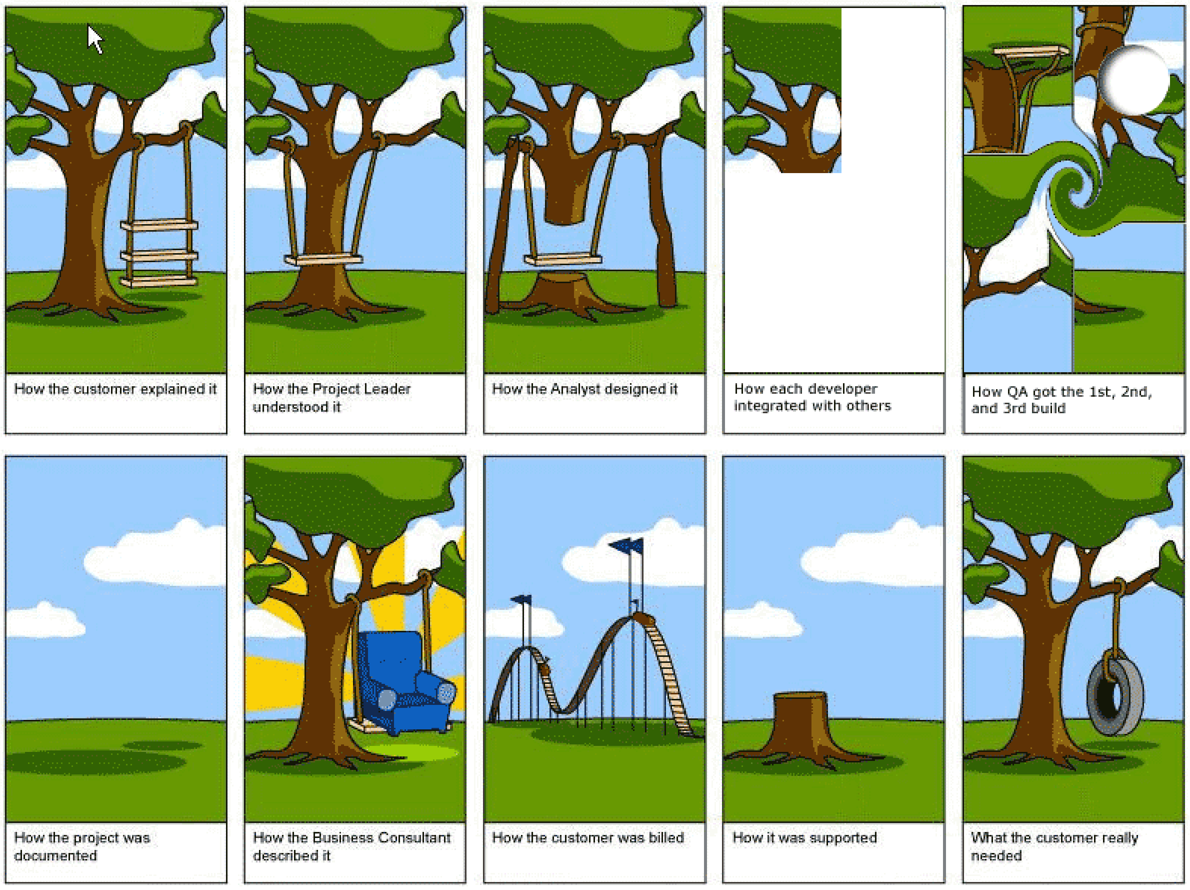
DECISION FRAMEWORK FOR SELECTING A SUITABLE SOFTWARE DEVELOPMENT PROCESS

Delft University of Technology
Faculty of Technology, Policy and Management
Systems Engineering, Policy Analysis and Management (SEPAM)

August, 2009

By Itamar Sharon


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How each developer integrated with others
How QA got the 1st, 2nd, and 3rd build
How the project was documented
How the Business Consultant described it
How the customer was billed
How it was supported
What the customer really needed

DECISION FRAMEWORK FOR SELECTING A SUITABLE SOFTWARE DEVELOPMENT PROCESS

SPM 5910 Master Thesis

Faculty of Technology, Policy and Management
Section: ICT

By Itamar Sharon

Student Number: 1192922

August, 2009

PREFACE

This Thesis report has been written in the period of February – August 2009. In this report I describe the research I have conducted during my internship at Operations and IT Banking ING (OIB ING) which is the final facet of my MSc education in Systems Engineering, Policy Analysis and Management at the faculty of Technology, Policy and Management (TPM), Delft University of Technology. Performing this research and writing this Thesis was, at some periods, a struggle and intellectual challenge. However, most of the times, I felt delighted of the work that I did and the contribution it will finally make.

For this research, I could not have wished for a better environment to perform in. The department in which I was stationed (Solution Delivery Centre Call Face NL) was very comfortable as a research setting. I experienced its employees as flexible, interested and most of all very helpful. The most difficult aspect of this graduation project was to conduct it individually. In the entire MSc education programme, every project consisted of multiple project members. I got quite used to working in a collaborative setting, and I therefore expected some difficulties along the way of this individual project. However, the employees at OIB ING made me lose this feeling. They would gladly make time for me to discuss and reflect on my research and therefore make me feel welcome. I would like to thank most of all Christien Bergman (supervisor and part of the graduation committee). The enthusiasm and help she showed me was the basis for the work I did. Furthermore, I would like to thank Jeroen Hendriks for providing me the opportunity to conduct this research. Finally I would like to thank all the employees at SoDC Call Face NL for making the internship such a pleasant experience.

I am also very grateful to the supervisors at the Delft University of Technology. Michel dos Santos Soares (first supervisor, section Information and Communication Technology) took all the time I needed for discussing problems and asking questions. I could not have wished for a better first supervisor. Christien, Michel, together with Joseph Barjis (second supervisor, section Systems Engineering) and Jan van den Berg (chairperson, section Information and Communication Technology), provided a great graduation committee. For that, I am very grateful.

SUMMARY

Software development processes intend to help project teams in organizing and structuring software projects. These processes are based on three generic models – the sequential model, the evolutionary model and the component-based software model. Software projects can differ in size, complexity and maturity. To cope with these software projects, software development processes are developed since the 1960's. Many different and similar software development processes exist.

Large organizations involved in developing software, struggle with the usage of processes. Even more important, many large organizations struggle in completing software projects on time and within budget. An example of such a large organization is OIB ING. In this organization only two software development processes are in use. These are the Waterfall model and the Rational Unified Process (RUP). This organization is aware of the fact that projects are not as successful as they wish. Furthermore, they know that the Waterfall model cannot cope with complex and uncertain projects.

At this point in time, choosing a software development process is often based on previous experiences and organizational standards. However, because of the large amount of available software development processes and the large differences that could be present between each individual software project, the decision of which process to use should depend on the project at hand. For many organizations, as well as for OIB ING, analysis of all processes and even more important, the characteristics of each new project, is very time consuming. Therefore, choosing a software development process that is known to the organization and familiar for all employees, is often seen to be an appropriate decision. Because this manner of approaching software projects cannot be seen as effective, this research is conducted. This research is focused on designing a framework which combines the available software development processes with the project at hand. By using this framework, project managers can select the most appropriate software development process based on the project's characteristics. The main research question that is answered in this thesis report is: "*What decision framework will assist project managers in large organizations in choosing a suitable software development process?"* A combination of case study, literature research, interviews with users and experts, and design was used to develop this framework.

To develop the framework, two sets of information were imperative: characteristics, advantages and disadvantages of software development processes and the characteristics that define software projects and influence the suitability of the processes. An analysis was performed of eight software development processes – the Waterfall model, RUP, Spiral model, V-model, Extreme Programming (XP), Scrum, the Dynamic Systems Development Method (DSDM) and Feature Driven Development (FDD). These processes represent the traditional processes, the agile processes and the processes in between.

The second set of information needed for this framework was an overview of characteristics of software projects executed at OIB ING. By conducting interviews and analyzing literature and lessons learned documents, characteristics were found. By distributing a survey among working personnel at OIB ING the first list of characteristics was validated and adjusted where needed. The final characteristics are: Project size, team size, requirements maturity, team relationship, client's commitment, scope clearness, risk clearness, development stability, stakeholders flexibility, method of contracting and outsourcing.

Based on these findings a framework was developed. The information regarding the software development processes was mapped on the characteristics of software projects. By scaling the software projects from 1 to 5 (for example: Project Size can be very small, thus 1, or the Project Size can be very big, thus 5) it was possible to analyze the suitability of each process on each scale. This formed a framework for selecting a suitable software development process for the project at hand. The framework consists of a questionnaire to be filled in by the user. This shows the five scales of each characteristic. By filling in these characteristics, the software project of interest is analyzed. Based on this project a certain process (or processes) is (are) concluded to be most suitable. To assist the users, the calculation tab is included, which shows the scores of the processes on each individual characteristic, and an explanation tab of the framework and one with all processes.

The findings in this research, a decision support framework, were based on a case study executed at OIB ING. However, this framework can be used for all market segments and organizations which are involved in developing software. By users and experts this decision framework is concluded to be suitable and very useful.

CONTENTS

FIGURE INDEX

TABLE INDEX

1. Introduction

And by now it is generally recognized that the design of any large sophisticated system is going to be a very difficult job, and whenever one meets people responsible for such undertakings, one finds them very much concerned about the reliability issue, and rightly so.

-Edsger Dijkstra, *The Humble Programmer (1972)*

Software Engineering is an engineering discipline which concerns the entire process of software production, from setting up requirements to testing and maintaining the system (Sommerville, 2007). The term software engineering was proposed in 1968 (Naeur & Randell, 1968). By that period, software development encountered several critical issues. Problems existed such as late deliveries, high maintenance costs, over-budget projects and low reliability. This period of software development misery became known as the Software Crisis (Georgiadou, 2003). Edsger Dijkstra, mentioned that this period of time was one of the darkest in his professional career, in which the future of computer science was uncertain (Dijkstra, 1972). The late 1960's initiated a critically important study and research of the possible methods, techniques and tools to attempt to minimize the likelihood of failures in software projects (Georgiadou, 2003).

The Software Crisis resulted in a search for increasing quality and number of tools and methods for software engineering. Software processes became a helpful approach for controlling the process from creating requirements through the testing and maintenance of the final system. A software process entails a set of activities, methods, practices and transformations that are used within the process of producing a software product (Georgiadou, 2003). In this research the focus is on software process models. A software process model gives an abstract, simplified description of a software process and includes environmental factors, such as roles of people involved and other software products (Sommerville, 2007). Most of the software process models are based on three generic models – The sequential model, the evolutionary (iterative) model and the component-based software model (Sommerville, 2007).

The sequential model entails a process which uses linear phasing. This means that when one phase is finished within a project, a next phase follows. These phases usually contain setting up requirements, design, implementation and final integration.
Evolutionary development creates an iterative way of working. An initial idea is produced, which is then discussed with stakeholders and is finally refined through user comments (Sommerville, 2007). Evolutionary development is often seen as more effective compared to approaches such as the sequential model - the incremental approach of development is considered a major benefit.
Component-based software models apply the reuse of common parts within a project. By reusing software components, the amount of software to be developed is significantly reduced which lowers costs and risks (Sommerville, 2007).

Most software processes in use are based on one, or even on a combination, of these approaches. Furthermore, no one is considered the best. Often multiple dependencies of the organization and the project create a vague view of which process to use. Within Operations and IT Banking ING (OIB ING), which is the ICT development branch within ING, regularly two software processes are used – the Waterfall model (Royce, 1970) and the Rational Unified Process (RUP) (Kruchten, 2004). The

Waterfall model uses the sequential approach and the RUP uses a combination of all three approaches.

Choosing a software development process is often based on previous experiences. For managers, to choose an appropriate process is becoming increasingly difficult. The combination of process characteristics, the environment, actors, factors and evolvement of the organization or project, need to be ideal (Meulendijk & Oud, 2007). However, for many organizations, as well as for OIB ING, the analysis of all processes and even more important, the characteristics of each new project, are time consuming. Therefore, choosing a software development process that is known to the organization and familiar for all employees is often seen to be a suitable decision.

Unfortunately, this manner of approaching software projects and their processes is not suitable for medium to large systems. According to Georgiadou, a review of hundreds of software development projects in organizations indicated that around 80% of the projects were still considered unsuccessful (over budget, unreliable, late deliveries) and that around a third of the projects were cancelled (Georgiadou, 2003). The main reason for these repeating difficulties, is that organizations apply processes to projects without considering the characteristics of these projects and the possible other processes available. The success of using a software process for a software project heavily depends on the characteristics of the project itself and the characteristics of the process in use. There is no ideal process (Sommerville, 1996). Hans van Vliet as well recognizes the need for applying a certain software process based on the characteristics of the project (Vliet, 2008). A possibility to cope with this, is to tailor a process according to the organization and its environment. Unfortunately, using this approach, project managers need to monitor, analyze and adjust the process tailoring strategies and the project environment continuously during the project (Xu & Ramesh, 2008). Another solution is to use different processes for different projects. Then, the most appropriate process to use, depends on the characteristics of the particular project (Meulendijk & Oud, 2007). By providing project managers a framework for choosing the right process, valuable time is saved. Furthermore, projects that are implemented with the use of a suitable development process, may increase the quality of the final product.

In order to assist managers in choosing an appropriate software development process for the specific software project at hand, a framework is needed. This research is focused on finding and applying an appropriate framework to assist project managers in large scale organizations to decide which process to use for a particular project.

This thesis is organized as follows. Chapter two defines the problem further. The problem environment in which OIB ING finds itself is analyzed and the knowledge gaps in choosing the correct software development process for a particular project are given. Leading from this problem description, concrete research questions and research methodology are formulated as well. Chapter three consists of a research regarding the software development processes in use. This entails software development processes in use at OIB ING and their vendors. Additional interesting processes are taken into account as well. In the final section of this chapter an overview is provided of the characteristics, advantages and disadvantages of each process. Chapter four focuses on the projects done by OIB ING. The essential characteristics of these projects are analyzed. From these two overviews of characteristics a mapping solution is proposed in chapter five. This is then translated into a working framework. The framework is validated and is described in chapter seven. Finally the conclusion, reflection and recommendations are presented in chapter 8.

2. RESEARCH DEFINITION

This chapter further elaborates on the problem definition in section 2.1, which is the foundation of this research. In section 2.2, the research questions are provided. Section 2.3 gives an overview of which methodologies are used and why. An elaboration concerning OIB ING is given in section 2.4. The organizational architecture is shown as well as their environment and business process. Finally in section 2.5 the processes which are explored in this research are given.

2.1 Problem description

In the last decades a significant increase in the number of software development processes occurred. The Waterfall model was the first recognized process of its kind. It described a sequential linear process where the development phases are followed step by step (Royce, 1970). The Waterfall model is a simple and well structured process. Unfortunately, also commented by Royce, the model has some significant shortcomings. The most critical lacking factor is iteration (Liu & Horowitz, 1989). The Waterfall model was the base of all software development processes in the time to come (Georgiadou, 2003). The most important processes that followed the Waterfall model were the Spiral model, the V-model, the Rational Unified Process (RUP) and agile processes, such as eXtreme Programming (XP) and Scrum (Georgiadou, 2003).

Project managers at OIB ING often automatically choose to use the Waterfall model, and sometimes the RUP, without considering any other options. So why do the project managers never choose another approach? Even if the RUP is an enhanced and sophisticated approach, it does not necessarily mean that the RUP is the appropriate process to use (Runeson & Greberg, 2004). Every project has different characteristics, such as costs, time-span, number of users and maturity of requirements. Furthermore, each development process differs too, e.g. the use of phases, different characteristics and the difference in complexity. Instead of having the RUP or the Waterfall model as the main process, familiarization with other software development processes and a reflection on the type of project at hand might conclude to use another approach (Meulendijk & Oud, 2007). Unfortunately, this would be a time consuming process to do for each new project. The decision framework designed in this research should provide managers a major benefit for choosing an appropriate process for each new project.

2.2 Research questions

The problem description stated previously can be translated in the following main research question:

What decision framework will assist project managers in large organizations in choosing a suitable software development process?

To answer this research question, a process is followed that flows from descriptive research, to the application and further evaluation of the framework. First, research needs to be done regarding the software development processes available and the processes that are familiar within OIB ING and its vendors. The processes used by OIB ING in the past and present are analyzed. Possible upcoming processes that might be useful in the near future are analyzed as well. An overview is created of the characteristics, advantages and disadvantages of the researched software development processes.

The framework that is developed depends on the characteristics of software development processes as well as on the characteristics of the software projects conducted by OIB ING. Therefore an overview is needed with all possible characteristics of these software projects. Characteristics such as costs, deadlines, coding language and maturity of requirements are a necessity for the framework. This can be achieved by enquiring the managers and by analyzing software projects done in the past by OIB ING.

Following the descriptive part of the research is the phase where the framework is applied. This consists of three parts. First a framework design is selected that is suitable for OIB ING. This is then designed into a decision framework that assists the managers at OIB ING in choosing the appropriate software development process for the project at hand. The decision framework is validated by the managers, other personnel and experts. Finally, generalization possibilities are analyzed. This is done to find whether the framework can be used for other organizations and to what extent.

The research can be formulated in certain sub questions. These sub questions relate to the different components of the research and are stated below:

Category 1: Software development processes analysis

- Which software development processes are used by large-scale organizations?
- What are the characteristics, advantages and disadvantages of these processes?

Category 2: Software project analysis

- Which characteristics of the software projects done at OIB ING are imperative for choosing the most appropriate software development process?
- How are these characteristics weighed by OIB ING personnel?

Category 3: Designing the framework

- How should the framework be designed to be suitable for OIB ING?
- To what extent is it possible to generalize this framework to other organizations and their projects?

2.3 Research methodology

The research methodology consists of six different methods combined. These six methods are “Case study”, “Desk research”, “Surveys”, “Interviews”, “Case descriptions” and “Design research”. A combination of methods is used to provide a wide scope of information and opinions within the research (Verschuren & Doorewaard, 1999).

Case Study

The Case study was performed at OIB ING. Information from OIB ING is used to provide a practical dimension. Practical information is needed to obtain an overview of how processes are used and why. Processes are always adjusted when used in practice. Furthermore, for analyzing projects, information is needed from personnel at OIB ING. Opinions of personnel regarding significant characteristics are necessary. Furthermore, the weights personnel give to those characteristics are interesting as well. The decision framework is designed for and validated by OIB ING.

Desk Research

Desk research is a literature searching approach. This provides basic theoretical information concerning the processes and their characteristics. To provide OIB ING with a decision framework, advantages and disadvantages are needed of all processes within this research. Some of these characteristics are found by interviews, but others are found by doing literature research. This subject is well supported in literature and therefore it provides a wide variety of information.

Survey
Surveys are used to get overall fieldwork information and opinions of personnel. In practice, professionals are very busy. By creating surveys and distributing these on the work floor, personnel are not disturbed too much. Opinions are needed concerning the characteristics of projects and how they are weighted. This method provides a wide and generic overview of information. By letting personnel fill in question forms, and applying this on a wide scale throughout the organization, a better view is provided of opinions organization-wide, instead of only of one or two individuals.

Interviews
The surveys provide a generic source of information. Some detailed and specific information is needed as well. This in-depth information is provided by performing one-on-one interviews. These interviews were performed with personnel throughout the organization. The target of these interviews were the managers within the organization, personnel that is involved in choosing the processes and personnel that works with those processes. Furthermore, interviews with vendors were needed as well. They provided information from another view point regarding that of OIB ING.

Case Descriptions
Case descriptions generate detailed information regarding specific projects that are analyzed. For example information concerning budget and time span of projects should be retrieved from these descriptions. Most of the information is regarded tacit knowledge. However, the case descriptions provide a basic understanding of projects and some of their characteristics. The detailed and tacit information was obtained by doing interviews.

Design Research
The method design research was used to design the decision framework for OIB ING and possibly for other organizations. Based on the characteristics of the processes and projects, a framework is designed. However, a mapping problem existed. When this mapping problem is configured into a design, this design is validated by personnel and experts within OIB ING.

Figure 2-1 depicts the process of the research. This is presented by using the UML activity diagram (Booch et al, 1997). The sub questions are the leading activities within the process. The methodologies that are used within the process are depicted as well. This represents an overview of when, within the process, each method was used.

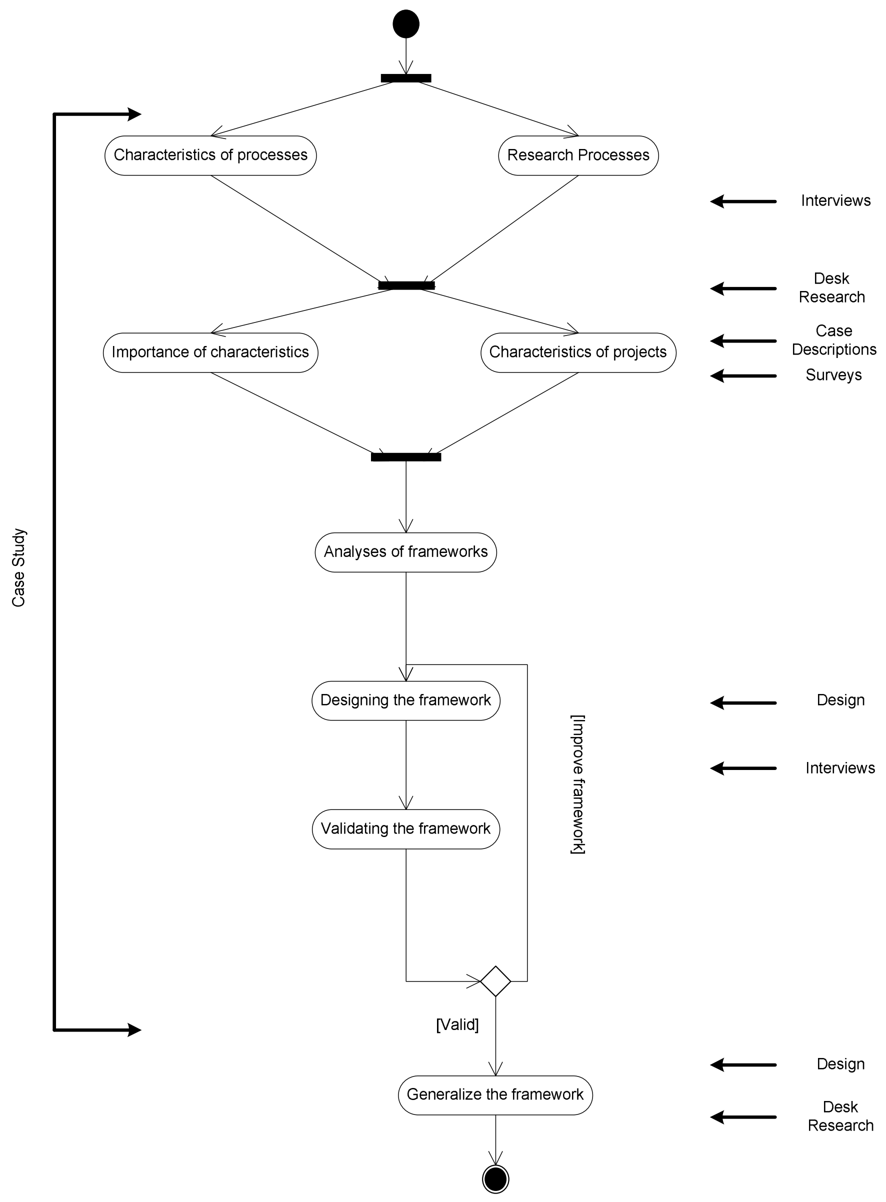


FIGURE 2-1 Research process UML

2.4 Research environment

This section provides an overview of the environment in which this research was conducted. As was mentioned before, the case study have taken place at OIB ING. To provide a better understanding of this organization an explicit description of the departments is given and their organizational structure. Furthermore, the process of delivering a project is explained as well as the key performance indicators of those projects. Finally, the imperative situational factors are specified.

2.4.1 Case study at Operations and IT Banking ING

Operations and IT Banking ING (OIB ING) supports commercial activities for Retail Banking and Wholesale Banking within ING. It is a single service provider that delivers value-added Operations and IT services to its business partners. The responsibilities of OIB ING contain the following services:

- Service Delivery for Wholesale Banking and Retail Banking
- Solution Delivery for Wholesale Banking, Retail Banking and Group Support Services, such as Human Resources, Finance and Risk Managements

- IT Infrastructure and data centres Service Delivery for Wholesale and Retail Banking worldwide, for Group Support Services and for all European based ING Insurance Units.

OIB ING should provide an effective and efficient baseline. Budgets need to be discretionary spent and managed. They are also responsible for a day-to-day operational service and solution delivery. Furthermore, OIB ING needs to provide end-to-end high quality solutions to be developed. A successful improvement in quality and maturity of processes ensures this. The mission of OIB ING is to provide their business partners a strategic advantage via efficient and effective employment of people, process and technology. For improving the solution delivery, OIB ING is using the industry standard CMMI for restoring reliability and improving efficiency (Chrissis et al, 2003).

To provide an overview of the organizational structure a schema is depicted in figure 2-2. Every vertical entails several solution delivery centres. The organizational structure of vertical Channels is depicted in figure 2-3. This research was conducted in this vertical, and mainly within SoDC Call Face Retail NL. The vertical Channels delivers services and products to the various distribution channels of Retail Netherlands and Belgium as well as Wholesale Banking. These distribution channels are Internet, Call and Face.

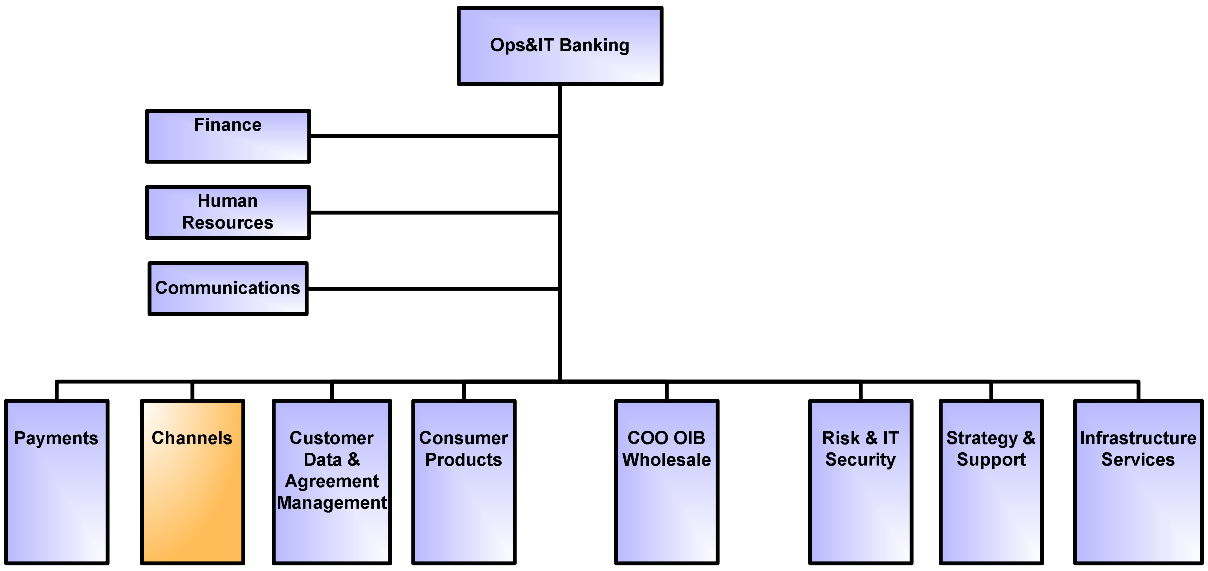


FIGURE 2-2 Organizational Structure Operations and IT Banking ING

Within the vertical Channels there are four solution delivery centres. These are Solution Delivery Centre Internet Retail NL, Solution Delivery Centre Call Face Retail NL, Solution Delivery Centre Wholesale Channels and the Solution Delivery Centre Retail Channels BE. This Master Thesis Research was conducted within Solution Delivery Centre Call Face Retail NL. Conclusions within this research could be used within the other departments as well. However, conclusions are based on findings from Solution Delivery Centre Call Face Retail NL.

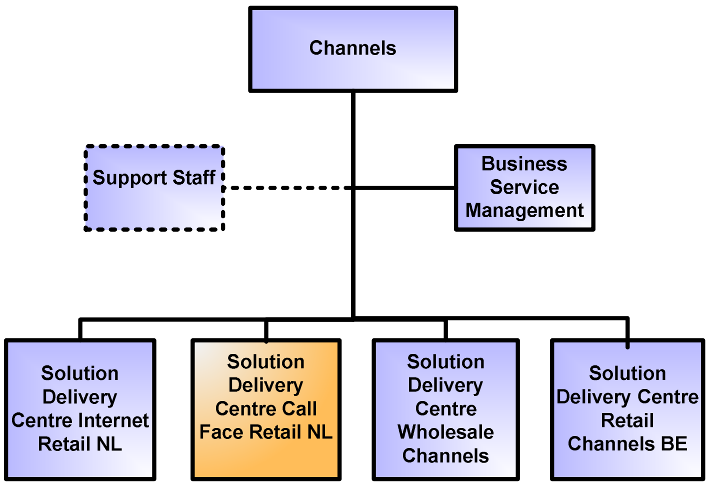

Channels
Support Staff
Business Service Management
Solution Delivery Centre Internet Retail NL
Solution Delivery Centre Call Face Retail NL
Solution Delivery Centre Wholesale Channels
Solution Delivery Centre Retail Channels BE

FIGURE 2-3 Organizational Structure Channels

Solution Delivery Centre Call Face Retail NL (SoDC Call Face NL) is end-to-end responsible for solution and service delivery of all channels (except Internet) within Retail Netherlands, i.e. Postbank and ING Bank (which are now combined into ING). This entails providing software services and products to their business partners.

The business partners listed below are the clients of SoDC Call Face NL:

- Branches ING
- Call centres
- Mail
- Complaints service
- Sales forces
  - Security
  - Mortgages
  - Insurances
  - Small to Medium sized Enterprises (companies from 5 to 6 employees)
  - Private banking (500.000 euro and more)
  - Personal banking (Between 75.000 en 500.000 euro)
  - Business banking (large Enterprises, no Multinationals (wholesale))

2.4.2 Business Process of Solution Delivery Centre Call Face Retail NL

As was explained before, this research has taken place at Solution Delivery Centre Call Face Retail NL (SoDC Call Face NL). This department within OIB ING creates applications for many other departments within ING, such as call centres and branches. In this section an overview is provided of how a project is processed within this department. Figure 2-4 depicts this process.

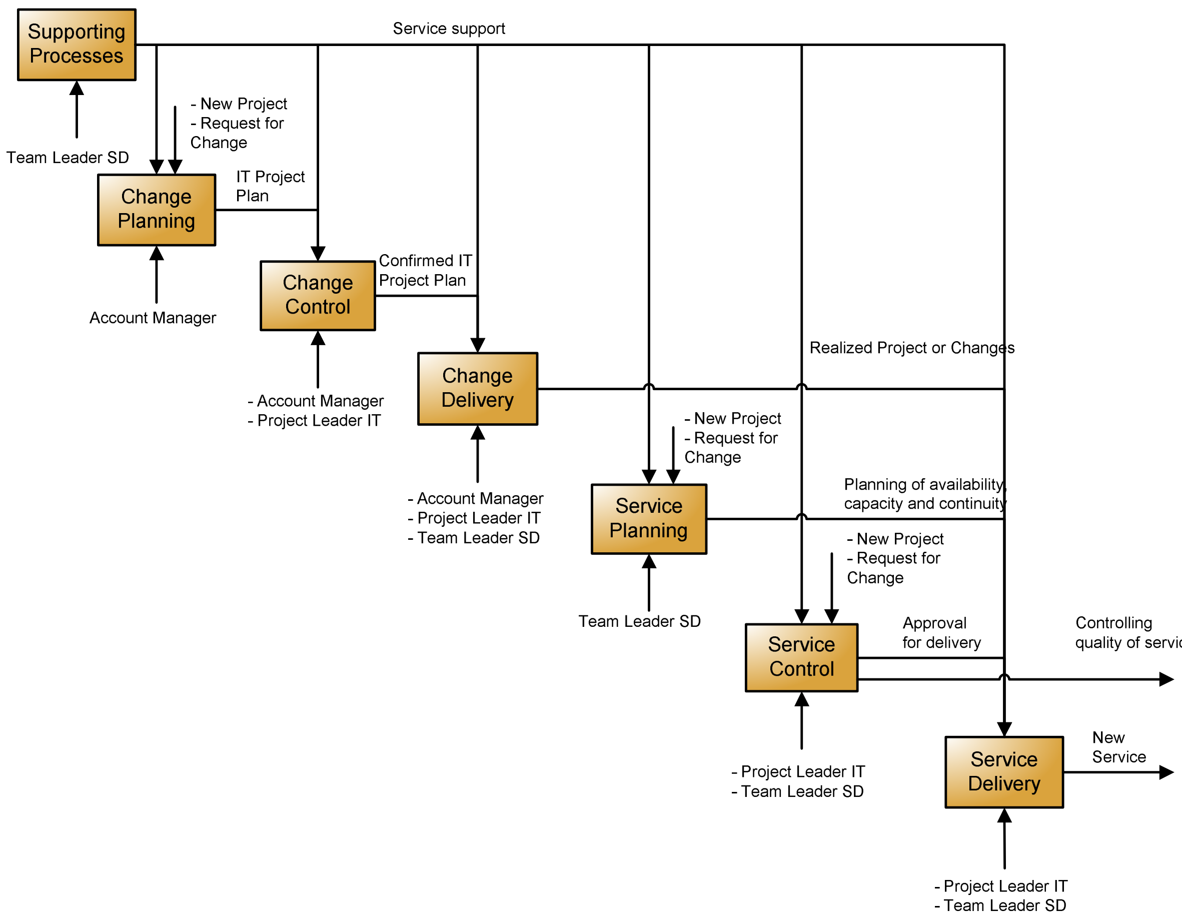


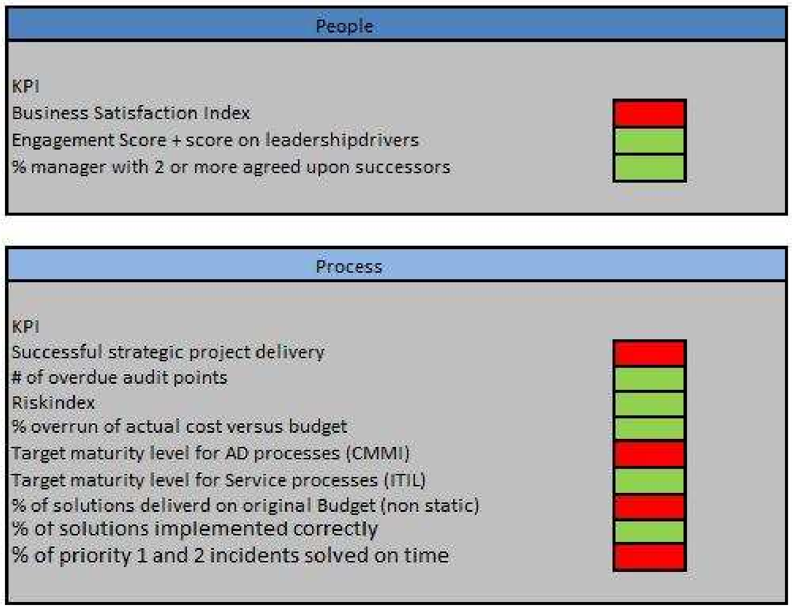
FIGURE 2-4 Business process of OIB ING

Assignments for new services or changes in existing services are delivered to “Change Planning”. An Account Manager is responsible for the contact with the business partner. They create a project plan. In this project plan a detailed overview is given which provides specified information regarding the time schedule, resources, requirements and deliverables. This project plan is assessed by “Change Control”. As soon as this is confirmed, “Change Delivery” executes the project. From then on the Project Leader IT leads the project team. They give feedback to the Account Manager and Team Leader. The Team Leader of solution delivery is responsible for the entire project, in the role of senior supplier. Especially in this area, the use of a process is of significant importance. When the new service is ready, it is delivered to “Service Delivery”. Combined with the service, a service level agreement is provided by “Service Planning”. This provides “Service Delivery” an overview of availability, capacity and continuity. The “Service Control” unit gives its approval to execute the new service. During this execution of the service, “Service Control” will guard the quality of the service and the realization of the agreements made. The “Supporting Processes” provides the entire process support, such as information regarding resources, SoDC budget, contract management and more. The exact descriptions of the responsibilities of the roles described above are presented in Appendix I.

Within SoDC Call Face NL the department Requirements Management is responsible for the process areas “Change Planning”, “Change Control”, and “Supporting Processes”. The Solution Delivery Teams are responsible for “Change Delivery”. These teams are given specific domains for which they are responsible. The department “Systems Management” is responsible for “Service Planning”, “Service Control” and “Service Delivery”.

2.4.3 SoDC Call Face NL project Key Performance Indicators (KPI's)

This research intents to help Project Managers to decide which process to use for a project. Currently projects at SoDC Call Face NL are not as successful as they want them to be. Figure 2-5 depicts the Key Performance Indicators. The most significant KPI is the business satisfaction index. They are given a red score for their services, mostly due to the lack of reliability in delivery. Furthermore the percentage of solutions delivered on original budget scored red as well. This provides a clear overview of the issues within their projects. The result of using software development processes should improve discipline to the development. This should help in improving issues such as low reliability and over budget. These below target scores might suggest that among other problems (i.e. lack of knowledge, miscommunication, unclear business processes) inappropriate processes are used or that the processes are used incorrectly. In this research the focus is on the possibility that inappropriate software development processes are used.



