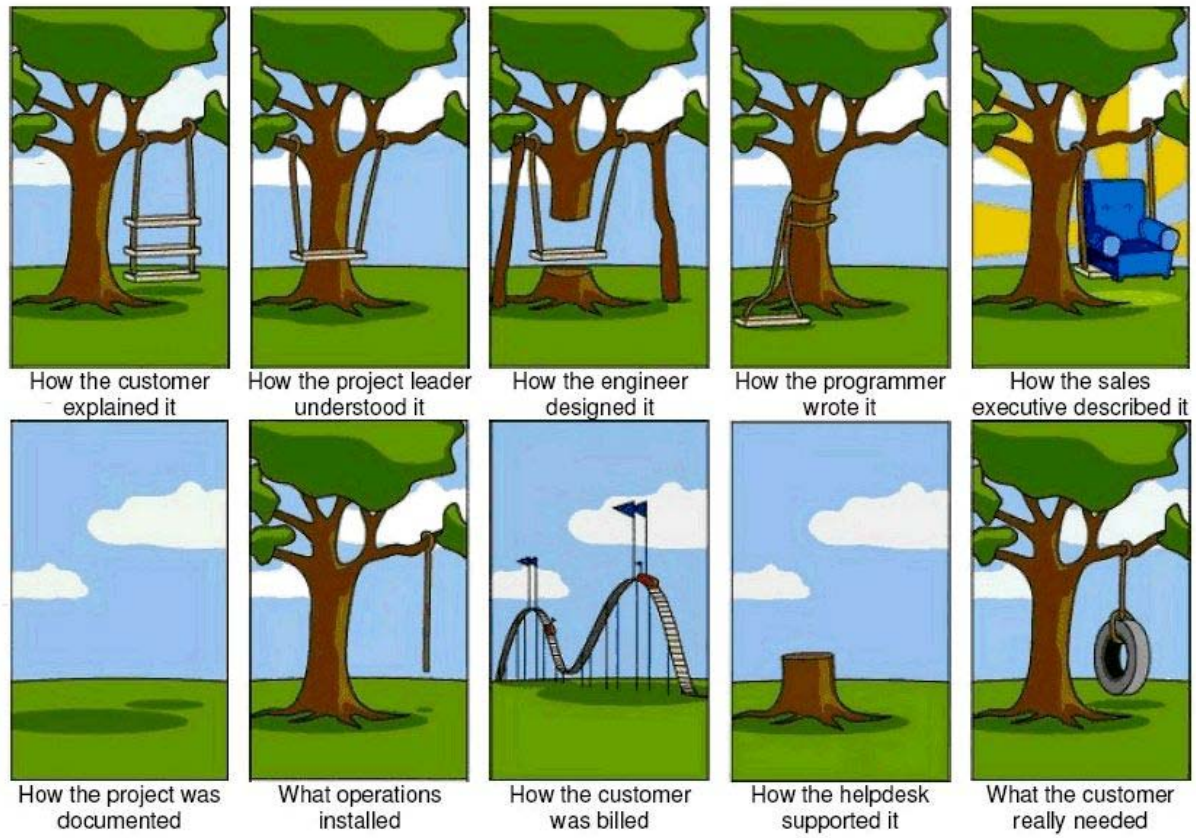


Figure 1: Why Requirements Engineering?

4.1.3. Two sub domains

Within the requirements engineering domain two sub domains can be distinguished, requirements development and requirements management. Requirements development focuses on eliciting, formulating and validating requirements. Requirements management focuses on the requirements lifecycle.

4.1.4. Requirements Development

> *The seeds of major software disasters are usually sown in the first three months of commencing the software project. – Capers Jones*

There are two principles by which requirements development can be performed. The first principle focuses on proper, clear and complete documentation of the requirements specification. It leans heavily on standard documentation and describes several techniques for eliciting and formulating proper requirements. Important aspect of this view on requirements is that this principle sees requirements engineering as a linear process [15]. Once you end the process and deliver the requirements specification, the set is fixed and requirements must not change in order to prevent cost-exceeding projects and project delays.

The second principle is more about communication. Supporters of this principle state that the most basic goal of requirements engineering is to make system designers understand what stakeholders

want. This view on handling requirements is subtracted from agile development methods. With the introduction of agile methods a shift from the first principle to the second is introduced. As Robertson & Robertson state *"Agile methods have influenced the way people develop software, with the result being that greater emphasis is placed on close customer relationships, and less emphasis is placed on documentation"*. More about agile software development in paragraph 4.4.

> *The first step to getting the things you want out of life is this: Decide what you want. - Ben Stein*

Requirements can be divided into three categories, Business-, User- and System Requirements. Business requirements concern the 'why' question. For example why something is relevant for the business and why something fulfils certain business goals or needs. Business requirements make sure that the solution (to be designed) will be in line with the (strategic) goals of an organization and the corresponding business needs. Business requirements can often be linked to business cases which justify the initiation of the project. User requirements, also called customer requirements, concern the 'what' question. For example what a solution must contain in order to fulfil the business requirements. With that said, a hierarchy arises. All the user requirements must be linked to one or more business requirements higher up the chain. This phenomenon is called vertical traceability. Vertical traceability is a very powerful method to ensure that all low level requirements contribute to higher requirements in terms of business needs. All the way to the bottom of the chain we find the system requirements. System requirements concern the 'how' question and describe what is needed in order to fulfil one or more user requirements higher up the chain. The following figure provides a schematic overview of the above principle.

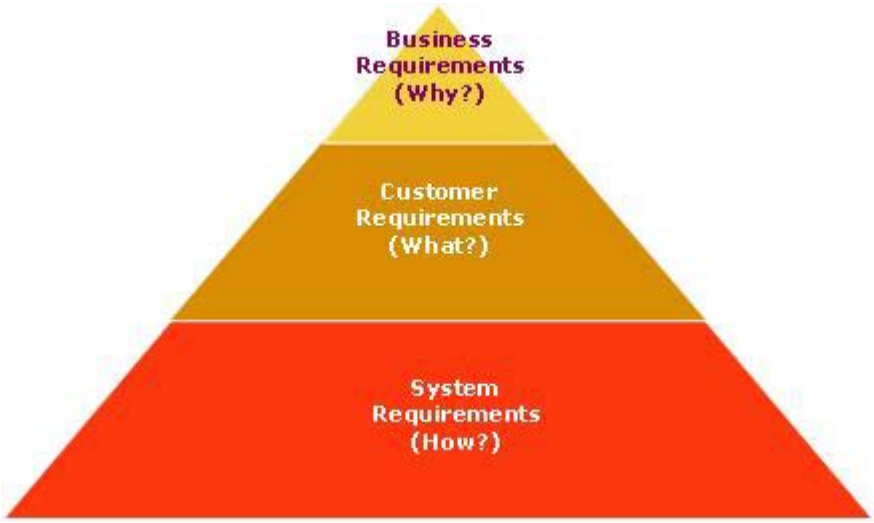


Figure 2: Requirements Triangle

In terms of enterprise architecture and architectural frameworks such as TOGAF the three levels of requirements can be compared with the business- application- and logical / technology layer [13]. The top layer of the enterprise architecture model concerns the business processes and functions. Business requirements describe a business need, which often justifies the necessity and relevance of the particular requirement in terms of contribution to the business processes. The middle layer of

the enterprise architecture model is the application layer [13]. User requirements describe what an application or system has to be capable of in order to fulfil the business requirements. The bottom layer of the enterprise architecture model concerns the logical layer or system layer. System requirements describe how in terms of technology or logical rules (not to be confused with functional design) a user requirement van be achieved.

4.1.5. The requirements engineering process

Requirements engineering consists typically of four phases [15]. The elicitation, analysis, specification and validation phase. Elicitation concerns the discovery and gathering of requirements. Several techniques like workshops, interviews and observations can be used to gather the requirements. Stakeholder identification can also be part of the elicitation phase. In the next phase, the analysis phase, all requirements will be refined and made very explicit in order to formulate them in the next phase. The specification phase concerns formulating all requirements down and forming a consistent requirement specification. In the final phase, the validation phase, the requirements specification is checked for validity by the stakeholders.

The most popular techniques for requirements elicitation are interviewing, organizing workshops, apprenticing, storyboarding and observing. When writing requirements down, they must be formulated according to SMART guidelines. SMART stands for Specific, Measurable, Actionable, Realistic and Time-bound, and forces the author to formulate the requirements in a clear and unambiguous way. Next to guidelines like SMART the use of (standard) templates, which are widely offered, is encouraged. Nowadays all kinds of tools are available for formulating, storing, sharing and even managing requirements.

Another commonly used technique for formulating requirements is Use cases [17]. Use cases express the necessary functionality by describing the communication between actors of the system and the system itself. The philosophy behind Use cases is describing stakeholders' needs in a readable and understandable way. Use cases model the requirements instead of fully writing all requirements down. Supported by business rules and scenario's use cases can be seen as a way of modelling requirements. Because use cases describe the interaction between users and the proposed system (in other words, it describes what actions user can perform on the proposed system) stakeholders without any technical knowledge can understand the requirements and check whether or not the requirements engineer understand their needs. This process is called requirements validation.

4.1.6. Simulating requirements

A very powerful way to validate requirements is simulating the proposed system based on the requirements. A commonly used technique for simulations is prototyping. Prototyping is a quick and dirty implementation of the proposed system, able to show the main functionality. Prototypes can be used to validate the requirements [15]. Stakeholders can see how you interpreted their needs and translated them into functional solutions. Misunderstandings caused by (textual) misinterpretations can be revealed where written requirements specifications would have kept them hidden. Prototypes also stimulate stakeholders to keep reflecting to see if this is what they really want the proposed system to be. Furthermore prototypes make requirements and functionality visible what enhances the stakeholders' involvement.

4.2. Requirements Management

In modern business environments and rapidly changing markets form a major challenge for Information Technology (IT) development. This means changing requirements are facts. Requirement management as a sub domain of requirements engineering, is responsible for managing fixed and changing requirements throughout the whole project and maybe even beyond project boundaries. In other words, requirements management concerns the requirements life-cycle.

In order to follow the requirements life-cycle throughout the software development process an explicit traceability needs to be maintained. In other words the system design (functional or technical) needs to have explicit links to the requirements in order to proof consistency. This type of traceability is called horizontal traceability. In order to match low-level requirements with higher-level requirements vertical traceability needs to be maintained. This way the relevance of the low-level requirements is proved. In other words it is proved that the low-level requirements contribute to the higher-level objectives (See figure 2). The concept behind these two kinds of traceability is called bi-directional traceability.

As stated earlier, there is a shift within requirements engineering from the use of documentation to a shared understanding of requirements by intense communication and customer involvement. This shift combined with agile development methods (paragraph 4.4) leads to highly flexible and complex requirements engineering processes. The biggest challenge is to find a balance between flexibility and adaptability in order to deal with changing environments while maintaining managerial control over the project.

4.2.1. Communication

Requirements management is mostly about communication. Therefore it's important to explicitly describe how requirements will be managed in a particular project. All parties involved in the project need to agree on how the team will deal with requirements. The document that describes how the project will deal with requirements is called the Requirements Management Plan (RMP).

Communication is important, but it is also what makes requirements practices difficult [1]. Requirements analyses often start with unclear and contradicting idea's of what a proposed system is to do. Communication is the mean by which these idea's are formed into a decent requirements specification. Why is communication that important? Communication not only clarifies idea's and enables the requirement analysts to write a requirements specification based on the stakeholders' needs and wishes, but it also stimulates customer involvement. Research has shown that the lack of customer / stakeholder involvement is the number one reason for project failure, while stakeholder and user involvement and clear requirements specifications are both top three project success factors [14]. Frequent communication keeps the project alive in the minds of the stakeholders and stimulates their involvement. The use of simulations and prototypes contributes in particular to stakeholders' involvement as stated above.

4.2.2. Changing requirements

Turbulent business environments involve (constantly) changing requirements. When viewing the requirements engineering process as a linear sequential process, the requirements manager can't change requirements forever. At a specific moment stakeholders have to agree on a requirements specification which will be frozen from that point on. A *baseline* is created and the project board decides whether or not the project will continue with this baseline [16]. If decided to go forth with the project the functional- and technical design documents will be constructed based on the requirements specification and finally the code is written. During these processes requirements can still change, but the impact on the project will be much greater (redoing work and the need for new analyses often result in extra costs and project delays). As a result, linear sequential requirements engineering processes results in inflexible requirements management due to the lack of options to adapt to changing requirements.

When viewing the requirements engineering process as an iterative sequential process, the requirements manager is able to change requirements. Iterative sequential process results into (functional) decomposition which enables an incremental model. While building the requirements concerning the functionality of the first increment, the requirements concerning the functionality of future increments can still change without affecting design or construction. When designing or constructing requirements can still change, but impact on the work of that particular increment must be taken into account.

Doubt is not a pleasant condition, but certainty is absurd. - Voltaire

4.2.3. Relation with project management

The relationship with project management is mostly an advisory relation. The project manager needs to make decisions concerning the lifecycle of a project. Often this decision depends on information gathered by the requirements engineers. When dealing with a project management framework like PRINCE2 [18], the project knows several decision points. The requirement engineers (both developers and managers) provide input concerning the requirements to be used in these decision points. A little bit more about PRINCE2 can be found in paragraph 4.5.

4.3. Software Engineering

Using linear software development models, like the waterfall model, for software development makes it very hard to adapt to the turbulent environments described in the above section. This model perceives the requirements development phase completely separate from the design, construction and testing phase. De set of requirements has to be complete and formulated and all parties involved in the project commit to this requirements design before moving on to the next phase. The waterfall model can be visualized as in fig 3 below.

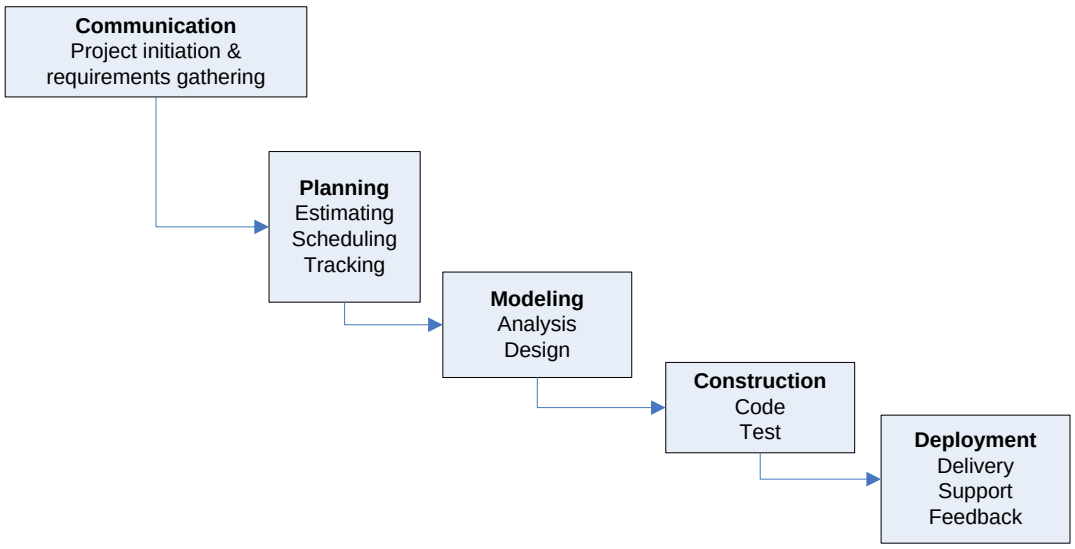


Figure 3: The Waterfall Model
Source [16] p.47

With the new and upcoming software development methodologies like agile software development the view on the function of requirements engineering is changing as well. New software developing methods focus more and more on flexibility and adaptability to cope with changing environments. These methods can be summarized under the general term *Agile Software Development* [16]. This shift in software development leads automatically to a shift within requirements engineering processes. Because of the dynamic characteristics of the software development processes, the same dynamics are expected from requirements engineering processes. This leads to a more flexible form of requirements management [15]. Flexible processes ask for intensive communication between all parties involved in a project [19]. This creates a perfect bridge to the next section.

4.4. Agile software development

Agile software development is all about adaptability in (highly) turbulent (business) environments. It can best be described by twelve principles [12].

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Business people and developers must work together daily throughout the project.

Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Working software is the primary measure of progress.

Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

Continuous attention to technical excellence and good design enhances agility.

Simplicity--the art of maximizing the amount of work not done--is essential.

The best architectures, requirements, and designs emerge from self-organizing teams.

At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

As stated by the twelve principles, agile software development is all about frequent software delivery, close collaboration between customers and developers and coping with changing requirements. Agile software development practitioners value working software over comprehensive documentation and responding to changes over following a plan. This makes agile software development highly flexible. The frequent delivery of software suggests an incremental, iterative model.