People	
KPI	
Business Satisfaction Index	🟥
Engagement Score + score on leadershipdrivers	🟩
% manager with 2 or more agreed upon successors	🟩

Process	
KPI	
Successful strategic project delivery	🟥
# of overdue audit points	🟩
Riskindex	🟩
% overrun of actual cost versus budget	🟩
Target maturity level for AD processes (CMMI)	🟥
Target maturity level for Service processes (ITIL)	🟩
% of solutions deliverd on original Budget (non static)	🟥
% of solutions implemented correctly	🟩
% of priority 1 and 2 incidents solved on time	🟥

FIGURE 2-5 Key Performance Indicators of SoDC Call Face NL

2.4.4 OIB environment

Many factors influence a market, an organization and even software development projects. The four most important factors are discussed here. These four factors play a significant role within the environment of OIB ING and thus this research. These factors are TANGO, dependencies of vendors, the manner of contracting and the current financial situation. The factors influence the opportunities of OIB ING, the possibilities for this research and possibilities for change. These four factors are shortly discussed as follows.

TANGO is the name for the transition of ING Bank and Postbank into ING. The TANGO program consists of 250 projects. All these projects contribute to the entire transition. For example, LISA (Leading ING Sales Application) provides an application for all sales within the branches of ING. OIB ING is responsible for implementing LISA and many other imperative applications within the TANGO program. This transition from old applications of both ING Bank and Postbank to entire new applications for the channels, caused a very stressful period. This program still has a major influence on the change portfolio of OIB ING projects.

OIB ING have chosen to outsource all of their technical activities such as coding. Major vendors are Logica, Capgemini, Getronics, Ordina and Atos Origin. These vendors play a significant role in the process of developing a system. When the project is outsourced a great part of control is lost. OIB ING is then dependent on the schedules and efficiency of those vendors. Furthermore, they are dependent on the process-environment and experience of these vendors. For example, if a vendor is used to apply RUP as standard software development process, difficulties can arise when OIB ING uses a different software model. Therefore, OIB ING and the major vendors mentioned before, decided to agree that they all use RUP as main methodology.

The method of contracting often in use at OIB ING could possibly be an obstacle for using other processes. The policy of ING is to apply fixed pricing with their vendors. This entails that ING and the vendor agree on a certain price. When projects go over budget, the additional costs are allocated to the vendor. This seems a solid method. However, fixed pricing encourages to apply the Waterfall model. Fixed pricing is most suitable when requirements are frozen. A vendor will never agree on a price when OIB ING is still free to change requirements during the entire process. Unfortunately this manner of contracting does not only occur between ING and vendors. Within ING, Retail applies this method as well. This method of contracting can therefore be an obstacle for introducing other processes which clash with this approach.

The final and very imperative and significant issue in the environment of OIB ING and this research is the current financial situation. In the first semester of 2009, the financial market is in crisis and so is ING. The value of ING is heavily diminishing on the stock exchange and insecurity hits. Budgetary cuts are needed and projects are being cancelled. This situation has a considerable influence on the changes and risks ING is willing to take. Especially for large projects, this need for control and safety definitely impacts the choice of processes to use and even more important the willingness to change these processes.

2.5 Processes

As was explained in the Introduction, many processes exist. This research was conducted in a large financial organization. Not every software development process that exists is viable for such an organization. In this section a demarcation is given of the processes that are analyzed in this research. This list of processes is based on the processes in use at OIB ING and interesting processes in literature.

First of all the Waterfall model and RUP are included in this research. These are the two main processes in use at OIB ING. Processes similar to these two are the V-model and the Spiral model. The V-model is based on the Waterfall model which might provide other insights. The Spiral model uses risk assessment intensively. Therefore these two are included in this research as well. The Agile methodology is also very interesting to be included in this research. These methods are based on an entire different approach compared to the Waterfall model. Agile methods are considered light approaches. Some of the most significant processes within this area of methodology are eXtreme Programming (XP), Scrum and Feature Driven Development (FDD). The final method which could be useful is the Dynamic Systems Development Method (DSDM). This process was used in the past by OIB ING and is also an Agile methodology.

Thus, the following processes researched in this thesis are:

- The Waterfall model
- The Rational Unified Process
- The Spiral model
- The V-model
- eXtreme Programming
- Scrum
- The Dynamic Systems Development Method
- Feature Driven Development

3. THEORETICAL BACKGROUND ON SOFTWARE PROCESSES

This chapter consists of an analyses of the software development processes stated in section 2.5. To answer the main research question it is imminent to provide a theoretical analysis of software development processes. In this chapter the following research sub questions are answered; "*Which software development processes are used by large-scale organizations?*" and "*What are the characteristics, advantages and disadvantages of these processes?*" The first eight sections present theory regarding software development processes. Within each section a description is given of the process as well as its characteristics and interesting features. Information regarding these processes is gathered by doing literature research and conducting interviews with employees within OIB ING. These analyses are summarized in the final section of this chapter, section 3.10. In this summary the reader is provided an overview of each process' characteristics, advantages and disadvantages. In section 3.9 an overview is presented which depicts whether or not the software process is in use at OIB ING, or at one of their vendors. The processes are presented in the same order as stated in section 2.5.

3.1 The Waterfall model

The Waterfall model, depicted in figure 3-1, became known as the first official software development process, and became a basis for future processes (Sommerville, 2007). The Waterfall model received its name from the cascading manner of following one phase to another. Each phase ends with one or more documents which are approved. The following phase in the process should not start until the previous phase is fully completed. This process should provide team members a clear overview of what work has priority and when it is allowed to follow the next phase. When using the Waterfall model, priority lies on a thorough analyses. This is especially important in this model considering that when constructing the system, no possibility exists for revisiting any requirements (Vliet, 2000).

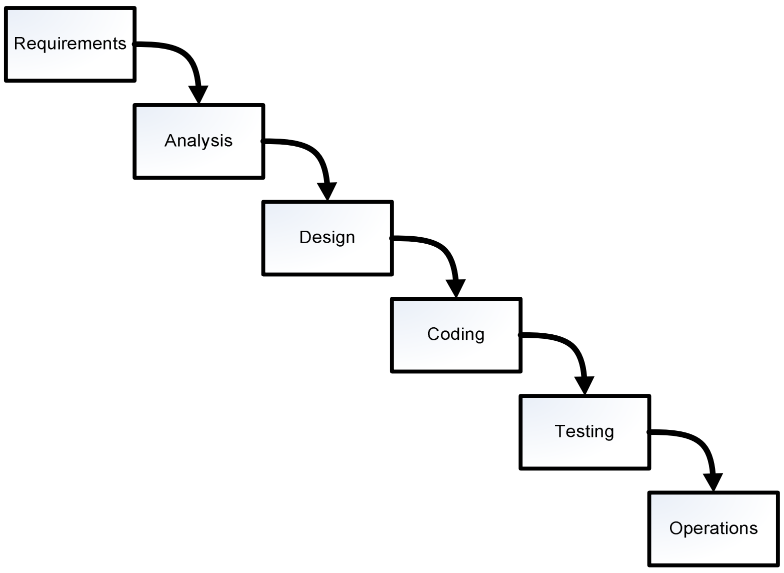


FIGURE 3-1 The Waterfall model

However, software projects do not usually follow such a strict linear process. For example, a minimum amount of software projects start with fully determined requirements. It is possible that the wishes of clients change or that they are misinterpreted. Furthermore, official laws can be adjusted or that through analyses certain issues become more clear in time. According to H. van Vliet ample quantitative evidence exists which shows that the Waterfall model is unrealistic. In many projects actual sequential steps are often not obeyed (Vliet, 2000).

After the original Waterfall model was described, Royce improved this model by including a form of iterations. This Waterfall model provides iterations between each successive step within the process. This approach of iterative processing is depicted in figure 3-2. The value of implementing this type of iteration is that as the design proceeds, the scope of changes is being kept in manageable limits (Royce, 1970).

As Winston Royce describes in the same paper, the Waterfall model is risky and invites failures (Royce, 1970). The most significant issue lies in the use of this type of iterations. When the project is situated in the testing phase of the process and errors are found which are located in the design, a major redesign is needed. This problem of inflexibility causes freezing of previous parts of the process in which errors occurred, meaning that problems that have occurred, e.g. in system requirements or analyses, are left for later resolution. This often results in problems being ignored or in problems being left for later resolution, resulting, obviously, in projects being badly structured and which do not conform the users wishes (Pressman, 2005; Sommerville, 2007).

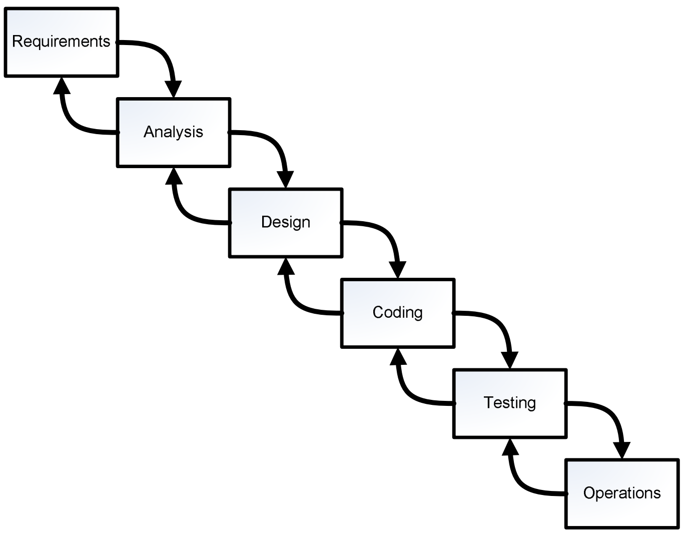


FIGURE 3-2 Waterfall model iteration

The Waterfall model should therefore only be used when dealing with an environment in which the requirements are determined and well-understood (which is rarely the case). Unfortunately, the Waterfall model is being used in great extent (Neil & Laplante, 2003). OIB ING uses the Waterfall model in 90% of its projects.

Waterfall model at OIB ING

According to Danny Wijnand and Ed van Kooten (both employees within SoDC Call Face NL), the Waterfall model within OIB ING is the preferred process for the greater part of the projects conducted (Appendix III). The model used by OIB ING is slightly adjusted in comparison with the original Waterfall model, but the linear sequence of phases is still present. The following phases are applied at OIB ING:

- Specification
- Designing and Building
- Integration and Testing
- Accepting the system
- Deployment

The Waterfall model within OIB ING uses less phases and names them differently. However, the content covered within the original Waterfall model corresponds to the content of the one in use at OIB ING. The content of each phase is discussed next.

The Specification Phase
The requirements and the probability of realizing those requirements are determined in this phase. The wishes of the client are specified and realized. Activities such as "the development of functional requirements" and "creating a cost overview" are a significant part of this phase.

Designing and Building
This phase entails the designing and testing of the system by the project team. Deliverables of this phase are a component model, a technical data model and the software of the system as a product.

Integration and Testing
The software is integrated in the entire system. Activities such as testing the integration and testing the system are crucial. Additional activities such as the creation of manuals provide the support the customer need for the actual usage of the system.

Accepting the system
The delivered products are accepted in this phase. Accepting the system is conducted by the users, maintenance personnel and client. Preparations for the actual implementation of the product is started.

Deployment
This phase entails the preparations for and the support of the actual implementation and usage of the system. An important aspect within this phase is the evaluation of the project and its process.

3.2 The Rational Unified Process

The Rational Unified Process (RUP) is a software engineering process developed and maintained by the Rational Software Corporation. On the 20th of February 2003, Rational was sold to IBM and has now become a separate division.

The RUP is a prescriptive well-defined software development process which provides organizations an iterative, requirements driven, software architectural approach to software development (Ambler et al, 2005). Software architecture entails a "top-level decomposition of a system into major components together with the characterization of how these components interact" (Vliet, 2000). The RUP defines these decisions and creates subsystems where they are decomposed in structural elements (Kruchten, 2004). Furthermore, the RUP provides a process framework to assign tasks and responsibilities to project members in a disciplinary approach (Kruchten, 2004). The RUP involves stakeholders and project members throughout the entire process. By using their roles, an overview is created of how decisions are divided and structured in the software project. Involving project members throughout the entire process, whether these members work with requirements, design, evaluation or configuration management, ensures that all members have a common language and view on how to approach the software project (IBM, 2005). A great advantage of the RUP is that it is based on a high amount of tool support (Kruchten, 2004).

Figure 3-3 shows the architecture of the RUP. The process has two dimensions. The horizontal axis (first dimension) represents the phases which occur in time. The vertical axes (second dimension) represents the core disciplines of each software project. These disciplines group the activities conducted in software projects. For each discipline, the effort throughout the process is depicted, e.g. Business Modeling costs relatively more effort in the Inception phase. However, Business Modeling continues in the Elaboration phase through to Construction.

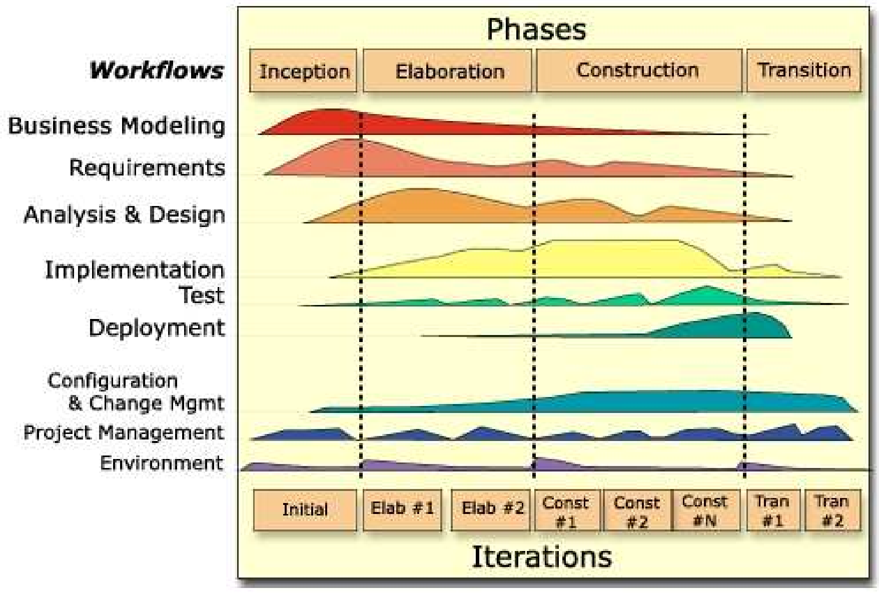


FIGURE 3-3 The RUP v2003 lifecycle.

Phases

As is depicted in figure 3-3, the RUP consists of four phases – Inception, Elaboration, Construction and Transition. Each phase has its own goals and its own well-defined objective. At the end of each phase, stakeholders assess the project. This assessment contains the executed work until that point, but also the plans for the coming phase(s). At this point a go/no-go decision is made whether to continue to the next phase. A no-go decision could entail rework on the previous phase or the plans for the coming phase, or even cancelation of the entire project (Ambler et al, 2005). The four phases of the RUP are explained next.

Inception

The Inception phase is the first phase of the RUP. This phase is of the utmost importance to create understanding among stakeholders, such as clients, vendors and users (Kruchten, 2004). This phase consists of creating an abstract starting point for the project. This is achieved by first developing high-level requirements at which stakeholder consensus is necessary. This delimits the project scope which provides stakeholders an overview of the becoming project. Then, if possible, the process needs to be tailored according to the environment of the project and the project team. Finally a high-level development plan is created (Ambler et al, 2005).

The objective for this phase is to create Lifecycle Objectives (LCO). Lifecycle Objectives are the initial requirements, the delimitation of the project, the development plan and the tailored process (Nazarenko & Beck 2002).

Elaboration
The objective of this phase is to analyze the problem, create an architectural design, create a project plan and elimination of the highest risks (Kruchten, 2004). In the Elaboration phase, the LCOs of the Inception phase are developed in greater detail. First, the high level requirements, defined in the previous phase, need to be specified further. These requirements are specified to address the risks of the project and to ensure that the scope of the project is accepted. The architecture that is used for this particular software project needs to be proven to the stakeholders (Ambler et al, 2005).

The final objective of the Elaboration phase is a Lifecycle Architecture (LCA). This entails that stakeholders assess the state of the project and accept the project's development plan. The requirements are agreed upon, the architecture satisfies the requirements, risk assessment is implemented and reasonable estimates are clear for the development of the project (Ambler et al, 2005)

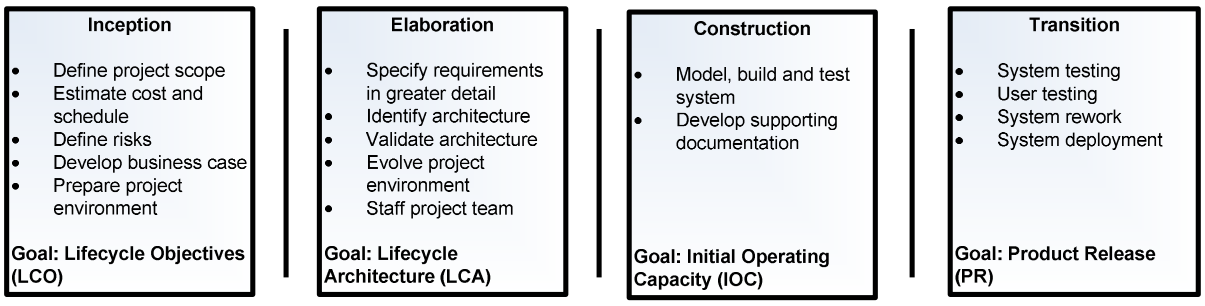
Construction
The Construction phase consists of the actual development of the system, till it reaches the phase where deployment is possible. The requirements, used throughout the process, should be reached. If not, a solution needs to be designed. Therefore, continuous analysis of these requirements is important. Furthermore, the Construction phase consists of coding and testing the actual software. It is possible to release prototypes for feedback from users and stakeholders (Ambler et al, 2005).

An Initial Operating Capacity (IOC) is the objective of this phase. This consists of an assessment of the project by all stakeholders. It must be agreed upon that this developed system is acceptable for release, i.e. the risks are controlled, current expenditures, and the expenditures for the near future are acceptable and a detailed plan is in place for the coming phase (Nazarenko & Beck 2002).

Transition
The purpose is to transfer the actual product to the users and client. This phase entails the actual deployment of the system. Tests are conducted in which the end-users of the system are involved. During the tests problems usually arise. These have to be corrected in this phase (Kruchten, 2004). Preparation for implementing the system in its environment is needed. The using and testing of the system and analyzing if it works as it should, is conducted at the actual client. Finally, support and training are provided in this phase as well.

The actual objective of this phase is Product Release (PR). The system is deployed when all stakeholders agree on the properties of the system. Stakeholders assess whether the project is ready for release. This involves supporting documentation, training and an overview of possible costs in the near future (Ambler et al, 2005).

These phases are summarized in figure 3-4. The specifications of each phase and the final goal are shortly described.



Inception
Define project scope
Estimate cost and schedule
Define risks
Develop business case
Prepare project environment
Goal: Lifecycle Objectives (LCO)
Elaboration
Specify requirements in greater detail
Identify architecture
Validate architecture
Evolve project environment
Staff project team
Goal: Lifecycle Architecture (LCA)
Construction
Model, build and test system
Develop supporting documentation
Goal: Initial Operating Capacity (IOC)
Transition
System testing
User testing
System rework
System deployment
Goal: Product Release (PR)

FIGURE 3-4 The RUP Phases

Iterations
As is depicted at the bottom of figure 3-3, the RUP uses iterations within its phases. Each iteration is focused on completing only a segment of the work done within that phase. A major benefit of implementing iterations within a process is that work does not have to be conducted in sequence. To start with a particular step, it does not necessarily mean the previous is completed (Ambler et al, 2005). During each iteration most of the imperative disciplines, such as implementation, testing etc, are included. This means that in each iteration most of the team members are involved. Each iteration builds up the entire system, continuing on the previous iterations. This parallel, iterative way of working has major benefits. First of all risks and errors are detected early on during the process. Iterations are addressed according to the priority of their risks. Secondly, Reuse of common parts is facilitated seeing that each iteration builds on the work of the previous. Finally, personnel is more effectively used, since each member involved in the major disciplines is involved in each iteration (Kroll & Kruchten, 2003).

Drawbacks of the RUP
Although the RUP uses an iterative and incremental approach, which should be very promising, some drawbacks are familiar within literature and on the work floor at OIB ING. Wolfgang Hesse states that the definitions and organization of the RUP lack clear structure and are too complex and overloaded for practical use (Hesse, 2003). According to Hesse, phases should not be the primary concept for structuring projects. Furthermore the objectives, or milestones, the RUP uses are based on the termination of a certain phase. However, according to Hesse, these milestones should be associated with the termination of certain activities. He finds as well that the iterations the RUP uses are linked to the phases. The iterations should rather be linked to architectural units (artefacts). Finally, he states that the disciplines in use by the RUP "are an overload, partly redundant concept and in particular, the former RUP core workflows were misnamed and their successor disciplines overlap the phases in a fuzzy and over-sophisticated way" (Hesse, 2003). According to other software engineering experts, such as Martin Fowler and Kent Beck, the vagueness of RUP is the main drawback. The process does not succeed in providing a clear indication of how to tackle a project. They have seen RUP being used as a Waterfall model or even as an XP process (Beck & Fowler, 2001).

Within OIB ING, some drawbacks of the RUP are found as well. From the interview with Eric Cardozo and Ed van Kooten (Appendix III.A) can be concluded that a transfer to the RUP within an organization is an intensive process for the project members. Within OIB ING, the Waterfall model was the main process in use. When they tried changing this to the RUP, resistance of project members was high. This change of development process is significant. It is even an intensive learning program for employees that are willing to cooperate. However, when forcing employees unwillingly to make this switch, considerable issues arise. Another significant issue found at OIB ING is the possibility of misinterpretation. The RUP is a light and easy to adapt approach. Although five suppliers of the ING agreed on using the RUP as software development process, it is still possible that each supplier uses a different approach. The possibility to interpret specific parts of the RUP as a

Waterfall approach is present. Danny Wijnand gives the following drawbacks of the RUP in an interview (Appendix III.B):

- Great learning curve
- If not supported by the environment, it is difficult to implement
- RUP is a very “big” process. It is possible to get lost in the process with its immense documentation
- Because it is so big and versatile, everyone has a different view of how to apply it. If one is not careful it may end up with 150 deliverables

3.2.1 eXternal Delivery Factory

The eXternal Delivery Factory (XDF) is the software development framework in use at OIB ING. It is largely based on the RUP. The XDF is used as a framework that describes a standard procedure for executing software projects in which outsourcing is included. Similar to the RUP, XDF uses an iterative and incremental development procedure. XDF is within OIB ING mostly used in modern development environments such as J2EE and .NET.

From the interview with Ed van Kooten and Eric Lopes Cardozo the main differences of XDF in comparison with RUP are (Appendix III.A):

- Contract management (outsourcing) is included in the process
- XDF stated the activities and rules more explicitly to clarify the responsibilities of OIB ING and its vendors

OIB ING outsources most, sometimes even all, of their technical elements, such as coding and testing. The RUP does not explicitly define rules for these procedures. Figure 3-5 depicts an overview which represents the percentage of work done by OIB ING and the percentage done by their vendors for each phase of the RUP. As is shown, the greater part of the construction phase is outsourced. The significant amount of outsourced work makes it necessary to implement this factor into XDF.

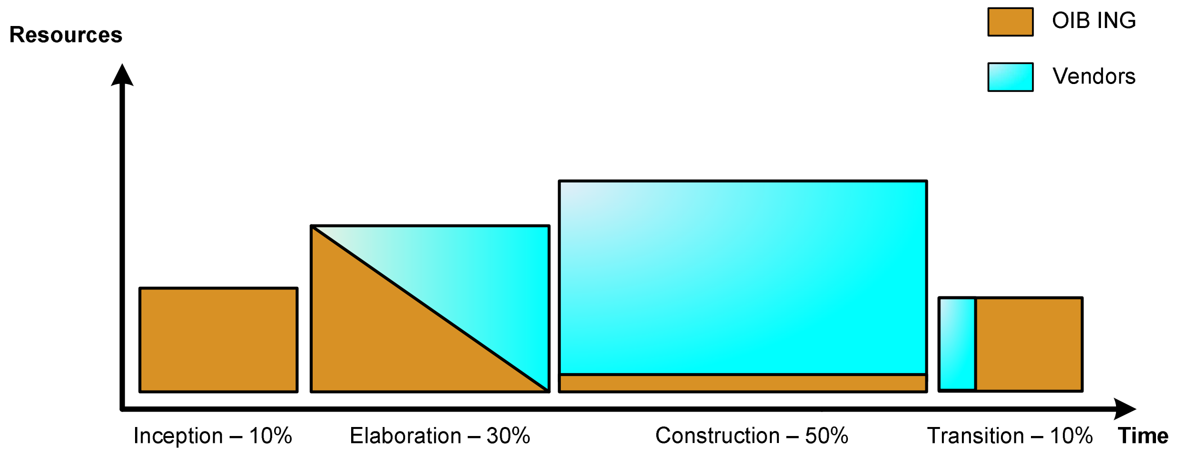


FIGURE 3-5 Outsourcing by OIB ING for each phase of the XDF

In addition to the differences described above, some other deviations are also present. XDF follows largely the RUP Workflows. However, some activities are changed or replaced. The activities within each discipline are shown in Appendix II. The other changes are described next.

Requirements – Timing of Completeness

A very significant change in XDF is the process of determining the requirements. All (for as far as possible) requirements are described at the end of the Elaboration phase. During Construction only changes within those requirements are expected. For the RUP, new requirements are also written during Construction. This deviation is created to support the process of outsourcing. The supplier does most of the work in the Construction phase.

Design – Reference Architecture, Candidate Architecture, High Level and Detailed Design
The final deviation concerns the design of the architecture. The RUP describes one Design Model, for which the Software Architect is responsible. XDF projects usually have two architects involved one from ING and one from the Supplier / Vendor. These architects have different responsibilities. The OIB ING Architect is responsible for the Candidate Architecture of the solution in accordance to the Reference Architecture, therefore also responsible for first drafts of the solution architecture and design at an abstract level. The vendor's software architect is responsible for detailing and realizing the Software Architecture within the boundaries of the Reference Architecture and (proposed) Candidate Architecture.

3.3 The Spiral model

The Spiral model, depicted in figure 3-6, represents phases not as sequential following steps but as loops. The first loop, for example, concerns requirements, the second loop represents analyses, the third concerns design and so on (Sommerville 2007). The concept of the Spiral model is that each cycle contains the same sequence of steps, whether this is requirements or actual coding (Boehm, 1988).

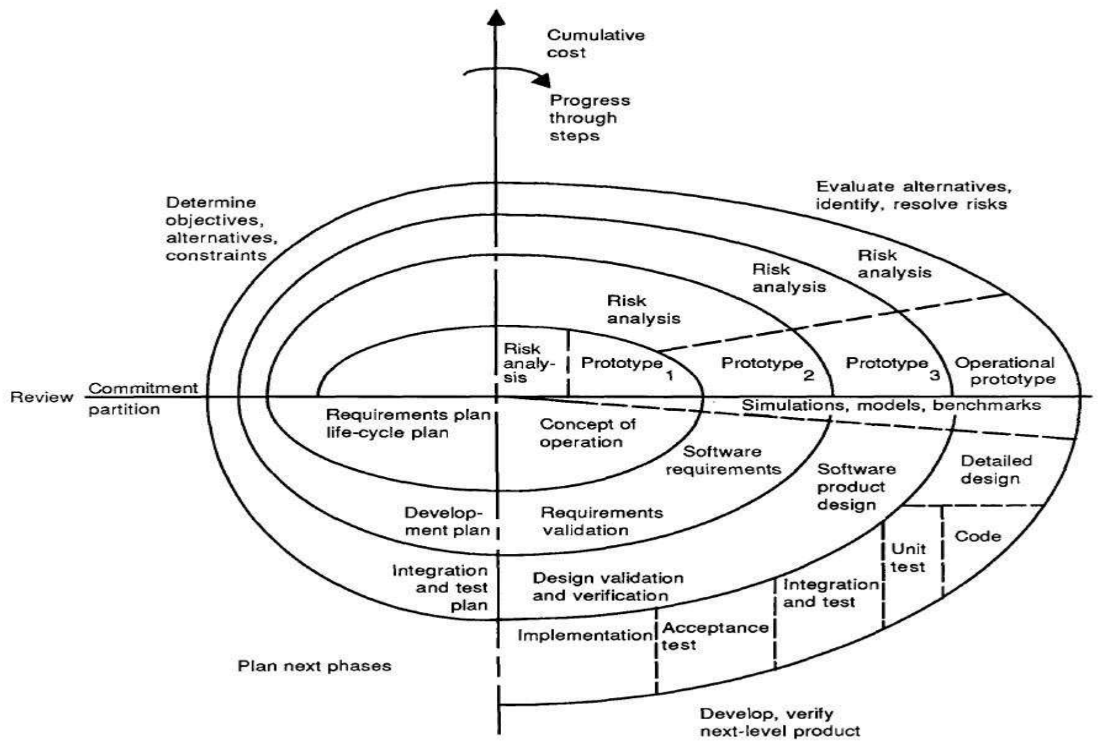


FIGURE 3-6 The Spiral model

As is depicted in the figure, four quarters are created. Each loop follows these four quarters. These quarters are discussed next.

Quarter 1: Determine objectives, alternatives and constraints
Each cycle starts with an identification of the portion of the project that is developed. Objectives, such as performance and functionality, should be recognized. A number of alternative solutions need

to be created. The spiral model makes use of alternative possibilities during the entire process. Each alternative should be analyzed to identify constraints such as costs, risks and interface (Boehm, 1988). Finally a management plan is created to guide the project team through the cycle (Sommerville, 2007).

Quarter 2: Evaluate alternatives, identify, resolve risks
Each alternative is evaluated based on the objectives stated in the first quarter of the process. This evaluation identifies the significant risks that accompanies these alternatives (Boehm, 1988). For the identified project-risks a detailed assessment is carried out. A cost effective strategy is needed to resolve those risks. Exemplary strategies are creating prototypes (especially helpful for identifying requirements) and reasoned modeling.

Quarter 3: Develop, verify next level product
In this quarter, development steps are taken. To provide guidance, a software development model needs to be chosen. This heavily depends on the characteristics of the risks. If user interface risks are dominant, evolutionary prototyping is recommended. However, the Waterfall model is a possible development model when confronted with sub-system integration risks (Sommerville, 2007). A mixture of processes is also available. This all depends on the identified and evaluated risks.

Quarter 4: Plan next phases
Within this phase reviewing plays a significant role. Each cycle is reviewed by all stakeholders involved. The review covers all products and plans created in the previous cycle (Boehm, 1988). At this point in the process a decision needs to be made whether to continue with a second cycle. If the review is negative, revision of the previous cycle is possible. If this is not feasible, cancellation is necessary. If the review is positive however, plans need to be created for the next cycle of the project (Sommerville, 2007).

As becomes clear when analyzing these quarters, the Spiral model heavily incorporates risk management, this in contrary to other processes such as the Waterfall model. This provides major benefits when dealing with large-scale projects. First of all, when risks are identified, prototypes can be created to resolve them. This means that prototypes are developed early on in the process. These prototypes provide clients an overview of the progress and requirements. Secondly, by providing all stakeholders the possibility to review the products and plans after each cycle, they feel involved and committed to the project. Finally, the incremental approach of development and the significant use of risk assessment provide project members the possibility to detect and eliminate errors early on in the process.

However, to fully benefit from this process extensive experience is needed in the field of risk assessment. The success of the Spiral model heavily depends on the risk assessment skills of each member in the project. Another problem that can arise is the lack of tailoring possibilities. The Spiral model cannot be adjusted to fit the organization. This lack of flexibility might create problems for rigid organizations or projects (Boehm, 1988).

3.4 V-model

The V-model, introduced in the 1980's, was created to represent the testing activities at all stages of a process. It is a top-down development process with a bottom-up implementation approach (Matthews, 2002). The phases used by the V-model are similar to those of the Waterfall model. Figure 3-7 depicts the V-model. The left hand side describes the requirements and design, while the right hand side represents implementation and testing. The final activity is maintenance and support.

Testing is based on the specifications of the activities on the other side. The horizontal dotted lines represent the connection (Georgiadou, 2003).

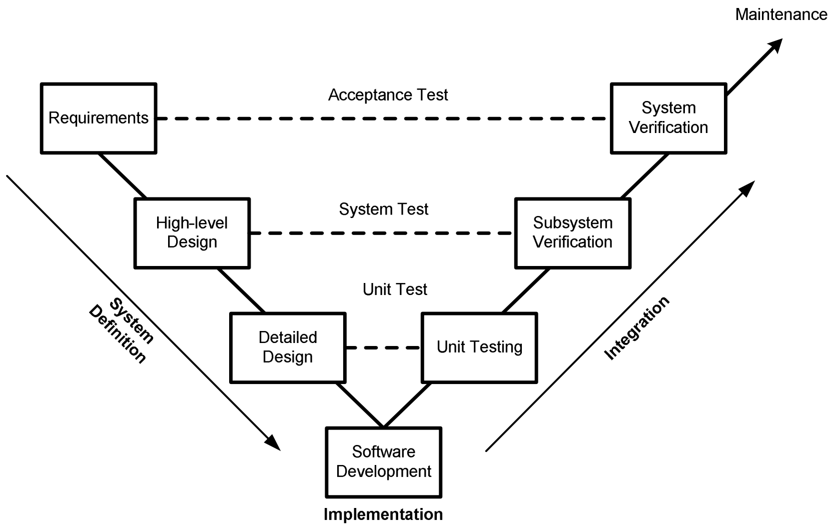


FIGURE 3-7 The V-model

The V-model introduces a comprehensive manner of testing. Each phase of integration starts with tests. This provides a solid integration of the software and system. Documenting the results of each test and comparing these with the documents retrieved from the activity which is connected by the dotted line, provides a clear overview of the results. However, similar issues as that with the Waterfall method can occur while using the V-model. In the beginning of the process, errors or misinterpretations concerning requirements and design often occur (Sommerville, 2007). The V-model does not explicitly state how to cope with these problems (Aughenbauch & Paredis, 2004).

3.5 Agile Methodologies

Agile software development entails processes that have been created by highly experienced practitioners (Dybå & Dingsøyr, 2008). Agile processes, in comparison with traditional processes, do not assume a makeable world in which problems are fully specifiable and in which optimal solutions exist. They rather rely on people and their creativity (Dybå & Dingsøyr, 2008). Agile processes maintain a “light” way of control. A strict process which should be followed does not exist, progress depends more on the understanding and development of the entire project group. Erickson defines agile processes as processes in which the heaviness, which exist in traditional processes, is stripped as much as possible to promote quick response to changes in the environment, user requirements and deadlines (Erickson et al, 2005). The Agile methodology has four core values (Dybå & Dingsøyr, 2008; Hunt, 2006):

- Individual thoughts and their interactions are more important than processes and tools
- Delivering working and tested software instead of ample documentation
- Instead of contract negotiation, collaboration with the client is needed
- Being ready for and responding to changes instead of following a strict plan

This all seems viable and useful. Indeed, agile processes are received well in this field of work and benefits are clear for many users. However, some practitioners criticize agile methods (Dybå & Dingsøyr, 2008; Booch, 2007; McBreen, 2003; Stephens & Rosenberg, 2003):

- There is a lack of focus on architecture and design
- There is little scientific support
- Only viable for highly experienced practitioners, not for novices

- The applicability of modeling languages within eXtreme Programming is minimal
- Larger, more complex projects and project teams do not suit agile

Four agile software development processes are investigated in this research. These are eXtreme Programming, Scrum, Dynamic Software Development Method (DSDM) and Feature Driven Development (FDD) (Dybå & Dingsøyr, 2008). Each process differs in the use of agility. These four processes are discussed next.

3.6 Extreme Programming

According to Beck (2005) XP is about social change. The habits created during the years of software development reduce the quality of work. XP is about freedom from those traditional habits and patterns (Beck & Andres, 2005). XP was originally designed to assist project teams of five to ten programmers, which tackled small development projects in which the environment consisted of uncertainty and changing requirements (Leffingwell, 2007). Four basic principles describe XP: Communication, Simplicity, Feedback and Courage (Hunt, 2006). It is an approach that uses a great amount of iterations. These iterations are stated as user stories. User stories consist of certain tasks. Each task is then developed and tested. After each release the cycle is evaluated. This manner of processing results in an incremental system (Sommerville 2007). These user stories are similar to the Use Cases applied in the RUP. The main difference with user stories is that these are written by the users themselves (Hunt, 2006). Figure 3-8 depicts the project lifecycle of XP.