> *If you limit your choices only to what seems possible or reasonable, you disconnect yourself from what you truly want, and all that is left is a compromise. - Robert Fritz*

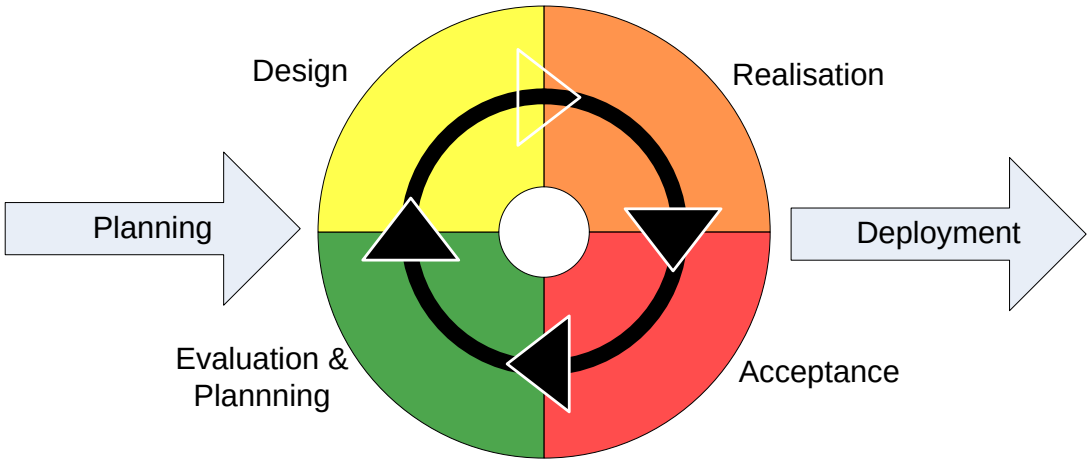


Figure 4 Iterative development
Source [19] p.17

Each iteration leads to a new increment in the realisation phase. This incremental design enables the project team to create islands of stability. Islands of stability are project milestones often in delivered software which can be used independently of future iterations. In case the project prematurely ends, the results of previous iterations will not be lost. This reduces the impact of project failure in terms of loss of money and time.

Iterative development processes also influences the requirement engineering processes. As Robertson & Robertson[15] state: *"The emphasis on iterative development means that the requirements phase is no longer completed before building begins"*. Therefore, many agile development methods define some kind of product backlog. This backlog contains (high-level) requirements that are on the waiting list to be built in future iterations. Often this happens by prioritizing the requirements and selecting the highest priority requirements for the next iteration.

You can use an eraser on the drafting table or a sledge hammer on the construction site – Frank Lloyd Wright

Agile software development methods often prefer the use of Use Cases over creating a full functional design [19]. Use Cases, like stated earlier, describe the interaction between an actor and the proposed system. Through the Use Case Specification clear *decomposition* can easily be achieved without the complex dependencies that can result from functional decomposition. Each Use Case in the Use Case Specification (which describes all the in Use Cases of the particular project) can be processed in a particular iteration. This way the results from a particular iteration are likely to be stand alone, which means they are independent of the results of future (or past) work. After each iteration an island of stability as well as a project milestone is achieved.

4.5. PRINCE2: Project Management Methodology

> *A project is like a road trip. Some projects are simple and routine, like driving to the store in broad daylight. But most projects worth doing are more like driving a truck off-road, in the mountains, at night.*
>
> *Cem Kaner, James Bach & Bret Pettichord*

Project based working becomes more and more popular over the past few years. PRINCE2, which stands for Projects in Controlled Environments [18], is a project management method focused on project based work. It defines an overall process and organization structure which can be used in (large) projects. PRINCE, the predecessor of PRINCE2, was originally meant for IT projects, PRINCE2 however is generally applicable.

In this thesis the emphasis will be on the process structure of the PRINCE2 methodology from a requirements analysis perspective. Therefore the organizational structure (like different roles and responsibilities and the relation between the project and the above lying programme) won't be discussed here. What will be discussed are the different phases during a project and the documents specified to support project management decisions.

PRINCE2 phasing

The PRINCE2 processes are defined by the PRINCE2 process model (see figure 5).

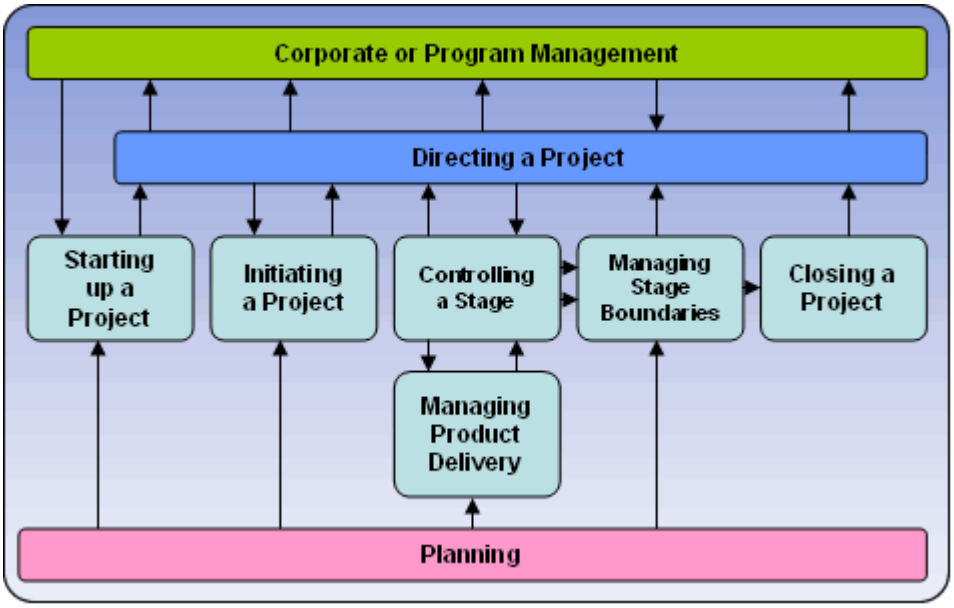


Figure 5: PRINCE2 process model
Source http://www.prince2.com

The process model looks like a sequential model with some alternation between 'Controlling a Stage' and 'Managing Product Delivery'. Requirements analysis is mainly important during the 'Initiating a Project ' phase and has some importance during the 'Starting up a Project' phase. As a consequence

of the sequential process these two phases only comes by once during a project. Therefore the focus is this thesis is limited to the first two phases (see chapter 7).

PRINCE2 documents

Before the 'Controlling a Stage' phase begins, three documents have been in order, the *Project Mandate, Project Brief* and *Project Initiation Document (PID)*. All three are briefly discussed below.

Project Mandate

The information in the project mandate is used to trigger the 'Starting up a Project' phase. It contains information about the project setup, project objectives and motivation. It also defines the scope of the project, the users and customers and a business case. When focussing on requirements. The project mandate provides an idea of who the solution is meant for and prescribes initial quality constraints (non-functional requirements).

Project Brief

The project brief is the result of the 'Starting up a Project' phase. At this point the information from the project mandate can be written down more specifically in terms of the project brief. In addition to the project mandate the project brief also contains information about the accepting criteria and the potential risks and their impact on the project. From a requirements perspective the more detailed business case in the project brief provides the first input for identifying the business requirements concerning why the business benefits from the result of the project.

Project Initiation Document (PID)

A budget tells us what we can't afford, but it doesn't keep us from buying it. - William Feather

The project initiation document (PID) is the result of the 'Initiating a Project' phase. In this document the whole project organization and structure is defined. The project is supported by a business case and the potential risks and quality assurance have been made explicit. From a requirements perspective the PID defines the high level business requirements. At this point the project's solution in terms of a system can be modelled and interfaces with other systems can be identified. By doing so, more money and time is spend on the first steps of the requirements analysis, before the project has even began.

5. Method

In this section explains and justifies the method used for conducting this research.

5.1. The research model

On a conceptual level, the research model van be described as in the following figure

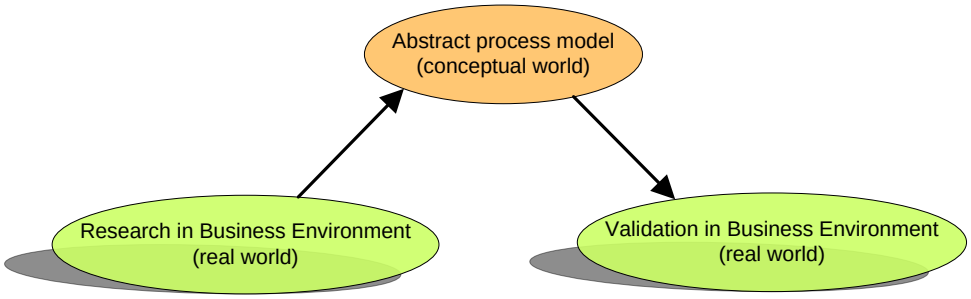


Figure 6: The Conceptual Research Model

> *Abstraction is one of the fundamental ways that we as humans cope with complexity – Grady Booch*

The research was conducted in a business environment which will be called the *real world*. In this real world the data needed for this research was gathered. The data consists of knowledge about the requirements engineering field and information about the *metarequirements*. The term metarequirements will be used to describe the requirements of a requirements engineering method. The research was done through several phases which will be described in the next section.

5.2. Phases

Figure 7 shows which phases are in order and through which steps the problem from data gathering to conclusion.

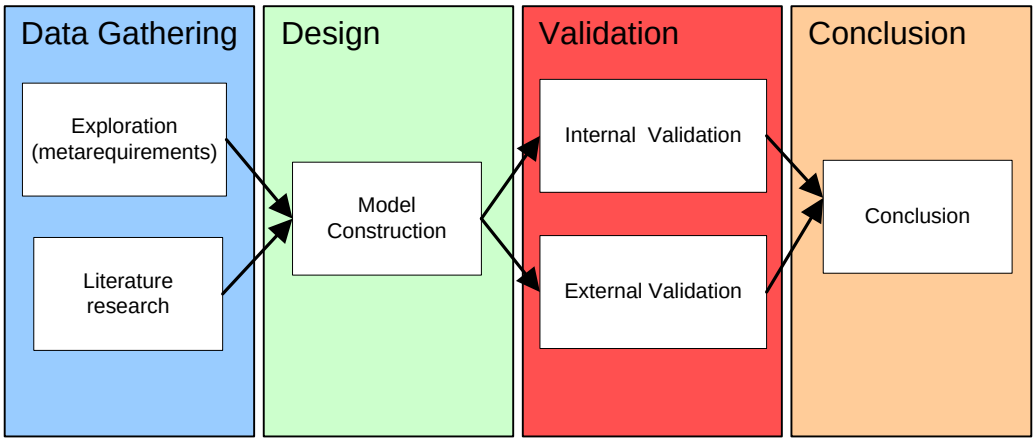


Figure 7: The Phased Research Model

5.2.1. Exploration phase

During the exploration phase information was gathered on what exactly concerns requirements development and management and what goals should be achieved. As stated earlier the research was conducted in a data warehouse environment and as a result specific needs raised above surface The data gathering was done by observing the current situation, interviewing employee's performing requirement analysis and by interviewing several RE process experts within the ING organisation and outside of the organisation. Based on the interviews, observations and partially the literature research set of metarequirements was constructed. The metarequirements are divided in general metarequirements and specific metarequirements and can be found in section 6.

5.2.2. Literature research

Books, articles and internal reports (business environment) were used to gather additional data about the RE as a research field. In addition this study introduced scientifically validated methods, concepts, success and failure factors, techniques and management frameworks that could be used in the design of the RE process model suggested in this paper. A summary of the most important subjects and concepts is presented in the last section, the theoretical framework. References of the articles and books that were used can be found in Chapter 12. For secrecy and compliance reasons the internal reports cannot be found in this section.

5.2.3. Model Construction

In the model construction phase the information gathered about RE processes from the literature and the interviews with RE experts were mapped on the metarequirements and a new model began to arise. At first the model was nothing more than a few fragments that covered specific metarequirements, but through an evolutionary process involving frequent feedback from both experts and stakeholders the model evolved into a full RE process description including different roles, activities, deliverables and performance measurement. Figure 6 conceives the perception of a sequential progress from one phase to another, but in reality progress was made in a much more dynamic and evolving way especially during the data gathering and design (model construction) phase that typically alternated. This way the model has grown towards the metarequirements resulting in an excellent (theoretical) fit.

5.2.4. Internal Validation

A theoretical fit. But in order to be a success the model has to be practically applicable. Therefore the model was both internally and externally validated. In the internal validation the model's consistency was checked. This was done by running a case study in the business environment. Not only did the case study validate the consistency of the model, it also provided input for the external validation.

5.2.5. External Validation

The external validation is all about insuring the model's applicability in general situations. Although the specific details of the model may be only applicable to certain situations or environments, the conceptual model behind the process model applies to requirements engineering issues in general. So when using the model (in different situations) a need occurs from a management perspective which can best be described by the questions *How well are we doing?*. In order to provide answers to this question a couple of parameters were defined that are critical to the success of requirements engineering processes. Input from both literature and the case study were used to define the parameters. Based on these parameters a *performance measurement system* (PMS) can be

constructed to measure how the requirements analysis goes. Finally a prototype of this PMS was build. For more information please see chapter 9.

5.2.6. Conclusion

Finally the results from the model construction, the case study and the performance measurement system were summarized as the results of this research.

> *By failing to prepare, you are preparing to fail.*
>
> *– Benjamin Franklin*

6. Metarequirements

He who asks a question is a fool for five minutes; he who does not ask a question is a fool forever – Chinese proverb

The requirements of the requirement engineering process model are called the metarequirements. These requirements are a high level description of what the process model needs to accomplish in order to succeed as a requirements engineering process model fulfilling the problem described in the problem statement. The requirements are the foundation on which the new RE method is built. For readability purposes the metarequirements have been divided into four categories.

6.1. General requirements

- Include the aspect that RE is both about problem investigating and solution specification
- Show how traceability is realized from problem investigation to solution specification
- Show forward and backward horizontal traceability
- Show upward and downward vertical traceability
- The method should begin with higher level requirements and work its way down to more refined system requirements
- The method should balance between flexibility and control
- The method should embrace changing requirements
- The method should prioritize requirements
- The method should be agile in order to prevent rework due to changing requirements as much as possible
- The method should describe the different roles, their tasks and their responsibilities involved in the requirements engineering process.

6.2. Requirements for communication

- The method should stimulate the communication between different stakeholders, requirements engineers and information analysts.
- The concepts of business requirements, customer requirements and system requirements should be made clear
- The method should stimulate stakeholders' involvement with the project
- The method should contain a vocabulary

6.3. Requirements for quality

- The method should enable a sufficient transition from requirements to functional design
- The method should raise the quality of the requirements specification by using templates and standards
- The method should stimulate an uniform perception of the problem at hand and the proposed solution

6.4. Requirements for applicability

- Use a framework to abstract from different project variants
- The method should be independent from different solution types (e.g. business or system solutions)
- The method should be flexible applicable depending on the project
- The method should not only describe what to do, but also how to do it

As one can see some (general) requirements hold for any requirements engineering method and some of the requirements listed above are based on the specific business environmental characteristic.

7. Method for Requirements Development & Management

The suggested process model that describes the new method for requirements development and management is mainly based on the metarequirements listed in last chapter. Next to the metarequirements the three sub problems stated in the problem statement form the three most important points of attention. Therefore this chapter will cover the coverage of those three pillars and the description of the model on a conceptual level. For a full and detailed description see appendix I.

> *Good ideas are not adopted automatically. They must be driven into practice with courageous patience. - Hyman Rickover*

7.1. The suggested process model

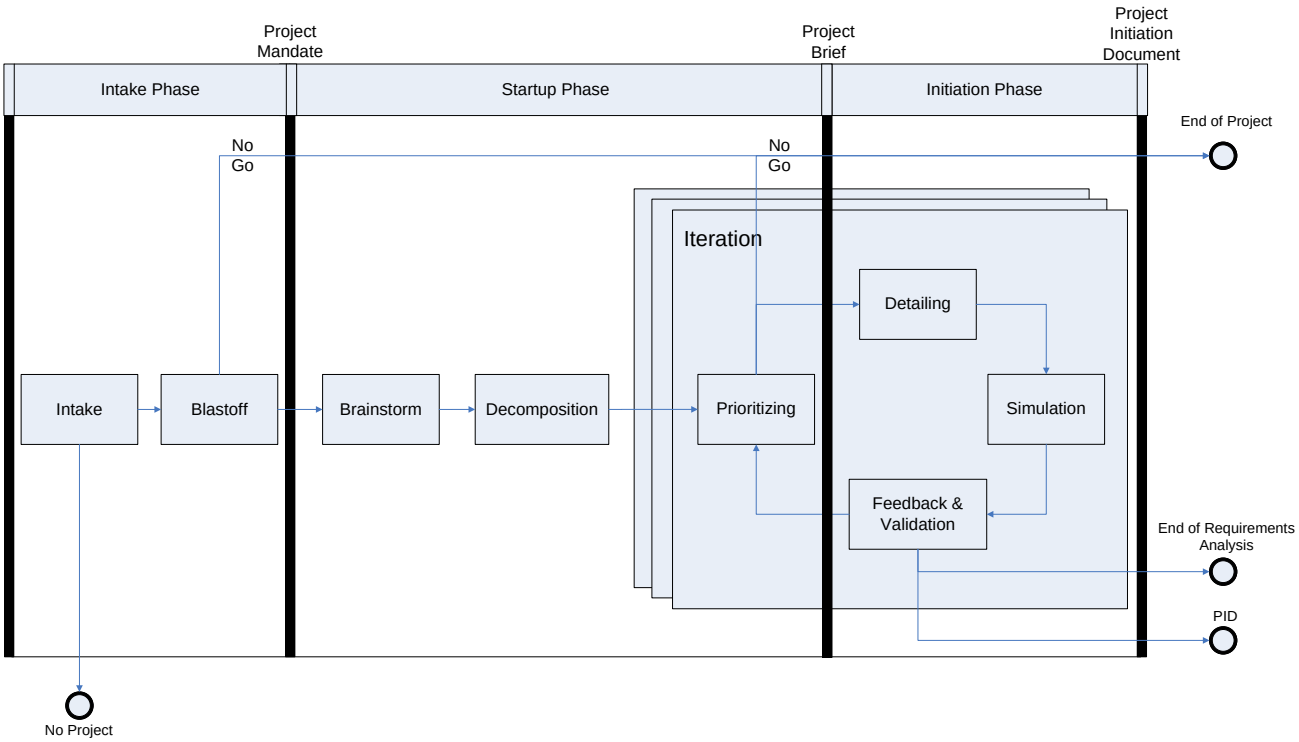


Figure 8: Process Model Projects

7.2. Balance between flexibility and control

The first challenge the model has to overcome is finding a balance between flexibility and control. The right balance differs from project to project and can be determined by the project management. In this section the model's key items for flexibility will be discussed as well as the key items for maintaining control.