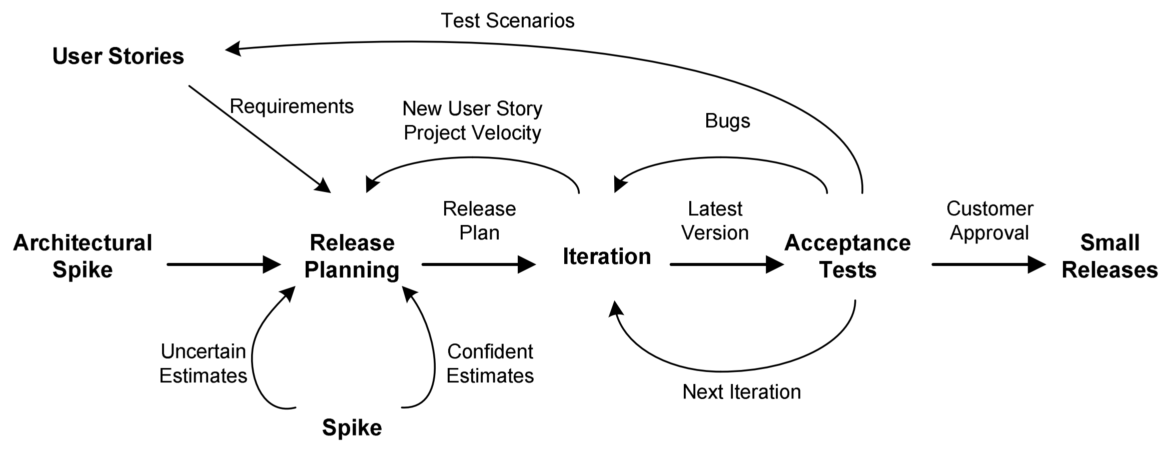


FIGURE 3-8 Extreme Programming lifecycle

The project lifecycle starts with an Architectural Spike. Each Spike entails investigating, researching and evaluating the problem. Imperative is to analyze the architecture early on. Disagreements concerning the architecture often occur in the beginning of a new project. Therefore, this should be discussed in the release planning meeting. The Release Planning meeting is where the project plan is discussed, i.e. which user stories are implemented, how many iterations are planned, and when are these delivered. This should be negotiated with all stakeholders. The next step is the Iterations. The higher number of iterations, the quicker the development team can react on changes or errors. Iterations within XP often consist of a couple of weeks (Hunt, 2006; Dybå & Dingsøyr, 2008). The base for each iteration is an Iteration Planning. This ensures that all stakeholders involved in the process can steer the process in the right direction. Negotiation is imperative in XP. Each iteration implements one or more user stories. These are the basis for the Acceptance Tests. The customer is asked to create scenarios which can be used to test the user stories. Acceptance tests need to be conducted as often as possible throughout the entire process. This ensures that the final deliverables correspond with the actual wishes of the client. Finally, Small Releases are provided to the client. By applying small releases, users can frequently test the deliverables and provide the developers with extensive feedback (Hunt, 2006).

XP practices
Above, the four basic principles of XP are described. Twelve practices are developed which translate the four core values in a way of working which aims at achieving XP objectives (Hunt, 2006). These twelve practices are also called the best practices which define XP. If these twelve practices are not included in the process, it can not be called XP. The twelve practices are (Beck & Andres, 2005):

- *The planning game*
- *Small releases*
- *Simple design*
- *Testing*
- *Refactoring*
- *Pair Programming*
- *Collective ownership*
- *Continuous integration*
- *On-site customer*
- *Coding standards*
- *40-hour week*
- *System metaphor*

Planning game
The planning game focuses on planning the next release. XP uses a very different approach of planning in comparison to traditional processes such as the Waterfall model. Planning in XP is an incremental and broad process.

Small releases
The system that is being developed should consist of many subsystems in which releases often occur. This provides end-users the possibility to give frequent and rapid feedback. These releases should be planned and scheduled in the planning game.

Simple design
The design needs to be as simple as possible. This ensures that the implementation of changes is a relatively easy task. Instead of freezing all the requirements and creating a design in great detail, XP recognizes the importance of a simple design. Changes often occur within projects. By making sure the design is simple and in less detail, these changes are manageable.

Testing
In XP, tests are written before actually developing the system. This ensures that developers first consider the actual objective of the system before thinking about how it should reach this objective. A developer creates one or two tests, then develops a piece of software and finally creates a test case. The software is then rewritten till it passes the final test case.

Refactoring
Refactoring simplifies the design. By applying the rewriting steps regularly, the system improves significantly. Short iterations allow rapid and frequent feedback . Refactoring should occur before and after implementing a new feature. This ensures improvement of the design and implementation of the systems as it continues (Hunt, 2006).

Pair Programming

Pair Programming is an intensively discussed element within XP. Pair Programming means that all the developers work in couples. This provides continuously peer review and code inspection. These couples will regularly change in composition. Unfortunately, working in pairs is not always received well (Hunt, 2006).

Collective ownership
Every developer within the project is involved in the entire project. This means that all developers can adjust any part of the system when they find it necessary. This provides the possibility to tackle found issues immediately by the ones that discovered it.

Continuous integration
The small releases used in XP were discussed before. In this phase these small releases are integrated Continuous integration entails that every completed task should be integrated into the current build. By applying this method, an incremental system is build up, in which each piece of software is tested and integrated

On-site customer
The client and users need to be present during the entire process. Preferably on the same work floor. By having the users available during the process, code is written which is in consensus with the actual users. The iterations within XP provide the possibility to develop small pieces of software which can be tested by the user. By evaluating these pieces of the project, the bigger view can be evaluated as well.

Coding standards
It was mentioned before, that every developer owns all the code. To provide a smooth development each developer should use the same coding standard.

40-hour week
In cases of high pressure, many developers start working additional hours. This causes exhaustion, tiredness and no enthusiasm. To keep everyone fresh and passionate 40-hour weeks should be applied.

System metaphor
The system metaphor is a high level overview of what the overall system should be. This should help people understand how each piece of software fits in the final big picture. The system metaphor is less detailed then the architectural design. It is simply there to provide a basic overview.

Imminent issues for XP
XP is a widely discussed process. Within the community mixed receptions exist of the actual usefulness. If an organization wants to apply XP for a project, certain imminent issues need to be taken into account (Hunt, 2006):

- Determining what to introduce
- Clarifying terminology to simplify communication with the rest of the company
- Avoiding underestimating the effort needed to introduce and adapt XP
- Introducing the practice of continuous testing early
- Having the customer on site
- Documentation is of less importance
- Programmers should work in pairs
- Team member should be skilled with solid domain knowledge
- All team members need to trust each other
- A focus is needed on human and social factors

- Finally, XP is not a full specified development method, but rather a philosophy

According to John Hunt, to actually apply XP to larger projects, the potential for failure, especially in inexperienced teams, is indeed greater. In literature most reviewed practices occurred in small project groups (less then 10 developers) on projects lasting less than 6 months. Hunt concludes with the following rules of thumb for when to use XP: Only for small projects of less than 10 people, experience within the domain is imperative, well-established architectures and scalability should not be an issue (Hunt, 2006).

3.6 Scrum

The Scrum process uses an iterative, incremental and lightweight approach. This process received its name due to its manner of collaboration. According to this process, just as with Rugby (which provided this process its name), a project team should work as a tight, integrated unit, in which every individual understands his/her role. Teams are self-directed and empowered to execute the process (Leffingwell, 2007). This team focus is the essential element of the Scrum process (Rising & Janoff, 2000). Scrum does not necessarily needs to be the main process for a particular project. It is possible that when a project is executed while applying the RUP to use elements of the Scrum process. This is especially useful when being caught in a team that does not seem to collaborate well.
The process of the Scrum is depicted in figure 3-9.

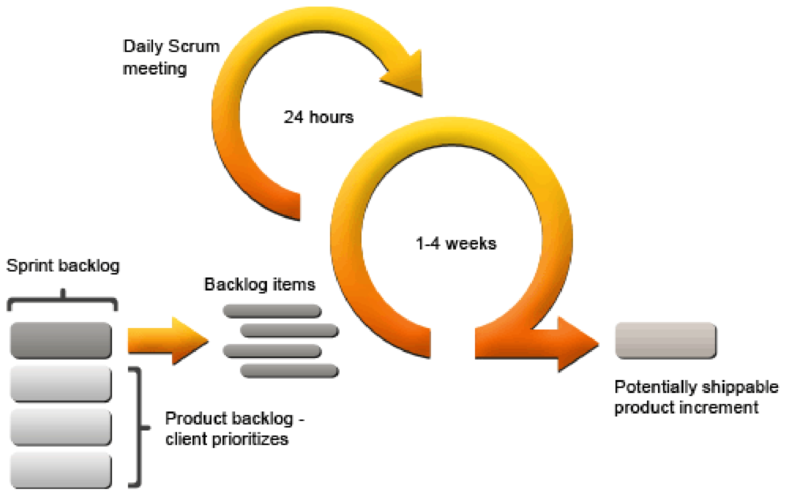

Daily Scrum meeting
24 hours
1-4 weeks
Sprint backlog
Backlog items
Potentially shippable product increment
Product backlog - client prioritizes

FIGURE 3-9 Scrum process

Phases
The Scrum process consist of multiple phases – Pregame, Game and Postgame. The Pregame phase consists of a planning, in which analyses and conceptualization are conducted, and of designing an architecture. This includes system architecture and a high level design. The phase Game consists of Development Sprints. Within these sprints new releases of functionality are developed. There are multiple iterative Development Sprints, which together slowly build up the system. The Postgame phase is the closure. A preparation for release, documentation, testing and release are the main elements within this phase (Schwaber, 1995). The steps conducted in each phase are discussed next.

Pregame – Planning
Within the Planning phase it is necessary to analyze the environment of the project and how the project will be conducted. This consists of conducting a risk assessment, estimating the costs, creating a project-team, defining the delivery date, developing a backlog list and finally verifying with management (Schwaber, 1995). A backlog list drives the teams activities. In this list all currently

identified activities are captured within. Before each sprint, the project-team should update the backlog and prioritize the activities described there (Rising & Janoff, 2000).

Pregame – Architecture/High Level Design
This phase mainly consists of creating a design of how the items on the backlog list will be implemented. Significant steps within this phase are a review of the assigned backlog items, an identification of the possible changes, refinement of the system architecture, an identification of the problems in implementing the changes and finally the design of a review meeting. The review meeting consists of a presentation of each team of what their approach is for implementing the necessary changes (Schwaber, 1995).

Game - Development Sprint
The development of the system is an iterative approach. Before starting the actual sprint, the team meets to review the release plan. When this is discussed, iterative sprints are conducted until the system is completed. Each sprint produces a visible and usable product which consists of a number of user interactions with the system (Rising & Janoff, 2000). This means that each sprint delivers functionality. A sprint is conducted over a predetermined period, usually around two weeks. The risk within each sprint is assessed throughout the process. The following steps need to be performed within each sprint; Develop, Wrap, Review and Adjust (Schwaber, 1995). A very significant element is that after each sprint a team review is conducted in which the whole team, including management and other stakeholders, participate. This review consists of discussing the deliverables of each sprint. The backlog list can be adjusted and a date for the next team review is decided.

Closure
This phase creates a system which can be released. This consists of integration- and system tests, user documentation and training and marketing material. This should only occur when management agrees on the current system and its variables (Schwaber, 1995).

Key practices of Scrum
Just as with Extreme Programming, the Scrum process also contains key practices. These are (Leffingwell, 2007):

- Cross-functional teams of eight or fewer team members
- Sprints are fixed iterations which should take 30-days
- The work within each sprint is planned and fixed
- The teams are self organizing and are self-directed, however, a Scrum Master mentors and manages the teams
- All the work, e.g. requirements, workload, design activities, are noted in the Product Backlog
- The Product Owner manages the Product Backlog
- The main communication method contains a daily 15-minute meeting
- Scrum heavily focuses on time-boxing
- Scrum is an iterative approach which allows requirements, architecture and design to emerge over time

Characteristics of projects
The Scrum process has very specific characteristics. Therefore, only certain projects with certain characteristics are viable to execute with the Scrum process. If these characteristics are not found within a particular project, this should not be executed with the Scrum process. These characteristics are (Schwaber, 1995);

- Deliverable should be flexible – deliverables are dictated by the environment
- Flexible schedule
- Small teams – not more than 6 people within each team. Multiple teams are possible

- Reviews should be conducted frequently – one to four weeks
- Collaboration within the organization and with the stakeholders should be possible
- The project should follow Object Oriented principles

3.7 Dynamic Systems Development Method

DSDM is the only agile process actively used by OIB ING. It is a framework of controls that centers on delivering projects quickly, in which it is supported by guidance on how to use those controls (Stapleton, 2003). DSDM is a Rapid Application Development (RAD) method (Beynon-Davies et al, 1999; Martin, 1991). By applying certain principles, a fast and effective development of software systems is provided, within deadlines and manpower resources (Rook et al, 1999). This method describes the following aspects of system development - project management, estimating, prototyping, testing, quality assurance, risk management. The intention of DSDM is to deliver software systems rapidly. Figure 3-10 depicts the DSDM framework. DSDM has five phases – feasibility study, business study, functional model iteration, system design and build iteration and implementation. These phases are discussed next.

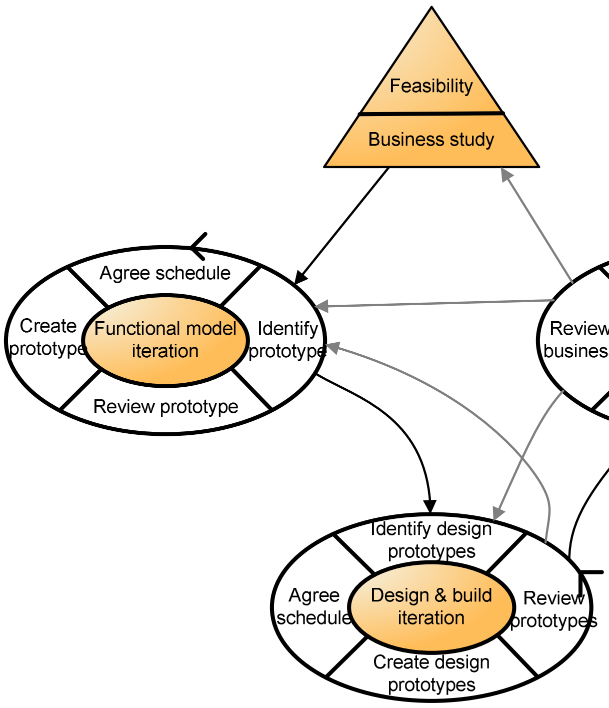


FIGURE 3-10 DSDM process diagram

Phases

Feasibility study

The first phase of the process provides a starting point for the project. In this phase questions such as "should we do the project?" and "is it technically possible?" should be answered here. However, this phase should also be used to answer whether or not DSDM is an appropriate process to use for this project (Stapleton, 2003). When and why to use the DSDM process is discussed later on in this section. Products that are delivered in this phase are a feasibility report, outline plan for development and a fast prototype.

Business study

When having decided that DSDM is in fact an appropriate process to use, a business study is conducted. This provides the foundation on which all subsequent phases are based on. The main objective for this phase is to create understanding of the business processes and information needs (Stapleton, 2003). As was explained earlier, this process provides rapid development. Therefore, instead of performing one-on-one interviews, group workshops are given to provide an environment in which knowledge can be quickly shared and stored. A result from this phase is the business area definition. This is a high level view of the processes to be automated. The requirements are based on the MoSCoW (Must haves, Should haves, Could haves and Won't haves) prioritization rules (Clegg & Barker, 2004). Other products provided at the end of this phase are a system architecture definition and a development plan (Stapleton, 2003).

Functional model iteration
This phase aims at refining the business aspects of the system. As it is stated in the name of this phase, iterations are used. Each iterations consists of four cycles: Identify what you are doing, agree on how you are going to do it, do it and then evaluate it (Stapleton, 2003). The functional model consists of analysis models as well as software components. These components should be tested continuously. The non-functional aspects are tested in the "design and build iteration". Other products created in this phase are prioritized functions, functional prototyping review documents, non-functional requirements and an implementation plan.

Design and build iteration
The design and build iteration phase consists of constructing the system. The system should be of such quality that it can be delivered to the users. Testing of this system is conducted throughout this iteration (Stapleton, 2003).

Implementation
The product that is delivered in this phase is the entire system containing all the necessary documentation. This includes training the users, hand guides and the actual software. The other product this phase delivers is the increment review document. This summarizes all that is achieved in terms of objectives. It explicitly reviews the requirements in combination with the actual resulting system. If the system is not ready for implementation, iterations of previous phases should be executed (Stapleton, 2003).

The underlying principles
The foundation of DSDM lie in nine underlying principles. These principles should all be applied in a project for DSDM to be successful. If some seem difficult to accomplish, using DSDM for that particular project should be reconsidered (Stapleton, 2003; Hunt, 2006).
These principles are:

- Active user involvement is imperative
- DSDM teams must be empowered to make decisions
- The focus is on frequent delivery of products
- Fitness for business purpose is the essential criterion for acceptance of deliverables
- Iterative and incremental development is necessary to converge on an accurate business solution
- All changes during development are reversible
- Requirements are baselined at a high level
- Testing is integrated throughout the lifecycle
- A collaborative and co-operative approach between all stakeholders is essential

When to use DSDM
DSDM is not a process that fits all projects. It depends on a number of critical issues. If the following six factors seem viable for the project at hand, DSDM is a very useful process. If not, it is wise to chose another.

First of all, the functionality of the user interface should be reasonably visible. Users are involved throughout the process in which they should be capable of verifying that the system is performing according to their wishes. Secondly, it is imperative that all end-user classes can be clearly identified. All user views within the development team should be completely covered. The third factor for the project is that the application should be relatively simple, computationally wise. However, deciding whether or not something is complex or simple is an intricate issue. The fourth factor is that the application should not be potentially large. If it is, it should be possible to split it in smaller subsystems. Otherwise the incremental and iterative approach is not beneficial. The fifth is that the project should consist of strict time-constraints. If not, the system contains too many unnecessary elements and delay is inevitable. The final factor is that the requirements should be flexible and of high level. If requirements are strict and clear, major benefits of DSDM are lost (Hunt, 2006).

3.8 Feature Driven Development

Feature Driven Development (FDD) is yet another process that falls under the agile methods. However, there is one major difference that separates FDD from the other agile processes, that is management of projects. For most organizations, it is imperative for management to have a view on how the project progresses and how this relates to the planning. FDD is a process which is based on short duration projects in which requirements change monthly, if not weekly (Hunt, 2006; Nebulon Pty Ltd.; Palmer & Felsing, 2002). FDD, as opposed to Scrum and XP, and more comparable to RUP, was developed for systems of great scope and scale. It provides a more agile perspective on developing large scale systems (Leffingwell, 2007).

FDD consists of five processes. The process of Feature Driven Development is depicted in figure 3-11 (Palmer & Felsing, 2002). The first step in the process is to create an overall model with the help of Domain Experts. This model is considered to be a domain object model. The next step is to develop a categorized features list (the features are discussed later in this section). When this list is developed, a development plan is drawn up. This plan is based on the features and their prioritization, developed in the previous step. Supported by this plan, small groups of features are taken in design and build iterations. These iterations should take no longer than two weeks, but are often shorter. These iterations are repeated till there are no more features (Palmer & Felsing, 2002).

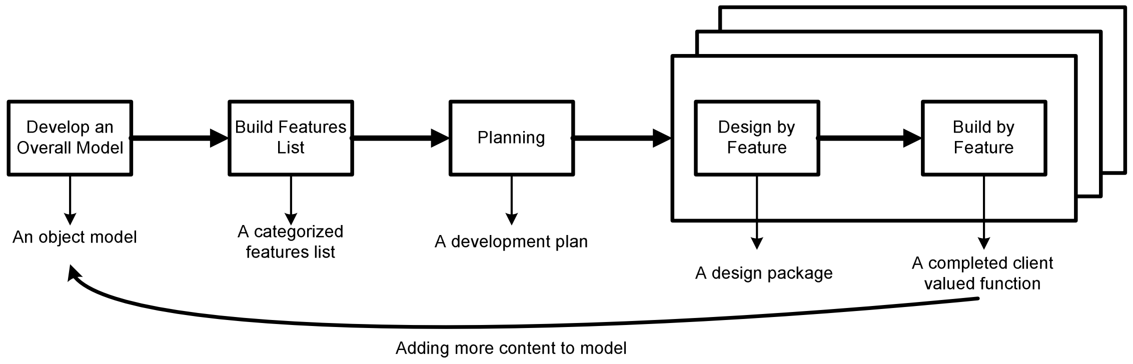


FIGURE 3-11 Feature Driven Development process

A feature
As was mentioned before, FDD is based on features. A feature is a schedulable requirement. These requirements are connected with the activity that will release it (Hunt, 2006). For each feature a cost and priority is given. This provides a mix of units of requirements with units of management. Therefore, management is still able to follow the progress by analyzing the features and not just the requirements.

Features have the following characteristics (Hunt, 2006):

- Small
- According to the systems stakeholders
- They can be collected into business-related groupings
- Prioritized
- Schedulable
- They have an associated cost
- Can be grouped into short iterations

The feature approach provides a clear overview of the total progress of a project. Below an exemplary table is presented in which a summary report is given based on the features. Because units of management are combined with the requirements, progress is managed much easier.

TABLE 3-1 Progress totals summary report

Feature Set	Number of Features	Number not Started	Number in Progress	Number Completed	Percentage Complete
Ordering a New Car	16	9	4	3	30%
Selling a Forecourt Car	7	0	0	7	100%
Selling an Approved Car	7	0	0	7	100%
Trading in an Old Car	15	5	0	10	66.7%
Arranging Financing	13	1	3	9	78%
Total	**58**	**15**	**7**	**36**	**63,9%**
…	**…**	**…**	**…**	**…**	
Complete Project Total	**164**	**64**	**32**	**58**	**45%**

The five processes of FDD

The process depicted in figure 3-11, consists of a number of steps. For each step a certain protocol exists. The description of these steps and tasks corresponding are explained next.

Develop an Overall Model

The Domain Experts perform a high-level walkthrough of the system. This is followed by a detailed domain walkthrough. Based on these domain walkthroughs, small groups create their own model and present their results for peer review and discussion. One of these proposed models (or a combination) is then selected to become the model for that domain area. The main tasks within this process are (Palmer & Felsing, 2002):

1. Form the modeling team
2. Conduct a domain walkthrough
3. Study documents
4. Develop small group models
5. Select one as team model
6. Refine the model in iterations
7. Write notes
8. Keep verifying with the client and users
9. Deliver an object model

Build a Features List

All the features that support the requirements of the client are identified. Each domain is divided into major feature sets. These major feature sets are then broken into feature sets (activities). Each step within that feature set is a feature. This results in a list of hierarchically categorized features. The main tasks are (Palmer & Felsing, 2002):

1. Form the features list team

2. Build the features list
3. Verify continuously with other team members and the Domain Experts
4. Deliver a list of features

Plan by Feature
In this step of the process a development plan is created. The order in which the features are to be implemented are decided in this process. This is based on the feature dependencies, load across the development teams and the complexity of the features. The tasks in this process are (Palmer & Felsing, 2002):
1. Form the Planning Team
2. Determine the sequence of features development
3. Assign feature sets to Chief Programmers
4. Assign classes to developers
5. Deliver a list of features with completion dates

Design by Feature
Before the final phase of the project (building), each feature is designed. The teams develop detailed sequence diagrams for the selected features. The tasks in this phase are (Palmer & Felsing, 2002):
1. Form a feature team
2. Conduct a domain walkthrough
3. Study the documents
4. Develop sequence diagrams
5. Refine object model
6. Write class and method prologue
7. Design inspection
8. Deliver an inspected design package

Build by Feature
Each feature is then build which result in functional elements requested by the client. Tasks corresponding to this phase are (Palmer & Felsing, 2002):
1. Implement classes and methods
2. Conduct a code inspection
3. Unit test
4. Promote to the build
5. Deliver functions

3.9 Usage of processes

In this section an overview is given of the processes in use at different organizations. This provides insight in which processes are known in the market. Unfortunately, only one supplier and of course OIB ING provided information. In an interview with Ron Stal from Cap Gemini, he explained that RUP is a market standard. A different interesting remark was that agile processes are occasionally used. However they do prefer to use package processes for smaller projects. (Appendix III.H). Package processes entail certain predefined elements that usually originates from the RUP. These are used and tailored to fit a certain organization. Cap Gemini has multiple package possibilities. OIB ING, as was explained before, uses XDF (tailored RUP), the Waterfall model, and have used DSDM in the past. Initially it was intended to provide an overview of possible processes that might arise in the near future. Unfortunately, this information was not found. In the interview with Ron Stal, he mentioned that, in his opinion, RUP is extensively used, and will be for the coming years. Table 3-2 provides an overview of the processes used in different organizations.

TABLE 3-2 Usage of processes

Process	Vendors	OIB ING
Waterfall model	X	X
Rational Unified Process	X	X
Spiral Model		
V-model		
eXtreme Programming	X	
Scrum	X	
Dynamic Systems Development Method		X
Feature Driven Development		

3.10 Overview of advantages and disadvantages

In the previous sections, each software development process included in this research, was analyzed. For developing a decision framework, an overview is needed which provides the basic characteristics, advantages and disadvantages of each software development process. This is provided in this section.

Before presenting the summary concerning the characteristics of each software development process, an overview is provided which discusses the differences between two main groups of processes – Traditional processes, such as the Waterfall model and V-model, and the Agile processes, concerning XP, SCRUM, DSDM and FDD.
The differences between the traditional processes and the agile processes are presented in table 3-3.

TABLE 3-3 Differences in traditional and agile development

	Traditional processes	Agile processes
Fundamental Assumption	System is fully specifiable and predictable and is built with extensive planning	System is developed by teams using continuous design, improvement and testing based on rapid feedback
Management Style	Command and control, hierarchy	Collaboration
Knowledge Managements	Explicit	Tacit knowledge
Communication	Formal	Informal
Desired Organizational structure	Aimed at large teams. Bureaucratic and highly formalized	Flexible, cooperative and social action
Quality Control	Heavy planning and strict control	Continuous control

The main difference between these two categories of processes is the view they have on projects and the environment surrounding them. Traditional processes see the world as fully specifiable, in which the environment is stable. Agile processes however, admit that changes always occur, especially in such a dynamic market. These processes are designed in such a way that the specification of elements occurs in a latter stage of the process, in which more knowledge has been gathered concerning the project and the environment. This implies a totally different approach of the organization and management of a project and project team. Traditional projects can be coordinated tightly in which every member of the project team can work individually. Agile processes need a much more flexible project team. The team members should collaborate and review each other. This provides a possibility to share knowledge among each other and their business partners.

As was mentioned before, OIB ING does not use an agile process. The only processes used are the Waterfall model and RUP. However, the differences in traditional and agile processes can also be found between these two processes. According to Jeroen Hendriks, manager within SoDC Call Face NL, the RUP, considers the development of a software project as a journey, in which gradually knowledge is gathered on the entire scope of the project. Floris Zwart, architect within SoDC Call

Face NL/Requirements Delivery, considers RUP to be a process which is more suitable for projects in an unstable and innovative environment. The Waterfall model however, is more suitable for projects which are familiar within the organization and in which the environment is stable and secure.

These differences in software processes and software process categories are also found in the characteristics, advantages and disadvantages of each individual process. This is provided in an overview depicted in table 3-4. The information presented in this table is based on literature research discussed and analyzed in the previous sections and on interviews with OIB ING personnel as well as TU Delft researchers and professors.

TABLE 3-4 Overview of processes with characteristics, advantages and disadvantages

Process	Main characteristics	Advantages	Disadvantages
Waterfall model	• Linear approach • Strict sequential steps • No iteration	• Clarity to the client • Requirements are specified early which provides a better idea of the project • Works well with fixed pricing • Well-known within many organizations (organizational knowledge)	• Changes are hardly possible to implement • Errors found late in the process. This leads to high costs to fix • Errors or needed changes are often neglected • Requirements are specified too early
RUP	• Iterations • Adjustable • Incremental process • Architecture centric	• Focus is on teamwork • Stakeholders are included to provide better overview • Parallel processing • Adjustable to many organizational structures • Errors can be detected early on in the process • Market standard • Heavy tool support	• Heavy complexity • Too many interpretations are possible • Users need intensive training • Separation in phases is outdated • Does not work well with fixed pricing
Spiral Model	• Iterations • Combination of Waterfall and iterations • Frequent reviews • Risk assessment • Incremental process	• Rapid development of prototypes • Risk evaluation • Users can evaluate frequently by reviewing • Errors can be detected early on in the process • Stakeholders are involved in entire process	• Extensive experience in risk assessment is a necessity • Lack of flexibility to adjust the process to the organization
V-model	• Top-down development • Bottom-up implementation • No iteration • Multiple testing phases at the end, each connected to an earlier design step	• Connection of testing with design provides a better overview • Requirements are specified early on which provides a better overview of the project • Works well with fixed pricing	• Changes are hardly possible to implement • Intensive testing at the end, errors are not early detected • Requirements are specified too early
XP	• Very light process • Heavy user involvement • Frequent reviewing • Emphasis on collaboration • Incremental process • Iterative process	• Increases communication within an organization • Enhances team productivity • Users and clients are in control • More freedom is offered to	• More validation is necessary • Management skills need to be high for Project Leader • The final goal might be

	• More suitable for small teams	team members	lost during the process • Very dependent on teamwork • Users and clients intensive involvement is not always feasible • Pair Programming is often very exhausting • Lack of attention to design and architectural issues
Scrum	• Agile process • Used for project management • Applicable for small teams and smaller projects	• Acknowledges that requirements can change during a process • Can be combined with other processes • Communication is improved within teams and with stakeholders • Errors are identified early • Very flexible • Freedom for creativity	• Very dependent on teamwork • Scrum master must be experienced
DSDM	• Agile process • Iterations • Incremental processing • Heavy user involvement • Most suitable for "simple" applications • Reviews throughout the cycles • Tests throughout the cycles • Suitable for smaller projects • Rapid Applications Development	• Frequent product delivery • Errors are detected early in the process • Everything is reversible • Clients and users can easily review and evaluate products throughout the entire process	• All underlying principles need to be feasible within a project • Collaboration among all stakeholders is not always feasible • Requirements are base lined at a high level, which gives little clarity to clients • Very dependent on teamwork
FDD	• Agile process • Design by features • Iterations • Suitable for smaller projects • Between agile and traditional • Frequent delivering of results	• Emphasis on status reporting • Estimations are more effective • Good control of change • Very manageable for an agile process • Provides a clear overview of the progress • Client can clearly see the impact on planning when adding or changing requirements • The process clearly shows whether or not a feature is new or it is a change • Authors have experience in doing projects at banks • Adaptation to fit an	• More validation is necessary • Dependent on teamwork • Dependent on the correct prioritization of "must-haves" and "nice-to-haves" requirements • Client needs to invest time • Possible to get lost in features

		organization is possible • Prioritization of requirements	

4. Characteristics of software projects

The decision framework, which is the objective of this research, consists of information regarding software development processes and information regarding the characteristics of projects. Chapter three presented the necessary information for the framework, regarding the software development processes, by analyzing their characteristics, advantages and disadvantages. This chapter analyzes the projects that are being conducted within OIB ING. The following sub questions are answered: "*Which characteristics of the software projects done at OIB ING are imperative for choosing the most appropriate software development process?*" and "*How are these characteristics weighed by OIB ING personnel?*" To answer these questions case descriptions and lessons learned documents are analyzed. Furthermore, by conducting interviews and distributing a survey, opinions and visions of personnel can be gathered as well.

This chapter is structured in three sections. In section 4.1 the process, within OIB ING, of selecting a software development process for a certain project, and how this ideally should look like is analyzed. Section 4.2 focuses on the characteristics of projects within OIB ING. These characteristics should be found by performing literature research, an analyses of lessons learned documents and by conducting interviews. Finally, in section 4.3 the characteristics found are reviewed by discussing these with OIB ING personnel. A survey is distributed to collect extensive information.

4.1 Decision process

In this section a short overview is provided of how decisions are made at the start of a project at OIB ING. This is imperative to get insight in the elements that play a role during the start-up phase of a project. From these elements, factors can be found which can influence the decision making of which software development process to use. As was explained before, currently OIB ING does not explicitly choose a software development process. However, for every new project a PID (Project Initiation Document) needs to be written. Within this PID an overview is provided which shows which activities should be prepared and when. This project plan is in accordance with the software development process. In this section the process of creating such a plan is discussed.

In an interview with Floris Zwart, an architect within SoDC Call Face NL/Requirements Delivery, who is often involved at the beginning of projects, the start-up process of a project was elucidated (Appendix III.F). The process can be summarized as follows (figure 4-1): A business partner of OIB ING comes with a request for a new software application. This business partner has some ideas of what this application should contain and how it should perform. However, no concrete requests are made. At this point, the Account Manager, as well as the Software Architect continuously meet with the client to get the requirements as clear as possible, as well as the architecture. Based on these meetings, a PID (Project Initiation Document) is written which, among other elements, contains the chosen requirements, the man hours needed for the project, and the project plan. Based on the man hours stated in the PID, a cost agreement is made. As was mentioned before, the agreement is a fixed price. When the PID is finished and agreed upon, the application is outsourced to a supplier.

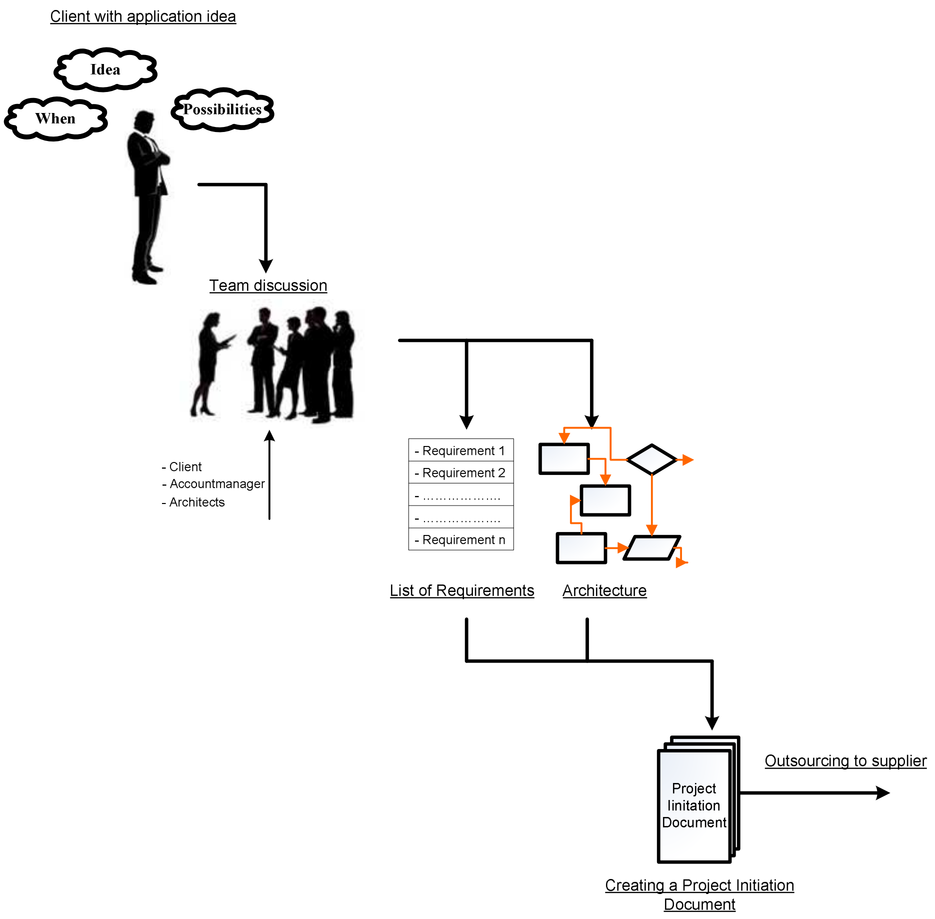


FIGURE 4-1 Business process of conducting projects

The team discussion currently entails discussing requirements and architecture. However, at that point in the process also a discussion regarding the software development process that is suitable needs to take place. By doing this, first of all, an appropriate process is chosen which is in itself a major benefit, secondly, everyone knows when and what is expected and thirdly, the client knows what is expected from him or her.
A different imperative factor in the decision process is the usage of outsourcing. According to Floris Zwart, based on the architecture and functional design, which are created within OIB ING, the solution architecture is developed by a supplier. There is enough feedback from the supplier to make sure the solution architecture is in line with the architecture developed at OIB ING. The main problem is that when a business partner wishes to adjust or add requirements, new agreements need to be made between OIB ING and the supplier. This is because agreements between OIB and the supplier are based on the contracting method “fixed pricing”. As was mentioned before, this method does not easily allow changes. Therefore, a factor such as the relationship with the supplier is imminent.

This decision process should be adjusted to let the use of software development processes be more effective. Instead of making a certain process standard or just keep using the process which is most convenient, a process should be selected based on the characteristics of a project. To redesign this business process, an analysis is needed concerning those characteristics. In the coming sections overviews are provided of imperative factors that could be of influence on this decision process. These characteristics are found by analyzing literature, project evaluations, conducting interviews and distributing a survey.

4.2 Characteristics of projects done by OIB ING

To decide which process fits the project at hand, an overview is needed of the characteristics of such projects. For each characteristic it is then possible to decide whether or not a certain process is suitable. To find these characteristics, a 'golden triangle' is analyzed in subsection 4.2.1. Based on this triangle characteristics are found. These are provided by performing three methods. These three methods are separated by each subsection. Subsection 4.2.2 describes the characteristics found by analyzing literature. The third subsection (4.2.3) describes characteristics found by analyzing evaluation reports which are called lessons learned projects. In subsection 4.2.4 characteristics found by doing interviews are discussed. These interviews were conducted with personnel from SoDC Call Face NL. The final subsection provides a summary of the golden triangle, and the characteristics related to it.