7.2.1. Creating flexibility

Like stated before, not the whole model is explained in this section, therefore we jump to the fourth phase, the decomposition phase. In a turbulent business environment changing requirements are likely to be facts. While dealing with changing requirements, rework is inevitable. Therefore it is unwise to fully crystallize and specify all the requirements discovered during the brainstorm phase.

Consider a situation in which you did fully specify all the requirements in detail and just when you finished the last requirement something happens in the world that heavily affects the requirements you specified first. Most of the energy put to specifying this requirements is now lost and you have to redo a lot of work. Much wiser is it to decompose the total amount of work into smaller simplified pieces and specify only one or a few pieces at a time. This way changing requirements only affect the work of a small portion and the amount of rework is minimized. The process of decomposing the total amount of work and simplify it into small controllable pieces is done during the decomposition phase (see model). Of course this method only holds when the specified requirements from a particular are taken to the next phase of the process model and not put aside until all parts are specified. In conclusion, the decomposition phase enables partitioning the total amount of work into smaller pieces enabling to work more flexible with the different pieces.

The great art of learning is to understand but little at a time - John Locke

Once the partitions are made the set of requirements have to be prioritized. By doing so, the requirements with the highest priority can be select to be specified in detail, put into simulation and be validated by the stakeholders. Once this process is done and the stakeholders are satisfied with the functionality described by the requirements be build. Once the building process starts, a new set of requirements can be selected and follow the same route through the process. This results into the iterative structure as displayed in figure 8.

When requirements change over time three situations can be described.

1) the changes affect requirements that have not been fully specified yet
2) the changes affect the requirements currently being specified
3) the changes affect requirements that are already in the building stage

In the first case, the changes can easily be taken into account while only little time and effort have been invested and the requirements are only described on high level. In the second case some rework might be necessary, but since we decomposed the work into small pieces the rework should be minimal. In the third case the requirements cannot be adapted easily, since the building process already started. At this point, those requirements may not change anymore and the necessary changes have to be written into new requirements. The new requirements can be selected and the management can choose which requirements to select for a next iteration. In all three cases flexibility is realised as good as possible. Of course the level of flexibility that can be achieved depends on the project characteristics.

In order to strengthen this principle the selected work for a particular iteration should be as small as possible. The shorter the iterations, the more flexibility is gained. However, it is important to realise that the requirements selected for a particular iteration are independent of requirements outside the iteration. If this independency is realised, the functionality created when the requirements are built can be used independently from future work. An *Island of Stability* is created.

7.2.2. Maintaining control

Scattered across the model there are certain decision points. Once arrived on this point the information gathered during the prior process will be used to decide whether or not to continue the process from here on. The decision points are indicated by the thick black vertical lines (see figure 8). The information gathered during the process must be gathered and formed into the documents *Project Mandate*, *Project Brief*, and *Project Initiation Document*.

These documents do not only contain information about the requirements, but mostly project management information. However the Requirements Manager is responsible for advising the Project Manager from a requirements perspective[2].

When the first two decision points are marked as 'Go' the iterations start. After each iteration there will be another decision point to decide whether or not to execute another iteration. There is no decision document necessary for this decision although it should be taken carefully. After all, a project cancelled halfway or even at the start-up is not necessarily a bad executed project. It may just be the right decision at the given moment.

Due to the frequent decision points prior and in between different iterations, project management is able to maintain vision and control over the project. When projects are cancelled, the loss should be minimal and sometimes the results made so far are still useful. This way project management will be able to monitor the progress frequently, identify upcoming risks and can steer the project in a flexible way. Project management stays in control over the project.

7.3. Applicability to both small and big projects

Of course the requirements analysis needs to be done carefully, but for smaller projects it may also lead to an overkill of analysis with unnecessary loss of time and money as a result. Therefore the model needs to be applicable to both smaller and bigger projects. Applicability to both small and big projects is realized in two ways. The first way is to distinguish different kinds of projects, the second way concerns the iterative characteristics of the model.

7.4. Distinction between types of projects

In order to be able to make a distinction between different project some criteria were defined. These criteria were defined together with management from the business environment. As a result five parameters were identified. Their values can lead to three types of projects which will be called *Job, Change* and *Project*. The parameters are *Number of stakeholders*, *Project Size, Use of Data, Timeframe* and *Using Frequency*.

Number of stakeholders[3]

[2] More information about different roles can be found in appendix I

The numbers of stakeholders often indicates the complexity of the requirements and their dependencies. The more stakeholders there are, the higher the risk of changing and conflicting requirements, so when the number of stakeholders is high, a more careful requirements analysis is in order. In the specific business environment in which the research was conducted, the amount of stakeholders was set to five meaning that if there were more than five different stakeholders, the type of project resulted in *Project*.

Project Size

The size of the project in terms of budgeted hours gives insight in how much time and money is at stake. Therefore the size of the project is a good indicator how to qualify the problem at hand and the necessary requirements analysis. During the research the amount of hours was set to 200 hours, meaning that if the budgeted hours exceeded the 200, the type of project resulted in *Project*.

Use of Data

This indicator specifically applies to information systems in a data warehouse environment, but it may also apply in more general sense. The use of data indicates whether new data has to be gathered or the existing data will be sufficient to solve the problem. This indicator indicates whether or not much building capacity is required to solve the problem.

Timeframe

The timeframe indicator estimates the time span between the start and completion of the project. The larger the time span, the higher the risk of changing requirements.

Using frequency

The last indicator determines how many times the solution will be used. Sometimes solutions to a highly important business problem will only be used once. These kind of problems ask for an immediate solution and an excessive requirements analysis may be redundant. This assumption can also be a different one, so this indicator comes last in the decision tree which looks as follows.

[3] The type of stakeholders here are limited to those who provide the requirements

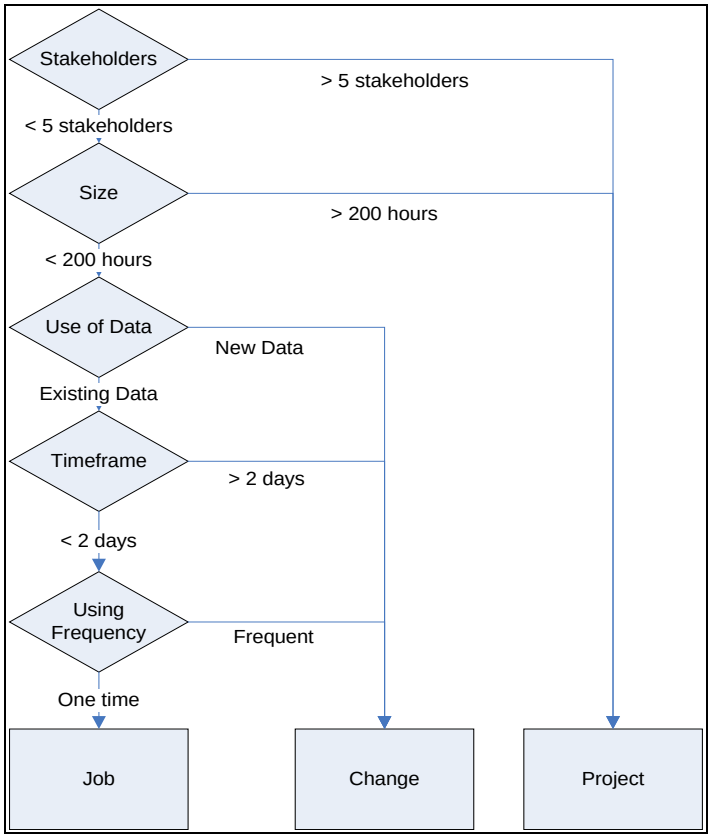


Figure 9: Project type decision tree

These decisions are done during the intake phase of the model. Therefore not much information is at hand already so the value of the indicators are mostly based on estimation. However the thresholds of the indicators can differ in different environments and be tuned accordingly. This requires experience and insight in the specific business environment.

The tree finally results in one of the three types of projects. If a project is of type *Job*, no structured requirements analysis is required. Ad hoc solutions can be provided in line with the small size and low complexity of the project. If a project is of type *Project* all the phases of the process model (figure 8) should be run though. If a project is of type *Change* the project requires a decent requirements analysis, but not all phases are necessary and some steps can be combined. This leads to the process model beneath[4].

[4] For more information about the process model for Change projects can be found in appendix I

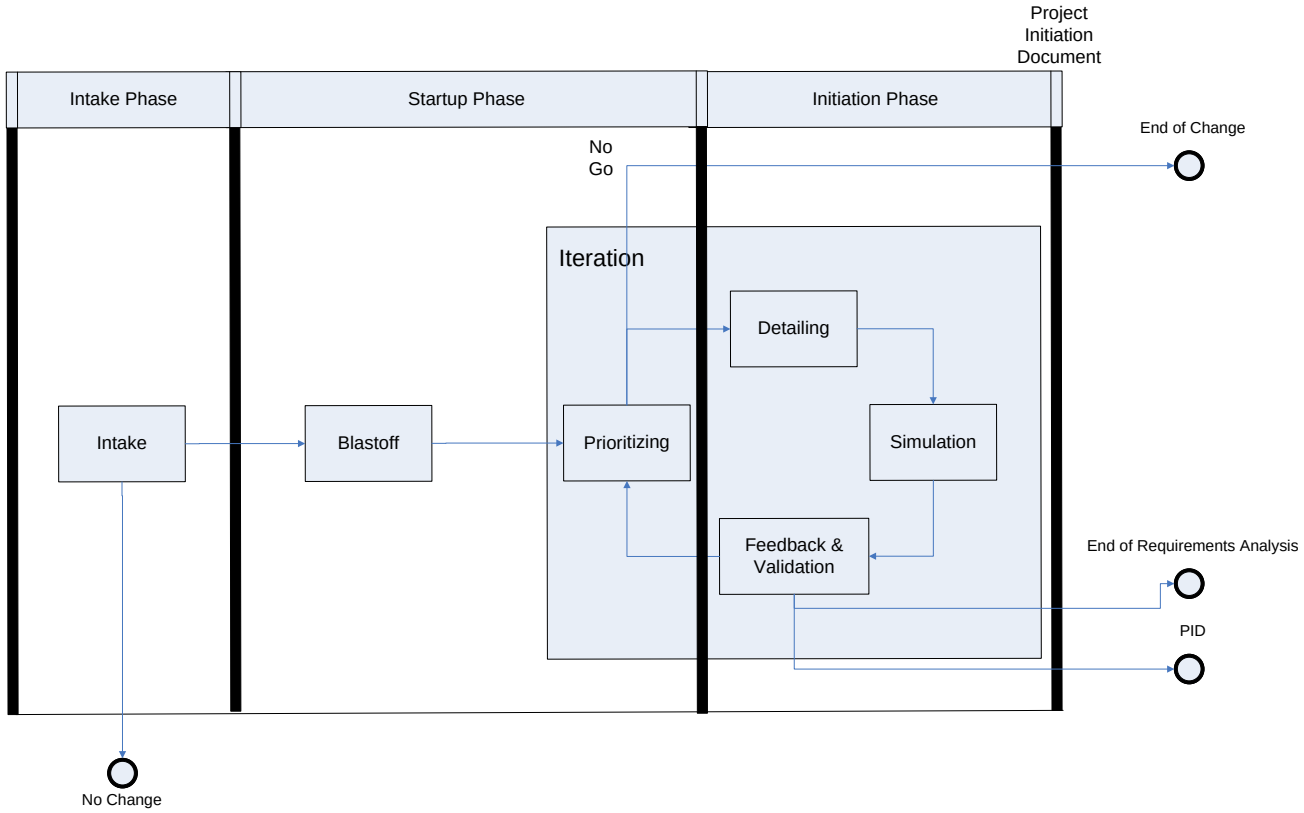


Figure 10: Process Model Change

7.4.1. Iterative characteristics

The iterative characteristics of the model introduces a wide applicability to various kinds of projects. As visualized in the model(s) some initial steps are always required (like identifying stakeholders and conducting an initial brainstorm to discover the first requirements), but the model does not prescribe how many iteration cycles should be taken. This depends on the project at hand and the result of the decomposition phase (in case of project type *Project*). Depending on the project the stopping criteria may differ as well. For example the iteration cycles may stop when all budget is used or when all the business requirements have been specified into customer- and system requirements. In conclusion the iterative structure of the process model strengthens the applicability to both small and larger projects.

Have no fear of perfection - you'll never reach it. - Salvador Dali

7.5. Maximize the quality and completeness of the requirements specification

As stated in the second sub problem the quality and completeness of the requirements specification needs to be maximized. This means that the process model should lead to an output specification in which all (or nearly all) requirements are specified in such way that they are clear and unambiguous. It is also very important that all requirements contribute to solving the business problem and any unnecessary or unimportant requirements are left out.

7.5.1. Maximize the completeness of the requirements specification

Maximization of the completeness of the requirements is realized in two ways. The first is by maintaining explicit bi-traceability. The process model prescribes that all system requirements should be captured in customer requirements, and all customer requirements should be captured in the business requirements. This is done by giving low level requirements explicit references to higher level requirement. By doing so all the requirements directly or indirectly contribute to the solution of the business problem. When customer requirements are identified which cannot be pointed to a higher level business requirement the customer requirement me either be excessive or it leads to new business requirements that haven't been identified yet. This forms an extra stimulus to maximize the completeness.

Think like a wise man but communicate in the language of the people. - William Butler Yeats

The second way is by intensive collaboration between stakeholders, requirements engineers and information analysts. Due to the frequent contact moments and collaboration during requirements gathering and validation sessions, the project is kept alive in the minds of the stakeholders. This stimulates the creation of new idea's ultimately leading to new requirements. It also leads to early discovery of changing requirements and new insights which reduces the risks of exceeding planning and budget. Finally the intense collaboration leads to strong mutual understanding of how to proposed solution is going to work. Especially the attendance of the information analyst is important in this matter. After all, the information analyst is the one responsible for translating the requirements specification into design. The mutual understanding not only contributes to the completeness, but also to the quality of the requirements specification and the requirements analysis as a process.

7.5.2. Maximize the quality of the requirements specification

> *Language is the source of misunderstandings.*
> *- Antoine de Saint-Exupery*

The quality of the requirements specification can be defined in terms of how well does the specification as an instrument transfer an understanding of what the proposed system should do. In other words, how capable is the requirements specification in creating a common perception of the proposed system among its readers. The process model tries to maximize the quality of the specification in two ways. The first way is by providing guidelines for formulating requirements. One of these guidelines is formulating requirements SMART[5]. This way ambiguity in the written requirements is prevented as much as possible. Another guideline is the use of templates that ensure a structured and uniform way of writing down requirements.

The second way is by simulation the requirements in the simulation phase. By simulating the requirements the mutual understanding of the functionality is put to the test. The information analyst will simulate the requirements according to his or her interpretation. When this interpretation is inappropriate, it will be signalled during the validation of the simulation. This way different people will share a common understanding even if the written requirements might leave room for different interpretations. Simulating requirements also makes the validation process much easier, because stakeholder can see the result and check if it matches their true needs.

7.5.3. Roles & Responsibilities

Two other subject worth mentioning are the *Roles & Responsibilities* and the *Stakeholders' involvement*. The document describing the Requirements Development & Management process model also contains a chapter in which the different roles and their responsibilities are explained.

8.

> *The ideal engineer is a composite ... He is not a scientist, he is not a mathematician, he is not a sociologist or a writer; but he may use the knowledge and techniques of any or all of these disciplines in solving engineering problems. - N. W. Dougherty*

[5] See Appendix I

Case Study: Customer Profile Service (CPS)

The Requirements Development & Management process model was put to the test in a case study performed in the business environment. In this chapter the case study's situation, approach and conclusions are discussed.

8.1. Situation

At the beginning the drawing board wasn't completely empty. The problem space, idea of solution and some project constraints were already known. Nevertheless the process started right at the beginning and walked through all the prescribed phases. The case concerned a service providing a customer profile to various operational systems. The scope was limited to the content of the service and its interface (service oriented). The integration of the service in the applications using the service were considered outside the scope. The system can be schematically represented as done in figure 11.

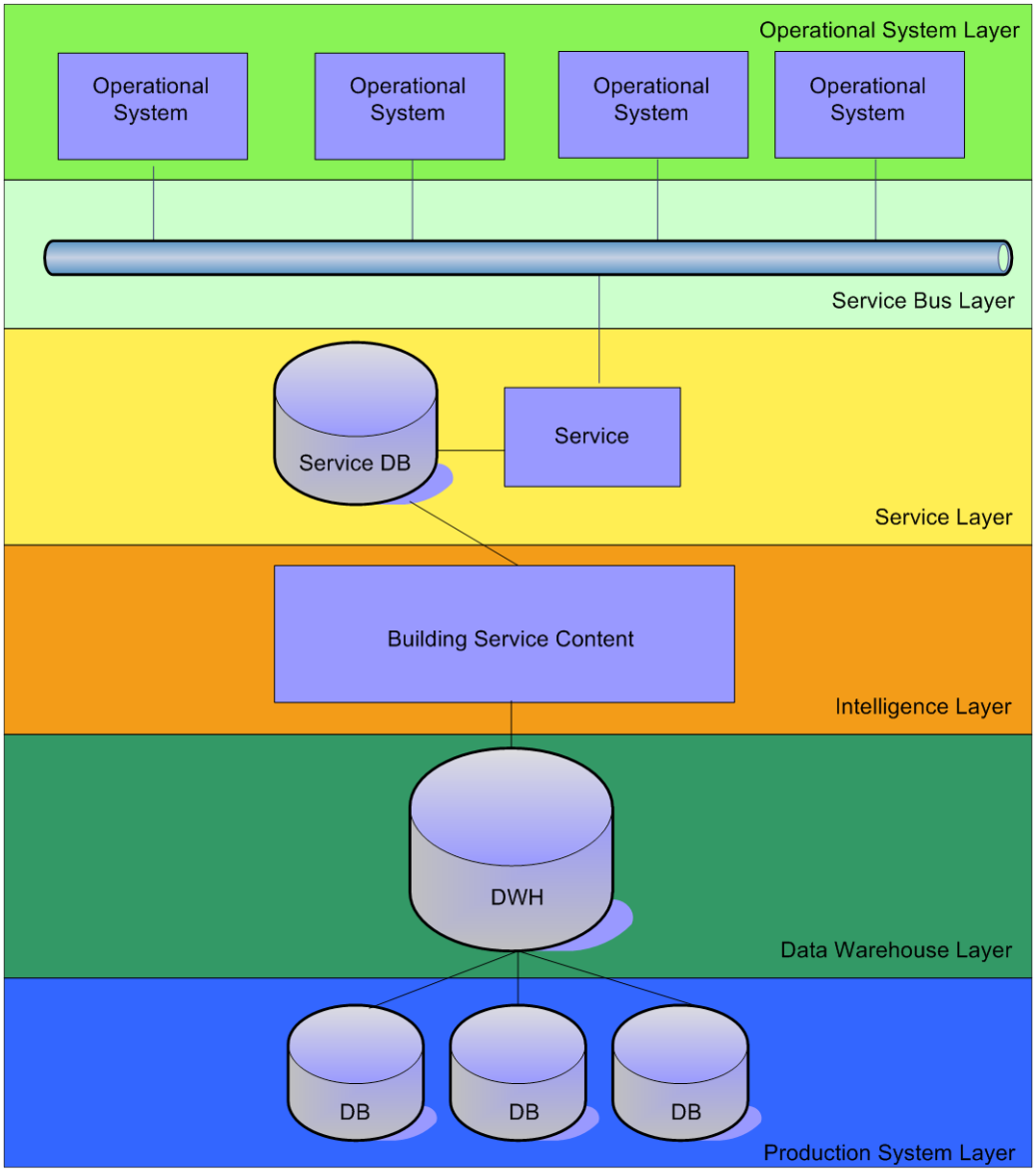


Figure 11: The CPS system sketch

> *Computers make it easier to do a lot of things, but most of the things they make it easier to do don't need to be done.*
> *- Andy Rooney*

Starting at the bottom layer in figure 11 the *Production System Layer* can be found. This layer concerns all the production systems from various business units. In the *Data Warehouse Layer* this information is gathered, combined and stored. The data from the data warehouse can be processed in the *Intelligence Layer* to create the desired content of the database in the *Service Layer*. This particular content can be spread real-time across the service bus in the *Service Bus Layer*. Finally the operational systems in the *Operational System Layer* can use the information in their applications. The scope of this case study was limited to the *Service Layer* and the *Intelligence Layer*. Textually expressed the scope was limited to defining the content of the service and the intelligence (mostly mathematical) to construct the content from the data in the data warehouse.

8.2. Approach

At the beginning of the case study five stakeholders were identified and introduced to each other. As the process model prescribes, the requirements identified during the brainstorm session were decomposed into several subjects. Each subject (containing one or more requirements) was detailed during an iteration. In total four workshops took place. The intention was to start each workshop with an evaluation of the results of last workshop, consider changing requirements and select and process the requirements for the upcoming iteration.

In order to measure the process all the participants have been asked to communicate via e-mail, through a single point of contact. This way, all the communication could be monitored, inside and outside the formal meeting points. The objective was to find any abnormalities, bottlenecks and occurring problems. The monitoring of the process was conducted by the researcher supported by an experienced requirements manager. Finally the results of the case study (in terms of deliverables) and experiences with the way of working were evaluated with all parties.

8.3. Conclusions

The conclusion of the case study twofold. On the one hand the case study can be concluded in terms of achieved deliverables and on the other hand the case study can be concluded in terms of how well the process to achieve the results unfolded. In other words how well the new requirements development & management process performed.

8.3.1. Deliverables

The deliverables were all quite satisfying and all the stakeholders involved were happy with the results in terms of specified requirements. This satisfaction can be seen as the validation that their needs and wishes have successfully been translated into a requirements specification. The information analyst who was involved during the whole process indicated that the requirements were clear and understandable enough to make the functional design.

One deliverable that was failed to achieve during the case study was the full simulation of all the requirements. The evaluation and validation of the requirements was performed by discussing textual requirements, data models and decision trees, but not by actual simulation. In the particular case the impact of this loss was only minor due to the fact that the CPS is meant to be used by

application and not directly by the stakeholder's themselves. Nonetheless the fact that the team failed to simulate the requirements on time did have some consequences on a methodological level.

8.3.2. The method

On a methodological level the conclusions were both positive and negative. Like stated above the simulations of the requirements could not be delivered in time. After some evaluation the team concluded that the failure was caused by the lack of dedication of the information analysts to the project. The CPS wasn't the only project the information analyst was involved in and other project gained higher priorities during the time being. In practice this is almost always the case and it is very exceptional to have exclusive dedication from all the parties involved. The iterative way of working as suggested by the process model will therefore only work when all parties show enough dedication to the project. This might lead to participating in less project at the same time, but finishing projects faster before starting another. This conclusion, although an interesting one, is more related to project management issues than requirements engineering issues.

When purely looking at the requirements engineering process the method came out quite positive. The frequent interaction enabled strong collaboration between all the stakeholders. Their communication, especially between the customers and the information analyst, resolved interpretation issues and communicational problems. The frequent workshops also kept the projects alive in the minds of stakeholders and positively stimulated their involvement. Practically this resulted in e-mail communication after working hours and new idea's and improvements at each workshop. Changing requirements occurred, but their impact remained at a minimum.

In conclusion the case study proved the suggested method as very useful to the business. The communication with the information analyst remains a point of concern, but the expectations are it will grow better over time. The results and experience collected during the case study also provided input for determining key success factors. When able to measure these factors, a project team can determine how well they are performing in the process. A tool to perform this measurement is presented in the next chapter.

9. Measure the model's performance

The objective this chapter is to measure the method in terms of how well the requirements engineering process is performing. In order to do so 11 parameters were identified. In the second paragraph of this chapter the model's consistency will be argued with the help of these parameters. The parameters are mapped to three of the phases described in the process model. These are the *Intake phase*, the *Startup phase* and the *Initiation phase*. The philosophy behind this mapping is that some parameters only concern a particular phase, while others are relevant in all the phases. The parameters are measured by indicators in terms of questions given to the project team (typically to the requirements manager) . The answer to these questions measures the situation at hand and the tool presented in the last paragraph of this chapter displays this situation in a graphical and quantifiable way.

The input leading to the parameters came from three perspectives. The first perspective is a scientific perspective [verwijzing]. In their research, Emam & Madhavji identified 34 key success factors of requirements engineering processes in information system development. Some of those factors were already specified into metarequirements (see chapter 6), while other factors involved more human aspects. The factors relating more to project management than requirements management were left out of consideration. The second perspective is an empirical perspective. The experience collected during the case study helped to identify and understand some of the parameters. The third perspective is an expert perspective. During a three hour workshop with eight requirements engineers from the business environment the parameters were discussed, supplemented and improved.

9.1. Parameters & Indicators

In the table below the parameters and their indicators are listed. In the *Argumentation* column a brief motivation for the particular indicator is given as why this indicator measures the parameter. The *Phases* column shows the phases in which the parameter is applicable.

Parameter	Indicator	Argumentation	Phases
Stakeholder involvement	How much input do the stakeholders provide?	The more input stakeholders provide, the higher their involvement to the project.	Intake Startup Initiation
	How many stakeholders attended the workshop(s)?	The more stakeholders (of the total stakeholders) attending the workshop(s), the higher their overall involvement.	
	Who initiated the project?	When the stakeholders initiated the project themselves, their involvement will be higher than if someone	

		else initiated the project.	
	What drives the stakeholders?	If stakeholders have to participate (e.g. orders from the boss) their involvement will be rather low. If stakeholders participate because the projects concerns their field of expertise their involvement will be rather high.	
	Is there any communication between stakeholders outside the formal meetings?	If so, the stakeholders show more involvement to the project.	
Stakeholder quality	How well do the business stakeholders communicate?	Measure the quality of the business stakeholders in terms of communicational skills.	Intake Startup Initiation
	How well do the IT stakeholders communicate?	Measure the quality of the IT stakeholders in terms of communicational skills.	
	How much knowledge do the business stakeholders have?	Measure the quality of the knowledge of business stakeholders.	
	How much knowledge do the IT stakeholders have?	Measure the quality of the knowledge of IT stakeholders.	
Stakeholder Mandate	Do the stakeholders have the power to make decisions concerning the project?	Measure the mandate the stakeholders have.	Intake Startup Initiation
Conflicting requirements	Did any conflicting requirements occur during the workshops?	Measure if the conflicting requirements exist at all.	Startup
	What is the impact of the conflicting requirements?	Measure the impact of the conflicting requirements to the project.	
Independency of	Can the selected	If the answer to this question	Initiation

requirements	requirements be simulated without future work?	is yes, the selected requirements are independent of future work during the simulation phase.	
	Can the selected requirements be built without future work?	If the answer to this question is yes, the selected requirements can be built independent of future work.	
Controllable amount of work	How many percent of the total requirements are selected for the current iteration?	The higher the percentage, the higher the risk of uncontrollable amounts of work.	Initiation
	What is the complexity of the requirements?	The higher the complexity, the higher the risk of uncontrollable amounts of work.	
Document validation	How many percent of all the requirements been validated by the stakeholders?	The higher the percentage, the higher the validation.	Initiation
	How many percent of all other documents (models etc.) been validated by the stakeholders?	The higher the percentage, the higher the validation.	
Requirements simulation	How many percent of the selected requirements have been simulated?	The higher the percentage, the better the requirements simulation is performing.	Initiation
Changing requirements	How many percent of the selected requirements have significantly changed during the iteration?	The higher the percentage, the more changing requirements among the selected requirements.	Initiation
	How many percent of the total requirements have significantly changed during the iteration?	The higher the percentage, the more changing requirements among the total requirements.	
Rework	How much rework is necessary as a result of the changing requirements?	Measure the amount of rework.	Initiation
	How many percent of the	Measure the range of the	

	total requirements needed rework.	rework.	
Deliverables	How many percent of all the agreed deliverables have been delivered?	The higher the percentage, the less missing deliverables.	Intake Startup Initiation

The questions used to determine the indicators and to measure the parameters can be answered by the requirements manager and project manager. The managers provide the answers according to their perceptions of the project. This way the managers are forced to think about how the project is developing and the parameters are nothing more than the expression of their thoughts in a quantifiable way. However the strength of this mechanisms is that managers are forced to think of aspects that often are taken for granted. A requirements manager can have the feeling that there are some changing requirements, but the focus is often laid on dealing with the changes rather than to discuss the potential impact on the project. When made explicit that the changing requirements can form a serious obstacle in the projects' progression the team can take the necessary actions to deal with the changing requirements or try to prevent them from reoccurring.

9.2. Argumentation

So why are these parameters? A method can be seen as a transforming mechanism. Something goes is called input, the input is processed through the method and transformed into output. Considering a requirements engineering method a business problem goes in (input) and a requirements specification of what the solution to this problem should look like comes out (output). Methods commonly processes the input through certain steps and activities. When looking at the method in this research these steps and activities form the process model (as shown in figure 8).

To show the model's reliability and argue that the model is sound the transitions from input to output will be reasoned. This reasoning will be done on both content and quality level. On a content level the main output of the model is the requirements specification. In chapter 7 and appendix I is explained why the model leads to the requirements specification and how the transitions (in terms of deliverables) from phase to phase should go[6]. On a quality level the transitions level will be discussed with the help of the parameters. The reasoning on a quality level is done to avoid the 'rubbish in, rubbish out' effect.

> *The avoidance of mistakes establishes the certainty of victory, for it means conquering an enemy that has already been defeated.*
> *- Sun Tzu, The Art of War*

There are three major transition points in the model. These points are located at the end of each phase (e.g. Intake, Startup or Initiation). It is arguable that potential risks to the project within a particular phase can be solved by the team due to their intensive collaboration and communication. By mutual adjustments most problems within a phase can be

[6] To see how the phases elapse in detail see Appendix I

easily solved by the project team[21]. Between phases the importance of the different roles change and also the focus on the deliverables changes (from organizational to high level business requirements to detailed specifications[7]). Therefore it is arguable that the crux lies after each phase and the parameters should zoom in on these transition points[8]. This is the main reason why the parameters in the table above are linked to one of these phases. Next will be explained why these parameters are applicable to the particular transition point(s).

9.2.1. Intake phase

Stakeholder involvement

During the intake phase the first contact is made with the stakeholder. It is important to notice how involved the stakeholders are in the project. Their involvement determines how well the requirements can be addressed in the Brainstorm phase. In addition, research has shown that the lack of stakeholder involvement is the number one reason for IT project failure [14].

Stakeholder quality

The quality of the stakeholders, both on communicational level as from a knowledge perspective plays an important role as well. Of course the stakeholders have to posses the business knowledge to determine what the best solution should look like, but the stakeholders also have the posses the communicational skills to express it. If this is not the case, the requirements engineer needs to question every detail and take none for granted for it may just be a false assumption.

Stakeholder Mandate

The method is based on short cycles and intensive collaboration in order to flexibly deal with changing requirements and to derive quick results. The lack of stakeholder's mandate can cause project delays while every major decision needs to be discussed with the person who actually has the proper mandate. The person with the proper mandate should be a stakeholder in the project.

Deliverables

In order to guide the project team through a structured way of working, all the agreed[9] deliverables should be yielded.

9.2.2. Startup phase

Stakeholder involvement

---

[7] and finally to functional or technical design

[8] Two exceptions: The parameters *Independency of requirements* and *Controllable amount of work* are mainly applicable at the beginning of the initiation phase. They warn about potential risks during the iteration.

[9] This can either be all the deliverables prescribed by the method (see appendix I) or just the deliverables the project manager and requirements manager selected.

The same reasons of why stakeholder involvement is important apply during the startup phase as they did during the intake phase. In addition the stakeholder involvement now can predict how well the requirements can be specified during the first iteration.

Stakeholder quality
This parameter is still applicable for the same reasons as during the intake phase.

Stakeholder Mandate
This parameter is still applicable for the same reasons as during the intake phase.

Conflicting requirement
Conflicting requirements need to be solved and it is better to solve them while they are still high level requirements than to wait until they are all specified. The iterations need to run as smooth and fast as possible, so conflicting requirements should be signaled and dealt with before.

Deliverables
In order to guide the project team through a structured way of working, all the agreed deliverables should be yielded.

9.2.3. Initiation phase

Stakeholder involvement
During this phase the stakeholders need to focus on the detail while specifying the high level business requirements into low level system requirements. A good thought about the details is essential for the quality of the requirements specification. Stakeholders with high involvement in the project think more carefully about these details.

Stakeholder quality
This parameter is still applicable for the same reasons as during the intake phase.

Stakeholder Mandate
This parameter is still applicable for the same reasons as during the intake phase.

Independency of requirements
The independency of requirements concern the selected requirements for a particular iteration. In order to create *Islands of Stability* and to deal with changing requirements the selected requirements need to be independent of future work.

Controllable amount of work
Like stated earlier, the iterations must be as short as possible in order to maximize the adaptability and flexibility. Therefore the amount of work selected for a particular iteration needs to be controllable and as simple as possible.

Document validation
All the documentation (requirements and other documentation) need to be validated by the stakeholders in order to avoid misinterpretations of the documentation. An iteration should only be finished when all documentation has been validated.

Requirements simulation
Like stated earlier, the simulation of requirements helps stakeholders visualize the result of their requirements and makes the validation process more precise. Sometimes it won't be necessary that all requirements have been simulated, but even then it might contribute to make this visible.

Changing requirements
This parameter gives insight in the changing requirements and the impact on the project. It's a very important parameter that can best be measured at the end of the iteration. When performing low on this parameter discussion with the stakeholders is required and sometimes the project manager should decide to move back to the drawing table or redoing the iteration before moving on to the development.

Rework
This parameter determines how much rework is resulted from changing requirements and can therefore be best measured at the end of an iteration. When the project team performs low on this parameter the requirements and project managers might reconsider the chosen decomposition.

Deliverables
In order to guide the project team through a structured way of working, all the agreed deliverables should be yielded.