4.2.1 The 'Golden Triangle'

Whether a project has been successful depends on the perception of every stakeholder involved. Often, this perception is based on the answer on three significant questions:

1. Was the project within estimated costs?
2. Was the project delivered on time?
3. Are all the requirements present in the delivered application?

In literature, these three characteristics are also mentioned as an influence on software development processes and software projects. According to Dybå & Dingsøyr (2008) and Reifer (2001), project size (here stated as budget and duration) are imperative for software development. Furthermore Sommerville (2007) explained in his book that the clearness of the requirements and the changes that have occurred during the project can make a software project a success or a failure (Appendix VI). In the CHAOS report of Johnson (1994), the basis for their analyses was also this triangle of factors (Johnson, 1994). In an interview, Floris Zwart indicated as well that three elements are critical for every project and every PID – budget, time and requirements. Furthermore, his opinion was that these characteristics are influenced by software development processes. However, these characteristics also influence the suitability of the software development processes (Appendix III.F). Not only do the factors and processes influence each other, the characteristics individually influence each other as well. Figure 4.2 presents these findings.

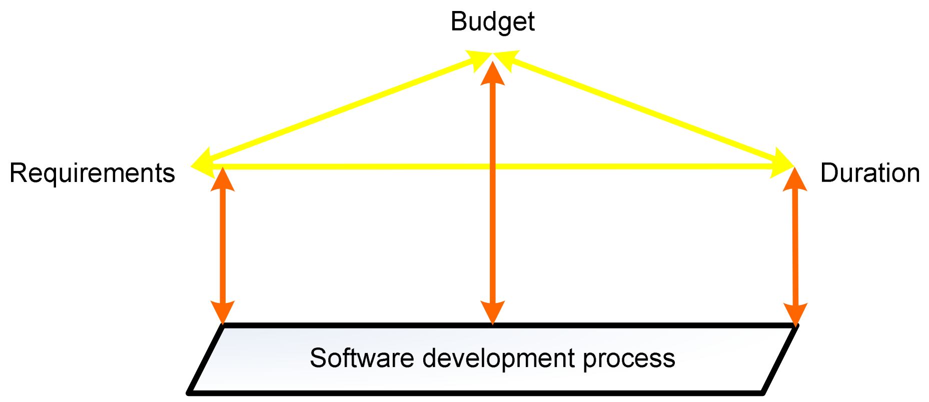


FIGURE 4-2 "Golden Triangle"

It is clear that this triangle is the base of all characteristics that influence the suitability of software development processes. As is depicted in figure 4.2, the software development process chosen and the golden triangle influence each other. If a process is very suitable, the budget and time schedule are within the estimations made, and all the necessary requirements are included. However, the suitability of the process depends on, for example, how clear the requirements are, or how big the project is.

From these three factors, significant characteristics can be found which influence the suitability of the software development process. By researching literature, lessons learned documents and performing interviews, it is possible to derive characteristics from the three major factors of the golden triangle. This is presented in the next subsections. Every subsection is allocated to each specific method of research – literature, lessons learned documents and interviews.

4.2.2 Characteristics found from literature

Software development processes are discussed intensively in literature. This is also presented in chapter three. Unfortunately, the connection between projects and the suitability of a process is hardly made. However, this does not mean that characteristics can not be found in literature. When analyzing the software development processes in chapter three, an overview of characteristics, advantages and disadvantages was made. The characteristic of a process can also imply that this characteristic should be present in the project. For example, agile processes are in literature described as processes suitable for projects in which the team size is small. Therefore, the team size of a project has influence on the suitability of software development processes. This and other characteristics found in literature are discussed in this subsection. The characteristics below are presented for each major factor individually. If characteristics are found which are not connected to any of these factors, they are stated under "other characteristics of interest". In subsection 4.2.5 a summary is given of all characteristics. An overview of sources found for each characteristic is presented in Appendix VI.

Characteristics influencing Project Budget:
Risk Clearness
Every project is confronted with risks. The degree of risks differs significantly. For projects based on previous executed projects, the risks are reduced notably. However, for projects in which entire new concepts are applied, risks can definitely have a significant impact. Software development processes should pay attention to the risks. Some processes do this more effective than others. In literature, risk mitigation is often discussed as a characteristic of a software development process. For example, the Waterfall model is in literature criticized by its lack of risk mitigation (Royce, 1970; Sommerville, 2007). The Spiral model is praised for its risk mitigation (Boehm, 1988). If high risks exist, and a software development process is chosen which is not suitable for handling this, costs and duration of the project will increase. This characteristic can therefore also be derived from the factor *Project Duration.* This is also mentioned in interviews.

Characteristics influencing Project Duration:
Risk Clearness

Characteristics influencing Requirements
Requirements Maturity
The characteristic "Requirements Maturity" entails how sure the business partner is concerning his/her wishes. When a business partner comes to OIB ING with a new project, a discussion takes place regarding the requirements. Some business partners know exactly what they want. However this is rare. Usually, the business partner has an idea of what he or she wants, although, no clear requirements are stated. This creates uncertainty for the entire project. Furthermore, when requirements are not mature, changes can easily occur while building the actual system. This often results in over budgeted and very long projects. Vliet (2008) states that this influences software development processes and that incremental development is very viable when requirements are unclear.

Changes Expected

This characteristic could be linked with the previous: “Requirements Maturity”. It is possible that a business partner is not sure concerning the requirements. However, based on previous projects and the relationship with this particular partner, it is possible to state that although the requirements are not mature, when the explicit requirements are finally found, changes are not likely to occur. Based on previous experiences it is also possible to state that, although the requirements are very mature, it is assumed that the requirements are most likely to be adjusted. Dybå and Dingsøyr (2008) state in their paper that this characteristic is indeed present in many software projects. According to them, agile methods are more suitable to cope with possible changes (see chapter three).

Client’s Commitment
The commitment of the business partner entails how involved the client is with the project and its progress. It is possible that a client just wants to state the requirements. However, some business partners find it more satisfying to be involved in the entire progress. This commitment can be very important for projects (Sommerville, 2007). Some software development processes are dependent on a strong commitment. Especially agile processes demand the client to be on site. If a client is not committed to this particular project, agile processes are less viable (Hunt, 2006).

Other characteristics of interest
Team Size
As was mentioned before, some characteristics are stated in literature as characteristics of software development processes (see Chapter three). This is the case for Team Size. In literature it is mentioned that agile processes are especially suitable for projects in which 6-8 team members are involved (Leffingwell, 2007; Sommerville, 2007; Runeson & Greberg, 2004). Team Size is definitely a characteristic of a software project which influences the suitability of software development processes.

Team Relationship
This is always important. This characteristic entails all possible team members, i.e. members from other departments, from the supplier, from OIB ING and the client. Regarding the other departments, it is not only possible to have an uncomfortable relationship with a certain department, it is also possible that certain departments are geographically scattered. If one of these issues plays a role, more documentation is needed during the process. By using more documentation, face to face contact is of less importance. Processes such as the Waterfall model or the V-model are both based on extensive documentation (Leffingwell, 2007; Royce, 1970). Regarding the supplier, important suppliers for OIB ING are Cap Gemini, Logica, Getronics, Ordina and Atos Origin. With most of these suppliers OIB ING has a significant history of cooperation. However, other suppliers are contracted as well. It is possible that the relationship between the supplier and OIB ING is not as smooth as it used to be.

4.2.3 Characteristics found from lessons learned documents
Some departments within OIB ING evaluate the project done. From these evaluation, lessons learned documents are written. In these documents it is stated what went right, what went average, what went wrong and what should be improved. These documents help in finding characteristics of projects. Characteristics found in these documents are stated below.

Characteristics influencing Project Budget:
Scope Clearness
The scope of a project is imperative. If the scope is unclear, team members can get lost in the project and time and budget overruns are imminent. In many lessons learned documents, the clearness of the scope was mentioned as point of improvement. Often, when the scope is unclear, requirements are likely to change. As was mentioned before, some processes handle these changes significantly

better than others. This characteristic can also be derived from the factor *Project Duration.* If the scope is not clear, delays are imminent and project duration will be longer.

Characteristics influencing Project Duration:
Scope Clearness

Characteristics influencing Requirements
The characteristics stated below are all found while searching literature. These are explained in subsection 4.2.2. The fact that these are also mentioned in lessons learned documents shows their relevance.
Client's Commitment, Requirements Maturity, Changes Expected

Other characteristics of interest
Team Size
This characteristic is already explained in subsection 4.2.2, as it was found in the literature analysis.

Departments Influence
Within a large organization such as ING, multiple departments might play a role in a large project. Within OIB ING the following departments might be involved in a project: Payments, Channels, Customer Data & Agreement Management, Consumer Products, COO OIB Wholesale, Risk & IT Security, Strategy and Support and Infrastructure Services. It is imperative to know how much influence these departments have in a particular project. For example, if Payments is significantly involved in the project, dependency exists on their planning. Usually, they apply the Waterfall model (as do many other departments). This process is used because they are focused on main-frame projects. If Channels decide to use RUP for example, this dependency might become an issue, especially concerning the integration.

4.2.2 Characteristics found by performing interviews
Besides analyzing written reports and literature, interviews with personnel are imperative. By discussing tacit knowledge, opinions and practice related issues are uncovered which are hardly described elsewhere. In this subsection, characteristics are discussed which are found by conducting interviews. These interviews are presented in Appendix III. Some characteristics discussed here are already described earlier. However, these are still mentioned, thereby emphasizing their importance for this research.

Characteristics influencing Budget:
Risk Clearness
This characteristic is already explained in subsection 4.2.2. This characteristic was found, besides in interviews, in literature as well. This explicitly shows the relevance of this characteristic for the framework.

Environmental Stability
Environmental stability entails the development environment of the software development within the organization. Are there new techniques used? Are new suppliers used? How is the development infrastructure? These and other questions have to be answered regarding this characteristic. If the development environment is not stable, more changes and risks can be expected.

Stakeholders Flexibility
OIB ING has many different possible business partners. Furthermore, each business partner is managed by a number of project managers. Each individual project manager has its preference regarding the phasing of the project. This is imperative to know regarding software development

processes. If a stakeholder prefers a traditional method of processing, an agile process is not applicable. Agile processes require the client to be on site for almost the entire project. If a stakeholders wants to perform the project in a traditional manner, he/she prefers a situation in which face to face time is limited and where documentation is the method of communicating.

Characteristics influencing Time Schedule:

Risk Clearness

This characteristic is already explained in subsection 4.2.2. This characteristic was found, besides in interviews, in literature as well. This explicitly shows the relevance of this characteristic for the framework.

Application's Familiarity

Projects done at OIB ING can differ in difficulty. A major factor for this is whether or not the application that is requested by the client is already familiar within OIB ING. It is possible that an already existing application needs to be adjusted in any way. These projects are already somewhat further on the experience lifecycle. However, if an application is completely new, they have to start from the beginning. How familiar the project is, influences the suitability of software development processes. For example, agile processes handle uncertainty better than the traditional processes.

Environmental Stability, Stakeholders Flexibility

These are explained above. They can both be derived from the factors project budget as well as project duration.

Characteristics influencing Requirements

These characteristics are already explained in subsection 4.2.2.

Client's Commitment, Requirements Maturity, Changes Expected, Stakeholders Flexibility

Method of contracting

There are two main methods of contracting, Time & Material and Fixed Pricing. Time & Material entails that the final price is simply dependent on the costs of the project, which is based on hours and material. The Fixed Pricing method states that, based on the predefined requirements, a price is negotiated between the supplier and the client. Everything above this price is allocated to the supplier. The difference between these two methods is that Time & Material is more flexible than Fixed Pricing. Fixed Pricing is better suitable when all the requirements are determined and frozen. Currently, many, or even all, organizations in the financial sector apply the Fixed Pricing method. According to these organizations, this method provides predictability and reliability. Unfortunately, software development processes are dependent on the applied contracting method. Agile processes for example do not comply with Fixed Pricing. Agile processes require flexibility which means that requirements should not be frozen.

Other characteristics of interest

Team Relationship

This characteristic is already explained in subsection 4.2.2.

4.2.5 Summary of currently found characteristics

In the subsections above characteristics were found by analyzing literature and lessons learned documents and by conducting interviews. These characteristics were found by connecting them to the golden triangle of factors, Budget, Duration and Requirements. In table 4.1 an overview is provided of all characteristics and their connection to the factors just mentioned.

TABLE 4-1 Overview of Characteristics

Project Budget	Project Duration	Requirements	Other
Risk Clearness	Risk Clearness	Requirements Maturity	Team Size
Scope Clearness	Scope Clearness	Changes Expected	Team Relationship
Environmental Stability	Applications Familiarity	Client's Commitment	Departments Influence
Stakeholders Flexibility	Environmental Stability	Stakeholders Flexibility	
	Stakeholders Flexibility	Method of Contracting	

In Figure 4-3 an overview is depicted of all the previous subsections. As was explained before, the three major factors (Golden Triangle) influence the software development process and software development process influences them. The factors influence each other as well. From these factors characteristics were derived by analyzing literature, lessons learned documents and by conducting interviews. These characteristics influence only the software development process. Therefore, the characteristics and the factors both influence the suitability of a software development process.

However, the interviews were performed only with a limited amount of managers. In the next section these characteristics are reviewed by conducting a survey among personnel. This reaches a wider scope and therefore provides more information.

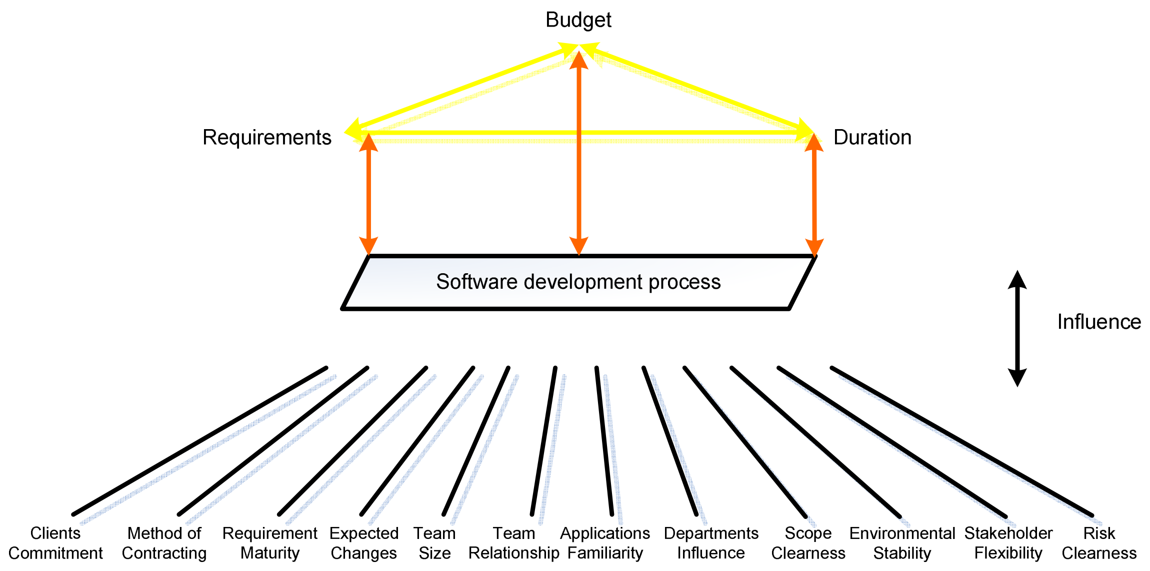


FIGURE 4-3 Final characteristics for decision framework

4.3 Review of characteristics

Opinions regarding software development processes differ greatly, not only within literature, but in practice as well. A number of interviews within OIB ING were conducted to get an overview of how software development processes were used in practice, and what the consequences were. However, for finding characteristics of projects which influence the suitability of software development processes, literature and interviews did not suffice. This particular part of the research was barely discussed in literature. Furthermore, in practice, very differentiating opinions came forth. Therefore, a survey was distributed among personnel (Appendix IV). By selecting employees in different sections of OIB ING, the survey provided extensive information regarding their opinions about the use of software development processes, characteristics that influence the suitability of these processes and how important they find each of these characteristics. In this subsection the opinions of personnel regarding the characteristics of projects that influence the suitability of software development processes are presented and discussed. The summary of results of the entire survey is presented in Appendix V. The questions asked hold a subjective, open and qualitative character. This is chosen to obtain representative opinions regarding the characteristics of software projects.

4.3.1 Changes in characteristics

In the survey, thirteen characteristics of software projects were presented. These characteristics were based on Lessons Learned Documents and Interviews. Every employee had to state whether or not he or she found each characteristic to be an influence on the suitability of a software development process. Furthermore, if he or she found that characteristics were missing, they could present them as well. Based on the results of the survey (Appendix V) and a review of these results, a number of changes have occurred in the list of characteristics. Each change is discussed next.

Project Budget & Project Duration changed into Project Size
In the former list, two characteristics were applied to indicate how large a particular project was. These were “Budget” and “Time Schedule”. However, through the analysis of the surveys, it became clear that some employees did not make this association. Some found budget not important but made a remark that the size of the project should be included. Others, who did make the connection between budget and size of the project, stated that it is a very important characteristic. To prevent any misinterpretation regarding these characteristics, “Budget” and “Time Schedule” are both replaced by “Project Size”. “Project Size” can be expressed in multiple ways. Not only the amount of money for the project or the amount of time available are possible expressions, but also the number of function points or man hours.

Merging of Requirements Maturity and the Changes Expected
In the survey, for both characteristics, almost equal weights were given. The employees that gave comments actually copied the comment of “Maturity of Requirements” to the characteristic “the Number of Changes Expected in the Requirements”. Thus, these two characteristics are correlated in such an extensive manner, that it would be best to combine the two. This characteristic is now noted as “Maturity of Requirements”.

Departments Influence deleted
Three employees responded that the characteristic “the Role of Other Departments” does not play a role when selecting a software development process. Furthermore, this characteristic scores lowest on importance in comparison to all the other characteristics. As was described in the previous subsection, this characteristic was included because when multiple departments were involved, integration and teamwork is more complex. This has especially consequences for the suitability of agile processes. Danny Wijnand explained that when other departments are involved, indeed the integration benefits might be lost. However, the benefits for the individual department that applies an agile process or RUP are still significant (Appendix III.I). Therefore, this characteristic is not significant enough to base the choice of process on.

The characteristic Outsourcing is added
A number of respondents mentioned that the factor outsourcing should be included. This characteristic was not taken into account because OIB ING always applies outsourcing. However, it still has influence on the suitability of certain processes. The agile processes are less viable when outsourcing is applied. Even RUP is less suitable. When there is no outsourcing, the Waterfall model and the V-model are less suitable because of their high overhead and documentation. Because it has indeed influence on the processes, this characteristic is included in the decision framework.

Applications Familiarity deleted

This characteristic is complex. Based on literature and interviews, it can be concluded that there is no unity concerning the influence of this factor. Some say that when an application is completely new, an iterative approach is most viable. However, others say that, when it is very new, clear steps need to be taken first to get enough documented information before starting the iterative approach. These two opinions are both very feasible. Furthermore, some employees from OIB ING stated that they do not think that this characteristic has any influence. Based on these two situations, it was decided to remove this characteristic.

4.3.2 Other interesting results

Besides these changes, a number of other possible changes were mentioned by the employees. Based on a personal review of these recommendations, it was chosen not to include these changes. The most significant recommended changes (which are not applied) are discussed to support the decision made.

Experience

One employee remarked that the experience the team/organization/stakeholder(s) has with the software development processes is of major importance. This is obviously true. It is not recommended to apply a process in which none of the team members has experience. However, with this decision framework, an organization should get a clear view on which processes should be used in which situations. If experience would be included in the framework, the processes that would result as suitable would most likely be the Waterfall model or RUP, i.e. the processes which are used currently at OIB ING. By removing the experience factor, the organization is confronted with the fact that, for a particular project, a certain process is more suitable than the processes presently used. This should encourage management to include the process in its development centre, and start training the employees. This is especially the case if this process is suitable in multiple executed projects.

Client's Commitment

Personnel within OIB ING could not agree on the importance of the characteristic "Client's Commitment". Some stated that it was very important and others found that it should not play any role. It is decided to still include this characteristic in the framework. For agile processes it is of the utmost importance that everyone is involved. Even for RUP, the customer is expected to play a significant role. If the customer is not at all involved in the project, it is much more difficult to work iteratively. Processes in which documentation is used extensively are then much more applicable.

Method of Contracting

Three employees remarked that the characteristic "Method of Contracting" does not have any influence on the suitability of software development processes. However, those statements are disputable. Agile processes are very dependent on incremental and iterative development. These processes approach the development of software as a journey in which every step results in more information. When Fixed Pricing is used, the development team is not allowed to make this journey. They have to decide at the beginning what the end product will be. Therefore, agile processes are less suitable when this method of contracting is used. This characteristic definitely influences suitability and shall be included in the decision framework.

Finally, when analyzing the responses of the employees, other interesting issues, which do not concern the characteristics, were found as well. An overview is provided next regarding these interesting issues and remarks.

Applying software development processes now and in the future

In the survey various open questions were asked to the employees (Appendix IV). Two questions in particular are very interesting for the research. The first question concerns software development now at OIB ING. The employees were asked to state whether or not the application of software development processes for projects is going well. Eight out of ten responses were "No". Employees at OIB ING are not satisfied with how processes are being used. This supports the problem description stated in chapter two. The second question that is very interesting is whether or not the employees want to have multiple processes to choose from when starting a particular project or if they rather have a standard process which is adjusted according to the project. Eight out of ten employees responded to this question clearly. Four responded to prefer multiple processes and the other four preferred a standardized process. In this thesis the possibility of multiple processes is researched. The standardization possibility is discussed in chapter nine.

Interesting remarks from employees
Within the survey some remarks were made which are very interesting for this research. For example, some employees responded on the questions which process they prefer to apply within OIB ING. Danny Wijnand, a Delivery Unit Manager within OIB ING, answered on this question: "the Rational Unified Process has my preference, since it is a market standard. Agile processes have my preference when something very new has to be made in which many uncertainties are present." Roel Noback, an IT Architect, responded on the same question very similar: "RUP has certainly my preference. RUP can be adjusted to both traditional methods and agile methods, depending on the situation. I had enough bad experiences in using the Waterfall model, especially while applying outsourcing. The risk mitigation aspect of RUP is a major value for these types of projects."
Another question in the survey was whether or not the employee found that software development processes are being applied effectively within OIB ING. Roel Noback, commented that "within OIB ING, if RUP is applied, it is applied in name only. When the Waterfall model is applied, they make a big mistake." Wouter Blokdijk responded that "when pressure from higher management is felt, the process part of the project is lost. At that point more focus lies on the final product than on the process of getting there."
As was explained before, some employees prefer to have multiple processes present, from which one can be chosen for a particular project. Others preferred a standardized process which is adjustable. Ed Schimmer, Senior Business Analyst, preferred multiple processes: "Multiple processes should be present. In many situations, traditional methods such as the Waterfall method shall be applied, for example at Payments systems. RUP should be promoted more, especially for projects in which outsourcing is applied. For smaller projects, short cyclic development processes are interesting, such as DSDM." Floris Zwart, senior Architect at OIB ING, responded differently. "From the governance point of view, one process is preferred. One standardized process increases transparency concerning budgets, outsourcing, deliverables etc." All the interesting remarks from employees stated in the survey are presented in Appendix V.

4.3.3 Final list of characteristics
In this section some changes have been made to the list of characteristics presented in subsection 4.2.5. The final list consists of eleven characteristics which influence the suitability of software development processes. These characteristics are used within the decision support framework, which is designed in chapter five. The list of characteristics is presented in figure 4-4.

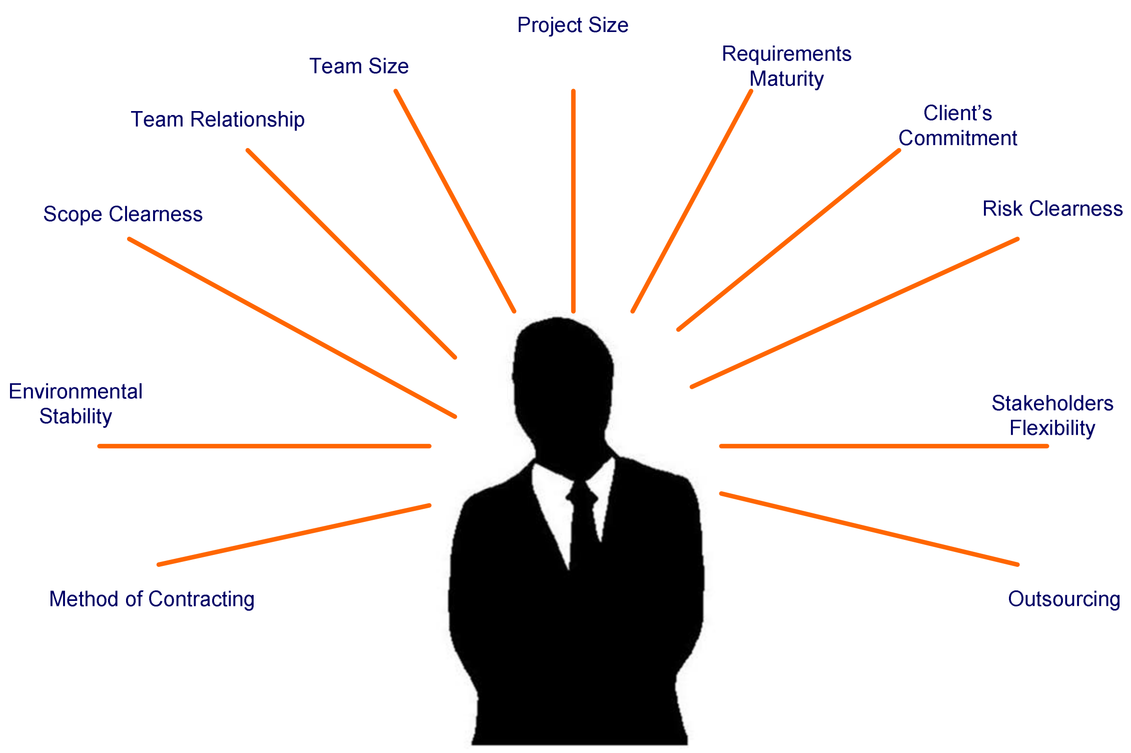


FIGURE 4-4 Final characteristics for decision framework

4.3.4 Weights given to characteristics

In Appendix IV the survey distributed among OIB ING personnel is presented. Appendix V presents the results. In this subsection the importance of the characteristics found in subsection 4.3.3 are discussed. Personnel that responded on the survey had to answer how they score the importance of each characteristic on a scale from 1 to 5, 1 being the lowest, and 5 the highest. Table 1 presents the average round off weights of the list of characteristics presented in subsection 4.2.3.

TABLE 4-2 Weights of Characteristics

Characteristics	**Average**
Project Size	4,1
Team Size	2,9
Requirements Maturity	4.4
Team Relationship	3
Stakeholders Flexibility	3
Clients Commitment	3.3
Scope Clearness	3,8
Risk Clearness	4.3
Environment Stability	2.8
Method of Contracting	2.9
Outsourcing	3.5

A significant result is that the characteristic "Maturity of Requirements" is weighed most important with a score of 4.4. Number two and three of the most important characteristics are "Risk Clearness" and "Project Size". The lowest scoring characteristic is "Stability of the Environment" with a score of 2.8. These scores are taken into account when including the final weights into the decision framework. However, it is possible to change these weights for each project. For certain projects, project groups, or situational factors, some characteristics are of more, or less, importance than decided in this research. The final weights are presented in chapter 6.

5. DESIGNING THE DECISION FRAMEWORK

In the previous chapters information is gathered to be applied in the framework for OIB ING. In chapter three research was conducted concerning the significant characteristics of the software development processes. In chapter four analysis was done of imminent characteristics of projects conducted at OIB ING. By mapping the suitability of each process (based on their main characteristics) on the characteristics of projects, a basis for the framework is made. The mapping of the processes are conducted in section 5.2 and 5.3 of this chapter. Section 5.1 explains the basic design of the decision support framework. This chapter answers the research sub question: *"How should this framework be designed to be suitable for OIB ING?"*

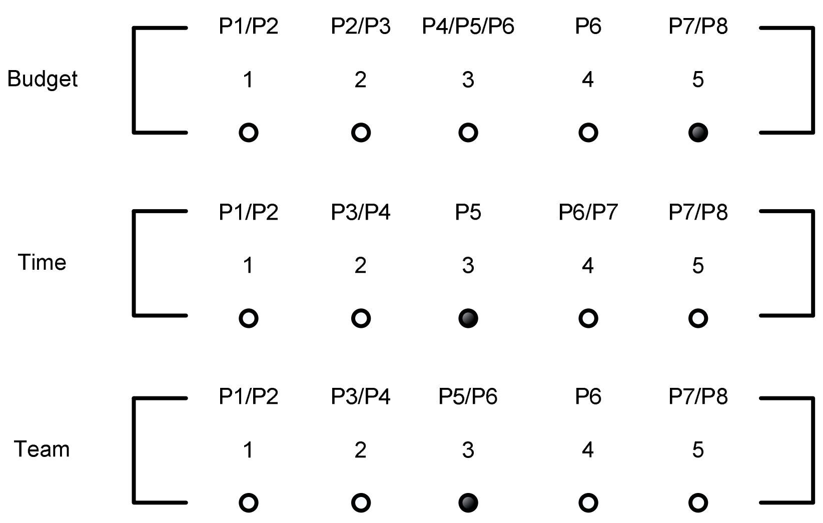
At the end of this chapter a framework is presented to assist project managers at OIB ING in selecting a suitable software development process based on the characteristics of the project. The first step is to analyze whether or not the software development processes are useful when a particular characteristic reaches a certain scale. This is presented in a table in section 5.2. Section 5.3 is similar to section 5.2. However, in section 5.3 a suitability factor is given whenever a software development process fits a certain scale. Some processes are possible when a particular characteristic reaches a certain scale, however, it might be less suitable than other processes. Finally in section three the mapping of the processes is applied in the framework. Screenshots, clarification, design choices and conclusions are given.

5.1 Design of the framework

This research is focused on designing a decision framework which can assist project managers at OIB ING in selecting an appropriate software development process for the project at hand. This decision framework should be based on the main element on which the effectiveness of a process is dependent, i.e. the characteristics of software projects. This section explains the basic design of the decision framework, which choices are made and what this entails for the matching of software processes with the characteristics of projects.

The previous chapter discussed the significant characteristics of projects that play a role when selecting a software development process. This framework should, by indicating the scale of each characteristic, provide an answer of which process or processes should be used concerning this particular project. The design chosen for this particular framework is a questionnaire form. For each characteristic, the project manager can select the appropriate scale. This scale goes from 1 to 5.

Figure 5-1 presents the first rough sketch of the proposed framework. For each characteristic, the project manager can select the appropriate scale. This scale goes from 1 to 5. For example, if the budget is very low, the project manager selects, in the questionnaire, the number 1. If the budget is very high, the 5th scale is selected. On each scale, certain processes are linked which are suitable for that particular scale. For each characteristic a weight can be given. By multiplying the processes resulted from each characteristic with the weights of each characteristic, the framework provides a certain score, for each process. The process with the highest score should be selected for that particular project.



Formula: 1 * (P7 + P8) + 1 * P5 + 1 * (P7 + P8)

Formula with Weights: ((1 * (P7 + P8)) * 0,7) + ((1 * P5) * 0,6) + ((1 * (P5 + P6)) * 0,8)

Best Suitable Process: P5 = FDD

Legenda:		
P1	-	Waterfall
P2	-	V-model
P3	-	Spiral
P4	-	The RUP
P5	-	FDD
P6	-	DSDM
P7	-	Scrum
P8	-	XP

FIGURE 5-1 First sketch of possible decision framework

5.2 Mapping the processes to the characteristics

This section focuses on the main issue of this thesis. The significant characteristics of each process and the characteristics of projects conducted at OIB ING were found in the previous chapters. For the framework, these two information sources need to be combined. In this section the software development processes are matched with the characteristics of software projects. By analyzing for each characteristic whether or not the process fits, a conclusion can be made based on all the individual scores combined.

The characteristics of software projects are in the framework scaled from 1 till 5, 1 being very low and 5 being very high. On each scale a decision needs to be made whether or not a process fits this. A certain process might only be useful when a characteristic, for example, the characteristic team size, is very low and low. This could entail agile processes. Agile processes have as characteristic being suitable for projects with a small project group. When the team size is very high, an agile process is less suitable. For each characteristic the processes are matched on the five scales. This overview is presented in tables 5-1 and 5-2. Table 5-1 represents the mapping of the software development processes Waterfall model, V-model, Spiral model and the RUP. Table 5-2 consists of the processes DSDM, XP, Scrum and FDD.

The decision whether or not a process fits a particular scale is based on theoretical information regarding this process and on conducted interviews. During this entire research opinions in literature, and opinions in interviews were gathered. At this point in time understanding concerning the subject is very scarce. For example, the creators of FDD mention that agile processes are suitable not just for small project groups, but for larger groups as well (Nebulon Pty Ltd., 2009). However, Dyb and Dingsøyr (2008) and Erickson et al (2005) state that agile process are limited to smaller projects with a small amount of team members. This difference in opinion does not only limits itself to this characteristic or these processes. Every author and every employee at OIB ING has its own opinion regarding software development. However, in this research a framework had to be made. Therefore, certain decisions needed to be made regarding the suitability of software development processes. The following tables present an overview of when a certain process is suitable regarding the project's characteristics. These choices are discussable and are not to be seen as fact. These are choices made based on own expertise, interviews and on literature research.

TABLE 5-1 Process categorization part 1

	Waterfall	V-model	Spiral	RUP
Project Size	High, Very High	High, Very High	High, Very High	Low, Medium, High, Very High
Team Size	Large, Very Large	Large, Very Large	Medium, Large	Small, Medium, Large
Requirements Maturity	Very Mature	Very Mature	Mature, Very Mature	Little Mature, Medium Mature, Mature, Very Mature
Team Relationship	Very Dire, Dire, Medium, Good	Very Dire, Dire, Medium, Good	Dire, Medium, Good, Very Good	Dire, Medium, Good, Very Good
Client's Commitment	Very Little, Little, Medium, High, Very High	Very Little, Little, Medium, High, Very High	Little, Medium, High, Very High	Little, Medium, High, Very High
Scope clearness	Clear, Very Clear	Clear, Very Clear	Medium, Clear, Very Clear	Little Clear, Medium, Clear, Very Clear
Risk clearness	Clear, Very Clear	Clear, Very Clear	Not Clear, Little Clear, Medium, Clear, Very Clear	Little Clear, Medium, Clear, Very Clear
Environmental Stability	Stable, Very Stable	Stable, Very Stable	Little Stable, Medium, Stable, Very Stable	Little Stable, Medium, Stable, Very Stable
Stakeholders Flexibility	Not Flexible, Little Flexible, Medium, Flexible	Not Flexible, Little Flexible, Medium, Flexible	Medium, Flexible, Very Flexible	Little Flexible, Medium, Flexible, Very Flexible
Method of contracting	Fixed	Fixed	Fixed	Fixed Pricing, Time & Material
Outsourcing	No, Yes	No, Yes	No, Yes	No, Yes

TABLE 5-2 Process categorization part 2

	DSDM	XP	Scrum	FDD
Project Size	Very Low, Low	Very Low	Very Low, Low	Very Low, Low, Medium
Team Size	Very Small, Small	Very Small	Very Small, Small	Very Small, Small, Medium
Requirements Maturity	Little Mature, Medium, Mature, Very Mature	Not Mature, Little Mature, Medium, Mature, Very Mature	Not Mature, Little Mature, Medium, Mature, Very Mature	Little Mature, Medium Mature, Mature, Very Mature
Team Relationship	Good, Very Good	Very Good	Very Good, Good	Good, Very Good
Client's Commitment	High, Very High	Very High	Very High	High, Very High
Scope clearness	Not Clear, Little Clear, Medium, Clear, Very Clear	Not Clear, Little Clear, Medium, Clear, Very Clear	Not Clear, Little Clear, Medium, Clear, Very Clear	Not Clear, Little Clear, Medium, Clear, Very Clear
Risk clearness	Not Clear, Little Clear, Medium,	Not Clear, Little Clear, Medium,	Not Clear, Little Clear, Medium,	Not Clear, Little Clear, Medium,

	Clear, Very Clear	Clear, Very Clear	Clear, Very Clear	Clear, Very Clear
Environmental Stability	Not Stable, Little Stable, Medium, Stable, Very Stable	Not Stable, Little Stable, Medium, Stable, Very Stable	Not Stable, Little Stable, Medium, Stable, Very Stable	Not Stable, Little Stable, Medium, Stable, Very Stable
Stakeholders Flexibility	Flexible, Very Flexible	Very Flexible	Very Flexible	Flexible, Very Flexible
Method of contracting	Time & Material	Time & Material	Time & Material	Fixed Pricing, Time & Material
Outsourcing	No	No	No	No, Yes

The main conclusion based on these tables is that smaller projects, low budget, small time schedule, and small team size fit agile processes better than the more traditional processes. Furthermore, agile processes are more capable of handling projects surrounded by an uncertain and risky environment. The RUP process, now together with the Waterfall model in use at OIB ING, fits regularly all the moderate scales. For example, the RUP is not suitable for very small projects and very large projects, however, it does fit everything in between.

Based on these tables it is possible to create a basic framework in which, when selecting for each characteristic which scale the project has, the outcome will be a suitable software development process. However, these tables only represent whether or not a process is possible when a factor has a certain scale. Multiple processes are matched on the same scale of a characteristic. By adding a suitability factor to these matches more dept is created. How effective each process is on each particular scale, is presented next in section 5.3

5.3 Separation of processes by suitability

In the previous section the software development processes were mapped on the projects characteristics. This is presented in tables 5-1 and 5-2. For example, concerning the characteristic Team Size, Scrum is only applicable when the team size is very small. However, the RUP can be used when the team size is small, medium or big. In these tables no separation was made between the processes that matched a certain scale. If two processes can both be applied when the team size is medium, this does not necessarily mean they are equally suitable. In this section the processes that fit a particular scale, are separated by suitability.

In the overview below (table 5-3), this separation is presented. These separations are based on theoretical information (information regarding the characteristics, advantages and disadvantages of software development processes presented in chapter three) and interviews (Appendix III). As is depicted below, a suitability factor is given to each colour. This means that when a process is suitable on a certain scale, it gets the colour green. However, when a process fits the scale but is less suitable than other processes, they get a lower factor, for example yellow or orange. When the colour green is given, the score (which is dependent on the weight given) is multiplied by alpha. When the colour yellow is given the score is multiplied by beta and orange presents a multiplication of gamma. These three scores can be set by the user. In this research the respective scores given to these suitability factors are 1, 0.75 and 0.5.

For example, in the first table the characteristic budget is presented. When the budget is very low, Scrum or XP, which are very light processes, are both very suitable. DSDM and FDD, which are a little more complex, are suitable, but not as much as Scrum or XP. Therefore they get the colour yellow. The sequence of names do not represent any preference, only the colours do. Scrum and XP in the first scale of budget are therefore equally suitable. This manner of separation is conducted on all scales and on all characteristics mentioned in tables 5-1 and 5-2. The choices made here concerning the suitability of a process is based on the characteristics of processes found in literature and described in chapter 3, and on interviews and opinions. However, because there is no information regarding the suitability of processes on certain projects, these choices are discussable and are not to be seen as fact.

TABLE 5-3 Suitability of processes on characteristics

α	🟩 *1
β	🟨 *0,75
γ	🟧 *0,5

Suitability Factor

	🟩 Scrum	🟩 FDD	🟩 RUP	🟩 RUP	🟨 Waterfall
	🟩 XP	🟩 RUP	🟨 FDD	🟨 Waterfall	🟨 V-model
	🟨 DSDM	🟨 DSDM	…	🟧 V-model	🟨 RUP
	🟨 FDD	🟧 Scrum	…	🟧 Spiral	🟧 Spiral
Project Size	1	2	3	4	5

	🟩 Scrum	🟩 FDD	🟩 RUP	🟩 RUP	🟩 Waterfall
	🟩 XP	🟩 DSDM	🟩 FDD	🟨 V-model	🟩 V-model
	🟨 FDD	🟨 RUP	🟨 Spiral	🟨 Waterfall	…
	🟨 DSDM	🟧 Scrum	…	🟧 Spiral	…
Team Size	1	2	3	4	5

The suitability of a process often depends on the extent of the project. Whether the size of a project is expressed in function points, deliverables or man hours is left open in this characteristic. Only the scale of the size needs to be stated.

When a project is very big, lightweight processes such as the agile processes, lose their suitability. For large projects often more documentation is needed because face-time is limited. Therefore, for the characteristic project size, as well as team size, the agile processes, such as Scrum and XP are only suitable on smaller scales and the Waterfall process or the V-model only for the larger scales. This is also confirmed by Runeson & Greberg (Runeson & Greberg, 2004).

	🟩 Scrum	🟩 FDD	🟩 RUP	🟩 RUP	🟩 RUP
	🟩 XP	🟩 DSDM	🟩 FDD	🟩 FDD	🟩 V-model
	…	🟩 Scrum	🟩 DSDM	🟩 Spiral	🟩 Waterfall
	…	🟩 XP	🟩 XP	🟩 DSDM	🟩 Spiral
	…	🟨 RUP	🟩 Scrum	🟩 Scrum	🟩 FDD
	…	…	…	🟩 XP	🟩 XP
	…	…	…		🟩 Scrum
	…	…	…		🟩 DSDM
Requirements Maturity	1	2	3	4	5

Software development processes handle the process of defining requirements differently. Traditional processes, such as the Waterfall model, require that the requirements of a project are determined early on in the process. However, this is not always a possibility. If a client does not know exactly what his/her wishes are, a solid definition of requirements will become an issue. Agile processes accept this fact and furthermore, understand that the environment is very dynamic. Therefore, agile processes have the ability to handle uncertainties much more effectively. For the characteristic "Requirements Maturity", this means that agile processes are always possible to apply. However, the Waterfall model and even the RUP require more mature requirements to be suitable. (Dybå & Dingsøyr, 2008; Leffingwell, 2007).

	Waterfall	Waterfall	RUP	RUP	RUP
	V-model	V-model	Spiral	FDD	FDD
	…	RUP	Waterfall	Spiral	DSDM
	…	Spiral	V-model	DSDM	Spiral
	…	…	…	Scrum	XP
	…	…	…	Waterfall	Scrum
	…	…	…	V-model	…
Team Relationship	1	2	3	4	5

	Waterfall	Waterfall	RUP	RUP	RUP
	V-model	V-model	Spiral	FDD	FDD
	…	RUP	Waterfall	Spiral	DSDM
	…	Spiral	V-model	DSDM	Spiral
	…	…	…	Waterfall	XP
	…	…	…	V-model	Scrum
	…	…	…	…	Waterfall
	…	…	…	…	V-model
Client's Commitment	1	2	3	4	5

The characteristics "Team Relationship" and "Client's Commitment", describe the teamwork within the entire organization. These characteristics are imperative for the suitability of a process. Processes in which teamwork is of great importance, the relationship within the team must be excellent and the client must be committed greatly. This is the case for agile processes. Scrum and XP for example need the client to be onsite for the entire project. If the relationship within the team is dire, the processes lose their suitability (Palmer & Felsing, 2002; Rising & Janoff, 2000). The Waterfall model depends heavily on documentation and less on face to face time. Therefore, when team work is difficult, and the client is not at all committed, documentation is a possible solution. This heavy use of documentation however, results in much overhead when the project only concerns one department in which the relationship is excellent. The RUP also depends on a committed client and a good relationship within the team.

	XP	XP	Scrum	Spiral	Spiral
	Scrum	Scrum	RUP	RUP	RUP
	DSDM	DSDM	DSDM	FDD	Waterfall
	FDD	FDD	FDD	DSDM	V-model
	…	RUP	XP	Scrum	FDD
	…	…	Spiral	XP	DSDM
	…	…	…	V-model	Scrum
	…	…	…	Waterfall	XP
Scope Clearness	1	2	3	4	5

	Spiral	Spiral	Spiral	Spiral	Spiral
	Scrum	Scrum	RUP	RUP	RUP
	DSDM	DSDM	DSDM	FDD	Waterfall
	XP	FDD	FDD	DSDM	V-model
	FDD	XP	XP	Scrum	FDD
	…	RUP	Scrum	XP	DSDM
	…	…	…	V-model	Scrum
	…	…	…	Waterfall	XP
Risk Clearness	1	2	3	4	5

In dynamic markets, software projects are often covered with risks. Software development processes should be able to cope with these risks. However, not every process does this effectively. The Spiral model is a process that does cope with risks. Within this process, risk assessment is of significant importance (Boehm, 1988). Agile processes, and RUP as well, have great risk mitigation. Because of the iterative and incremental approach, risks are found early in the process. Unfortunately, the Waterfall model and the V-model do not effectively tackle these risks (the V-model does have a better approach of validation and verification comparing to the Waterfall model). When applying these processes, risks are often found at the end of the project, where costs for repair are significant. Therefore, the Waterfall model as well as the V-model can only be used when the risks are very clear (Royce, 1970; Sommerville, 2007).

	Scrum	Scrum	Scrum	Scrum	V-model
	XP	XP	XP	XP	Waterfall
	DSDM	DSDM	RUP	RUP	RUP
	FDD	FDD	FDD	FDD	Spiral
	…	RUP	DSDM	Spiral	DSDM
	…	Spiral	Spiral	DSDM	FDD
	…	…	…	V-model	XP
	…	…	…	Waterfall	Scrum
Environmental Stability	1	2	3	4	5

	Waterfall	Waterfall	RUP	RUP	RUP
	V-model	V-model	Spiral	FDD	FDD
	…	RUP	Waterfall	Spiral	DSDM
	…	…	V-model	DSDM	Spiral
	…	…	…	Waterfall	XP
	…	…	…	V-model	Scrum
Stakeholders Flexibility	1	2	3	4	5

As was mentioned before, a project is very dependent on its stakeholders. This also counts for software development processes. Agile processes require the client to be very flexible. The client needs to accept that requirements are not defined at the very beginning. Furthermore, a cost overview and a time schedule cannot be frozen as well, because these might change during the process. Therefore, the client must accept that he has more uncertainty concerning his deliverables. This is not the case for traditional processes. The Waterfall model states everything the client needs to know at the beginning of the project.

	RUP	Waterfall
	FDD	Spiral
	DSDM	V-model
	Scrum	RUP
	XP	FDD
Method of Contracting	Time & Material	Fixed Pricing

The suitability of software development processes depends on the method of contracting (explanation in chapter four). If the method time/material is applied, all processes are suitable. However, when fixed price is applied, certain processes lose their effect. The processes that lose their effect are the agile processes. An agile process is not applicable when the price, and thus the requirements, are frozen (Paetsch et al, 2003).

	🟩 RUP	🟩 Waterfall
	🟩 FDD	🟩 V-model
	🟩 DSDM	🟨 Spiral
	🟩 Scrum	🟨 RUP
	🟩 XP	🟧 FDD
	🟩 Spiral	…
	🟨 Waterfall	…
	🟨 V-model	…
Outsourcing	No	Yes

In relationship to the previous characteristic is the characteristic “Outsourcing”. In many organizations outsourcing is extensively used. Especially for software developing organizations many cost benefits as well as expertise are available when applying outsourcing, or even offshore outsourcing (outsourcing to overseas countries). This however heavily impacts the suitability of software development processes. If there is no outsourcing, all software development processes are suitable, however, the Waterfall model and the V-model are less suitable. This is caused by its extensive use of documentation which is of less importance when no outsourcing is applied. However, when outsourcing is applied, the Waterfall model and the V-model are most suitable. The other processes, depending more on teamwork and on site clients, are less suitable (agile and RUP).

5.4 Final decision framework

Based on the mapping solution presented in the previous sections, a final framework is developed. This framework consists of three elements. The first element is a questionnaire form in which all characteristics found in chapter four are included, the second is a calculation tab and finally a conclusion tab. These three components are separately presented by screenshots and discussed in the next subsection. In subsection 5.4.2, the design choices made are explained. Finally, in subsection 5.4.3, a conclusion is given concerning the framework.

5.4.1 Screenshots

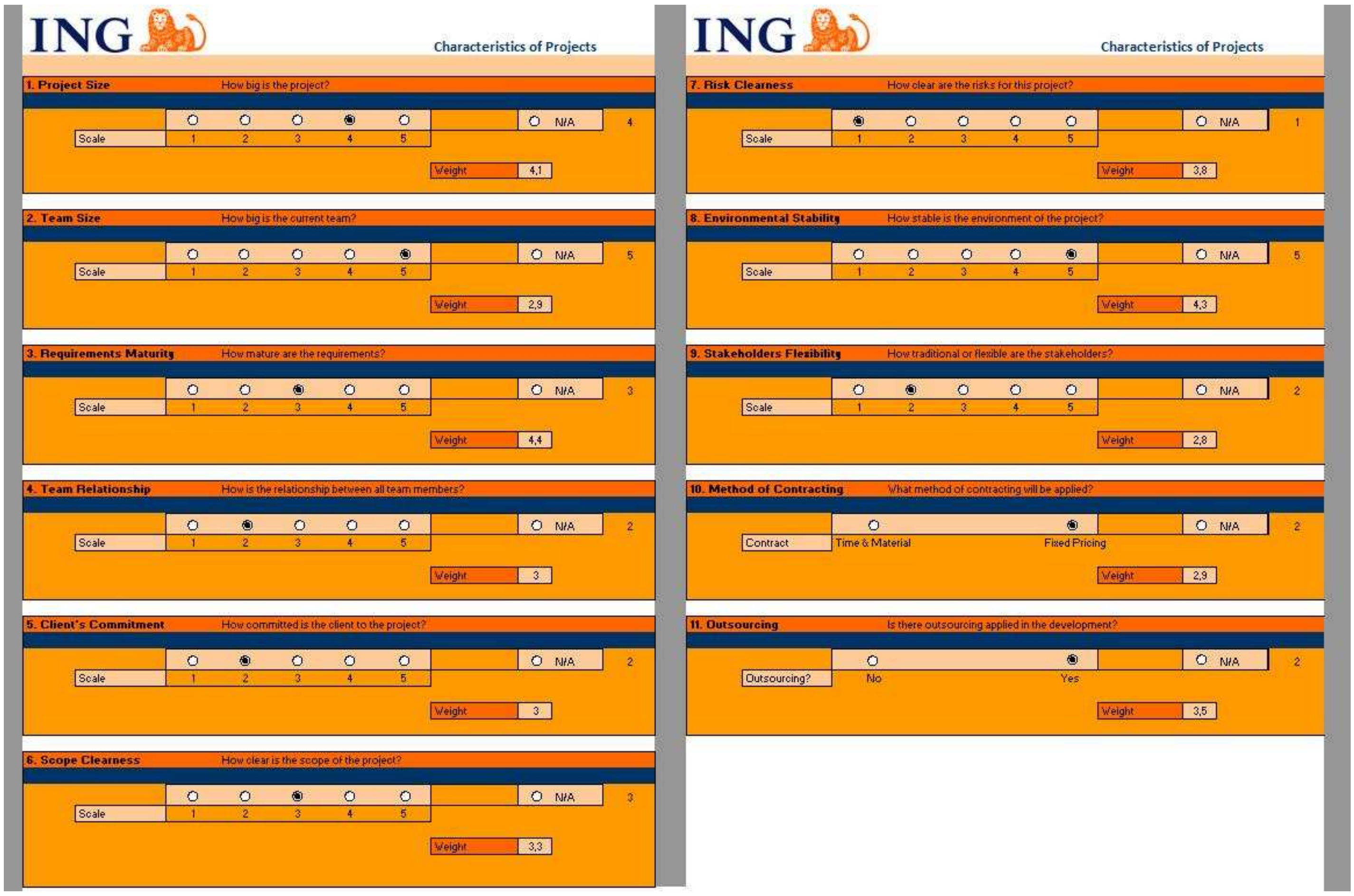
In chapter four, eleven characteristics of software projects were found which influence the suitability of a software development process. Each characteristic can differ in scale (as discussed in previous sections). In the first element of the framework a questionnaire form is presented. In this form it is possible for a user to state the scales of each characteristic for the particular project. However, in this final framework an average project at OIB ING is presented to fully show how the framework functions. The scales and weights for this average OIB project are presented in table 5-4.

TABLE 5-4 Scales and weights of average project at OIB ING

Characteristic	Scale	Weight
Project Size	4	4,1
Team Size	5	2,9
Requirements Maturity	3	4,4
Team Relationship	2	3
Client’s Commitment	2	3
Scope Clearness	3	3,3
Risk Clearness	1	3,8
Environmental Stability	5	4,3
Stakeholders Flexibility	2	2,8
Method of Contracting	Fixed Pricing	2,9
Outsourcing	Yes	3,5

In figure 5-2 a screenshot is presented of the questionnaire form in the final framework. The user can indicate the scale of the characteristic by selecting the circle. The goal of this component of the framework is that the project manager should indicate for each characteristic which scale it has. For example, the project size can be very big, there are a lot of team members involved and the requirements are still very immature. These eleven characteristics are all the relevant characteristics that represent a software project (relevance as in "influencing the suitability of software development processes). However, it could be a possibility that a certain characteristic is not of interest for the project manager regarding a particular project. Therefore, a "Not Available" option is included in the framework. If this is selected for a certain characteristic, it is excluded from the calculation.

For every characteristic a certain weight is given. This represents the importance of the factor (which is collected in the calculation). This weight is based on a survey distributed among the employees of OIB ING and own expertise. However, it is possible to change this if the project manager and client agree on others. This is automatically changed in the calculation as well.



ING — Characteristics of Projects

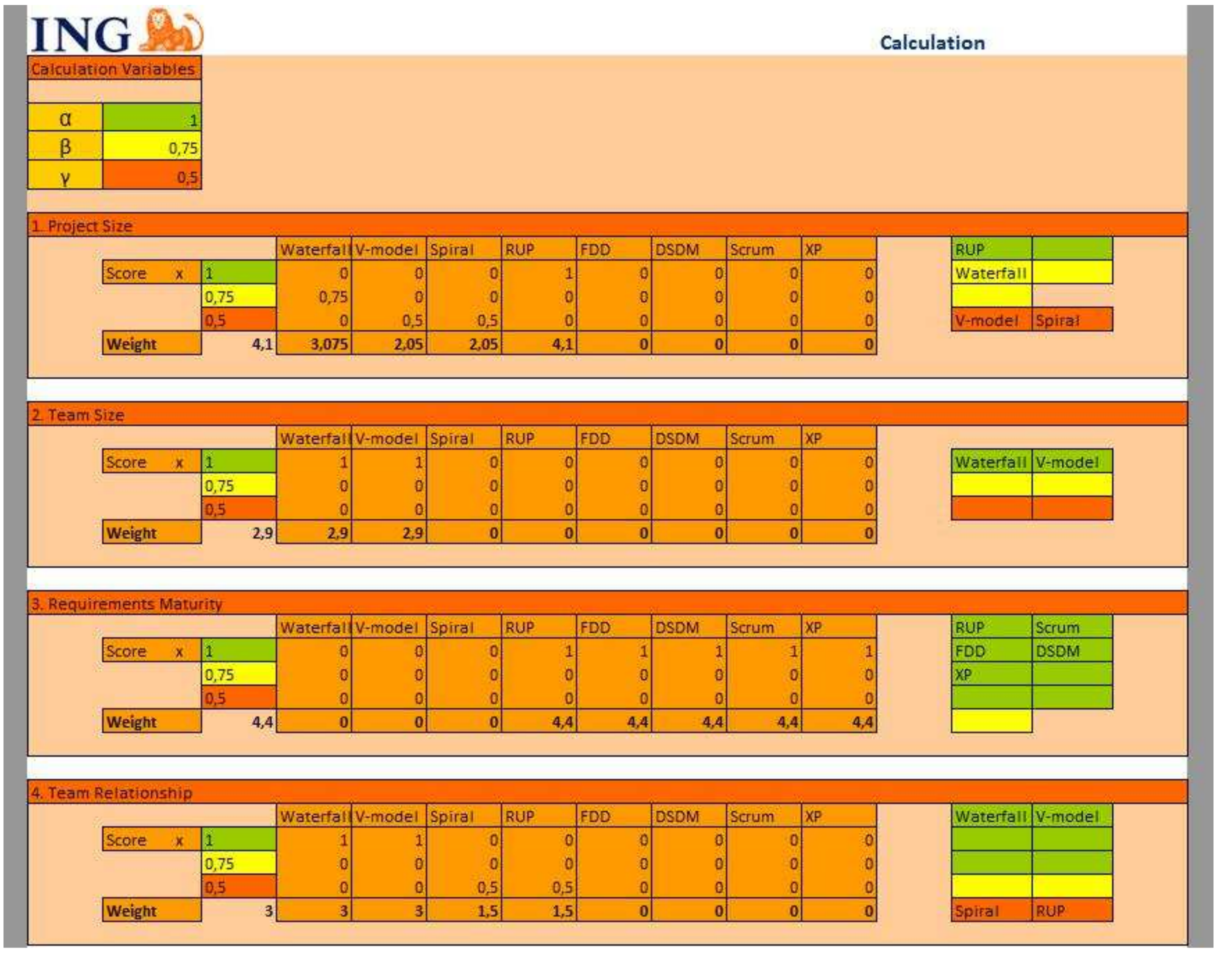
1. Project Size	How big is the project?	Scale: 1 2 3 [✓ 4] 5	N/A	4	Weight 4,1
2. Team Size	How big is the current team?	Scale: 1 2 3 4 [✓ 5]	N/A	5	Weight 2,9
3. Requirements Maturity	How mature are the requirements?	Scale: 1 2 [✓ 3] 4 5	N/A	3	Weight 4,4
4. Team Relationship	How is the relationship between all team members?	Scale: 1 [✓ 2] 3 4 5	N/A	2	Weight 3
5. Client's Commitment	How committed is the client to the project?	Scale: 1 [✓ 2] 3 4 5	N/A	2	Weight 3
6. Scope Clearness	How clear is the scope of the project?	Scale: 1 2 [✓ 3] 4 5	N/A	3	Weight 3,3
7. Risk Clearness	How clear are the risks for this project?	Scale: [✓ 1] 2 3 4 5	N/A	1	Weight 3,8
8. Environmental Stability	How stable is the environment of the project?	Scale: 1 2 3 4 [✓ 5]	N/A	5	Weight 4,3
9. Stakeholders Flexibility	How traditional or flexible are the stakeholders?	Scale: 1 [✓ 2] 3 4 5	N/A	2	Weight 2,8
10. Method of Contracting	What method of contracting will be applied?	Contract: Time & Material [✓ Fixed Pricing]	N/A	2	Weight 2,9
11. Outsourcing	Is there outsourcing applied in the development?	Outsourcing?: No [✓ Yes]	N/A	2	Weight 3,5

FIGURE 5-2 Screenshot of Framework – Questionnaire

When a selection is made for every characteristic and the weight is given, the calculation tab shows each individual score. This tab is presented in figure 5-3.

The first element of this tab is the "Calculation Variable". As was explained before, software development processes can differ in suitability for a certain scale. In this framework, the suitability scores are 1, 0,75 and 0,5. This can be changed by any user. However, it is necessary that the green colour (alpha) is higher than the other two and that the yellow colour (Beta) is higher than the orange colour (Gamma). The second and main element of the this part of the framework is the calculation of scores for the software development processes on each individual characteristic. At the right of each individual calculation table a colour table is presented. This colour table represents the tables depicted in section 5.3. In this particular figure, the scale 4 is selected for "Project Size". This means that the project is rather large. In the table presented in section 5.3, it is shown that when the project size has scale four, the RUP is most suitable. The Waterfall model is suitable as well, but less. Therefore this is presented by a yellow colour. The V-model and Spiral model are also suitable nevertheless, again, less than the other two software development processes. Therefore they are given an orange colour. This is calculated in the bigger table. As the figure shows, when a process has a high suitability, it receives the score 1. If the process is presented in the yellow cells, it receives the score 0.75. Orange receives 0.5 as score. These scores are then multiplied with the weight given.

This is conducted for every characteristic presented in the questionnaire form. The score for every individual characteristic is then summed up, and presented in the conclusion tab. Figure 5-4 depicts this tab.


ING

Calculation

Calculation Variables

α	1
β	0,75
γ	0,5

1. Project Size

		Waterfall	V-model	Spiral	RUP	FDD	DSDM	Scrum	XP
Score x	1	0	0	0	1	0	0	0	0
	0,75	0,75	0	0	0	0	0	0	0
	0,5	0	0,5	0,5	0	0	0	0	0
Weight	4,1	**3,075**	**2,05**	**2,05**	**4,1**	**0**	**0**	**0**	**0**

RUP	🟩
Waterfall	🟨
🟨	
V-model	Spiral

2. Team Size

		Waterfall	V-model	Spiral	RUP	FDD	DSDM	Scrum	XP
Score x	1	1	1	0	0	0	0	0	0
	0,75	0	0	0	0	0	0	0	0
	0,5	0	0	0	0	0	0	0	0
Weight	2,9	**2,9**	**2,9**	**0**	**0**	**0**	**0**	**0**	**0**

Waterfall	V-model
🟨	🟨
🟧	🟧

3. Requirements Maturity

		Waterfall	V-model	Spiral	RUP	FDD	DSDM	Scrum	XP
Score x	1	0	0	0	1	1	1	1	1
	0,75	0	0	0	0	0	0	0	0
	0,5	0	0	0	0	0	0	0	0
Weight	4,4	**0**	**0**	**0**	**4,4**	**4,4**	**4,4**	**4,4**	**4,4**

RUP	Scrum
FDD	DSDM
XP	🟩
🟩	🟩
🟨	

4. Team Relationship

		Waterfall	V-model	Spiral	RUP	FDD	DSDM	Scrum	XP
Score x	1	1	1	0	0	0	0	0	0
	0,75	0	0	0	0	0	0	0	0
	0,5	0	0	0,5	0,5	0	0	0	0
Weight	3	**3**	**3**	**1,5**	**1,5**	**0**	**0**	**0**	**0**

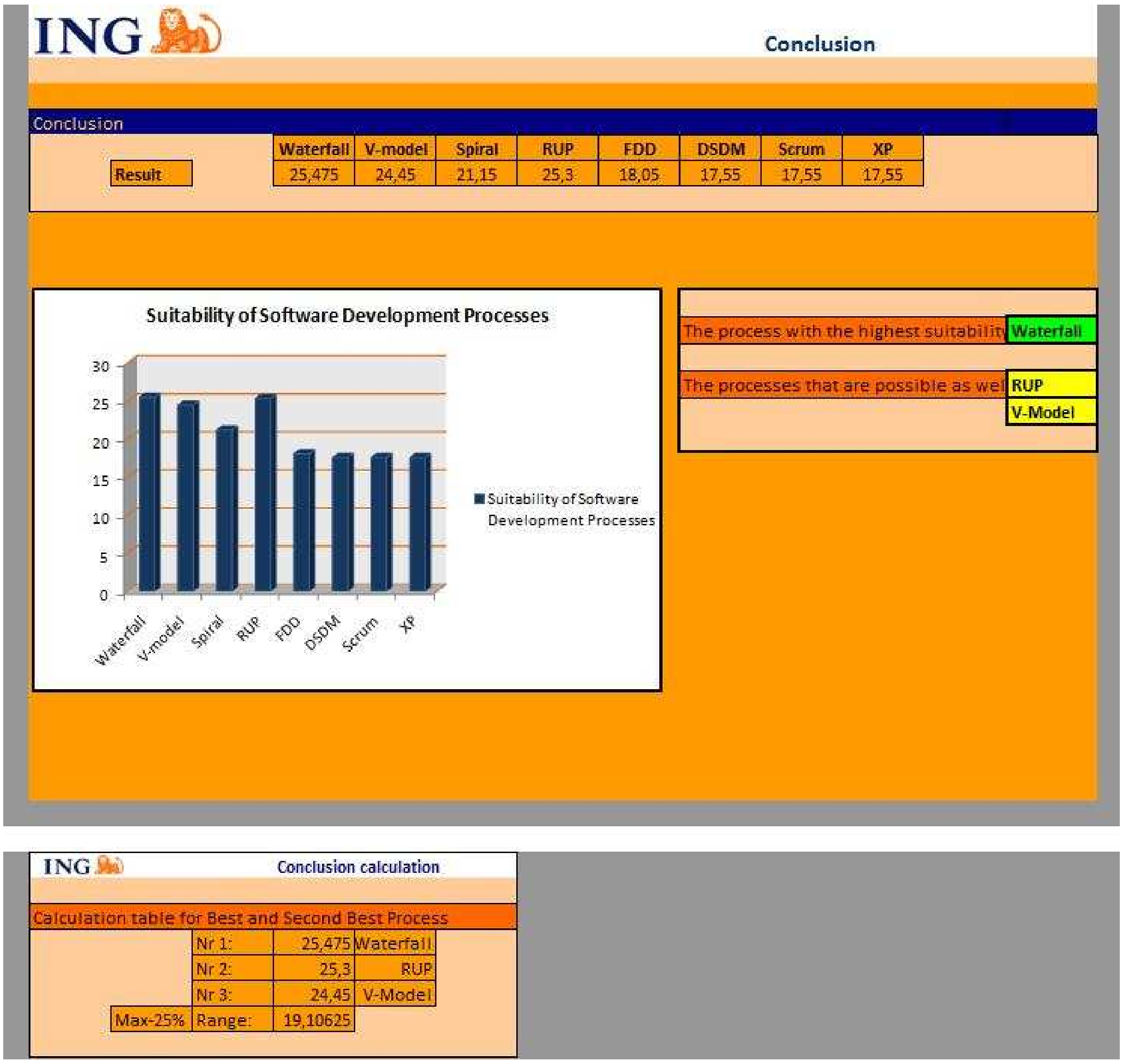
Waterfall	V-model
🟩	🟩
🟩	🟩
🟨	🟨
Spiral	RUP

FIGURE 5-3 Screenshot of Framework – Calculation

The graph presented in this tab gives a better overview on how the processes have scored.

In figure 5-4 a second table is presented (Calculation table for best and second best processes). Because the choices made in this section regarding the suitability of the processes are not factual, other high scoring processes are presented as well. In this framework a maximal difference of 25% is chosen. However, this is possible to change. If the score of a particular process finds itself between the maximal score and the maximal score minus 25% this is presented as "other possible processes".

This is also presented on the right side of the figure. The process with the highest score receives a green colour. The processes, (maximal of two) which are within the 25% score range, are presented by the colour yellow.


ING

Conclusion

Conclusion

	Waterfall	V-model	Spiral	RUP	FDD	DSDM	Scrum	XP
Result	25,475	24,45	21,15	25,3	18,05	17,55	17,55	17,55

Suitability of Software Development Processes
30
25
20
15
10
5
0
Waterfall
V-model
Spiral
RUP
FDD
DSDM
Scrum
XP
Suitability of Software Development Processes

The process with the highest suitabilit	Waterfall
The processes that are possible as wel	RUP
	V-Model

ING

Conclusion calculation

Calculation table for Best and Second Best Process

	Nr 1:	25,475	Waterfall
	Nr 2:	25,3	RUP
	Nr 3:	24,45	V-Model
Max-25%	Range:	19,10625	

FIGURE 5-4 Screenshot of Framework – Conclusion

5.4.2 Design Choices

For this framework certain design choices are made. In this subsection, these choices are discussed. Almost all aspects are based on specific arguments and are deliberated on great extent. The following design aspects are discussed:

- Colours for the framework
- The positioning of the score tables
- The Calculation Variables
- Allowing weights to change
- The “Not Available” option
- The graph in the conclusion tab
- The multiple possible processes in the conclusion

Colours of the framework
The colouration of the entire framework represents the style of ING. To provide the users with a framework which fits their comfort zone was considered very important. Furthermore, by using these colours, it clearly shows that this particular framework was designed for ING usage. The colour orange is used throughout the entire report, and it is appropriate to apply this as well to the framework.

The positioning of the score tables
The score tables are the smaller tables presented in figure 5-3, which present the suitability of the processes (green/yellow/orange). As is shown in chapter five, the original thought was to position these tables in the questionnaire form. However, it was decided to change this. In the original design, it was possible to see directly which consequences each scale has on the suitability of processes. Therefore, it could be possible that users would change their scales to select their favourite process to use. This overview of scores should still be present. However, it should not be obviously connected to the selection of the scales. Therefore, these score tables are moved to the calculation tab (which should not be of interest for the user).

The Calculation Variables
In the framework calculation variables are included. When multiple processes fit on a certain scale, it is possible that they differ in suitability. These calculation variables indicate their suitability. The actual variables are alpha, beta and gamma. In this particular framework these variables are given a score. These are 1, 0.75 and 0.5. It is possible to change these scores. However, it is imperative that the ranking of suitability and thus score stays the same.

Allowing weights to change
The weights presented now in the framework are based on a survey conducted at OIB ING. However, it could be possible that a project manager, together with a business partner, decide that other weights should be given. This could for example be dependent on the current market situation. Therefore, the possibility to change these weights is included in the framework. These changes can be made in the questionnaire form. These changes are then automatically applied in the calculation tab.

The "Not Available" option
In the framework, the not available option is added. In certain projects, particular characteristics might not be of interest or unknown. This can be applied by selecting the N/A option. When this is selected, the entire characteristic is removed from the calculation.

The graph in the conclusion tab
In the conclusion tab a graph is added which presents the scores of each software development process on all the characteristics. This graph is added to provide the user with a better overview of the scores. Furthermore, the comparison between processes is more clear and the results can be communicated better to other stakeholders.

The multiple possible processes in the conclusion
In the conclusion tab it is possible that multiple processes are presented. This copes with the uncertainty in the framework. Usually one process comes out to be most suitable. However, if a second or third process is very close as well, this does not immediately mean that these are not suitable. Therefore a range is given to cope with this. When other processes fall in the range of the maximal score minus 25% they ought to be seen as suitable. This range is chosen because it deletes certain processes which are not suitable at all, but still leaves some room for discussion.

5.4.3 Conclusions
The framework designed is based on the entire research conducted. By analyzing the software development processes, significant characteristics, advantages and disadvantages became clear. These were mapped on the characteristics of software projects found by analyzing literature and conducting interviews. A survey was distributed among employees to review and verify the characteristics found. This mapping resulted in the framework presented in this chapter. The framework designed supports project managers at OIB ING in selecting a certain software development process based on the characteristics of the particular project.

This framework needs to be validated. Validation of the framework is conducted at OIB ING and TU Delft and are discussed in the next chapter. After the framework is validated, it is possible to analyze the possibilities to generalize this framework to other organizations, other markets or maybe even other decision processes. This is presented in chapter six. The final conclusions are presented in chapter seven.

6. VALIDATION

An imperative step in designing a framework which should assist project managers in the decision making consists of the validation of the framework. Validation aims at finding whether or not the framework does what it is intended to do. However, absolute validation is a myth (O'Keefe & O'Leary, 1993). To be as sure as possible that the framework is valid, years of testing and experimenting is inevitable. This is not feasible for this research. Therefore, in this chapter, only three validation techniques are applied. The result of these tests will not indicate whether or not the framework is valid but it will show whether or not it is a useable tool. To find out if the framework is doing what is intended, it is necessary to state the expectation: The framework designed in this research should

…assist project managers in selecting an appropriate software development process in accordance with the particular project at hand.

Validation is useful on many levels. The most significant one is often the elimination of risks. The existence of risks is often seen as most important to asses. However, not only for reducing the risk of failure is validation imperative to apply (Illgen & Gledhill, 1999):

Credibility
In this field of research a clear understanding concerning the suitability of software development processes is scarce. There is not a well established theory on which this framework is based on. Decisions made are principally based on interviews and own expertise. These are however partly confirmed by a literature review. Another reason to increase credibility is that in literature as well as in practice, conflicting opinions exist. A highly established author can move one opinion, which is then refuted by a different highly established author. Therefore, to increase credibility, validation is a necessity.

Confidence
For an organization, and even more important the employees within that organization, implementing a change in their comfort zone, their way of working, is difficult. Not only does validation increases the credibility of a designer, but also the confidence from others in the designer and just as important the confidence in yourself. A well executed validation which results in positive conclusions, confirms the added value to the users.

Wider Acceptance
Because of the increase in credibility and confidence, and the decrease in failure possibilities, the design will be widely accepted. A design is only successful if it is actually applied. Support from the organization and their users are of significant importance.

In this research a framework is designed which should assist the project manager to select a suitable software development process based on the characteristics of the project at hand. Three imperative aspects should be questioned in this section of the research. First, a tool should be easy to use and clear for the actual users. They should understand what is expected and which handlings should be executed. The second important aspect is the actual outcome of the framework. When the scales of the characteristics are indicated in the questionnaire, certain processes should result as most suitable to apply. Important to know is if these resulting processes are in accordance with the expected outcomes. If this is not the case, it is interesting to know why they differ. Finally, this validation process should indicate whether or not the framework is usable. Does it have the added value that is expected? Summarized, the three validation questions are:

1. *Is the design of the decision framework appropriate for users at OIB ING?*
2. *Are the conclusions in accordance with the expectations?*
3. *Is the decision framework an appropriate tool to assist project managers in selecting a suitable software development process based on the software project at hand?*

Multiple validation techniques exist to validate a design. In the first section different techniques are discussed. Furthermore, the three techniques used in this research are explained. In section 6.2 the results of the validation process are discussed. Finally a conclusion regarding the validation of the decision framework follows in section 6.3.

6.1 Validation techniques

Choosing a validation method depends on multiple characteristics of the product and on a number of environmental factors. A very important characteristic is the type of content the to-be validated product has. A model that produces quantitative information should be validated differently than a model that only consists of qualitative information. A quantitative model should be validated by comparing the numbers from the model, with already existing empirical data. Environmental factors have a big influence as well. Especially resources, such as time, people and money can be critical for the amount of possibilities. Next, a number of validation techniques are named (Illgen & Gledhill, 1999):

- Face validation – *Based on look and feel of the models results. This is executed with the help of experts.*
- Trace validation – *An evaluation of each possible path in the model. A very strong but time consuming method.*
- Bottom-up validation – *A method in which validation starts at the lowest level. Very good with bottom-up development.*
- Multistage validation – *Develop assumptions. Compare these assumptions with the validated assumptions*
- Internal validation – *The distribution of results are compared to the distribution of expected values.*
- Sensitivity validation – *Input variable and initial conditions are changed. The impact on the results are then evaluated*
- Comparison – *This is very useful if multiple models exist. Compare models by evaluating distribution differences*
- Interface testing – *Tests data, model and user interfaces. This ensures correct importing and exporting of data.*
- Graphic display – *Compares the graphical results of the model with the real system. Very similar to comparison validation*
- Turing tests – *An expert evaluates two sets of output data, that of the real world and that of the system. The expert differentiates between the two.*
- Experimental Validation – *Together with users test and evaluate the system.*

Based on the validation questions stated before three validation techniques are chosen: Sensitivity validation, face validation and experimental validation. These are however not the most effective techniques. Most appropriate would be to apply the framework on an actual project, asses the entire process of developing the project and compare end results with previous similar executed projects. However, because of time restrictions, this technique is within this research not viable. The three techniques selected are however effective enough considering the available resources. The three techniques and the motivation for using these techniques are discussed in the following subsections.