When measuring the parameters potential risks and problems can be identified[10] as soon as possible and the project and requirements manager will be able to respond accordingly. How they exactly should respond falls outside the scope of this research. Main reason for this is that the correct responds depends on the project, the environment and the people involved in the project. These factors can often be best be overseen by (experienced) managers.

9.2.4. Go / No Go decisions

Right at the transition points, the method prescribes a go / no go decision point. Measurement of the parameters can help decide whether or not to continue the project, even when all the goals on content level are met. When the quality of the input can't be ensured, the quality of the output can't either.

9.2.5. Reverse Engineering

The final argumentation for the method's consistency by using the parameters will be shown by *Reverse Engineering*. The desired output will be translated back through the parameters into the necessary input.

9.2.5.1. Initiation phase

At the end of the iteration phase a requirements specification should be created and validated by the stakeholders. Next to this document it is important that the information analyst knows what the stakeholders need. It is arguable to state that the desired output will be met during an iteration if the selected requirements are controllable and clear (on high level) and the team works closely together

[10] Due to low performance on certain parameters

with stakeholders to detail, simulate and validate the requirements. Taking multiple iteration cycles into account, changing requirements and rework should be minimized by short cycles.

The necessary input can be summarized as non-conflicting high level business requirements and the stakeholders' involvement and collaboration.

9.2.5.2. Startup phase

At the end of the startup phase the input for the initiation phase should be delivered. This holds a decomposed (for selection purposes) set of non-conflicting high level business requirements. It is arguable to state that this deliverable will be delivered when the project team together with the stakeholders brainstorm to identify the business requirements. Close collaboration will prevent conflicting requirements and solve these conflict when they occur. In order to let the startup phase perform well the project's organization, scope and constraints should be clear. Of course stakeholders involvement and mandate play an important role as well.

9.2.5.3. Intake phase

The intake phase will provide clearness to the project's organization, scope and constraints as soon as the deliverables are met. Of course the quality of the deliverables depends on the quality, involvement and mandate of the stakeholders.

9.2.6. Argumentation (summary)

The argumentation for the consistency of the model is argued with the help of 11 parameters that cover the transition points that occur between the different phases of the method. After the parameters have been introduced and the indicators by which the parameters can be measured have been presented, the argumentation starts by linking the parameters to the different phases (on a quality level). Next the link between the parameters and the decision points are made clear and finally the phases are linked together starting at the desired output and working down towards the intake of the project. In the next paragraph a dashboard is presented by which the parameters can be measured and visualized.

9.3. Performance Measurement Dashboard

As stated before. The parameters can be used to detect potential risks and points of attention during the requirements engineering process. In this paragraph a tool is presented by which the parameters can be measured according to the indicators and questions shown in the table on paragraph 9.1. After answering the questions the answers are translated into figures and combined with tunable weights (several values determine one parameter) the final values of the parameters are constructed. These values are presented in the tool as a percentage. The higher the percentage, the better the project team is performing on the parameter. In order to see in one glance how well the overall project is doing the values of the parameters are graphically shown by faces. The happier the face looks, the better the team is performing on the parameter. The happiness of the faces is supported by the color of the face (green, orange or red), so if red faces start to occur the team needs to take action accordingly. For a screenshot of the dashboard tool see below.

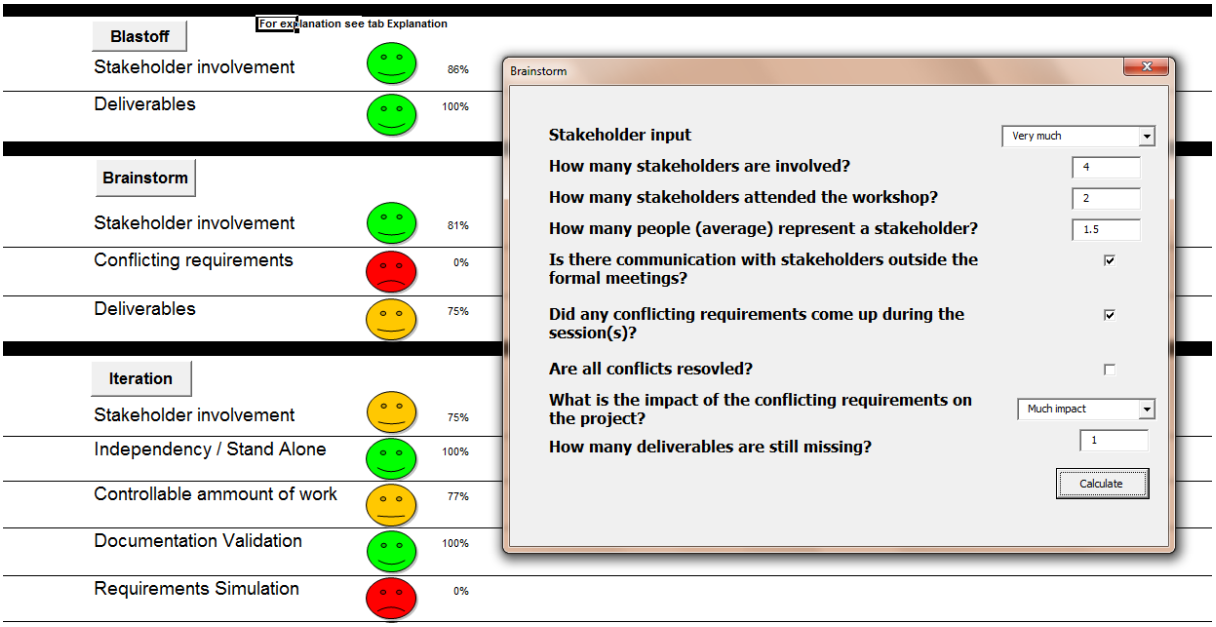


Figure 12: Dashboard screenshot

10. Discussion and further research

10.1. Discussion

Before moving on to the conclusion of this thesis some points of discussion are worth considering. The first point of discussion is the true goal of requirements engineering processes. Requirements engineering is about translating the whishes and needs from stakeholders into software design or solutions. In plain words, the software designers and developers need to understand what they should design or build. There are many ways to achieve this, nevertheless requirements engineering prevails as a mean to achieve this underlying goal. The requirements specification or models as deliverables of the requirements engineering processes and their quality is the central issue in many research as well as in this theses. However it is arguable that less attention should be paid to the actual documentation and more to the ways people in different roles communicate and collaborate. Simply put, the gap between business and IT (because this issue is broader than just requirements engineering) should in my opinion be closed by the way people in organizations collaborate than just by passing documents to each other. The role of requirements managers in this process will be to facilitate this collaboration. How this role should be fulfilled is likely different in every project and therefore very hard to measure and reason about. Although this thesis addresses the collaboration aspect, the input and output of the model and the consistency check of the method are based on the quality of documents, not directly on the way of working.

The second point of discussion the applicability of the method. The method focuses on highly turbulent environments and projects with uncertainty as a constant obstacle. It may not be very applicable to projects that are more routine and simple. Although a simplification of the full method[11] is presented as well, the method still demands certain steps and documents that might lead to an requirements engineering overkill in certain projects.

The third and final point of discussion is about the amount of control and restrictions on the method. When looking at true agile software development, documentation is left out as much as possible and the main focus is on achieving results in terms of working satisfying software. The method presented in this research still relies on a lot of documentation and can therefore never be as flexible as true agile processes. Evidently the control mechanisms used in the method are aimed at giving control to project- and requirements management. Big question here will be if this amount of control is still necessary when stakeholders and software developers are truly working together to achieve working software? Currently there is still a need for a man in the middle, a bridge between both worlds. This brings us partially back to discussion point one for the true problem is the way of working that creates the gap business and information technology, not the lack of quality of documents that suppose to close the gap.

> *There are only 10 types of people in the world - those who understand binary and those who don't. - Author unknown*

[11] see Appendix I

10.2. Further research

Resulting from this thesis, further research can be conducted in three different areas. The first area will be requirements management across projects. This research focuses on the lifecycle of requirements during the requirements analysis phase in software development. When using incremental software development models the lifecycle of the requirements extent to the whole software development process. However unaddressed in this research is the extension of the requirements lifecycle across projects.

A second point for further research is the performance measurement system for requirements engineering processes. The indicators and the weights used to value the parameters are not proved to be correct. Further research should focus on the alignment of these indicators and their weights to measure requirements engineering processes in turbulent environments more precisely.

The third and final point for further research is decreasing the amount of documentation while staying in control of the project and taking functional maintenance afterwards into account. Documentation should be used mainly as reference when dealing with maintenance issues and the function to distribute knowledge to others within a same project should disappear. It is very interesting to perform research to how everybody involved in the project should work together to stay in control and how this affects the organizational design of projects.

11. Conclusion

The conclusion of this thesis will be given by answering the research questions as stated in the problem statement. The main research question was:

"What should the process model of a method for requirements development and management, capable of handling changing requirements while maintaining project control, look like."

This problem was split up into three sub problems that together answer the research question. In this conclusion the answers to the sub problems will be summarized.

The first sub problem was the applicability to both small and big projects.
In chapter 7 the applicability to both small and big projects is handled in two ways. The first way is by the iterative nature of the model. Small projects ask for one or few iterations, big projects ask for more, the method stays the same in both cases. The second way is by introducing a simplification of the original process model for smaller and more routine projects.

The second sub problem is to maximize the quality and completeness of the requirements.
The completeness of the requirements is realized through frequent feedback to stakeholders and spending relatively much time on the Blastoff and Brainstorm phases. The quality is maximized by the structured way of working presented by the method. Not only is shown what steps need to be taken, but also how they should performed.

The third sub problem, the most important sub problem, is about finding a balance between flexibility and control. The iterative way of working involving short cycles and frequent feedback moments allow the project team to identify changing requirements in an early stage and process them accordingly. This way flexibility is created. On the other hand, the decision points accompanied by measurable parameters enables management to measure the team's performance and stay in control over the project.

12. References

[1] Cheng, B.H.C., Atlee, J.M., *"Research Directions in Requirements Engineering"*, Future of Software Engineering 2007, pp. 285-303, 2007

[2] Seligmann, P.S., Wijers, G.M., Sol, H.G., *" Analyzing the structure of I.S. methodologies an alternative approach"*, Proceedings of the First Dutch Conference on Information Systems, 1989

[3] Paetsch, F., Eberlein, A., Maurer, F., *„Requirements Engineering and Agile Software Development“*, Proceedings of the 12th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, pp. 308-313, 2003

[4] El Emam, K., Madhavji, N.H., *“An Instrument for Measuring the Success of the Requirements Engineering Process in Information Systems Development”*, Empirical Software Engineering, vol. 1, pp. 201-240, Kluwer Academic Publischers, Boston, 1996

[5] Niazi, M., Hickman, C., Ahmad, R., Babar, M.A., *“A Model for Requirements Change Management: Implementation of CMMI Level 2 Specific Practice”* pp. 143-157, Springer-Verlag Berlin Heidelberg, 2008

[6] Frederiks, P.J.M., Van der Weide, Th. P. *“Information modeling: The process and the required competencies of its participants”*, Data & Knowledge Engeneering, vol. 58 pp. 4-20, Elsevier B.V., 2006

[7] Hayes, J.H., Dekhtyar, A., Sundaram, S., K., Howard, S., *“Helping Analysts Trace Requirements: An Objective Look”*, Proceedings of the Requirements Engineering Conference, pp. 249-259, IEEE Computer Society, 2004

[8] Muthu, S., Whitman, L., Cheraghi, S. H., *“Business Process Reengineering: A Consolidated Methodology”*, Proceedings of the 4th Annual International Conference on Industrial Engineering Theory, pp. 17-20, 1999

[9] Dyba, T., *“An Instrument for Measuring the Key Factors of Success in Software Process Improvement”*, Empirical Software Engineering, vol. 5, pp. 357-390, Kluwer Academic Publishers, Boston, 2000

[10] Nuseibeh, R., Easterbrook, S., *"Requirements Engineering: A Roadmap"*, The future of software Engineering, ACM & IEEE Computer Society Press, 2000

[11] Van der Aalst, W. M. P., Song, M., *“Mining Social Networks: Uncovering Interaction Patterns in Business Processes"* International Conference on Business Process Management, vol. 3080, pp. 244-260, Springer-Verlag, Berlijn, 2004

[12] Beck, K., Beedle, M., Bennekum, van, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmit, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, J., Sutherland J., Thomas, D. *“Manifesto for Agile Software Development”*, http://www.agilemanifesto.org, 2001

[13] The open group, "The Open Group Architecture Framework Book 9", The open group, 2009 ISBN 97-90-8753-230-7

[14] The Standish Group, *"CHAOS Report"*, 1995

[15] Robertson, S., Robertson, J., *"Mastering the Requirement Process"*, Second Edition, Addison-Wesley, 2006 ISBN: 9780321419491

[16] Pressman, R.S., *"Software Engineering: A Practitioner's Approach"*, Sixth Edition, McGraw-Hill Higher Education, 2005 ISBN: 0-07-285318-2

[17] Kulak, D., Guiney, E., *"Use Cases"*, Secon Edition, Pearson Education, 2003 ISBN: 978-90-430-128-3

[18] Janssen, P., *"PRINCE2 Compact"*, Pearson Education, 2nd edition, 2009, ISBN 978-90-430-1284-3

[19] Dekker, E., Collaris, R.A., *"Rub op Maat"*, 2nd edition, 2009, ISBN 978-90-12-12602-1

[20] Cannegieter, J.J., Van Solingen, R., *"De kleine CMMI voor ontwikkeling"*, 3rd edition, Academic Service, 2009, ISBN: 9789012581288

[21] Mintzberg, H., *"Structure in fives: Designing effective organizations"*, Prentice-Hall, 2009, ISBN: 9780138541910

Appendix I

A method for Requirements Development & Management in turbulent environments

Instruction document

13. Introduction

This appendix describes the full method for requirements development and management in turbulent business environments. First the three challenges will be shortly described. These challenges are equal to the three sub problems of the research document. Secondly the different roles will be described together with their tasks and responsibilities. In chapter 4 the process model for projects will be discussed. This process model forms the most extensive version of the method. All the different phases, activities and steps will be explained in terms of what to do and how to do it. In chapter 5 the simplified process model is explained and differences with the full model are highlighted. In the final chapter a brief vocabulary can be found.

14. Aiming points

The method was specifically aimed at three challenges.

14.1. Completeness and correctness of the requirements

An absolute guarantee of completeness and correctness of all the requirements seems a Utopia. Nevertheless the process model is constructed in such way that the requirements can be identified and specified in a careful and controllable way. Bi-directional traceability plays an important role in this matter. Bi-directional traceability holds that requirements are traceable in both horizontal and vertical way. On a horizontal level this means that all the requirements should be explicitly traceable during the whole process. For example in Use Cases and other model and even later on in design and actual code[12]. On a vertical level the distinction is made between Business-, Customer- (also called User) and System requirements. These types of requirements concern respectively the 'why' 'what' and 'how' questions. So business requirements concern why something is needed for the businesses (what business goals are obtained), customer requirements concern what is needed in order to achieve the business requirements and the system requirements concern how it is achieved. Note that system requirements narrowly relate to (functional) design, but they are still requirements. Vertical traceability ensures that all system requirements relate to upper customer requirements and all customer requirements relate to upper business requirements. This way maximization of the completeness is achieved and unnecessary requirements (that can't be linked to upper lying requirements) can be identified and tossed out.

> *We spend a lot of time – the majority of project effort – not implementing or testing, but trying to decide what to build – Brain Lawrence*

The correctness of the requirements in terms of quality and understandability is achieved by uniform formulations and formulation guidelines.

[12] This document focuses only on the traceability during the requirements analysis

14.2. Flexible procession of changing requirements

Changing requirements are facts in turbulent business environments. Flexible procession of changing requirements is realized in two ways. The first way concerns the emphasis at the beginning of the analysis. When performing the analysis in a structured and careful way from start on, a lot of miscommunications, misinterpretations and confusions can be prevented. Inconsistencies and confusion of tongues can be identified in an early stage of the process reducing their impact on the project. The early discovery of these mala fide occurrences is realized by facilitation heavy collaboration and communication in the early stages of the analysis.

The second way concerns the iterative characteristics of the process model. By specifying the requirements incrementally and validating the particular requirements after an increment has been completed the impact of changing requirements can be reduced. For example if we have three business requirements and after specifying the first requirements into customer and system requirements new insights are gained which rule out the second business requirement. Hardly any effort has been put into specifying the second requirements so far, so minimal time is lost as a result. If the outcome of the validation process shows that the requirements are misinterpreted by the requirements engineer, these insights only affect the particular increment and since we chose to keep the increments as small as possible, minimal time is lost.

Specifying requirements incrementally in an iterative way will only be successful if decomposed and prioritized properly. More about these steps in chapter 4.

14.3. Applicability to large and small projects

Not all projects have the same size. It can't be the intention to blow up relatively small project by excessive requirements analysis. Sometimes projects that start relatively small grown bigger into full scale project. Often this is signaled when discovering the requirements. Therefore the first step of the process model is the intake of the problem. Depending on this intake step the problem is classified into either *Job*, *Change*, or *Project*[13]. A job can be executed without any structured requirements analysis, a project requires the full process model. When dealing with a change application of the simplified version of the full model will be sufficient. If the characteristics of the problem changes while performing the requirements analysis the project manager might consider to change tactics.

The iterative way of working also enlarges the applicability to different projects. Big projects can have multiple iterations while smaller projects (that are still considered as projects) can have one or few iterations. This way the emphasis in the beginning of the process is maintained.

[13] More information about the classification can be found chapter 4

14.4. Overall challenges: controllability and quality

Overall the method also focuses on the quality and controllability of the process. On a quality level the goal of requirements engineering processes is bilateral. On the one hand the goal is to achieve a complete and correct requirements specification of what intended system is supposed to do, while on the other hand the goals is make the system designers and developers understand what the business needs. The latter is the actual goal and the first can be seen as a mean to achieve it.

Making designers and developers understand what the business need is all about creating a uniform perception of what the intended system is supposed to do. The method supports this process by prescribing workshops and frequent feedback sessions due to short cycles. This leads to intensive and frequent collaboration and keeps the project alive in the minds of the people involved. This way any flaws and misconceptions can be identified more easily and dealt with.

Controllability forms the counterpart of flexibility. Flexibility is wanted as long as the project management can stay in control over the project. This means controlling time, money and quality. The insurance of the controllability is achieved by certain decision points. These decision points are linked to the three phases of the project.[14] At each of these moment the project management decides whether or not to continue the project. Due to the incremental delivery of specified requirements the functionality can be incrementally build as well. When a project is discontinued the results achieved so far can still be useful. The abortion of a project always feels as a failure, but it might just be the right decision at that time. Better to turn back halfway than to get lost altogether.

> *I love deadlines. I like the whooshing sound they make as they fly by.*
>
> *– Douglas Adams*

[14] The three phases are the *Intake*, *Startup* and *Initiation* phase and are based on PRINCE2

15. Roles

In this chapter the main roles involved during the requirements analysis are described. When looking at requirements the requirements engineer and the requirements manager have a central role. Therefore the tasks and responsibilities of these roles are described as well.

15.1. Stakeholder

A stakeholder is the person or agency who has a stake in the project. It are the combined whishes, needs and restrictions of the stakeholders that lead to the requirements of the proposed system. The project's sponsor as specific stakeholder takes the go / no go decisions based on the information and the advice given by the project manager and the requirements manager. Below a list of most common stakeholders.

- Project's sponsor
- Customer
- End user
- System developer
- Information analyst
- System architect
- Tester
- Law

15.2. Project Manager (PM)

The project manager is responsible for the course of the project. This responsibility is larger than just the requirements analysis phase. With the focus on the requirements the project manager is responsible for controlling time, money and quality. The project manager has the responsibility over the entire requirements analysis and takes the final decision in how the project team should handle requirements. De requirements managers plays a supporting role in this matter.

15.3. Requirements Manager (RM)

The requirements manager's main responsibility is the preparation and facilitation of all activities related to discovering, formulating and maintaining requirements in the requirements specification. The requirements manager supports the project manager in making decisions concerning requirements issues. This also applies to dealing with changing requirements[15].

[15] Outside the scope of this document the requirements manager is also responsible for the requirements' lifecycle outside the project. For example for reuse purposes in future projects.

15.3.1. Tasks and Responsibilities

- Determines the specific requirements processes. These processes need to fit into the project management framework applied to the particular project and need to be written in the Requirements Management Plan[16].
- Keeps an eye on changes in the environment that might influence the project's requirements
- Solves all conflicts and inconsistencies together with the stakeholders
- Checks if all activities are conform the Requirements Management Plan.
- Manages the requirements and changes to the requirements
- Checks and is responsible for the quality of the requirements
- Checks and is responsible for maintaining bi-directional traceability of the requirements
- Advises the project manager about dealing with changing requirements
- Reports to the project manager about the progressions and quality of the requirements.

15.3.2. Combination with different roles

Combination with project manager

Advantage: Involved with stakeholder management and affinity with requirements management

Disadvantage: Focus more on time management and budget instead of focusing on requirements

Combination with Business analyst / Information analyst

Advantage: Focus on requirements and knowledge of the domain and environment

Disadvantage: Business - and information analyst are not always manager types

15.4. Requirements Engineer (RE)

The main responsibility of the requirements engineer is to elicit, analyze and specify the business-, customer- and system requirements. The requirements engineer signals inconsistencies in the collection of requirements and brings structure this collection[17]. Often the requirements engineer will lead the workshops and conduct interviews with stakeholders.

15.4.1. Tasks and responsibilities

- Elicit, analyze and specify all level requirements
- Signal conflicts and inconsistencies among the requirements
- Analyze the impact of changing requirements

[16] See chapter 4

[17] for the improvement of explicit traceability

- Maintain bi-directional traceability
- Report to the requirements manager
- Determines the acceptance criteria of the requirements
- Facilitates the feedback to the stakeholders
- Supports the requirements manager

15.4.2. Combination with different roles

Combination with requirements manager

Advantage: Heavy focus on requirements and has an interest in qualitative strong requirements

Disadvantage: Has to focus on the details while keeping an overview on project level

Combination with senior user

Advantage: Focus on the needs of the end users and expert knowledge about the domain

Disadvantage: Thinking more in terms of solutions than in terms of requirements.

15.5. Information analyst (IA)

The information analyst is responsible for translating the requirements into system design. The goal of the whole requirements analysis is to make the information analyst understand what the stakeholders want. Therefore the information analyst needs to be involved in the project from an early stage on. Preferably the information analyst simulating the requirements and creating the design is the same person.

15.5.1. Tasks and responsibilities

- Understanding the requirements
- Simulating the requirements for validation purposes
- Translating the requirements into design
- Validation of the design (after the requirements analysis)

People forget how fast you did a job, but they always remember how well you did it.

– Howard Newton

15.6. Combination matrix

In the matrixes below the possible role combinations are displayed. An 'A' stands for advisable and the 'P' stands for possible. Combining roles will lead to a reduction of the people involved in a project which might benefit smaller projects.

15.6.1. Combination matrix Project

	PM	**RM**	**RE**	**User**[18]	**Information Analyst**
PM		P			
RM	P		P		
RE		P			P
User					P
Information Analyst			P	P	

In projects it seems reasonable that the project management role and the requirements management role can be combined in order to address all management tasks to one person. The requirements management role can also be combined with the requirements engineer role in order to address all responsibilities concerning requirements to one person. Combining roles also knows its disadvantages and is therefore not advised in projects. When all the roles are addressed to different persons the mutual control enhances the quality of the requirements.

15.6.2. Combination matrix Change

	PM	**RM**	**RE**	**User**[19]	**Information Analyst**
PM		A	P		P
RM	A		A		P
RE	P	A			A

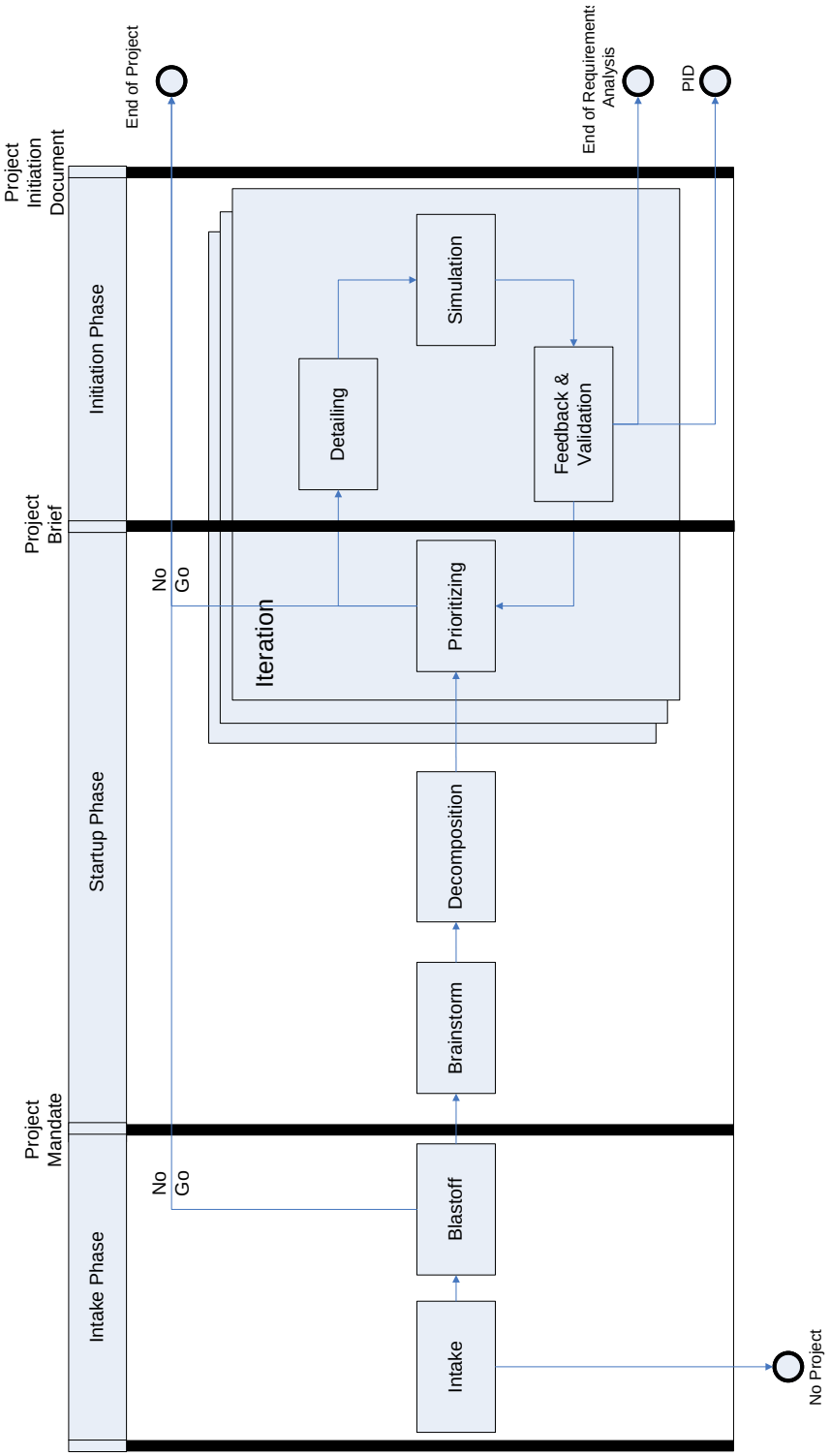
[18] (Senior) end user

[19] (Senior) end user

User			P		
Information Analyst	P	P	A		

In change assignments the environment is more familiar and the complexity is much lower than in projects. This leads to more space for combining different roles and some combinations are even advised to reduce the number of needed people to reduce the deployment of human capital. Combining roles might still lead to wrong implicit assumptions leading to flaws in the requirements specification.

Process model requirements development and management in Projects



16. Process Model Project

> *If you cannot describe what you are doing as a process, you don't know what you're doing.*
>
> *– W.E. Deming*

On the last page the process model of requirements development and management during projects is presented. The different steps (blocks) are linked to the different phases. In this chapter all the steps and their deliverables will be discussed. Each phase will be ended with a go / no go decision support by three documents that form the connection between requirement engineering and project management.

16.1. Intake (Intake phase)

The intake forms the first contact between the requirements engineer and the stakeholders. Based on this contact has to be decided what the motivation and possible size of the project is. Five parameters need to be valued in order to determine whether the project team will deal with a job, change or project. The following decision three can be used.

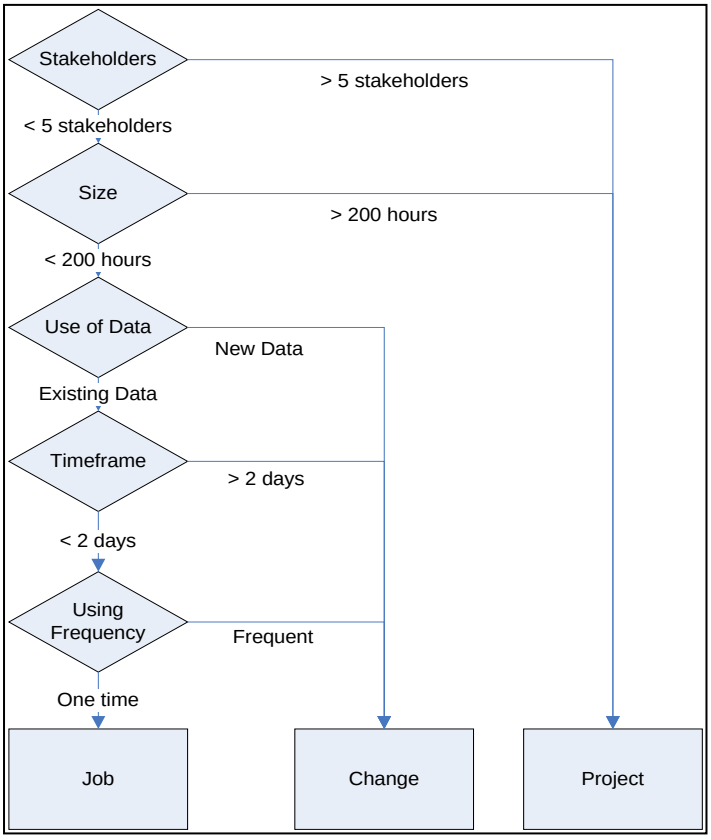
Stakeholders
When more than five different stakeholders are identified the chance of encountering inconsistencies is large. Chances are that changing requirements will occur and therefore the project will be classified as project.

Size
Size in terms of an estimation of the amount of hours needed to solve the problem

Use of Data
If the problem can be solved with existing data the problem is less complex than if new data needs to be gathered.

Time Frame
Defines the time span of the project

Using frequency
The frequency by which the solution will be used



16.2. Blastoff (Intake phase)

> *If there is no struggle, there is no progress.*
>
> *- Frederick Douglass*

The Blastoff can be seen as the start of the requirements analyses. The Blastoff consists of one or more workshops in which the project team and the stakeholders are represented. At the end of the Blastoff some subjects need to be described and delivered as input to the Project Mandate.

16.2.1. Activities

1. Organize one or more workshops with the following parties present
    - Project Manager
    - Requirements Manager
    - Requirements Engineer
    - Stakeholders
    - Information Analyst
2. During the workshop(s) construct the deliverables of the Blastoff

16.2.2. Deliverables

- Description of the project goal
- Inventory of the stakeholders
- Requirements Management Plan
- Visualization of the problem space
- Estimation of the hours needed to perform the Startup phase
- Context model
- Vocabulary
- Overview of all relevant assumptions, constraints and dependencies

16.2.3. Description of the project goal

The project goal needs to be discussed and described to make sure that all parties involved know why the project is executed and how they benefit from the results. A clear description of the project goal can often help to identify the first business requirements. The project goal needs to be described in the project mandate.

16.2.4. Inventory of the stakeholders

To maximize the completeness of the requirements all the stakeholders need to be identified and listed in an inventory before moving on to the requirements. In the first workshop the team should think about which (potential) stakeholders are not represented yet. A context model can help to identify missing stakeholders. The inventory of stakeholders should be added to the project mandate.

16.2.5. Requirements Management Plan

The requirements management plan describes how the project team will deal with requirements. The techniques used to elicit and formulate the requirements must be described as well as appointments regarding how the project team will deal with changing requirements. The quality criteria and the traceability should be described as well. The project manager and requirements

manager can choose which techniques, models and activities they will use during the requirements analysis. They are even allowed to diverge from the original model if they think it's appropriate for the particular project. All of this needs to be described in the requirements management plan and every party involved needs to commit to this plan. The requirements management plan should contain the following subjects.

Requirements development

- How will the business requirements be gathered and formulated?
- How will the customer requirements be gathered and formulated?
- How will the system requirements be gathered and formulated?
- How will the non-functional requirements be gathered and formulated?
- How will the vertical traceability be maintained?
- How will the requirements be validated?

Requirements management

- What method will the project team use? (which steps will be taken, which won't and why?)
- How will the project team deal with the decision points?
- How will the project team deal with changing requirements?
- How will the horizontal traceability be maintained?

The requirements management plan needs to be added to the project mandate.

16.2.6. Visualization of the problem space

A sketch, model or other visualization of the problem space and the suggested main functionality provides an overview for the project team and the stakeholders. It's relatively easy to make one and it helps clarify the situation and environment to everyone involved. The visualization does not have to be added to the project mandate, if created it can be added to the project's documentation.

16.2.7. Estimation of the hours needed to perform the Startup phase

En estimation of the hours needed to perform the Startup phase is necessary for the go / no go decision at the end of the Intake phase. The estimation needs to be added to the project mandate.

About the time we can make the ends meet, somebody moves the ends.