6.1.1 Sensitivity validation

Sensitivity validation is a very applicable tool when no (or limited) case studies are present (O'Keefe & O'Leary, 1993). Sensitivity analyses entails changing input variables and research the change in output variables. It is important that the output variables do not change extremely when initial conditions are only changed a little.

In this research a framework is developed. To be able to compare the results, an initial starting point is needed. This starting point corresponds with the average executed projects at OIB ING, and is developed with input from Christien Bergman (presented in Appendix VII). This framework is tested on sensitivity by changing the initial conditions. The framework has two sets of initial conditions which are viable to change. These sets concern the calculation variables, which multiply the score by a certain number depending on the suitability of the process on a particular characteristic, and the weights of each characteristic.

The initial scores given to the calculation variables were chosen randomly. It is however important that variable α is higher than variable β and that variable β is higher than variable γ. This should also be the case when changing the variables. The changes concerning the Calculation Variables are the following:

TABLE 6-1 Calculation variables

Variable	Original Value	Test Value 1	Test Value 2	Test Value 3
α	1	1	20	1
β	0,75	0,99	10	0,1
γ	0,5	0,98	1	0,05

An essential difference between the calculation variables and the weights is that the weights are not selected randomly, but are researched and discussed with experienced personnel. The initial weights given to the characteristics are defined by consulting employees at OIB ING. Therefore, these weights should not be changed within this framework. Furthermore, these weights are the basis for the framework. It is understandable that when these weights change, the final outcome will change as well. However, it is interesting to research what will happen to the conclusion if all the weights are equal, which therefore eliminates the weights. Table 6-2 will show the changes made.

TABLE 6-2 Weights variables

Characteristic	Weight Original	Weight Test 1
Project Size	4,1	2
Team Size	2,9	2
Requirements Maturity	4,4	2
Team Relationship	3	2
Client's Commitment	3	2
Scope Clearness	3,3	2
Risk Clearness	3,8	2
Environmental Stability	4,3	2
Stakeholders Flexibility	2,8	2
Method of Contracting	2,9	2
Outsourcing	3,5	2

6.1.2 Experimental validation

A suitable and solid validation method is to introduce the end user to the system. The users can, based on their practical experience, apply the framework as it would in "real-life" and evaluate the results. The method of experimentation by the end user is discussed next. The user of the framework (employees within OIB ING) will have to state the scales for each characteristic. This is done twice. First, the questionnaire is filled in by testing an actual executed project. The user can then compare the software development process actually used for this project with the software development process that resulted from the framework. Secondly, the user has to apply different fictive projects. For example, the user can scale the characteristics according to a small and innovative project. The results can be compared to the individual expectations (which is based on experience) and evaluated. Not only the result of the framework is interesting, but also the usability factor. It is of the utmost importance that each user understand what is expected from him or her and why. Furthermore, the steps taken by the framework should be understandable.

Three employees within OIB ING tested this framework. The three employees are Ed van Kooten (Product manager XDF within SoDC Call Face NL), Christien Berman (manager of requirements management within SoDC Call Face NL) and finally Wouter Blokdijk (architect within SoDC Call Face NL). These three are all very familiar with software development and software development processes. Furthermore, they are all involved in executing projects within OIB ING. They can therefore evaluate the results of the framework with previous experiences. Ideally, a high number of users are used. Unfortunately, because of time restrictions of the employees and this research more than three users was unviable.

The questionnaire presented to the users while testing the framework is presented in Appendix VIII. The results are discussed in section 7.2

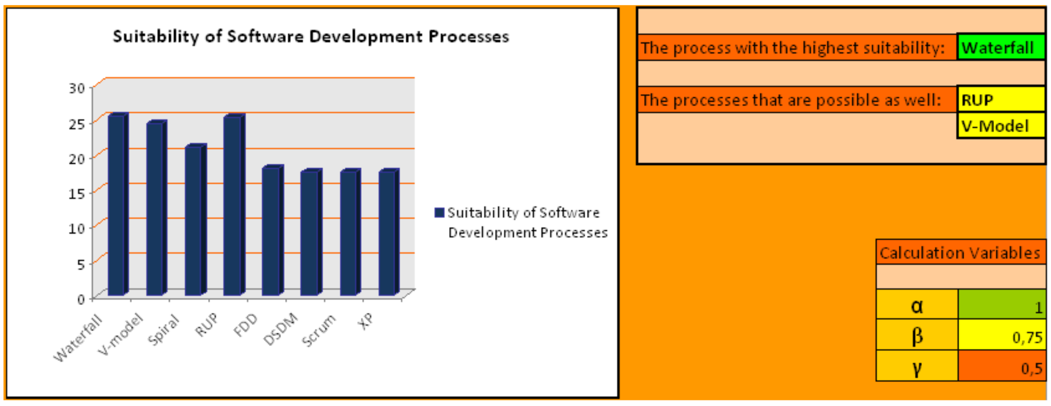
6.1.2 Face validation

Face validation involves experts (knowledge in high extent) regarding the subject. These individual experts were asked whether or not the framework itself and its behaviour are correct (Illgen & Gledhill, 1999; Sargent, 2005). Face validation is very suitable at the early stages of development where an informal review can be highly informative. In this framework certain choices were made, part based on literature and part on subjective opinions. Experts can evaluate these choices. For this method of validation two experts were invited to review the framework. These are Jeroen Hendriks (Manager within SoDC Call Face NL) and Jos Vrancken (Professor within the department of ICT, TU Delft). These very knowledgeable experts regarding software development and software development processes were asked to evaluate the framework by using it in practice and answering some questions. In Appendix VIII the face validation questionnaire as well as their answers can be found. These are discussed in the next section

6.2 Validation results

6.2.1 Sensitivity validation

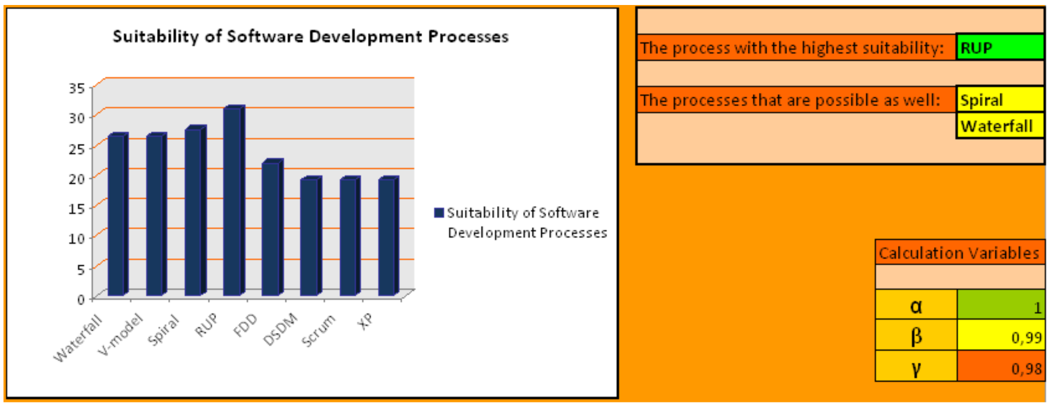
The goal of this method is to change the initial conditions, while analyzing the differences that occur in output or conclusions. When a system is robust, the changes in output will be minimal. The first sensitivity analysis test for this framework is to analyze the conclusion while changing the calculation variables. The graph, and processes with highest and second highest suitability score should be observed. This sensitivity validation consisted of three scenarios. Each scenario has different values for calculation variables. The conclusions of these three scenarios and the original framework conclusion are presented below.


Suitability of Software Development Processes
30
25
20
15
10
5
0
Waterfall
V-model
Spiral
RUP
FDD
DSDM
Scrum
XP
Suitability of Software Development Processes

The process with the highest suitability:	Waterfall
The processes that are possible as well:	RUP
	V-Model

Calculation Variables	
α	1
β	0,75
γ	0,5

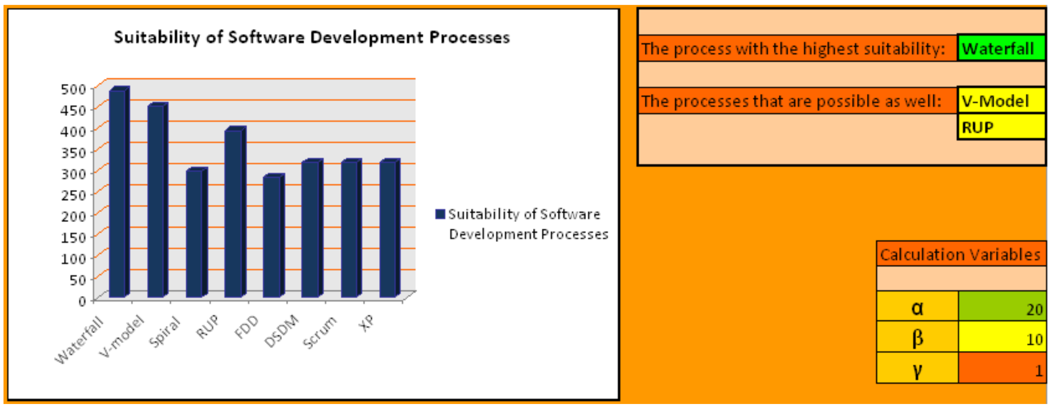
FIGURE 6-1 Conclusion with original calculation variables


Suitability of Software Development Processes
35
30
25
20
15
10
5
0
Waterfall
V-model
Spiral
RUP
FDD
DSDM
Scrum
XP
Suitability of Software Development Processes

The process with the highest suitability:	RUP
The processes that are possible as well:	Spiral
	Waterfall

Calculation Variables	
α	1
β	0,99
γ	0,98

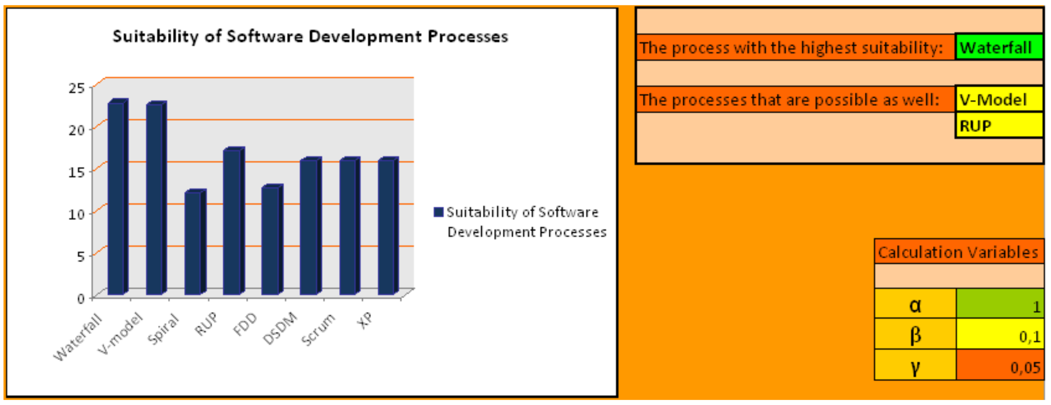
FIGURE 6-2 Conclusion with test value 1


Suitability of Software Development Processes
500
450
400
350
300
250
200
150
100
50
0
Waterfall
V-model
Spiral
RUP
FDD
DSDM
Scrum
XP
Suitability of Software Development Processes

The process with the highest suitability:	Waterfall
The processes that are possible as well:	V-Model
	RUP

Calculation Variables	
α	20
β	10
γ	1

FIGURE 6-3 Conclusion with test value 2


Suitability of Software Development Processes
25
20
15
10
5
0
Waterfall
V-model
Spiral
RUP
FDD
DSDM
Scrum
XP
Suitability of Software Development Processes

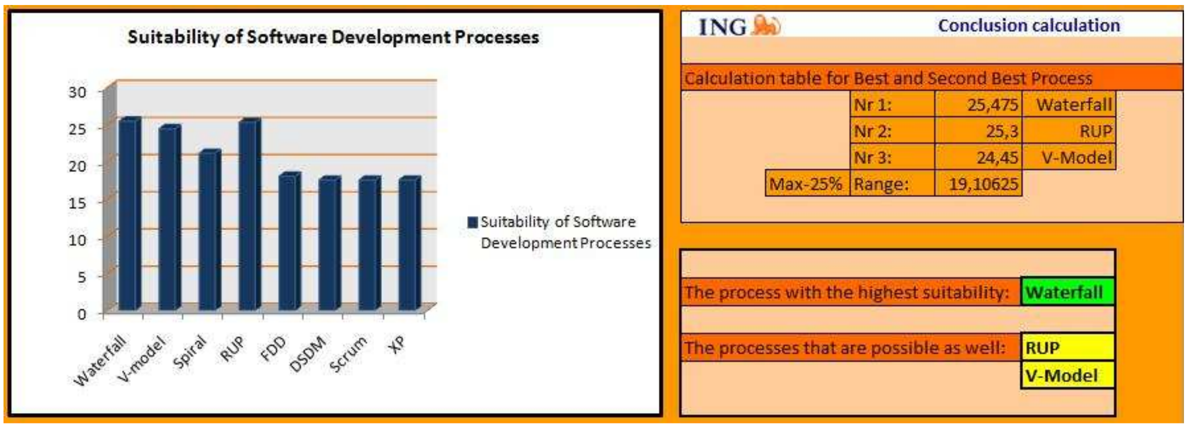
The process with the highest suitability:	Waterfall
The processes that are possible as well:	V-Model
	RUP

Calculation Variables	
α	1
β	0,1
γ	0,05

FIGURE 6-4 Conclusion with test value 3

From the first sensitivity analyses it can be concluded that the framework is quite robust. The second and third scenarios show the same results as the original conclusion. The graphs of all three scenarios show a similar behaviour. The only difference is the score each process receives. However, this is consistent with the changes in the calculation variables values. The conclusion of scenario 1 resulted somewhat different. However, the three processes with the highest suitability score are almost similar to those of the other scenarios and the original framework. From this analysis it can be concluded that the actual value of the calculation variables is not that significant. It is recommended to still use the original number because of the balanced ranking of suitability. However, the final users can decide themselves what the actual values will be.

The second sensitivity analysis concerns the weights of each characteristic. Because of the significant importance of the weights to the framework, big changes will not be analyzed. However, it is interesting to find how much influence the current weights have on the final score. Therefore, all the weights are equal. The results of this analysis is shown in figure 6-6. In figure 6-5 the original conclusion is depicted.



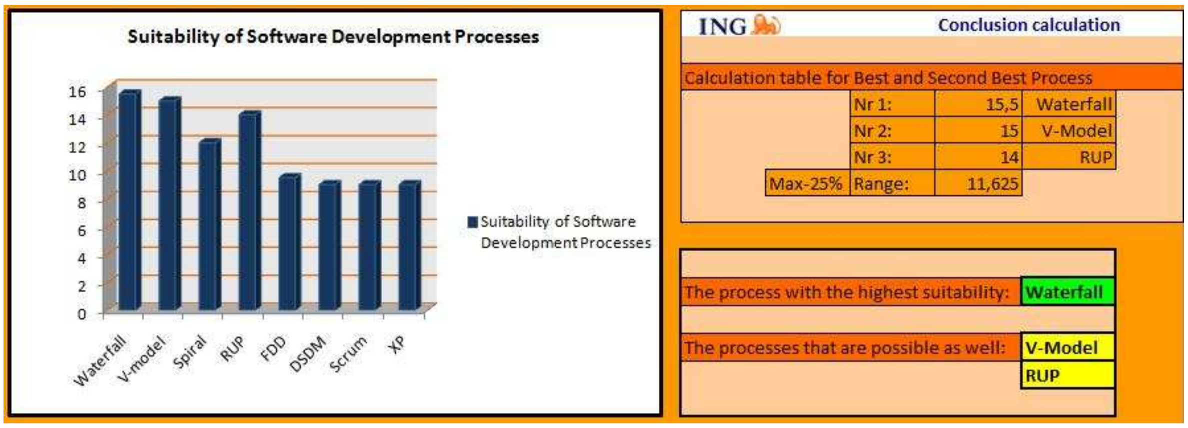
ING Conclusion calculation

Calculation table for Best and Second Best Process

	Nr 1:	25,475	Waterfall
	Nr 2:	25,3	RUP
	Nr 3:	24,45	V-Model
Max-25%	Range:	19,10625	

The process with the highest suitability:	Waterfall
The processes that are possible as well:	RUP
	V-Model

FIGURE 6-5 Conclusion with original weights



ING Conclusion calculation

Calculation table for Best and Second Best Process

	Nr 1:	15,5	Waterfall
	Nr 2:	15	V-Model
	Nr 3:	14	RUP
Max-25%	Range:	11,625	

The process with the highest suitability:	Waterfall
The processes that are possible as well:	V-Model
	RUP

FIGURE 6-6 Conclusion with all equal weights

This validation test shows that the current weights are not of a significant influence. It is obvious that when the weights are changed extremely, the result will as well. However, figure 6-6 shows that when the weights of all characteristics made equal, no significant changes occur to the final result. This again, pleas that the framework is quite robust.

6.2.2 Experimental validation

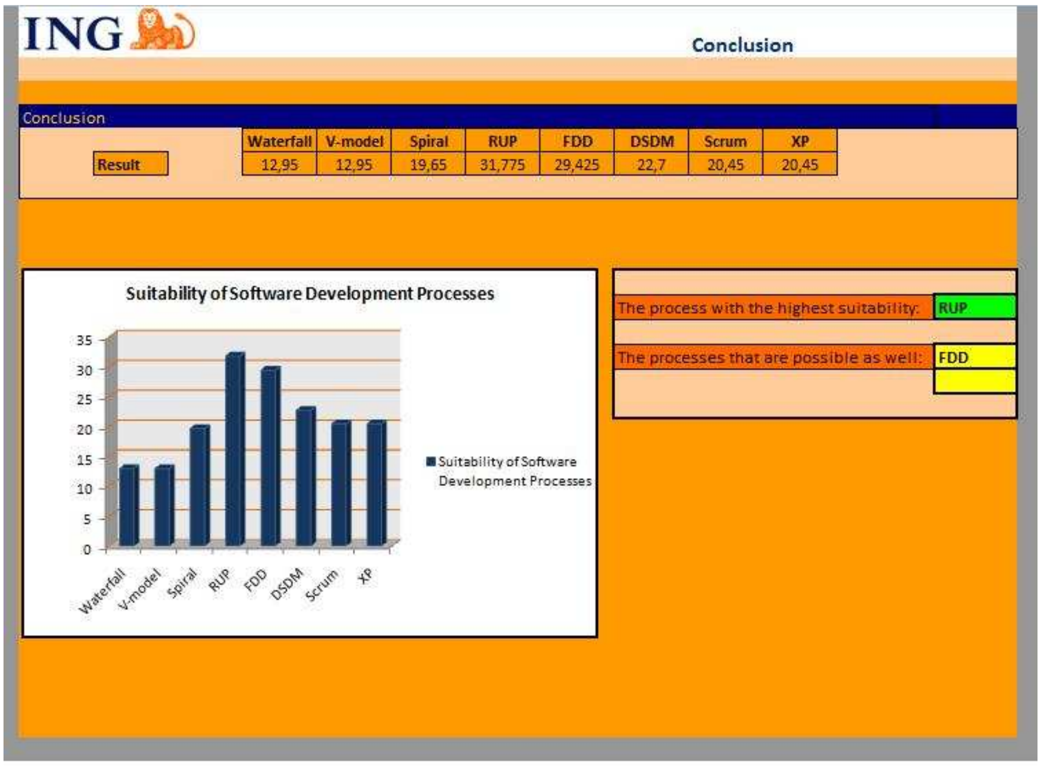
The experimental validation consisted of three subjects who used the framework and commented by filling in the questionnaire presented in Appendix VII. Each of the three subjects experimented with the framework by applying it to different projects. In this subsection three projects are discussed, one of each subject. This subsection is structured in three parts, each discussing the results and comments of one particular subject.

Ed van Kooten (Product manager XDF within SoDC Call Face NL)
Ed van Kooten is an experienced user of software development processes and is familiar with the projects executed in the organization of OIB ING. He is especially experienced in RUP. For this validation experiment this subject applied the framework to a project which is especially feasible for RUP, thereby testing the framework on its accuracy. The project characteristics are stated in table 6-3.

TABLE 6-3 Experimental project Ed van Kooten

Project Characteristic	Scale	Project Characteristic	Scale
Project Size	3	Risk Clearness	2
Team Size	3	Environmental Stability	2
Requirements Maturity	2	Stakeholders Flexibility	2
Team Relationship	4	Method of Contracting	Time/Material
Client’s Commitment	3	Outsourcing	Yes
Scope Clearness	2		

This project was, according to the respondent, a typical project for applying RUP. This project was tested on the framework and the following conclusion was made (figure 6-7). The conclusion of the decision support framework clearly corresponds to the expectations of the subject. However, this validation does not consist solely of testing the result. The respondent had the possibility to give comments on the entire framework, i.e. the design, usage, set-up. These comments are discussed next.


Suitability of Software Development Processes
35
30
25
20
15
10
5
0
Waterfall
V-model
Spiral
RUP
FDD
DSDM
Scrum
XP
Suitability of Software Development Processes

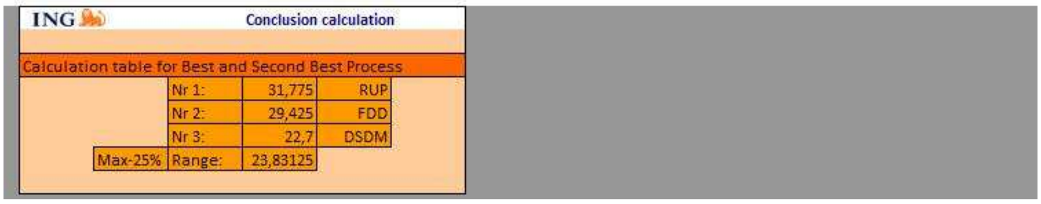
ING

Conclusion

Conclusion

	Waterfall	V-model	Spiral	RUP	FDD	DSDM	Scrum	XP
Result	12,95	12,95	19,65	31,775	29,425	22,7	20,45	20,45

The process with the highest suitability:	RUP
The processes that are possible as well:	FDD



ING

Conclusion calculation

Calculation table for Best and Second Best Process

	Nr 1:	31,775	RUP
	Nr 2:	29,425	FDD
	Nr 3:	22,7	DSDM
Max-25%	Range:	23,83125	

FIGURE 6-7 Conclusion experimental project Ed van Kooten

Positive remarks:
The respondent found the questionnaire to be well-organized. This was also a remark regarding the conclusion tab. The different approaches of showing the suitability of the processes was appreciated. Furthermore, the conclusions of the experimental projects were according to the respondents expectations. The respondent stated that this decision support framework is a very useful tool for starting projects. “It triggers the user to think about the project. Now, projects are often started without even having analyzed what the project actually contains” (Appendix VII).

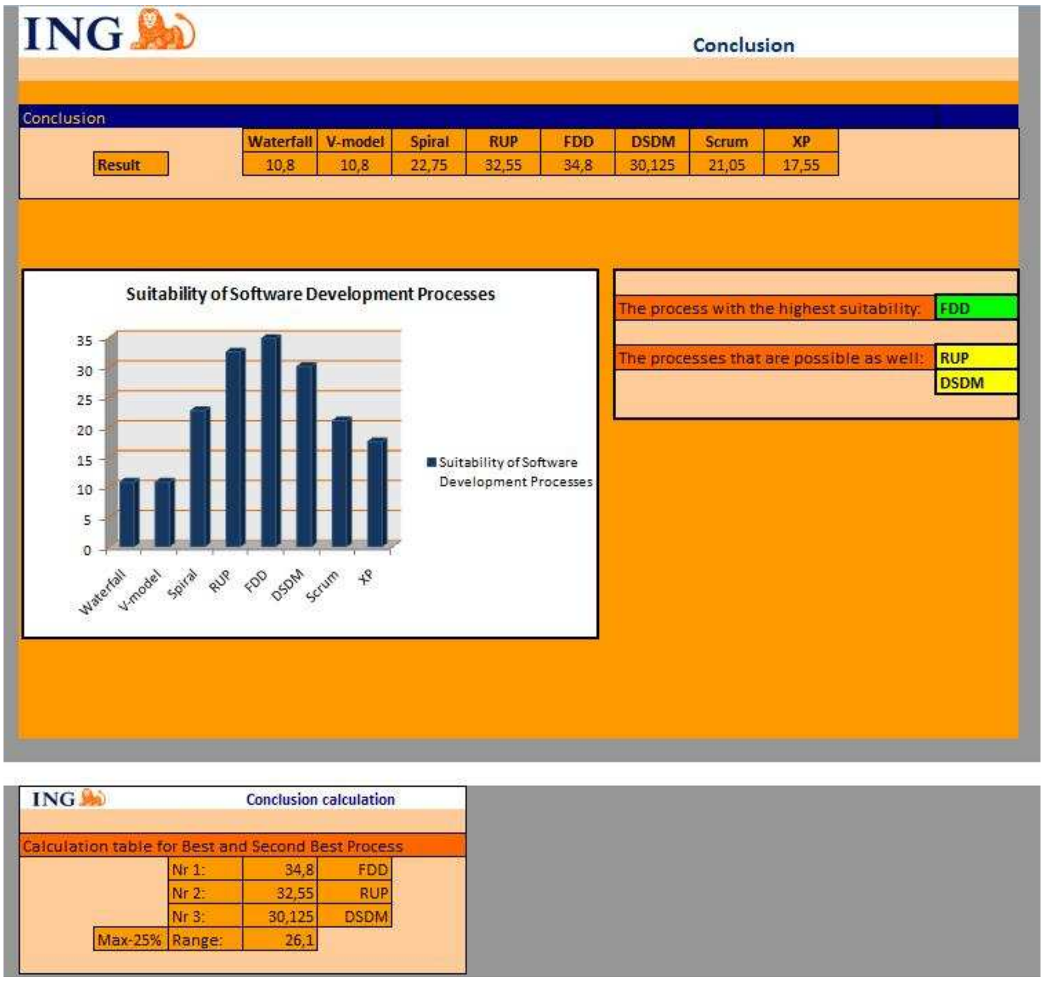
Suggested improvements:
The respondent also had some comments for improving the framework. The first part of the framework consists of a questionnaire. The scales used were not clear for the respondent. The following example was given: “For a certain user, who is often involved in smaller projects, a project of 200.000 Euros could be extremely big, while for someone else, this would count as an average sized project.” The respondent therefore suggested to use absolute numbers to indicate the scales. A different comment was that he missed an explanation. The framework should be a tool on its own, and useful without the report. Furthermore, this explanation should prevent users to interpret the framework differently. Finally, he commented that although the characteristics of the project are all very viable, relationships among these characteristics are missing. As example the respondent gave the following: “If the risks of a project are very unclear, other characteristics might change.” The respondent did indicate that the possibility of changing weights is a very important aspect, because when the risks are very unclear, this characteristic might become significantly more important than it is when the risks are very clear.

Wouter Blokdijk (architect within SoDC Call Face NL)
The second respondent for this validation phase is Wouter Blokdijk. He is employed as an architect and is therefore involved in the beginning of many projects. Furthermore, he is very knowledgeable concerning software development processes. The project stated next (table 6-4) represents the average project he conducts.

TABLE 6-4 Experimental project Wouter Blokdijk

Project Characteristic	Scale	Project Characteristic	Scale
Project Size	2	Risk Clearness	2
Team Size	2	Environmental Stability	2
Requirements Maturity	2	Stakeholders Flexibility	4
Team Relationship	4	Method of Contracting	Fixed Pricing
Client's Commitment	4	Outsourcing	Yes
Scope Clearness	3		

For this particular project the respondent commented that he expected an iterative process to be most suitable. Furthermore he mentioned that the Waterfall model was most likely to be the least suitable. The conclusion, presented in figure 6-8, is in accordance with the respondents expectations. FDD, RUP and DSDM are all processes which use iterative development. The respondent commented that he is familiar with DSDM and RUP and knows that these two are indeed viable for this particular project. Other comments are presented next.


ING
Suitability of Software Development Processes
35
30
25
20
15
10
5
0
Waterfall
V-model
Spiral
RUP
FDD
DSDM
Scrum
XP
Suitability of Software Development Processes
ING
Conclusion calculation

Conclusion

Conclusion

	Waterfall	V-model	Spiral	RUP	FDD	DSDM	Scrum	XP
Result	10,8	10,8	22,75	32,55	34,8	30,125	21,05	17,55

The process with the highest suitability:	FDD
The processes that are possible as well:	RUP
	DSDM

Calculation table for Best and Second Best Process

	Nr 1:	34,8	FDD
	Nr 2:	32,55	RUP
	Nr 3:	30,125	DSDM
Max-25%	Range:	26,1	

FIGURE 6-8 Conclusion experimental project Wouter Blokdijk

Positive remarks:
The approach used for this framework is, according to the respondent, appropriate. While doing projects, he experienced that projects were executed without actually analyzing the characteristics of that project. This framework triggers the user to think. Furthermore, this framework can be used as a tool to convince not only yourself for using a certain process, but even more important, the client.

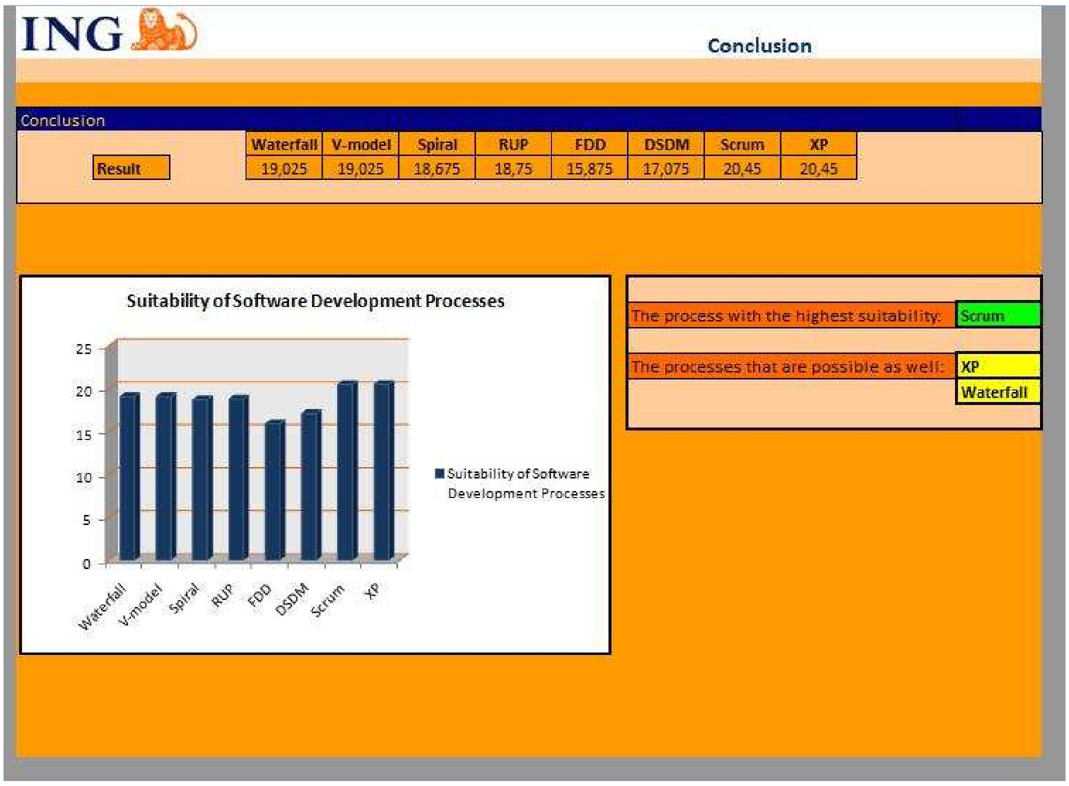
Suggested improvements:
The respondent also commented that some improvements are advised. While using the questionnaire form in the framework, the respondent noticed that the scales are not clear. Absolute numbers to indicate *Project Size* and *Team Size*, would be safer. An other issue he encountered was that some characteristics were unclear. To prevent own interpretation these should be adjusted. Especially the characteristic *Environment Stability* was mentioned. The respondent commented that this could be solved by adding an explanation tab, in which the entire framework is elaborated on. Finally, he noticed that some processes were unknown to him. In this particular experiment, FDD turned out to be most suitable. However, the respondent was not familiar with this process. He suggested to add a linking page to give the user the possibility to get more information regarding that particular process.

Christien Berman (manager of requirements management within SoDC Call Face NL)
The final respondent used for the experimental validation was Christien Bergman. This respondent is, as manager of requirements management, intensely involved in the starting up of a project and even more important, the evaluation of the project. Therefore this respondent has significant knowledge concerning projects performed at OIB ING and their results. Testing this on the framework is very interesting. Furthermore, this respondent has partly introduced RUP into an organization in which only the Waterfall model was used. She is knowledgeable concerning software development processes. The project that this respondent tested on the framework was a project executed earlier this year by OIB ING. As is shown in table 6-5 this project can be considered as very large with many uncertainties.

TABLE 6-5 Experimental project Christien Bergman

Project Characteristic	Scale	Project Characteristic	Scale
Project Size	5	Risk Clearness	2
Team Size	5	Environmental Stability	1
Requirements Maturity	1	Stakeholders Flexibility	4
Team Relationship	2	Method of Contracting	Time/Material
Client's Commitment	3	Outsourcing	Yes
Scope Clearness	1		

For this particular project the respondent reacted very surprised (Figure 6-9). According to the respondent, this particular project was a failure. The project was too big, and the uncertainties were too great. However, they still executed the project. If this was tested on the framework other conclusions might have been taken. Figure 6-9 shows the extreme conclusion regarding this project. This conclusion could be seen as a fault of the framework. However, the respondent commented that this conclusion was very clear. For her, it became clear that this particular project could not be executed. Because of the extreme uncertainties, agile processes, such as Scrum and XP were most suitable. However, the size of the project created a difficulty for agile processes and was more suitable for the Waterfall process. The respondent stated that, according to this result, the project should either be separated into smaller components, in which agile processes suffice, or the uncertainties should be resolved to create more suitability for the Waterfall model or RUP (Appendix VII).


ING

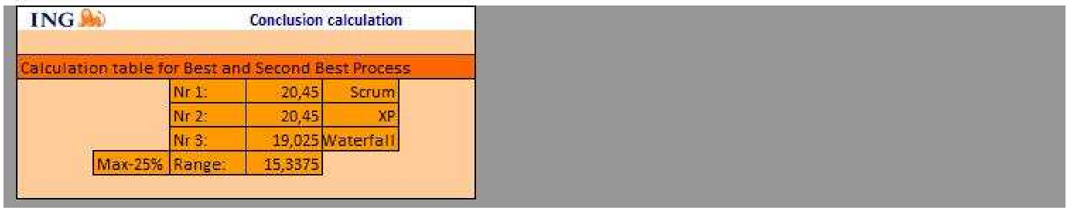
Conclusion

Conclusion

	Waterfall	V-model	Spiral	RUP	FDD	DSDM	Scrum	XP
Result	19,025	19,025	18,675	18,75	15,875	17,075	20,45	20,45

Suitability of Software Development Processes
25
20
15
10
5
0
Waterfall
V-model
Spiral
RUP
FDD
DSDM
Scrum
XP
Suitability of Software Development Processes

The process with the highest suitability:	Scrum
The processes that are possible as well:	XP
	Waterfall


ING

Conclusion calculation

Calculation table for Best and Second Best Process

	Nr 1:	20,45	Scrum
	Nr 2:	20,45	XP
	Nr 3:	19,025	Waterfall
Max-25%	Range:	15,3375	

FIGURE 6-9 Conclusion experimental project Christien Bergman

Positive remarks:

The respondent commented that, when initiating the framework, the style was in accordance with ING and therefore easy to use. Furthermore, the weights gathered from a survey were, according to the respondent, imperative. This will make sure users will not apply their own personal feeling into this framework. After experimenting with the framework, she commented that, in her opinion, the framework was correct. The processes she expected, based on her own experience, resulted as suitable. According to the respondent, this framework is very easy to use and accessible. She commented that it is a "real eye opener". Especially for increasing understanding and creating a certain image it is very useful.

Suggested Improvements:

As with other respondents, she commented that more explanation was needed regarding the framework, the characteristics and the usage of the framework. Furthermore, she commented that it would be interesting to include some sort of "red flag". For example: If every characteristic scores agile, except for the *Project Size* (which is very large and thus scores the Waterfall model), a certain warning should be stated which alerts the user for this particular issue. This will inform the user that, although Scrum scores most suitable, the user should consider the project size before taking this actual decision.

6.2.3 Face validation

Besides experimental validation, face validation was conducted. Two experts on this area were asked to test the framework and answer some questions. This was conducted to get in-depth information regarding the correctness and usability of the framework. The questions asked are presented in Appendix VII. For this validation element the focus was on two aspects:

1. Are the results according the experts expectations?
2. Is the framework appropriate and useful for large organizations?