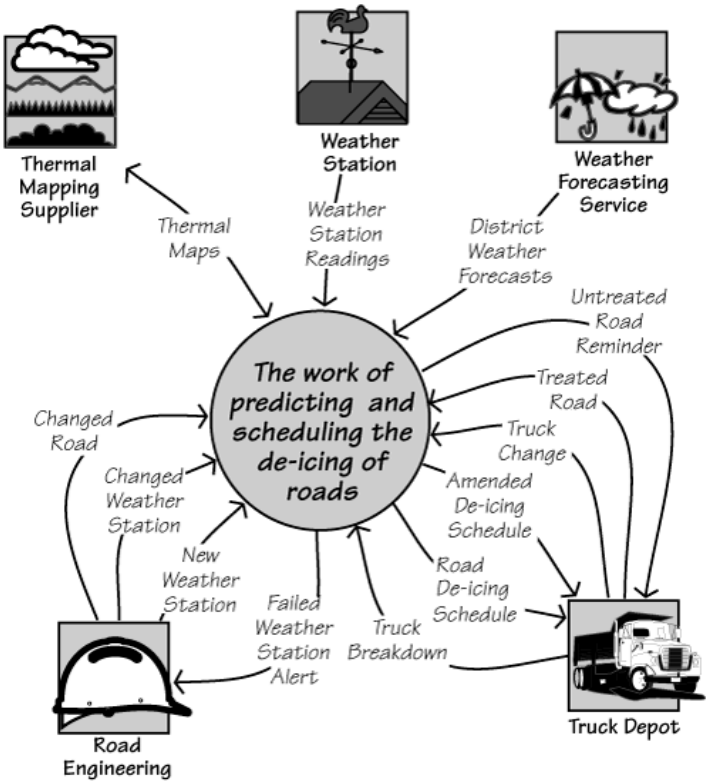
- Herbert Hoover

16.2.8. Context model

An easy pitfall in requirements engineering is thinking in terms of solutions instead of requirements. When performing a brainstorm session it is very important to have a clear view on the context of the proposed system. A context model can help. A context model describes the *work* that the proposed system is to do and the interaction with other systems. Using a context model prevents the team from thinking in terms of solutions (technical level) and helps the team to focus on the work (process level). Below you can find an example of the context model for "The work of predicting and scheduling the de-icing of roads."

The context model is not only applicable during the Blastoff step. It might help by identifying the first business requirements and be of use while determining the decomposition of the requirements set. When dealing with Changes a context model isn't mandatory, however if there's interaction with more than three other systems it is strongly advised.

The context model does not have to be added to the project mandate, but it must be added to the project documentation.



Source: Robertson & Robertson 2006

16.2.9. Vocabulary

Terminology is an obstacle in many projects. Especially during requirements analysis where people with different disciplines van to communicate and work together. Jargon or multi-interpretable concepts can be easily added to a vocabulary and provide clarity to everyone.

16.2.10. Go / No Go decision

At the end of the Intake phase the go / no go decision has to be taken whether to execute the Startup phase of not. The decision is taken by the project's sponsor based on the project mandate with the help of the project manager.

16.3. Brainstorm (Startup phase)

The scope and context is clear due to the execution of the Intake phase and now the team can focus on the requirements. In the Brainstorm step the first high level business requirements will be identified. A good way is to organize a workshop with all stakeholders. To discover as much requirements as possible all requirements are acceptable at this point. The quantity of the requirements will lead to quality later on. The requirements engineer is responsible for maintaining the scope and makes sure that the stakeholders stay on the right track.

16.3.1. Activities

- Organize a workshop with the following parties involved
    - Project Manager
    - Requirements Manager
    - Requirements Engineer
    - Stakeholders
    - Information Analyst
    - IT Architect (optional)
- Create a mindmap or other model of the proposed functionality
- List the actors and interactions they will have with the proposed system
- List the discovered functional business- and customer requirements discovered
- List the discovered non-functional requirements
- Ensure vertical traceability
- Create a Use Case Survey with a global Use Case description of each interaction
- Connect the customer requirements to the Use Cases (horizontal traceability)
- Validate all deliverables with the stakeholders

16.3.2. Deliverables

- Mindmap
- List of actors and interactions
- List of business- and customer requirements
- List of non-functional requirements
- Use Case Survey
- Global Use Cases

If you want to be incrementally better: Be competitive. If you want to be exponentially better: Be cooperative.

– Unknown Author

16.3.3. Mindmaps

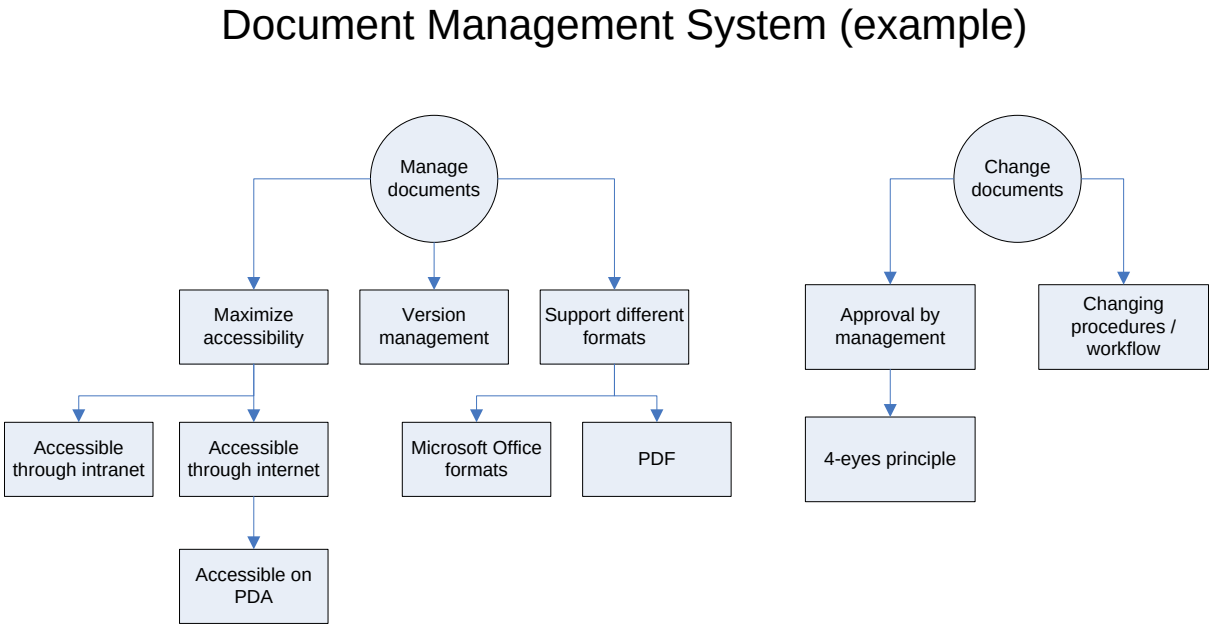
Mindmaps are graphical representations of the associations stakeholders have with the work. Unlike a context model which is aimed at surrounding systems, a mindmap represents how the stakeholders see the problem. Mindmaps provide great input for discovering requirements. They can be constructed in four steps. The first step is to choose the starting point, the central subject. The second step is to write down all associations. The third step is to detail the association in more depth. The fourth step is to cluster the associations. Below you can find an example of a mindmap concerning "Lead Generation".

Source: http://www.startwithalead.com

> *Not everything that can be counted counts, and not everything that counts can be counted.*
>
> *– Albert Einstein*

16.3.4. Product breakdown

Another possibility is to brainstorm with the main functionality as a starting point. One or a few main functionalities will be identified and all participants add something to them on turn. The end result may look as the figure beneath.



The product breakdown technique is highly efficient to find functional customer requirements. Of course all the above lying business requirements must be clear. If a customer requirements can't be traced back to a business requirement then either the customer requirement isn't a valid one or not all business requirements have been identified yet.

Note that although a product breakdown leads to specific business- and customer requirements, it will often narrow down the frame of mind. Be careful not to start thinking in solutions, but stick to the requirements

16.3.5. List the interaction with the proposed system

To prevent getting lost in a flood of requirements it is advisable to make an inventory of all the interaction between the actors and the proposed system. Actors can be people as well as systems. Possible interaction as for example "User places order", "Credit Card System validates transaction details" or "System sends order to warehouse". These interaction can be processed into a Use Case Survey (see below). Structuring these interactions can be helpful when dealing with decomposition.

16.3.6. List the business requirements and customer requirements

List all the discovered business- and customer requirements. Keep in mind that this is still a brainstorm phase, so keep the customer requirements on a global level and focus mainly on the business requirements. Mind the vertical traceability and connect the identified customer requirements to the listed interactions (which will develop into Use Cases later on).

At the end of the brainstorm session all the requirements need to be fitted within the scope of the project. Requirements that won't fit must be marked as 'Won't haves' (see paragraph 4.4). It is important not to pay only attention to the functional requirements, but to the non-functional requirements as well. Especially the IT architect can play an important role in this part., because non-functional requirements are often about quality and performance constraints on a technical level.

Non-functional requirements can be about:

- User interface
- Usability
- Performance
- Reliability
- Operational constraints
- Technical constraints
- Maintenance and service
- Security
- Privacy and compliance regulations
- Culture and politics
- Law and regulations

16.3.7. Use Case Survey

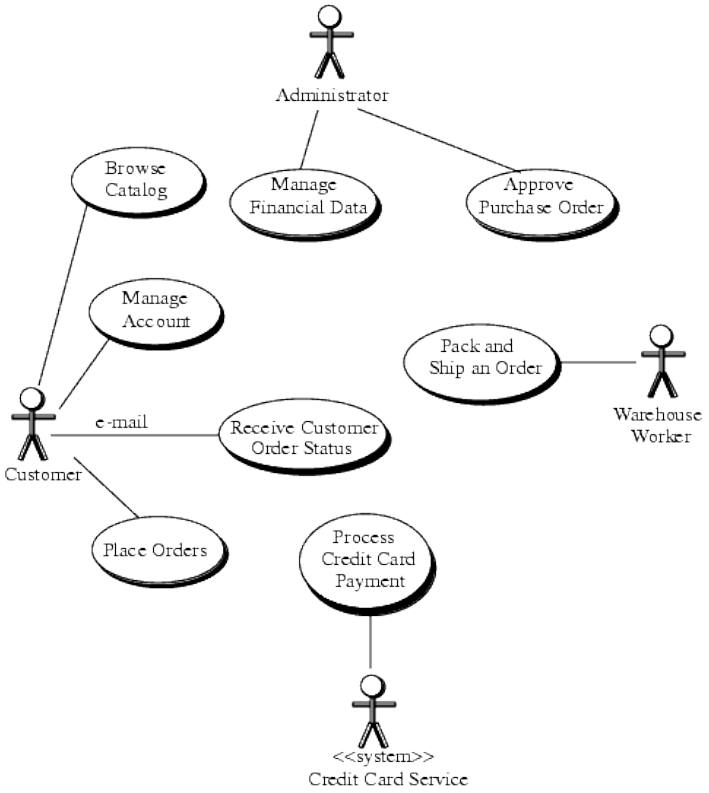
Use Cases are about specifying the interactions between the system and the actors. A Use Case Survey provides a global overview of all Use Cases and by using a Use Case Survey it is easy to see what interactions are present in the proposed system without having to specify each Use Case into detail. An example of a global Use Case to be included in the Use Case Survey can be found below.

Item	**Example (Coffee machine)**
Use Case Number	1
Use Case Name	Select beverage
Initiator (actor)	User
Description	The user enters his choice on the machine
Techinal priority	None

Business priority	Must have
Dependency	Dependant of Use Case #X
Source (Business requirement)	Business Requirement #1
Remarks	

16.3.8. Use Case Diagram (Mandatory when actors > 1 or Use Cases > 3)

In order to see the interrelationships between the different Use Cases a Use Case Diagram can be made. A Use Case Diagram forms the schematical representation of all the interactions. In the example the Use Case Diagram of an online store is presented.



Source: http://java.sun.com

> *Pay attention to what users do, not what they say.*
>
> *– Jakob Nielson*

16.4. Decomposition (Startup phase)

> *Make it as simple as possible. But no simlper.*
>
> *– Albert Einstein*

Complexity often forms a major obstacle in larger projects. When looking at requirements it is the responsibility of the requirements manager to create order in this complexity. This can be realized by decomposition. Decompositions means that a large complex problem will be divided into smaller simpler pieces. A common way of decomposition is functional decomposition. Functional decomposition means that the functionality (functional requirements) of the proposed system will be divided into smaller parts. The structure of this decomposition is comparable with the structure of a product breakdown.

Another possibility for decomposition is decompose the problem based on the interactions between actors and the system. These interactions can be translated to customer requirements (what the system should do). Use Cases are suitable for formulating the interactions.

Both forms of decomposition can be applied by the requirements manager. Different project can require different ways of decomposition. By choosing the type of decomposition it is important to make each piece 'stand alone'. This means that all the pieces are independent of each other. By doing so the iterative processing of the different pieces create *Islands of Stability*. An Island of Stability can be seen as a project milestone that won't be lost in case the project will be terminated. In other words, the functionality created so far can be used by the users, even if the total functionality won't be delivered. In some cases dependency between pieces is inevitable. In these cases correct prioritization is required, making that the conditional pieces will be processed first. Finally when decomposing the requirements set the aim should be to make the pieces as small as possible, while maintaining the stand alone property.

16.5. Prioritizing (Startup phase)[20]

In order to determine which Use Cases and corresponding requirements will be taken into an iteration cycle, all Use Cases should be prioritized.

16.5.1. Activities

- Prioritize all the Use Cases according to the MoSCoW technique
- Select the Use Cases that will be taken into the next iteration cycle
- Estimate the total amount of hours needed to perform the initiation phase

16.5.2. Deliverables

- List of the prioritized Use Cases
- List of the selected Use Cases
- Estimation of the hours needed to perform the initiation phase

16.5.3. MoSCoW

MoSCoW stands for Must, Should, Could and Won't have

- Must have
  These Use Cases must be selected for the next iteration, because they are either the most important Use Case in terms of business benefit or they are essential for other Use Cases.
- Should have
  These Use Cases are essential to the project, but not directly for the upcoming iteration.
- Could have
  These Use Cases are not essential to the project, but are nice to have. Important is that the realization of these Use Cases will never interfere with the realization of the must and should haves.
- Won't have / Would have
  These Use Cases do not have enough priority to be reconsidered in this project. However they might be interesting for future project or new releases. Therefore the W often stands for Would have as well.

16.5.4. The selection

The selection of the Use Cases will be based on the prioritization. Important aim is to make the selection stand alone in such way the result of the iteration can lead to working functionality, independent of future work.

[20] From this point on is assumed decomposition is applied by the different Use Cases

16.5.5. Go / No Go decision

At the end of the Startup phase has the decided whether or not to execute the Initiation phase (at least one iteration). The project brief is the document related to this decision. The project brief will contain the Use Cases and hours estimation for the upcoming iteration[21]. The project brief is only required before the first iteration. In following iteration this decision can be taken by the project management and the project sponsors.

16.6. Detailing (Initiation phase)

The selected Use Cases need to detailed into full Use Case descriptions and the corresponding cusomter- and system requirements.

> *Use cases are simply an aid to defining what exists outside the system (actors) and what should be performed by the system – Ivar Jacobson*

16.6.1. Activities

- Organize a workshop session with the following parties involved
    - Requirements Manager
    - Requirements Engineer
    - Relevant Stakeholders
    - Information Analyst
    - IT specialist

      The attendance of the IT specialist (the one who is actually going to simulate the requirements) leads to a better understanding of the requirements which is much more effective than just provide the requirements specification and written Use Cases. Preferably this person is the same person who will design the actual system after the requirements analysis.
- Fully specify the Use Cases
- Create a textual or visual scenario (optional)
- Formulate all customer- and system requirements (including non-functional requirements)
- Create a traceability matrix

16.6.2. Deliverables

- Fully specified Use Cases possibly provided with a scenario
- Full set of requirements corresponding to the selected Use Cases
- Traceability Matrix

16.6.3. Use Cases

Before the customer- and system requirements will be specified the Use Cases from the Use Case Survey have to be detailed. The following level of detail can be used.

Item	**Example (Coffee machine)**
Use Case Number	#1
Use Case Name	Select Beverage

[21] Aside from other project management data which won't be considered in this document

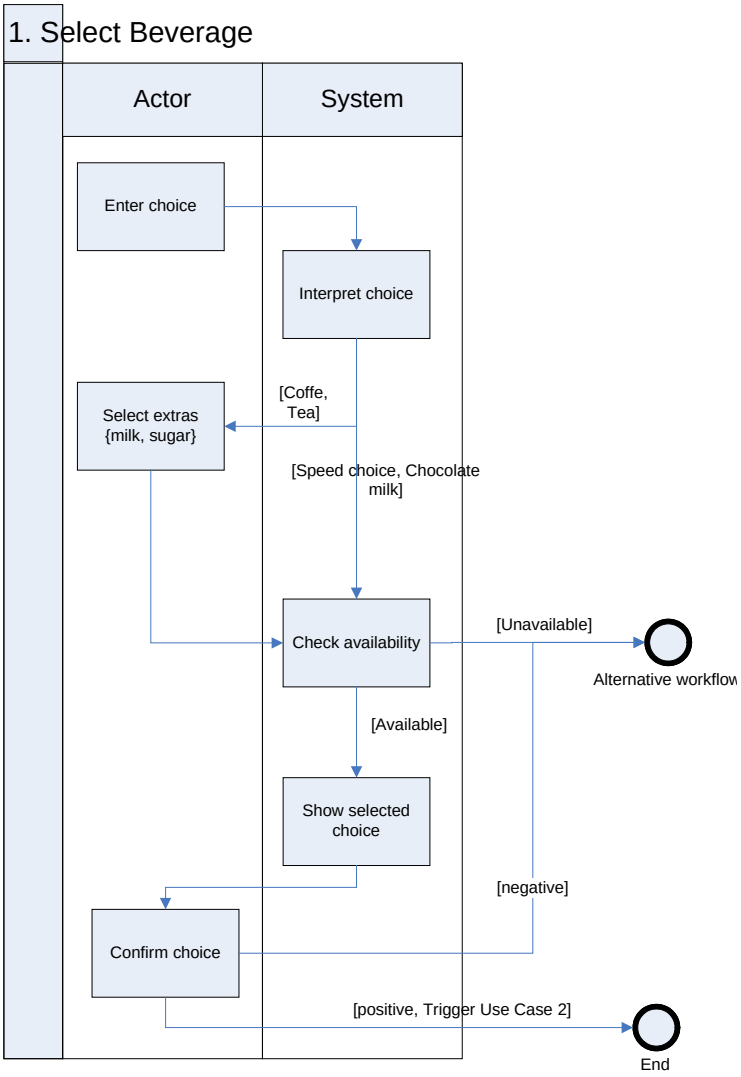
Description	The user enters his choice on the machine
Actors	Users
Triggers	The user wants a beverage from the machine
Preconditions	-
Course of event	1. The user selects his option in the menu 2. The user selects extras a. The user selects the type of extras b. The user selects the amount of extras 3. The user confirms choice
Exceptional Cases	Product out of stock
Post conditions	The user has selected the wanted beverage including extras
Source (Business Requirement)	[Reference for vertical traceability]
Covered by (Customer Requirements)	[Reference for horizontal traceabilty]

16.6.4. Scenarios

By using scenarios the Use Cases can be placed into context. In a scenario the course of the event of a Use Case is described in more detail. In other words, a scenario describes the story of a Use Case. Scenarios can describe alternative paths, exceptions and conditionality (if-else-then). Scenarios can be described in textual or graphical way (see examples).

Business Event Name	User selects beverage
Business Use Case Name and Number	#1. Select Beverage
Trigger	The user wants a beverage from the machine
Precondition	-
Actor	The user
Description of normal workflow	1. The user selects choice 2. If choice is 'coffee' or 'thee' then go to step 3 else if choice is 'Chocolate milk' then go to step 4 3. The user selects extras a. The user selects milk or sugar b. The user selects the amount of extras 4. The system shows the selected product 5. The user confirms the product
Description alternative workflow	1. The user selects speed choice 2. The system shows the selected product

	3. The user confirms the product
Exceptional cases	1. Product out of stock 1.1 The system notifies the user 1.2a The user makes new selection (restart Use Case) 1.2b The user stops selection (terminate Use Case).
Proper outcome	1. The user selected and confirmed his choice 2. Use Case #2 is triggered



Example of textual and graphical form

Creating scenarios is not mandatory, but it's advised when dealing with Use Cases that contain alternative paths or exceptional cases. Writing scenario's helps with the discovery of the system requirements and is suitable for communication with stakeholders. The requirements manager decides whether or not to make scenario for a particular Use Case.

16.6.5. Formulating requirements

To formulate requirements in a correct and unambiguous way a few guidelines can be used. Below a checklist can be found regarding the quality of requirements specification.

Requirements QA & Verification Checklist

Document name: ..... **Document Id.:** .....

Checked by: ..... **Date:** ...../...../.....

Check (+ explanation)	Ok	nOk	N/a
Generic checks for individual requirements			
1. The requirement is unambiguous (not multi-interpretable).			
2. The requirement is as small as possible so it cannot be divided.			
3. The requirement is verifiable. (you must be able to test whether the final solution meets the requirements).			
4. The requirement is consistent (does not argue against itself, e.g. by containing different statements).			
5. The requirement is specific (it uses specific, measurable values).			
6. The requirement is relevant (within the scope and adds value to the defined goals).			
7. The requirement describes not a solution (only a need; what is wanted). Note: Business requirement describes a 'Why' of a wanted change. User requirement describes a 'What' of a wanted change.			
Sentence construction			
8. The requirement must be brief (short and to the point).			
9. The requirement is written in an active sentence, according this syntax: [Actor] + [Verb] + [rest of description]. For instance: "The content manager can update the homepage text and pictures". Note: This syntax is not applicable onto quality			

Check (+ explanation)	Ok	nOk	N/a
requirements and constraints.			
10. The requirement is formulated positively (avoid "not" or denial/ negative constructions).			
11. In case the requirement contains a number, it must also contain the measurement units; Kilo, Minutes, m3, Euro, etc.).			
Attributes			
12. The requirement is fully described			
13. The attributes are filled correctly (They must have a value which is correct and within the value range of that attribute).			
14. The requirement is traceable			
Terminology			
15. The requirement uses known and defined terms.			
16. The requirement does not use words making it ambiguous or vague			

16.6.6. SMART requirements

Another guideline for formulating requirements is the SMART guideline. SMART stands for:

- Specific — Formulate requirements as specific as possible
- Measurable — Requirements need to be measurable
- Actionable — Requirements must be executable
- Realistic — Requirements must be realistic and feasible
- Time-bound — It must be clear when requirements are fulfilled

16.6.7. Snowcards and templates

There are many different templates than can be used to formulate requirements. In these template attributes like the priority, mandatory and traceability can be linked to the requirements. The bi-directional traceability is most important attribute and is mandatory for all requirements (if specified in a template). Concerning the horizontal traceability between customer requirements and the Use Cases multiple customer requirements can refer to multiple Use Cases.

Snowcards provide an overview of all attributes of a particular requirement in one view. Snowcards can be easily generated from the records in the template and promote the readability of the requirements. There are also tools available for writing, storing and managing requirements.

16.6.8. Traceability Matrix

The traceability matrix contains all bi-directional traces between requirements, Use Cases and other documents[22]. An example of a requirements traceability matrix can be found below.

Requirements Traceability Matrix

Project: *Example*

Requirements					Production Acceptance Test	
Req.ID	Short Description	Use Case	Design Documents	Build Components	Test plan	Test Case
	optional / mandatory	M	M	O	O	O
BR.1	Business Requirement One					
BR.2	Business Requirement Two					
CR.1	Customer Requirement One					
CR.2	Customer Requirement Two					
CR.3	Customer Requirement Three					
SR.1	System Requirement One					
SR.2	System Requirement Two					
SR.2.1	Detail of System Requirement Two					
END						

Requirements Traceability Matrix

16.7. Simulation (Initiation phase)

The system requirements defined in the detailing step of the current iteration can be simulated into a mock-up or a prototype. Because of the iteration cycles this simulation will incrementally build. Each iteration delivers a new increment of the simulation that must tested and validated by the stakeholders. Due to the short cycles the increments are as small as possible, but independent of future work. This way the increments are controllable and traceability can be maintained.

The goal of the simulation is to provide stakeholders (end users in specific) with quick results and let's them get affinity with the system and the functionality. Any flaws, misinterpretations and changing insights will head for the surface and their impact on the project will be minimized. There are two levels of simulation that can be used to validate the requirements with stakeholders.

16.7.1. Mock-up

A mock-up is a visual, non-functional simulation of the proposed system that simulates the functionality but doesn't enable users to actually work with the system. Examples of mock-ups are powerpoint sheets, (interactive) web pages and other interfaces. Creating a mock-up is relatively easy and especially useful when validating requirements concerning user interfaces.

[22] Like prototypes, functional design, technical design, test plans and end code

16.7.2. Prototype

A prototype in this context can be seen as a workable simulation of the functional requirements. A prototype is not excessively tested and probably won't meet all quality related constraints and non-functional requirements, but it does enable the stakeholders and end users to work (partially) with the system. The advantage of using working prototypes is that stakeholders can evaluate and validate the requirements better than by using a mock-up. The disadvantage is that the creation of a prototypes ask for more resources. It is reasonable that if the stakeholders are fully satisfied with the prototype, the prototype will be of great input for the design and construction of the actual system.

16.8. Feedback and Validation (Initiation phase)

In this final step the results of the simulation will be presented to the stakeholders. The stakeholders can try the simulated functionality (in case of a prototype) and their experiences and evaluation can be used as input for following iterations. Changing requirements must be listed and priorities must be reconsidered[23]. Afterwards the circle completes and a new iteration can be started.

16.8.1. Activities

- Validate the requirements simulation with the stakeholders
  - List all changing requirements and new insights
  - If the results is fully satisfying to the stakeholders a PID can be constructed and the particular increment can be designed and constructed.
- Decide whether or not to start a new iteration

Validation of the simulation is preferable done in a plenary session with all stakeholders. PID stands for *Project Initiation Document* and can be the result of the Initiation phase. The PID forms the decision document for the design, construction, testing and implementation of the increment specified during the Initiation phase.

> *Questions about whether design is necessary or affordable are quite beside the point: design is inevitable. The alternative to good design is bad design, not no design at all – Douglas Martin*

The project manager decides together with the project's sponsor if a new iteration will be started or the requirements analysis will end. There three possibilities why no new iteration will be launched and the requirements analysis ends.

1. All the Must, Should and Could haves are processed into requirements an simulated in either a mock-up or a prototype. Condition here is that no changing requirements were discovered during the feedback and validation step. Changing requirements can also be signaled as 'won't have' and be stored for future projects or system releases.
2. The budget limit or project deadline are close by. Due to careful prioritization and achieving results in short iterations, most functionality will be processed in either a mock-up or a

[23] Only when a new iteration is launched, during the Prioritization step

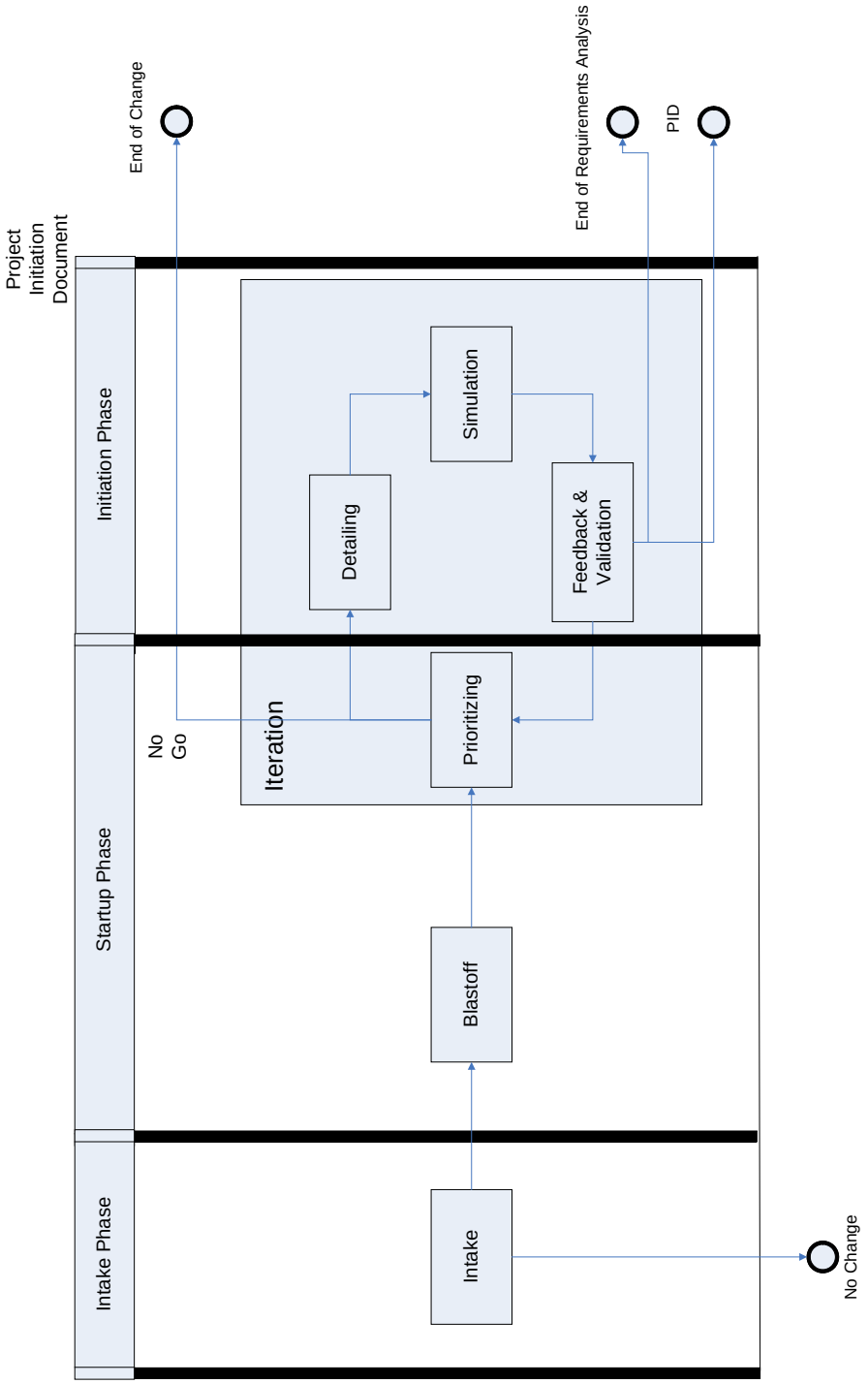
prototype. The requirements that won't be processed further must be marked as 'won't haves' to be used in future project or future system releases when new budget is obtained.

3. The project manager has another reason to end the requirements analysis. Just like with the second option, unprocessed requirements must be marked as 'won't have'.

Note that like stated earlier following iterations do not have to be provided with a Project Brief, the project manager together with the project's sponsor decide whether or not to continue with a new iteration. When the decision is made to process an increment into design and construction a PID is always required.

I have not failed. I've just found 10.000 ways that won't work
– Thomas Edison

Process model requirements development and management in Changes



17. Process Model Change

When dealing with Changes a simplified model is applicable. Unlike when dealing with Projects not all steps of the original model will be taken. Another difference is the absence of the decision documents. The PID at the end of an iteration is still applicable although it is reasonable that other assignment documents are sufficient as well. Nevertheless the Go / No Go decisions are still applicable, just without the support of the decision documents.

17.1. Intake (Intake phase)

The intake step is equal to intake step in Projects. The Go / No Go decision will be taken solely on the information gathered during this step.

17.2. Blastoff (Startup phase)

In the process model Change the Blastoff step is combined with the Brainstorm step and falls within the Startup phase.

17.2.1. Activities

- Organize one plenary workshop with the following parties attending[24]
    - Requirements Manager
    - Requirements Engineer
    - Stakeholders
    - Information Analyst
- Create a list with discovered Business Requirements and interactions
- Create a Use Case Survey containing global Use Cases
    - Process each interaction into a global Use Case
    - Trace the Use Cases to the business requirements
    - With more than one actor or more than three Use Cases create a Use Case Diagram
- Validate the Use Case Survey by the stakeholders
- Create the deliverables below

17.2.2. Deliverables (Mandatory)

- Description of the goal and scope of the Change
- Inventory of the stakeholders
- Requirements Management Plan
- List with business requirements
- Use Case Survey

17.2.3. Deliverables (Optional)

- Use Case Diagram
- Vocabulary
- Other relevant assumptions or constraints (if applicable)

[24] In Changes it is thinkable that certain roles are combined. For more information see chapter 3

17.3. Prioritizing (Startup phase)

Due to the size of Changes explicit decomposition is unnecessary. Nevertheless the Use Cases have to be prioritized and a selection must be made which Use Cases will be processed in the Initiation phase.

17.3.1. Activities

The activities are equal to those in Projects.

17.3.2. Go / No Go decision

The Go / No Go decision is similar to the one in Projects.

17.4. Detailing (Initiation phase)

The detailing step is equal to the detailing step in Projects.

17.5. Simulation (Initiation phase)

The simulation step is the same the simulation step in Projects. Because Changes mostly concern changes on existing systems the creation of prototypes can be easier and will probably require less resources than in Projects.

17.6. Feedback and Validation (Initiation phase)

The feedback and validation step is equal to the feedback and validation step in Projects and the same decisions are made at the end of the step (also the end of the Initiation phase). One important point of attention here is to carefully consider changing requirements. New insights and changing requirements might lead to an expansion of the problem and the intended solution. The new situation should be reconsidered, for example with the help of the decision three from the Intake phase, to see whether the team is still dealing with a Change, or a Project approach may be more appropriate.

> *Progress is made by correcting the mistakes resulting from the making of progress.*
>
> *– Claude Gibb*

18. Vocabulary

Requirement	*Written form of a wish or need of a stakeholder*
Business Requirements	*Requirements concerning the 'why' question. Why is the requirement relevant for the business*
Customer / User Requirements	*Requirements concerning the 'what' question. What is the proposed system to do.*
System Requirements	*Requirements concerning the 'how' questions. How will the customer requirements be fulfilled.*
Stakeholder	*Someone with a stake in the project*
Mindmap	*Schematical representation of entities and addociations*
Workshop *collaborate*	*Communicative work form in which people involved to achieve a common goal*
Template	*A mold (e.g. for documents)*
Use Case *actors*	*A description of the interaction between a system and its*
Use Case Survey	*An overview of all global Use Cases*
Use Case Diagram *relations*	*A schematical representation of all Use Cases and their with actors*
Requirements Engineer *validating*	*The person responsible for eliciting, formulating and requirements*
Requirements Manager	*The person responsible for the requirements' lifecycle*
Requirements Management Plan	*A document containing how requirements will be managed*
Scenario	*An event-based description of a Use Case*
SMART	*Quality framework for formulating (requirements)*
MoSCoW *and*	*A method for prioritization, respectively Must, Should, Could Won't have*
Functional Requirements	*Requirements describing a functional part of the proposed system*
Non-Functional Requirements	*Requirements describing non-functional parts of the proposed system. Often related to required quality and usability*

Snowcard	*Clear view of one requirements and its attributes.*
Mock-up *the*	*A (visual) simulation of the proposed functionality, without functionality truly existing*
Prototype	*A visual simulation of the proposed functionality with the functionality truly existing in a usable way*
Traceability (bi-directional)	*Horizontal and Vertical traceability of requirements to higher level requirements or other project documentation*