The first issue that came across was the lacking of information. Each characteristic should be thoroughly explained to prevent individual interpretation. Furthermore, the questions asked in the questionnaire are not clear. It is recommended by both experts to include an extensive information sheet which explains each element of the framework and the steps taken. One expert recommended to include information “clouds” which will appear when scrolling over a particular element of the framework.

The second issue that was mentioned by one of the experts was the lacking of interdependency between characteristics. In this framework each characteristic is discussed individually and processes are mapped on each characteristic individually as well. According to the expert, it could be possible that a certain characteristic changes when a scale of another characteristic is changed. It is recommended by the expert to research these interdependencies and include them in the framework.

On the final conclusion on the usability of the framework both experts were positive. They both stated that for a software project, such a tool would be a great benefit. One expert commented that the insight you get of the project is of great importance. By using this tool, the user is confronted with the characteristics of a project. By thoroughly analyzing and discussing these characteristics the view and knowledge about the project are significantly increased. One expert tested a project conducted at Rijkswaterstaat (the project concerned Road Traffic Management). Also this test was successfully completed according to the expert. This shows that the framework is suitable outside of OIB ING and the financial market.

6.3 Validation Conclusions

The overall validation result is positive concerning the framework. In the introduction of this chapter, three main validation questions were stated. These were:

1. *Is the design of the decision framework appropriate for users at OIB ING?*
2. *Are the conclusions in accordance with the expectations?*
3. *Is the decision framework an appropriate tool to assist project managers in selecting a suitable software development process based on the software project at hand?*

Three validation methods were conducted to test the framework: Sensitivity validation, experimental validation and face validation. The sensitivity validation focused on the sensitivity of the suitability scores and weights. This can be concluded as a positive result. Changes in the suitability scores as well as the changes in the weights of each characteristic did not (or minimally) effect the final conclusion of the framework.

The other two validation methods resulted in very useful comments. From these two methods it is possible to conclude that the framework is indeed an useful method for analyzing the project at hand, and finding an appropriate software development process. This was concluded by all interviewed users and experts. The major point of interested was, according to the users, the fact that the framework triggers the user to actually think and analyze the project. Furthermore, the possibility to communicate this with the client is a significant advantage. A different interesting aspect was the test with Christien Bergman. In this test the framework concluded that almost all software development processes are suitable. This indicates that the project is not specified thoroughly enough. This can be a positive influence on group discussion and negotiation.

Besides the positive general view on the framework, some helpful recommendations were stated as well. First of all, it became clear that the framework needs additional usage information. All users agreed that individual interpretation should be avoided. Secondly, the usage of scales was perceived to be a useful method, however the lack of indication of what each scale means was found to be a difficulty. A different suggestion was to include some sort of warning system: If a process is suitable on all characteristics except one, this should be notified to the user. The final recommended improvement was to include interdependencies between the found characteristics.

Based on the results of the total validation process some changes were made to the framework. The most significant changes are the dependencies that are included between Team Size and Team Relationship and between Requirements Maturity and Scope Clearness. The information tabs which explain the necessary steps in the framework and the processes, and the warning element which indicates when the most suitable process does not score at all on one more characteristic(s). It can be concluded that, after these changes were made, the framework is a useful and helpful tool for large scale organizations involved in developing software.

As was mentioned in the introduction of this chapter, this research is limited in time. It is therefore recommended to start with an ongoing validation process of testing and fine-tuning the framework.
One of the most significant validation methods is comparing the execution and delivering of projects that are executed with the help of the framework with projects that did not.

7. Conclusions

Software development processes should support organizations in developing software projects in a structural and organized manner. Unfortunately, many large organizations still struggle with applying these processes appropriately (Georgiadou, 2003). A large number of processes are available for organizations. However, just a few processes are often actually in use. Organizations apply processes for projects without considering the characteristics of these projects and the possible other processes available. The success of using a software development process for a software project heavily depends on the characteristics of the project itself and the characteristics of the used process. There is no ideal process (Sommerville, 1996). A solution is to use different processes for different projects. By providing project managers a decision framework for choosing the right process, valuable time is saved and the quality of the final product is increased. The main research question in this thesis research was:

What decision framework will assist project managers in large organizations in choosing a suitable software development process?

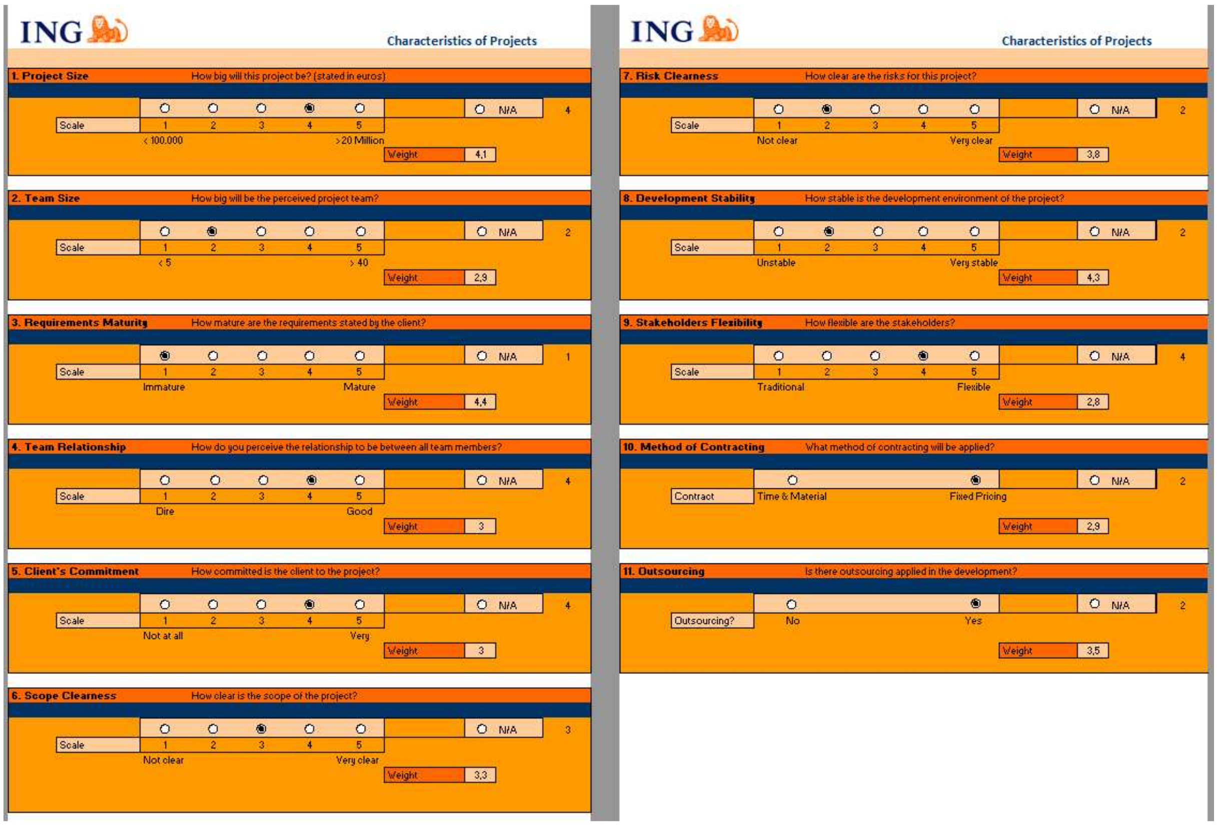
The first step was not only to analyze software development processes, but software projects as well. This case study at OIB ING provided enough information to get an overview of all the relevant characteristics of software projects. Literature was a great source of information for the software development processes. For this particular decision framework the main problem was to map the characteristics of software development processes and the characteristics of the software projects. This mapping problem was solved by analyzing the suitability of each process on each characteristic individually. By dividing the software project's characteristics in five different scales, an overview was created of the suitability of each process on each possible scale.

The developed framework is further elaborated on in section 7.1. Explanation and screenshots are presented. Section 7.2 reflects on this entire research. Questions such as "which issues were overcome and which were not?" are answered. Furthermore, generalization possibilities for the developed framework are presented. Finally, in section 7.3, recommendations are formulate with regard to this research. These consist of all the implications this research has on OIB ING. A separate subsection is devoted to recommended future research.

7.1 The framework

Based on the mapping solution explained before, a decision framework has been developed. The current framework consists of an MS Excel model. The decision framework is presented as follows by presenting screenshots of each element. A fictitious project is used to present the mechanics of the framework. In subsection 7.1.2, the usability of the framework is discussed.

7.1.1 Framework screenshots



ING

Characteristics of Projects

1. Project Size How big will this project be? (stated in euros)

Scale: 1 2 3 [✓ 4] 5 — N/A — 4

< 100.000 … >20 Million

Weight 4,1

2. Team Size How big will be the perceived project team?

Scale: 1 [✓ 2] 3 4 5 — N/A — 2

< 5 … > 40

Weight 2,9

3. Requirements Maturity How mature are the requirements stated by the client?

Scale: [✓ 1] 2 3 4 5 — N/A — 1

Immature … Mature

Weight 4,4

4. Team Relationship How do you perceive the relationship to be between all team members?

Scale: 1 2 3 [✓ 4] 5 — N/A — 4

Dire … Good

Weight 3

5. Client's Commitment How committed is the client to the project?

Scale: 1 2 3 [✓ 4] 5 — N/A — 4

Not at all … Very

Weight 3

6. Scope Clearness How clear is the scope of the project?

Scale: 1 2 [✓ 3] 4 5 — N/A — 3

Not clear … Very clear

Weight 3,3

ING

Characteristics of Projects

7. Risk Clearness How clear are the risks for this project?

Scale: 1 [✓ 2] 3 4 5 — N/A — 2

Not clear … Very clear

Weight 3,8

8. Development Stability How stable is the development environment of the project?

Scale: 1 [✓ 2] 3 4 5 — N/A — 2

Unstable … Very stable

Weight 4,3

9. Stakeholders Flexibility How flexible are the stakeholders?

Scale: 1 2 3 [✓ 4] 5 — N/A — 4

Traditional … Flexible

Weight 2,8

10. Method of Contracting What method of contracting will be applied?

Contract: Time & Material [✓ Fixed Pricing] — N/A — 2

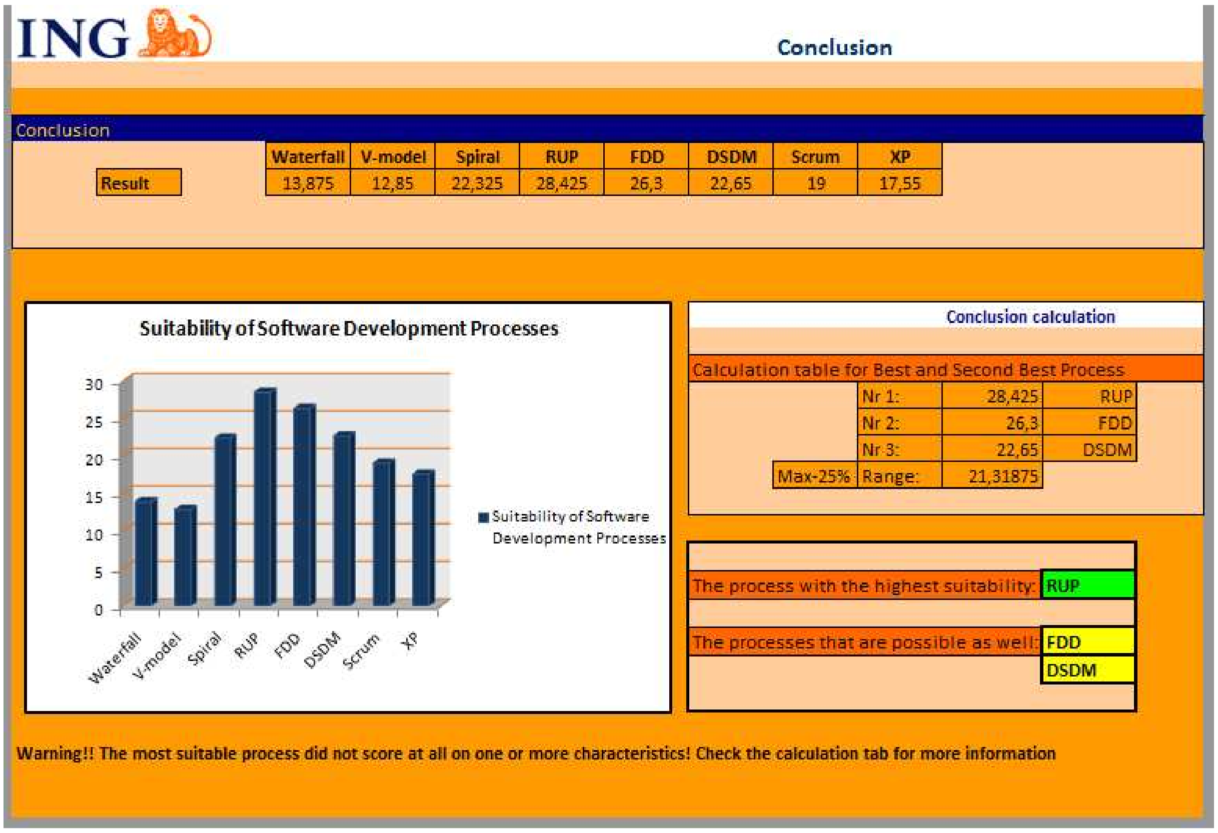
Weight 2,9

11. Outsourcing Is there outsourcing applied in the development?

Outsourcing?: No [✓ Yes] — N/A — 2

Weight 3,5

FIGURE 7-1 Framework Questionnaire


Suitability of Software Development Processes
30
25
20
15
10
5
0
Waterfall
V-model
Spiral
RUP
FDD
DSDM
Scrum
XP
Suitability of Software Development Processes

ING

Conclusion

Conclusion

	Waterfall	V-model	Spiral	RUP	FDD	DSDM	Scrum	XP
Result	13,875	12,85	22,325	28,425	26,3	22,65	19	17,55

Conclusion calculation

Calculation table for Best and Second Best Process

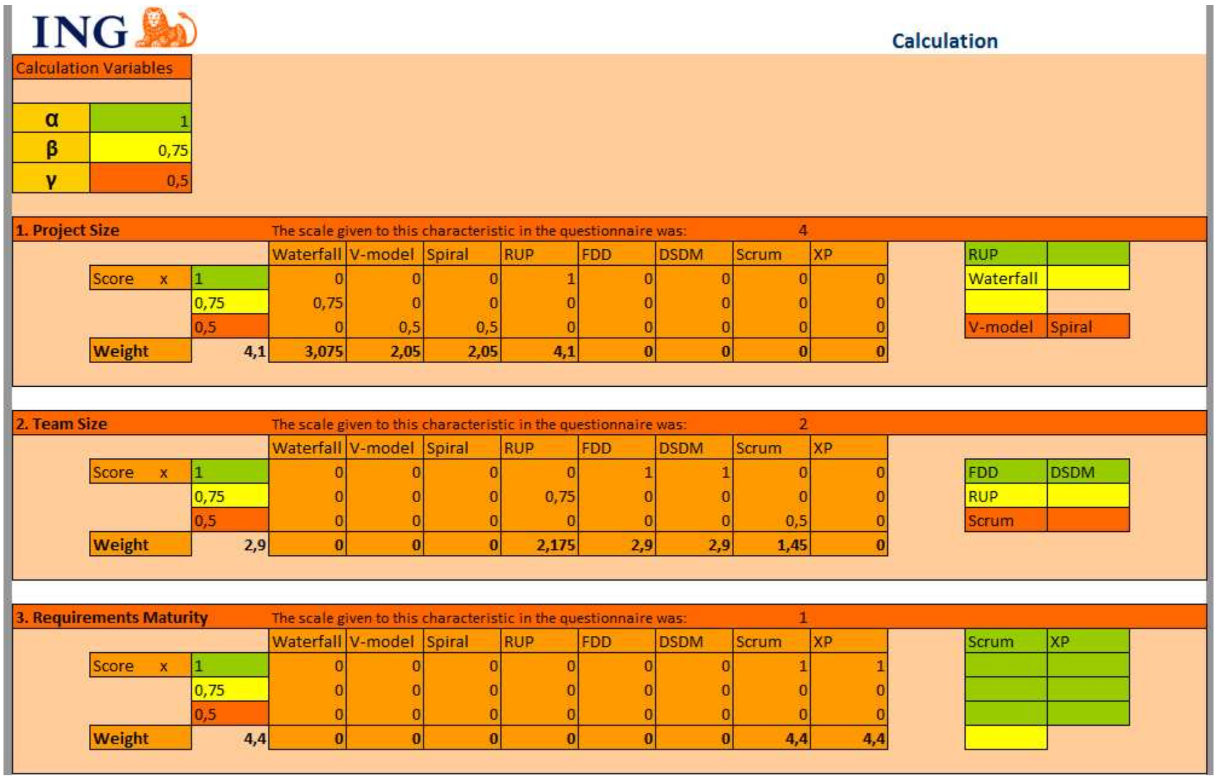
	Nr 1:	28,425	RUP
	Nr 2:	26,3	FDD
	Nr 3:	22,65	DSDM
Max-25%	Range:	21,31875	

The process with the highest suitability: RUP

The processes that are possible as well: FDD, DSDM

Warning!! The most suitable process did not score at all on one or more characteristics! Check the calculation tab for more information

FIGURE 7-2 Framework Conclusion



Calculation

Calculation Variables	
α	1
β	0,75
γ	0,5

1. Project Size The scale given to this characteristic in the questionnaire was: 4

			Waterfall	V-model	Spiral	RUP	FDD	DSDM	Scrum	XP
Score	x	1	0	0	0	1	0	0	0	0
		0,75	0,75	0	0	0	0	0	0	0
		0,5	0	0,5	0,5	0	0	0	0	0
Weight		4,1	3,075	2,05	2,05	4,1	0	0	0	0

RUP	
Waterfall	
V-model	Spiral

2. Team Size The scale given to this characteristic in the questionnaire was: 2

			Waterfall	V-model	Spiral	RUP	FDD	DSDM	Scrum	XP
Score	x	1	0	0	0	0	1	1	0	0
		0,75	0	0	0	0,75	0	0	0	0
		0,5	0	0	0	0	0	0	0,5	0
Weight		2,9	0	0	0	2,175	2,9	2,9	1,45	0

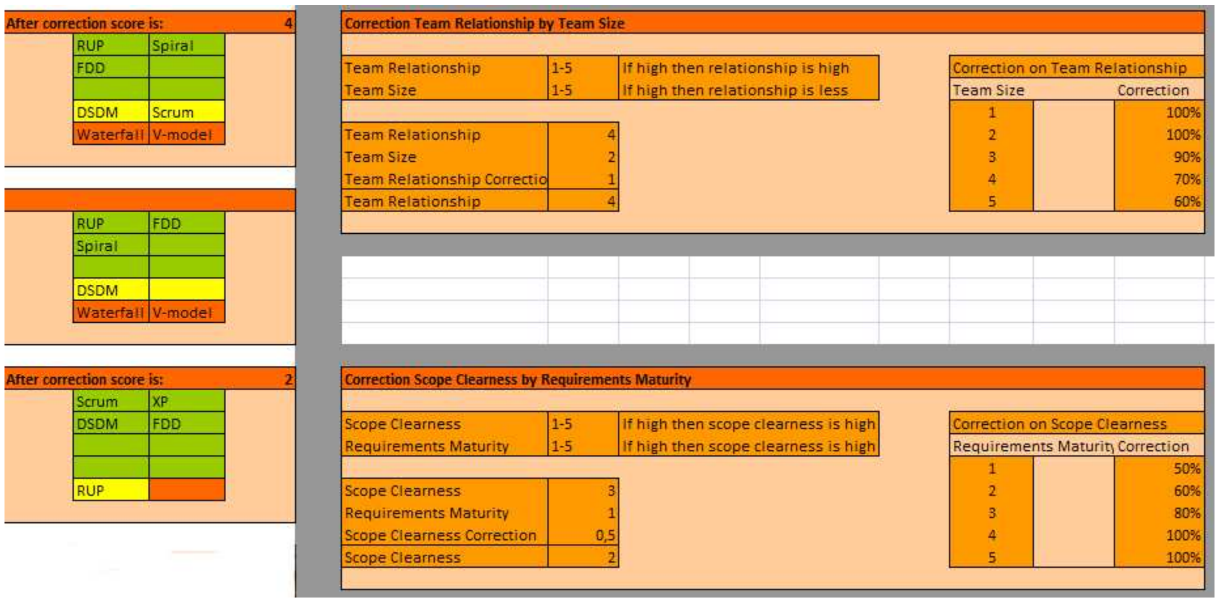
FDD	DSDM
RUP	
Scrum	

3. Requirements Maturity The scale given to this characteristic in the questionnaire was: 1

			Waterfall	V-model	Spiral	RUP	FDD	DSDM	Scrum	XP
Score	x	1	0	0	0	0	0	0	1	1
		0,75	0	0	0	0	0	0	0	0
		0,5	0	0	0	0	0	0	0	0
Weight		4,4	0	0	0	0	0	0	4,4	4,4

Scrum	XP

FIGURE 7-3 Framework Calculation Part 1



After correction score is: 4

RUP	Spiral
FDD	
DSDM	Scrum
Waterfall	V-model

RUP	FDD
Spiral	
DSDM	
Waterfall	V-model

After correction score is: 2

Scrum	XP
DSDM	FDD
RUP	

Correction Team Relationship by Team Size

Team Relationship	1-5	If high then relationship is high
Team Size	1-5	If high then relationship is less

Team Relationship	4
Team Size	2
Team Relationship Correctio	1
Team Relationship	4

Correction on Team Relationship

Team Size		Correction
1		100%
2		100%
3		90%
4		70%
5		60%

Correction Scope Clearness by Requirements Maturity

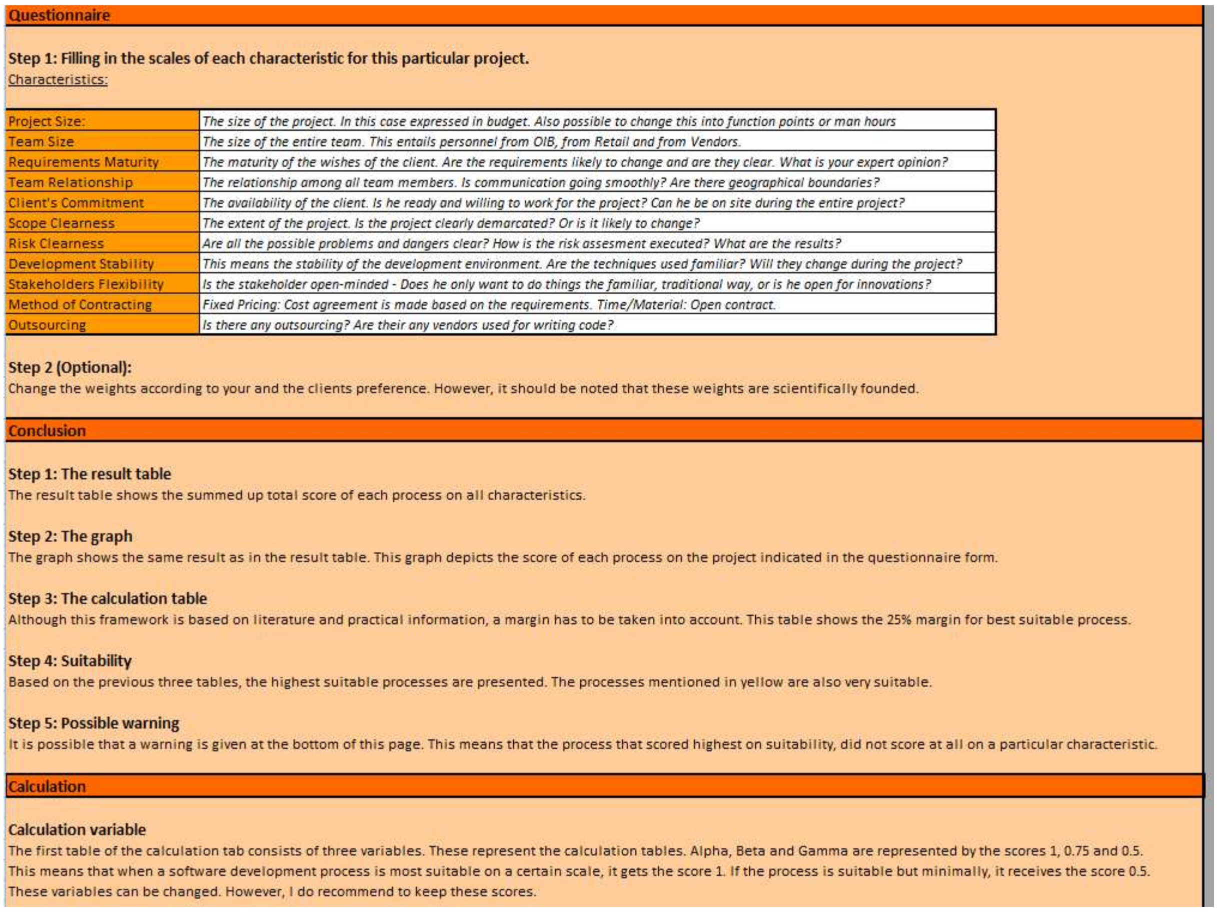
Scope Clearness	1-5	If high then scope clearness is high
Requirements Maturity	1-5	If high then scope clearness is high

Scope Clearness	3
Requirements Maturity	1
Scope Clearness Correction	0,5
Scope Clearness	2

Correction on Scope Clearness

Requirements Maturity		Correction
1		50%
2		60%
3		80%
4		100%
5		100%

FIGURE 7-4 Framework Calculation Part 2



Questionnaire

Step 1: Filling in the scales of each characteristic for this particular project.
Characteristics:

Project Size:	*The size of the project. In this case expressed in budget. Also possible to change this into function points or man hours*
Team Size	*The size of the entire team. This entails personnel from OIB, from Retail and from Vendors.*
Requirements Maturity	*The maturity of the wishes of the client. Are the requirements likely to change and are they clear. What is your expert opinion?*
Team Relationship	*The relationship among all team members. Is communication going smoothly? Are there geographical boundaries?*
Client's Commitment	*The availability of the client. Is he ready and willing to work for the project? Can he be on site during the entire project?*
Scope Clearness	*The extent of the project. Is the project clearly demarcated? Or is it likely to change?*
Risk Clearness	*Are all the possible problems and dangers clear? How is the risk assesment executed? What are the results?*
Development Stability	*This means the stability of the development environment. Are the techniques used familiar? Will they change during the project?*
Stakeholders Flexibility	*Is the stakeholder open-minded - Does he only want to do things the familiar, traditional way, or is he open for innovations?*
Method of Contracting	*Fixed Pricing: Cost agreement is made based on the requirements. Time/Material: Open contract.*
Outsourcing	*Is there any outsourcing? Are their any vendors used for writing code?*

Step 2 (Optional):
Change the weights according to your and the clients preference. However, it should be noted that these weights are scientifically founded.

Conclusion

Step 1: The result table
The result table shows the summed up total score of each process on all characteristics.

Step 2: The graph
The graph shows the same result as in the result table. This graph depicts the score of each process on the project indicated in the questionnaire form.

Step 3: The calculation table
Although this framework is based on literature and practical information, a margin has to be taken into account. This table shows the 25% margin for best suitable process.

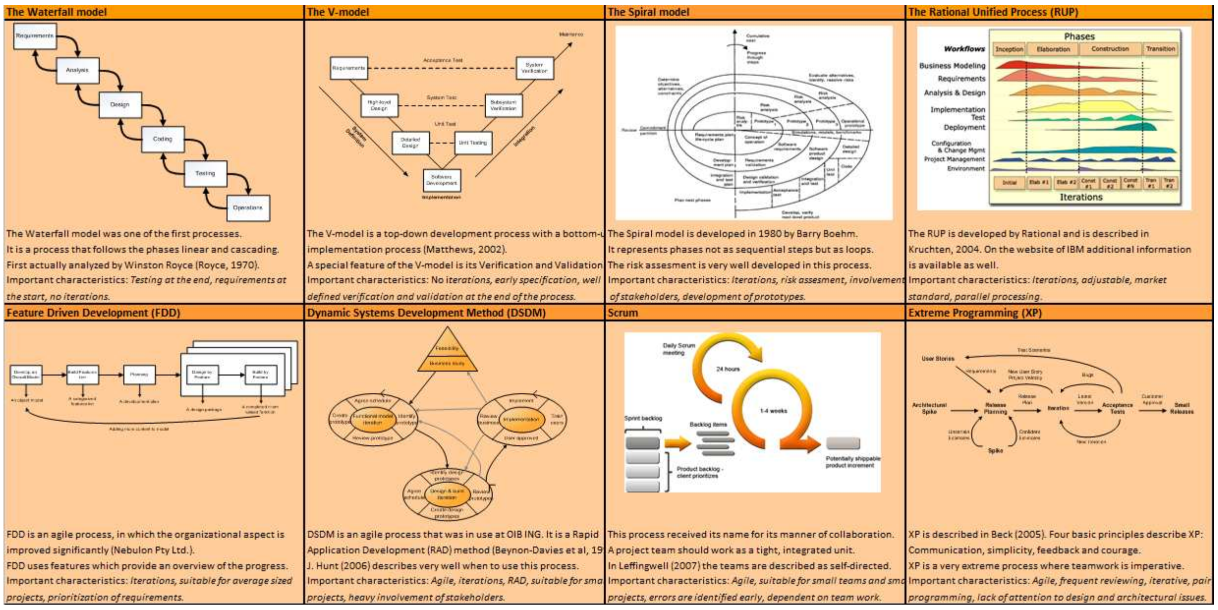
Step 4: Suitability
Based on the previous three tables, the highest suitable processes are presented. The processes mentioned in yellow are also very suitable.

Step 5: Possible warning
It is possible that a warning is given at the bottom of this page. This means that the process that scored highest on suitability, did not score at all on a particular characteristic.

Calculation

Calculation variable
The first table of the calculation tab consists of three variables. These represent the calculation tables. Alpha, Beta and Gamma are represented by the scores 1, 0.75 and 0.5.
This means that when a software development process is most suitable on a certain scale, it gets the score 1. If the process is suitable but minimally, it receives the score 0.5.
These variables can be changed. However, I do recommend to keep these scores.

FIGURE 7-5 Framework Explanation



The Waterfall model

The Waterfall model was one of the first processes.
It is a process that follows the phases linear and cascading.
First actually analyzed by Winston Royce (Royce, 1970).
Important characteristics: *Testing at the end, requirements at the start, no iterations.*

The V-model

The V-model is a top-down development process with a bottom-u
implementation process (Matthews, 2002).
A special feature of the V-model is its Verification and Validation
Important characteristics: No *iterations, early specification, well defined verification and validation at the end of the process.*

The Spiral model

The Spiral model is developed in 1980 by Barry Boehm.
It represents phases not as sequential steps but as loops.
The risk assesment is very well developed in this process.
Important characteristics: *Iterations, risk assesment, involvement of stakeholders, development of prototypes.*

The Rational Unified Process (RUP)

The RUP is developed by Rational and is described in Kruchten, 2004. On the website of IBM additional information is available as well.
Important characteristics: *Iterations, adjustable, market standard, parallel processing.*

Feature Driven Development (FDD)

FDD is an agile process, in which the organizational aspect is improved significantly (Nebulon Pty Ltd.).
FDD uses features which provide an overview of the progress.
Important characteristics: *Iterations, suitable for average sized projects, prioritization of requirements.*

Dynamic Systems Development Method (DSDM)

DSDM is an agile process that was in use at OIB ING. It is a Rapid
Application Development (RAD) method (Beynon-Davies et al, 19
J. Hunt (2006) describes very well when to use this process.
Important characteristics: *Agile, iterations, RAD, suitable for sma projects, heavy involvement of stakeholders.*

Scrum

This process received its name for its manner of collaboration.
A project team should work as a tight, integrated unit.
In Leffingwell (2007) the teams are described as self-directed.
Important characteristics: *Agile, suitable for small teams and sma projects, errors are identified early, dependent on team work.*

Extreme Programming (XP)

XP is described in Beck (2005). Four basic principles describe XP:
Communication, simplicity, feedback and courage.
XP is a very extreme process where teamwork is imperative.
Important characteristics: *Agile, frequent reviewing, iterative, pair programming, lack of attention to design and architectural issues.*

FIGURE 7-6 Framework Process Page

7.1.2 Framework usability

A significant element of this research was the validation of the framework. Is the framework easy to use, and does it actually do what it intends to do? Three important validation questions were formulated to research the validity of the framework:

1. *Is the design of the decision framework appropriate for users at OIB ING?*
2. *Are the conclusions in accordance with the expectations?*
3. *Is the decision framework an appropriate tool to assist project managers in selecting a suitable software development process based on the software project at hand?*

A validation process was developed to find an answer to these questions. After testing the system with actual users and experts a positive conclusion resulted. The users and experts all stated that the framework would indeed be a useful tool for conducting software projects. The users especially considered the framework to be a real eye-opener. By using the framework, projects would be conducted after thoroughly specifying imperative characteristics. An interesting result followed from one particular test conducted in the validation process. In this test almost all processes scored equally in suitability. This suggests that the project needs to be specified further. Therefore, the framework does not only indicate which processes are suitable, but also whether or not the project is specified enough. It helps in discussing unsuitable projects.

Although the final result was positive, some useful and interesting remarks were made during the validation process. The most interesting suggested improvements are the following:

- Framework explanation
- A page with a link to all processes for more information
- Questionnaire should be more clear, own interpretation should be minimal
- A warning when the most suitable process does not score on one (or more) characteristic(s)
- Dependences between characteristics

All these suggestions are included in the framework. It is therefore possible to state that the framework is a useful tool for large organizations and that it is in accordance with the wishes of users and experts. However, due to the lack of time, a comprehensive validation is still lacking. It is imperative to analyze the usability of the framework even further, as well as a data-comparison process between projects executed with the framework and projects without.

7.2 Reflection

In this section the research conducted is reflected upon. In subsection 7.2.1 issues are discussed that result from this research. Finally the generalization possibilities are presented in subsection 7.2.2.

7.2.1 Issues

Disagreement in literature and practice

A main issue that came across during this research was the lack of agreement between experts and practitioners regarding the characteristics of software development processes. In this research a decision framework had to be created in which the suitability of software development processes was presented. However, there is no uniformity in literature. Some sources stated for example that agile processes are useless when applying outsourcing. This was subsequently contradicted in other literature. Furthermore, in practice, at OIB ING, opinions differentiated as well. Therefore, the most difficult task in this research was to decide when a software development process was suitable or not. The lack of confirmation from the scientific society adds uncertainty to the framework. This should encourage the users to further validate the framework to eliminate risks. The framework at this point in time reflects the way employees at OIB ING work with a combination of some scientific background together with my own expertise.

The lack of literature regarding characteristics of software projects
Although there was no agreement in literature regarding software development processes, there was an abundance of information available. Unfortunately, this was not the case for the characteristics of software projects which influence the suitability of processes. The characteristics found are partly based on scientific sources. However most of these sources refer more to the characteristics of a process. Other great sources are interviews, surveys and lessons learned documents. This lack of scientific foundation regarding the characteristics of software projects could be an issue for the developed decision support framework. Although an extensive validation process was executed in this research, it is imperative to continuously test and validate the framework even further, especially because this field of research is quite new and uncertain.

Experience
The final issue for this research and the framework developed is the factor *experience.* In this research the experience an organization has with software development processes is not included. This means that the framework might suggest a software development process for a particular project which, however, is not known within the organization. This framework partly functions as an eye-opener for the user. If experience was included in the framework, probably a small selection of processes (which are already in use at the organization) will often be the outcome. By removing experience from the framework, users might be confronted with the fact that other processes, which are not in use at the moment, are far more appropriate to apply. The aim of this current framework is to show the users which other software development processes might be of interest, which possibly lead to the evolution of an organization such as OIB ING. It will be imperative that they include these interesting processes into their portfolio and mature their organization into a highly experienced software developing team with a broad array of experience. If this is all conducted in the coming years, it is possible to include experience into the framework.

7.2.2 Generalization possibilities
The framework developed in this research is based on a case study executed at OIB ING. In this subsection the possibilities to generalize this framework to other market segments or other organizations is discussed.

One of the aims of this research was to interview developers outside of OIB ING to find whether or not any differences exist in the manner of developing software. Unfortunately this was not feasible due to the lack of time. Therefore it is difficult to state the exact generalization possibilities for this framework. However, after the validation process conducted in this research, it seems that the framework is quite general. In chapter six of this research, an expert validation was conducted. Jos Vrancken tested the framework on a project executed by *Rijkswaterstaat.* According to this expert, the framework was indeed a very helpful tool. Although further testing is a necessity, I think this framework can be generalized to other organizations in the same market segment. Even for organizations outside this market segment, this framework offers some benefits. This framework is developed for ICT related projects. Software projects in the food industry will not differ significantly from projects in the financial industry. Furthermore, the same software development processes are often used. Although this framework is useful for any software developing organization, some changes are needed. First of all the style of the entire framework should be adjusted. The current framework is designed to fit the style of ING. Other changes, which are not necessarily needed, might be the weights and the processes. The weights currently used in the framework are partly based on the opinions and experience from personnel at OIB ING. This might differ in other organizations or market segments. The software development processes included in this framework could also change when generalizing it to other organizations. Some processes, such as DSDM, are included because of the experience the organization has with this particular process.

7.3 Recommendations

In subsection 7.3.1 the implications this research has on OIB ING, large organizations in general and the scientific society is discussed. Sections 7.3.2 discusses recommended future research.

7.3.1 Implications

The result of this research brings about many implications for OIB ING and for organizations in general. The first very important element is how projects should be initiated. By using this framework, team members, clients and stakeholders are forced to extensively reflect and analyze the project at hand. Each characteristic of the project is of great importance and should therefore be defined thoroughly. Furthermore making adjustments to the underlying calculations of this framework should be very limited. Although the framework might differ from the users own expert opinion and reflection, no changes should be made. These calculations are based on scientific research and practical validation. The result should be interpreted as an eye opener, not as an actual fact. However, if experience in using the framework increases, changes in factors and scales are possible.

A different and very important aspect is the usage of software development processes. At this point in time only the Waterfall model and the RUP are in use at OIB ING. The RUP is not even used throughout the entire organization. It is of the utmost importance to train employees in using RUP much more intensively. The Waterfall model is, according to this framework, a very unsuitable process for the majority of projects executed at OIB ING. Therefore, it is important to start expending the knowledge within the organization on RUP and possibly other processes as well.

Besides the RUP and the Waterfall model, six other software development processes are included in this framework. It is not necessary to be experienced in all these processes. However, it is highly recommended that, when a certain process is consistently considered most suitable, research concerning this process should be conducted and included in the organization. For some of these processes this means even changing organizational wide rules. To use agile processes, changes in an organization, such as changes in method of contracting and outsourcing, are imminent (Leffingwell, 2007).

7.3.2 Future research

In this research the focus was on developing a framework to help project managers at selecting an appropriate software development process for a certain project. During this research, other possibilities came forth. These possibilities were outside of the scope of this research. However, these are discussed here as they are recommended to be researched.

Single standard process

One possibility that was mentioned multiple times (especially in interviews and surveys) was to standardize one particular process. Especially the RUP was mentioned as a great process to standardize. This could be a possible solution. However, if one process is used as a standard process, it will be imperative to create three or multiple versions of this process. This means that for a smaller project, a tailored RUP process is used which fits smaller projects. If there are three different types of RUP processes developed (for small projects, average projects and large projects), this could definitely be a solid solution. It is recommended to research this possibility.

Multiple processes within one project
A different solution that was found during this research was to use multiple processes for one single project. Not only projects differ from each other, but also phases within a particular project could be very different as well. Therefore it might be a possibility to use two or three different processes for one project. For example, if the requirements are not clear, an agile process can be used for requirement management. However, because outsourcing is applied, the Waterfall model should be used in the coding phase. For this solution, the framework developed in this research could be very useful, as long as the project is correctly separated in different phases.

7.3.3 Validation
The conclusion of the validation process was generally positive. However, at this point in time it is not feasible to state that the framework is absolutely valid. Continuous testing is necessary. This framework focuses on an area in which literature does not suffice. Because of this, insecurity might become an issue. The most significant method to find whether or not the framework does what it intends to do, is to compare projects conducted while using the framework and projects conducted without using the framework.

Although the framework is not valid yet, it is still believed to be a helpful and interesting tool to use at this point. The fact that it forces the user to evaluate and discuss the project at hand is a significant benefit.

LITERATURE

Ambler, S.W.; Nalbone, J.; Vizdos, M.J. (2005). *The enterprise unified process: extending the Rational Unified Process*. Upper Saddle River: Prentice-Hall PTR, 2005.

Aughenbauch, J.M.; Paredis, C.J.J. (2004). The Role and Limitations of Modeling and Simulation in Systems Design. *Paper presented at the ASME International Mechanical Engineering Congress and RD&D Expo,* held in Anaheim, California, 13-19 November, 2004.

Beck, K.; Andres, C. (2005). *Extreme Programming Explained. Embrace Change.* Addison-Wesley, Boston.

Beck, K.; Fowler, M. (2001). Interview with Kent Beck and Martin Fowler. *InformIT, provided by Addison-Wesley Professional,* 2001. [Online] Requested March 2009. Available at http://www.informit.com/articles/article.aspx?p=20972

Beynon-Davies, P.; Carne, C.; Mackay, H.;Tudhope, D. (1999). Rapid Application Development (RAD): an empirical review. *European Journal of Information Systems,* Vol. 8, pp. 212-223, 1999.

Boehm, B. W. (1988). A Spiral Model of Software Development and Enhancement. *IEEE Computer*, Vol. 21, No. 5, pp. 61-72.

Booch, G. (2007). The Economics of Architecture-First. *Software, IEEE,* Vol. 24, No. 3, pp. 10-11, September-October 2007.

Booch, G.; Rumbaugh, J.; Jacobson, I. (1997). *The Unified Modeling Language user guide*: Addison-Wesley.

Brooks, F.P. (1987). No silver bullet: Essence and Accidents of Software Engineering. *IEEE Computer,* Vol. 20, No. 4, pp. 10-19.

Brooks, F.P. (1995). *The Mythical Man-Month: Essays on Software Engineering.* Addison-Wesley Pearson Eduction (1995)

Chrissis, M.B.; Konrad, M.; Shrum, S., (2003). *CMMI: Guidelines for Process Integration and Product Improvement*. Addison-Wesley, Boston, MA.

Clegg, D.; Barker, R. (2004). *Case Method Fast-Track: A RAD Approach*. Addison-Wesley.

Coley Consulting (2007). Reducing Your Acceptance Testing Risk. *Coley Consulting.* [Online] Requested March 2009. Available at http://www.coleyconsulting.co.uk/moscow.htm

Dijkstra, E.W. (1972). The Humble Programmer. *Communications of the ACM,* Vol. 15, No. 10, pp. 859-866, October 1972.

Dybå, T.; Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information and Software Technology,* Vol. 50, pp. 833-859, 2008.

Erickson, J.; Lyytinen, K.; Siau, K. (2005). Agile Modeling, Agile software development, and extreme programming: the state of research. *Journal of Database Management,* Vol. 16, No. 3, pp. 226-237, 2005.

Fayad, M.; Hamu, D.; Brugali D. (2000). Enterprise frameworks characteristics, criteria, and challenges. *Communications of the ACM,* Vol. 43, No. 10, pp. 39-46.

Georgiadou, E. (2003). Software Process and Product Improvement: A Historical Perspective. *Cybernetics and Systems Analyses,* Vol. 39, No. 1, pp. 125-142.

Göransson, B.; Lif, M.; Gulliksen, J. (2003), Usability Design – Extending Rational Unified Process with a New Discipline. *Proceedings of Design, Specification and Verification of Interactive Systems DSV-IS 2003*, Funchal, Madeira Island, Portugal, June 4 – 6, 2003.

Hesse, W. (2003). Dinosaur meets Archaeopteryx? Or: Is there an alternative for Rational Unified Process. *Softw Syst Model*, Vol 2, pp. 240–247, September.

Hunt, J. (2006). *Agile Software Construction.* Springer-Verlag, London.

IBM (2005). Rational Unified Process; Best Practices for Software Development Teams. *Rational Software White Paper.* [Online] Requested September 2008. Available at http://www-128.ibm.com/developerworks/rational/library/253.html.

Illgen, J.D.; Gledhill, D.W. (1999). 21st Century Verification and Validation Techniques for Synthetic Training Models and Simulations. *I/ITSEC (The Interservice/Industry Training, Simulation & Education) Conference 1999: Synthetic Solutions for the 21st Century.* Orlando. The United States of America, 29 November – 2 December 2007.

Johnson, J.H. (1994). *The CHAOS Report*. The Standish Group International, Inc., (1994).

Kleijnen, J.P.C. (1995). Verification and Validation of Simulation Models. *European Journal of Operational Research,* Vol. 82, pp. 145-162.

Kroll, P.; Kruchten, P. (2003). *The Rational Unified Process Made Easy: A Practitioners Guide to the RUP*. Addison-Wesley.

Kruchten, P. (2004). *The Rational Unified Process 3rd Edition: An Introduction*. Addison-Wesley Longman, Inc.

Leffingwell, D. (2007). *Scaling Software Agility. Best Practices for Large Enterprises.* Addison-Wesley, February 2007.

Liu, L.; Horowitz, E. (1989). A Formal Model for Software Project Management. *IEEE Transactions on Software Engineering.* Vol. 15, No. 10, pp. 1280-1293.

Martin, J. (1991). *Rapid Application Development*. MacMillan.

Matthews, T. (2002). Phase I – Systems Engineering Management Plan: A Process Review and Appraisal of the Systems Engineering Capability for the Florida Department of Transportation. *Florida Department of Transportation.* [Online] Requested March 2009. Available at http://www.floridaits.com/SEMP/Files/PDF_Report/030220-TMI-V2.pdf

McBreen, P. (2003). *Questioning Extreme Programming.* Pearson Education, Boston, USA, 2003.

Meulendijk, K.L.; Oud, S. (2007). Projectmanagement is risicomanagement*. Automatisering Gids*, Vol. 47, 2007.

Naeur, P.; Randell B. (1968). *Software Engineering; Report on a Conference by the NATO Science Committee.* NATO Scientific Affairs Division, Brussels, Belgium

Nazarenko, Y.; Beck, V. (2002). Managing Product Requirements with Evolutionary Lifecycle Model. *Paper presented at the RTO IST Symposium on "Technology for Evolutionary Software Development"*, held in Bonn, Germany, 23-24 September 2002.

Neil, C.J.; Laplante, P.A. (2003). Requirements Engineering: The State of the Practice. *IEEE Software* pp. 40-45, November/December, 2003.

Nebulon Pty Ltd. (2009). Feature Driven Development. *Feature Driven Development.* [Online] Requested March 2009. Available at http://www.featuredrivendevelopment.com/

O'Keefe, R.M.; O'Leary, D.E. (1993). Expert system verification and validation: a survey and tutorial. *Artificial Intelligence Review,* Vol. 7, pp. 3-42, 1993.

Paetsch, F.; Eberlein, A.; Maurer, F. (2003). Requirements Engineering and Agile Software Development. *Paper presented at the Proceedings of the Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'03).*

Palmer, S.R.; Felsing, J.M. (2002). *A Practical Guide to Feature-Driven Development*. Prentice Hall, 2002.

Pressman, R.S. (2005). *Software engineering; a practitioner's approach, 6th edition.* McGraw-Hill, New York, 1997.

Reifer, D.J. (2001). Software Management's Seven Deadly Sins. *IEEE Software*, pp.12-15, March/April 2001.

Rising, L.; Janoff, N.S. (2000). The Scrum software development process for small teams. *IEEE Software*, Vol. 17, pp. 26-32, 2000.

Rook, J.; Lloyd, I.J.; Turner, K.J. (1999). Book Reviews. *The Computer Journal,* Vol. 42, No. 2, pp. 150-152

Royce, W. W., (1970). Managing the Development of Large Software Systems. *IEEE WESCON*, pp. 1-9, August.

Runeson, P.; Greberg, P. (2004). Extreme Programming and Rational Unified Process – Contrasts or Synonyms? *In Proceedings of the Forth Conference on Software Engineering Research and Practice in Sweden*. Linköping University, October 21-22, 2004

Sargent, R.G. (2005). Verification and Validation of Simulation Models. *Paper presented at the Proceedings of the 2005 Winter Simulation Conference,* 2005.

Schwaber, K. (1995). Scrum Development Process. *presented at OOPSLA'95 Workshop on Business Object Design and Implementation*, 1995.

Sommerville, I. (1996). Software Process Models. *ACM Computing Surveys,* Vol.28, No. 1, pp. 269-271, March 1996.

Sommerville, I. (2007). *Software Engineering 8th Edition*. Pearson Education Limited, Harlow, England.

Stapleton, J. (2003). *DSDM, Business focused development. Second Edition.* Pearson Education Limited, Harlow, England.

Stephens, M.; Rosenberg, D. (2003). *Extreme Programming Refactored: The Case Against XP*. Apress, Berkeley, 2003.

Verschuren, P.; Doorewaard, H. (1999). *Designing a Research Project*. Lemma, Utrecht, Netherlands.

Vliet, van H. (2008). *Software Engineering; Principles and Practice.* John Wiley & Sons Ltd, USA.

Xu, P.; Ramesh, B. (2008). Using Process Tailoring to Manage Software Development Changes. *IT Pro,* July/August, pp. 39-45.

APPENDIX I – ROLES WITHIN A PROJECT

Account manager within SoDC Call Face NL
The account manager (AM) is responsible for the single point of contact with a client. He will receive new projects are requests for changes in projects and will manage the communications with the client and the project leader. The following list describes the rolls and responsibilities of the account manager:

- Point of contact with client
- Change Management for making of appointments concerning SoDC Call Face services
- Managing Project Leader IT at start-up phase. Start up phase consists of formulating Project Initiation Document
- Guaranteeing quality of Project Initiation Document
- Discussing service delivery in accordance with the Project Initiation Document with Team Leader Service Delivery (SD)
- Responsible for managing relationship with client
- Protect clients interests within SoDC Call Face NL
- Delivering input for portfolio management

Project Leader IT within SoDC Call Face NL
The Project Leader IT manages the service delivery. He has responsibilities within the start-up phase and the execution phase. There responsibilities are listed below:

Start-up phase:
- Report to account manager regularly
- Synchronizing content of Project Initiation Document with Team Leader SD
- Setting up a team plan which should be in accordance with the Project Initiation Document
- Creating commitment from supplier involved in the project

Execution phase:
- Deliverables should be according to the team plan
- Report to Team Leader SD
- Informing the account manager with project progress and issues/exceptions/requests for changes
- When deviating from Project Initiation Document authorization from account manager is needed

Team Leader SD within SoDC Call Face NL
The Team Leader within each project is responsible for the entire project. He will manage the team members involved in the project. The responsibilities are listed below:

- Providing resources
- Contributes to the creation of the Project Initiation Document
- Approves the final Project Initiation Document
- Responsible for the deliveries
- Responsible for the entire organization of the project

APPENDIX II – XDF DISCIPLINES

The eXternal Delivery Factory consists of ten disciplines. These disciplines are activities grouped together. OIB ING documented the meaning of these disciplines, the activities belonging to them and the team members that are involved. Below, each discipline is shortly described and the activities belonging to each discipline are presented. The order in which these disciplines are discussed is similar to that of the RUP scheme. Each activity is conducted by a certain team member. These roles are presented as well.

Business Modeling
The objective of this discipline is to understand the organization and the problem domain that is being addressed by the project. With the knowledge gained through this process a viable solution needs to be identified which can address the problem domain in which the project finds itself.

Activity	**Role**
Develop a Domain Model	Business Process Analyst

Requirements
This discipline identifies what the behaviour of the system should be. In the requirements discipline the needs and features are transformed into the Software Requirements; such as the use-cases and supplementary specifications.

Activity	**Role**
Requirements Management	System Analyst
Requirements Development	System Analyst
Maintain Requirements Attributes Repository	System Analyst

Analysis & Design
The objective of Analysis and Design is to create a bridge between the requirements, identified previously, and the technical implementation. This discipline focuses on creating and designing an architecture which helps in forming the bridge. While the Requirements discipline identifies what he behaviour of the system should be, this discipline identifies how the behaviour of the system is to be implemented.

Activity	**Role**
Select a Reference Architecture	Software Architect
Define a Candidate Architecture	Software Architect
Determine Interface Descriptions	Software Architect
Determine Components to be used	Software Architect
Manage Architectural Synthesis	Software Architect
Perform Architectural Synthesis	Software Architect
Refine the Architecture	Software Architect
Perform GAP Analysis	Software Architect
Design Architectural Significant Use Cases	Designer
Develop and Maintain the Design	Designer

Implementation

Within this discipline the designed model will be implemented. The model will be transformed into executable code. Furthermore, a basic level of unit testing is performed.

Activity	**Role**
Structure the Implementation Model	Software Architect
Structure and Implement the PoC	Software Architect
Initiate and Execute Knowledge Transfer to ING	Project Manager
Support ING for System Integration	Integrator
Plan the Integration	Integrator
Implement Components	Implementer
Integrate each Subsystem	Implementer
Integrate the System	Integrator
Implement Physical Data model	Database Designer

Test

To ensure quality of the project, the objective of the Testing discipline is to perform an evaluation. This evaluation includes finding defects and verifying and validating the system. Especially whether or not the requirements are met is critical.

Activity	**Role**
Define Evaluation Mission	Test Manager
Accept Test Responsibilities	Test Manager
Develop and Execute tests	Tester
Report Test Results and Test Evaluation Summary to Supplier	Test Manager
Report Test Results and Test Evaluation Summary to ING	Test Manager
Maintain (Master) Test Plan	Test Manager
Prepare for FAT	Test Manager
Prepare for UAT and PAT	Test Manager
Execute Functional Acceptance Test	Test Manager

Deployment

The deployment discipline focuses on making the system available to end users. To manage this, an execution plan has to be created.

Activity	**Role**
Plan Deployment	Deployment Manager
Manage Acceptance Test	Deployment Manager
Facilitate Beta Test	Test Manager
Produce Deployment Unit	Deployment Manager
Beta Test Product	Deployment Manager
Support ING Deploying the Executable	Deployment Manager
Develop Support Materials	Technical Writer
Support the UAT and PAT	Deployment Manager

Quality Assurance & Acceptance

This discipline protects the quality of the entire project. By creating responsibilities and activities for the stakeholder, an additional quality control is formed. These activities, such as creating an acceptance log, will be communicated to the project manager. Communication is in this case critical.

Activity	**Role**

Document Acceptance Criteria	Stakeholder
Execute Review	Stakeholder
Prepare and Plan for Acceptance	Stakeholder
Create Acceptance Log	Stakeholder
Conduct Review	Stakeholder
Deliver Review Records to Supplier	Project Manager
Finalize Acceptance Criteria	Project Manager
Deliver Metrics	Project Manager
Update Acceptance Log	Project Manager
Deliver Acceptance Log to Supplier	Project Manager
Update Project Measurements with Metrics Received	Project Manager
Facilitate Supplier Phase Assessment	Stakeholder
Conduct Phase Assessment with Supplier	Project Manager
Produce Review Records	Any One
Execute User Acceptance Test	Project Manager

Configuration Management

Configuration Management entails tracking, controlling and managing of changes to the projects artefacts.

Activity	**Role**
Write Configuration Management Plan	Configuration Manager
Establish Change Control Process	Change Control Manager
Set Up Configuration Management Environment	Configuration Manager
Define and Promote Baseline	Configuration Manager
Provide Supplier of Artefacts of Baseline	Configuration Manager
Manage Change Requests	Configuration Manager
Manage Baselines and Releases	Configuration Manager
Monitor and Report on Configuration Status	Configuration Manager
Change and Deliver Configuration Items	Any One

Project Management

The objective of this discipline is to direct the activities within the project. This includes managing risks, directing and coordinating individuals within and outside the scope of the project to be sure that it is delivered on time and within budget.

Activity	**Role**
Initiate a Project	Project Manager
Develop PID	Project Manager
Develop Project Plan	Project Manager
Develop Project Constraints for Supplier	Project Manager
Request for Proposal	Project Manager
Plan the Project	Project Manager
Controlling a Phase	Project Manager
Manage Phase Boundaries	Project Manager
Managing Product Delivery	Project Manager
Plan Kick-off	Project Manager
Accepting a Work Order	Project Manager
Authorizing a Work Order	Project Manager
Execute Work Order	Project Manager

Execute Kick-off	Project Manager
Sign-off Work Order	Project Manager
Delivering a Work Order	Project Manager
Close-out Phase	Project Manager
Develop Service Level Agreement for Maintenance Service	Project Manager
Conduct Customer and User Survey	Project Manager
Initiate Project Closure and Execute Lessons Learned Assessment	Project Manager
Prepare for Warranty Period	Project Manager

Environment

The Environment discipline is to support the project and the project team. This entails ensuring that the appropriate process, guidance and tools are available.

Activity	**Role**
Prepare Environment for Project	Process Engineer
Define Tooling Based on ING Toolset	Tool Specialist
Establish Project Environment	Project Manager
Prepare Environment for Iteration	Project Manager
Establish Knowledge Transfer for Delivered Components by ING	Project Manager
Establish Offsite Environment Based on ING Toolset	Tool Specialist

Appendix III – Interview transcript

During this research interviews were conducted with personnel within OIB ING and vendors. These interviews provided me a significant amount of interesting information. However, I will not include the full transcripts of those interviews. In this Appendix, only the significant citations are included. These citations are in Dutch, for the interviews were conducted with Dutch speaking personnel. The full transcripts are available upon request.

III.A Interview Eric Lopes Cardozo and Ed van Kooten concerning RUP

Position Ed van Kooten: Product Manager XDF
Position Eric Lopes Cardozo: Mentor for applying RUP in an organization
Interview concerning RUP and XDF
Operations and IT Banking ING, February 5, 2009

"XDF is een enigszins aangepaste versie van RUP die binnen OIB ING toegepast wordt.

- Contract management met leveranciers toegevoegd
- XDF heeft regels en activiteiten duidelijker en strikter uiteengezet in het kader van het toewijzen van verantwoordelijkheden aan OIB ING en leveranciers"

"OIB ING heeft een uitbesteding beleid. Dit houdt in dat (bijna) alle puur technische aspecten, zoals codering, uitbesteed zijn. Binnen XDF is deze uitbestedingfactor verwerkt. Met vijf leveranciers zijn contract afspraken gemaakt over het gebruik van processen. Projecten die uitbesteed zijn aan deze leveranciers moeten volgens het XDF principe uitgevoerd worden. Deze leveranciers zijn: Capgemini, Logica, Getronics, Ordina en Atos Origin."

"Over het algemeen worden projecten uitgevoerd met gebruik van het "Waterval model". Projecten die uitgevoerd worden door middel van dit model, resulteren over het algemeen in klassieke fouten. Vastzitten op het eind is een veel voorkomend probleem. Wij denken dat met het gebruik van XDF, projecten afgerond kunnen worden zonder significante problemen."

"In het geval van RUP wordt er veel meer aangesproken op het teambelang. Doordat projectmedewerkers veel intensiever in het project aanwezig zijn, en ook het hele proces doorlopen, wordt het individuele werk minder van belang, en het belang van het eindproduct veel groter."

"RUP is een uitgebreid proces met veel inhoud. Bedrijven die allemaal RUP gebruiken, kunnen daardoor toch nog een hele andere aanpak toepassen. Hierdoor wordt er vaak binnen de ING zelf een invulling gekozen voor RUP. Dit is in principe goed. Het op maat maken van het RUP raamwerk ten opzichte van de organisatie is noodzakelijk. Alleen is dit op maat maken helaas over het algemeen binnen de ING vaak te herleiden naar het Waterval model."

"Binnen een bedrijf als ING moet RUP altijd mogelijk zijn. Doordat RUP vrijheid biedt voor aanpassingen kan het proces altijd zo gevormd worden dat het past bij het project."

III.B Interview with Danny Wijnand

Position: Delivery Manager SoDC Call Face NL
Interview concerning the usage of RUP (or the lack of it)

Operations and IT Banking ING, February 19, 2009

“Er is een hele sterke tendens naar de Waterval methode, en er is een kleine groep die nu meer iteratief werkt.”

“Als XDF gebruikt wordt binnen de ING merk je dat de focus iets meer ligt op teamniveau. In de eerste fase van RUP creëert men een toename in teamgevoel en zorgt men dat het team zelf multidisciplinair is.”

“Een project dat uitgevoerd wordt, kan worden verdeeld over meerdere verticals. Payments doet het onderdeel payments, channels doet channels etc. Als wij RUP gebruiken en wij hebben een onderdeel af, willen we dit integreren met andere delen. Helaas gebruikt dan een collega afdeling, zoals payments, het Waterval model. Hierdoor loop je vast in de uiteindelijke koppeling en integratie in het gehele systeem.”

“In alle trajecten die ik doe probeer ik zoveel mogelijk XDF toe te passen. Er zijn echter genoeg projectleiders die iteratief werken niet kennen en de voorkeur hebben voor het Waterval model. De eerste reactie op XDF is; hoe plan je zoiets, wordt dit niet een zooitje. Ze zijn namelijk erg gewend om vaste deliverables te hebben.”

“Het Waterval principe wordt veelal toegepast in XDF. Het gevoel bij mensen dat de requirements zo snel mogelijk af moeten zijn komt vaak voor. Dit komt door twee dingen. Ten eerste door de managers van bovenaf. Die willen een soort van schijnzekerheid – “dit is af, dit is af en dit is bijna af”. Ten tweede komt het door de manier waarop contracten opgesteld worden. De business kant wil graag dat er zo snel mogelijk een fixed prijs bepaald is. Een fixed prijs forceert dat er veel dingen in het project bevroren moeten zijn. Hierdoor kom je vanzelf het Waterval model in. “

“Fixed price biedt veel meer zekerheid. Het is dan duidelijk; er is zoveel geld en zoveel tijd, en dit zijn de requirements. Hierdoor zou je er in principe voor kunnen zorgen dat men niet over de kosten gaan, want anders zijn die kosten voor henzelf en niet voor de klant. Helaas gaat dit dus wel tegen een methodiek als RUP in. En al helemaal tegen methodieken als Agile. Dus denk maar niet dat je met XP of Scrum aan kan komen zetten.”

“Je moet een hele volwassen organisatie hebben om XDF voluit toe te passen. Als je bijvoorbeeld XP of RUP gebruikt, zullen een aantal mensen bepaalde aspecten toch “vergeten”. Vooral in een grote organisatie is dit onmogelijk te controleren en gebeurt dit gegarandeerd. Dit hebben we gezien. Vrij en blij en veranderen wanneer en hoe je wilt. Als je het Waterval model gebruikt wordt je gedwongen door het proces zelf om dit te controleren. Maar bij Agile of RUP moeten mensen uit zichzelf dit uitvoeren en controleren.”

“Je merkt dat steeds meer mensen naar de RUP kant schuiven. Vooral ook omdat het een marktstandaard is en vele vendors dit gebruiken.”

Wat vind jij nou echte nadelen van RUP?

- Grote leercurve
- Als je niet je omgeving mee krijgt is het moeilijk om het er in te passen
- RUP zelf is heel erg groot. Als je niet oppast wordt je geleefd door het proces
- Omdat er zoveel beschreven is heeft iedereen andere ideeën over welke dingen gedaan moeten worden en welke gedaan mogen worden. Iedereen werkt daarom op net een andere manier en heb je de kans dat er 150 deliverables komen.

III.C Interview with Jeroen van Barneveld

Position: Manager of Systems Management within SoDC Call Face NL
Introducing interview
Operations and IT Banking ING, February 20, 2009

“Als er op het eind getest wordt met een systeem (zoals bij Waterval), ontdek je problemen die al dicht bij het beheer in de buurt komen. Mocht je RUP gebruiken zou je deze problemen al eerder treffen en zouden wij (als Systems Management) daar minder last van hebben.”

“De keuze van proces komt neer op de Solution Delivery teams. Wij kunnen wel onze voorkeur aangeven.”

“Ik heb RUP nog niet gezien in de praktijk, maar heb wel gezien dat Waterval niet werkt in de praktijk. Ik heb dus zeker een voorkeur voor niet waterval processen, vooral voor grote projecten.”

“Over-budget is iets wat vaak voorkomt. Dit komt misschien gedeeltelijk door onszelf omdat we de scope niet goed bepalen. Maar ook doordat requirements vaak veranderen. Als je niet zorgt dat de requirements bevroren zijn, is het lastig binnen de kosten te blijven. Tijd is ook een belangrijke factor. Vooral vanuit mijn rol als system manager vind ik dat een hele belangrijke. Als de tijdslijnen te krap worden om de system management processen uit te voeren, zoals acceptatietesten, moet dit versneld uitgevoerd worden, en ontstaat er veel druk.”

III.D Interview with Jeroen Hendriks

Position: Manager SoDC Call Face NL
Introducing interview
Operations and IT Banking ING, February 20, 2009

“Fixed pricing werkt als volgt. Stel “Retail” geeft aan; ik wil de applicatie voor 9 miljoen euro (prijzen zijn fictief). OIB gaat dan aan de hand van een Project Initiation Document (PID) een prijs aangeven. In dit voorbeeld blijkt uit de PID dat het voor 8.7 miljoen kan. Dit is dan een afspraak. Doordat de Retail Manager afhankelijk is van zijn budget, zal deze altijd blijven zeggen dat de afspraak 8.7 miljoen is.”

“Doordat de Retail Manager een bepaald budget heeft, is het voor deze noodzakelijk een duidelijk beeld te krijgen van wat de kosten zouden zijn van het aankomende project. Helaas is het lastig om een oordeel te geven naar aanleiding van de PID. In het begin van het project is de onzekerheid wat betreft de prijs het hoogst. Deze onzekerheid wordt kleiner naarmate het project vordert. Op het eind van het project kan je 100% zeker zeggen wat de prijs precies zal zijn.”

“Het RUP en Waterval methode kiezen allebei een andere methode om deze onzekerheid aan te pakken. Het RUP zegt, het ontwikkelen is een reis. De kosten zullen tussen de 300 duizend en de 50 duizend euro liggen, en we zorgen voor een duidelijke exit strategie. Met deze instelling begint men dan het project. Hierbij accepteert de klant de onzekerheid. Mocht uiteindelijk blijken dat de kosten veel hoger worden dan gepland, kan er gebruikt gemaakt worden van de exit strategie. Hierbij moet continue gecommuniceerd worden met de klant over in hoeverre het proces nog conform initiële verwachtingen loopt en hoe die worden bijgesteld (continu proces), of dat er eventueel gebruik gemaakt moet worden van de exit strategie. Het Waterval model zegt; we gaan beginnen met het opzetten van de requirements. Vanuit deze requirements schatten wij dat de kosten ongeveer 200 duizend zijn. Dit wordt dan geaccepteerd en het ontwikkelen begint. Op het eind, bij het testen, kan het duidelijk worden dat het dan toch duurder is. Dit gebeurt ook geregeld omdat bij de initiële inschatting er te weinig informatie was om een solide inschatting te maken (die dus gegarandeerd wordt gehaald).Helaas is de exit strategie dan allang verlopen."

"Het nadeel van fixed pricing is dat de leverancier kan gaan minimaliseren. Aangezien de extra kosten voor de leverancier zijn, zou het kunnen dat hij dit zo minimaal mogelijk probeert te maken. De kwaliteit van het product kan dan afnemen. Wat is meer gewenst; een kwalitatief goed product waarbij het risico aanwezig is dat de kosten iets hoger zijn, of wil je een product dat mogelijk half af is, maar waarbij er geen extra kosten gemaakt worden? Dit is onmogelijk te bepalen zonder rekening te houden met de karakteristieken van de omgeving – welke stakeholders zijn er betrokken, hoe is de relatie met hen, welke processen zijn ze gewend etc. Pas als dit bekend is, is het mogelijk een beslissing te maken betreffende fixed pricing of niet."

"Fixed pricing is gebaseerd op een maakbare wereld. Je hebt een bepaalde hoeveelheid zekerheid, een bepaalde hoeveelheid geld, dus dit wordt de prijs van het project. RUP echter gaat uit van een minder maakbaar geheel en staat, over het algemeen, op gespannen voet met fixed pricing."

"Het gebruiken van een bepaald proces is ontzettend afhankelijk van de cultuur binnen je organisatie. Binnen elk bedrijf heerst er een cultuur met zijn eigen comfortzone. De cultuur binnen een bank is een georganiseerde wereld waarin risico's geminimaliseerd worden. Het Waterval model is een duidelijk en georganiseerd proces. Dit komt overeen met de cultuur binnen een bank. Een bank draait op een wereld van betrouwbaarheid, voorspelbaarheid, integriteit, duidelijkheid. Dit is een comfortzone waar mensen binnen een bank in zitten. In hoeverre is het dan mogelijk om nieuwe processen in te voeren die de manier van werken, denken en voelen zouden kunnen aantasten? En hoe zou dit kunnen worden gerealiseerd?"

III.E Interview with Theo Brandenburg

Position: Manager Service Delivery Team II SoDC Call Face NL
Introducing interview
Operations and IT Banking ING, March 03, 2009

"Ik heb een aantal bedenkingen bij RUP. Tegenwoordig, in de huidige omstandigheden, wil een bedrijf weten wat ze krijgt voor een miljoen, bij wijze van spreken. Kan je dan voorzichtig iteratief werken zonder een duidelijk idee te hebben waar je naar toe gaat? Nee, tegenwoordig heb je een duidelijke en overtuigende business case nodig. En dan kom je al snel in de sfeer van de Watervalachtige methode, omdat je dan een veel meer geordende structuur krijgt van wat je gaat maken."

"De keuze van proces is situationeel bepaald. Als je kijkt naar een aantal projecten die we laatst gedaan hebben, zijn het allemaal volledig nieuwe applicaties. In die setting is het veel verstandiger een "proof of concept" te ontwikkelen. Als je dan eenmaal een bestaand platform hebt, waar iedereen meer ervaring in heeft, dan kan je hier met gebruikers over praten en is een proces als RUP beter toepasbaar. Iedereen begrijpt dan beter wat er te verwachten is. In hele nieuwe projecten heeft iedereen een heel ander idee en andere wensen. Je bevind je in een veranderlijke omgeving. Het is veel lastiger om dan zonder een duidelijk plan en een goede business case door te gaan."

"We hebben parallel aan het ontwikkelen van LISA (met behulp van XDF) een remodelling traject gehad, waarbij we gekeken hebben naar waarom LISA niet altijd voldoet aan de wensen die we voor ogen hadden. We hebben aan de hand daarvan een standaard ontwikkeld, die nu op de plank ligt voor volgende trajecten binnen LISA. In deze standaard staat bijvoorbeeld, wat je stopt in een use case, wat je stopt in je design model en wat waar thuis hoort. Hier ontstaat ook de discussie of je RUP zo zuiver mogelijk moet toe passen. Hierbij zou je dan zo laat mogelijk beslissingen moeten maken. Dit werkt lastiger met de leverancier. Ik ben dan ook niet een fan van de zuivere vorm van RUP. In een ideale situatie misschien wel, maar vind het nu niet goed toepasbaar. Ik zit meer aan de watervallige kant. De teamsamenwerking binnen RUP spreekt me aan. Maar in plaats van iteratief te werken zoals RUP dat ziet zou ik liever zien dat er stokjes worden doorgegeven tussen die teams en dan op die manier complete cyclussen verwerken."

"Bij fixed price/fixed date leg je het risico bij de leverancier neer. Die zal als reactie hierop heel streng zijn in de zin van zijn intake. Daarnaast is het inderdaad mogelijk dat als de budgetlijn in zicht is, het product afgeleverd wordt zonder dat alle afgesproken requirements behaald zijn. Daar krijg je dan ook discussies betreffende of dit element een fout is of een change. Doe je het time and material, dan zijn ze wat soepeler. Dan zou je misschien zelfs kunnen zeggen dat het, zeker in deze tijd, voordeliger is voor de leverancier als het project minder soepel verloopt. Dan blijven zij namelijk een project om handen hebben. Daarnaast zijn discussies voor de leverancier helemaal niet erg, extra tijd is dan mooi meegenomen."

"Of je dan de ene variant kiest of de andere, er zit altijd iets aan vast. Bij fixed pricing zijn ze scherp, ze lopen misschien wat harder. Bij time and material kan je je afvragen of zij wel de beste mensen op het project hebben staan, of dat die gepositioneerd worden op de fixed pricing projecten. Beide modellen hebben zijn voor en tegens. Mijn mening is dat je het meeste waar voor je geld krijgt als je een langdurige relatie krijgt met je leverancier. Hierdoor zou een leverancier de aanzet moeten krijgen goede producten af te leveren want dan behoudt hij zijn klant. Wij als ING zouden nog veel duidelijker moeten zijn in de evaluaties van de leverancier. Daar waar we vinden dat het minder is gegaan moeten wij ten eerste dit laten weten en ten tweede dit in verband brengen met die langdurige relatie. Als je zo een relatie ontwikkeld dan zou de methode van contracting niet uit hoeven maken."

"Ik zit er op dit moment watervallerig in. Omdat we dat nu nodig hebben, aan voorspelbaarheid en betrouwbaarheid."

"Er wordt tegenwoordig in al die evaluaties veel toegedicht aan het proces, het Waterval model. Het is fout gegaan, dus het ligt aan het proces. Dit is helaas niet het geval. Het ligt aan ervaring en omgaan met het onbekende. Het ligt niet aan de methode, maar aan de mensen die de methode toepassen."

"Afhankelijk van de complexiteit, is het aan het begin van het Waterval model moeilijk om in termen van oplossingen te denken, maar dat hoeft ook helemaal niet Als je zorgt dat je weet WAT je precies wilt en niet HOE je het wilt is het mogelijk het Waterval model te gebruiken. Door duidelijk te maken wat je business processen zijn en die analyseert, dan krijg je een beter overzicht van wat de wensen zijn."

III.F Interview met Floris Zwart

Position: Architect binnen SoDC Call Face NL/ Requirements Delivery
Exploratory interview
Operations and IT Banking ING, March 17, 2009

"Binnen de ING zijn twee processen in gebruik, het Waterval model en RUP. Als architect zijnde is RUP veel makkelijker. Je hebt dan wel veel druk in het begin maar daarna ben je klaar en hoef je alleen nog maar vanaf de zijkant mee te kijken. En met het toepassen van het Waterval model in projecten merk je dat je op het eind toch nog veel tijd kwijt bent in het bijstellen van elementen, keuzes heroverwegen etc. Dat maakt het een stuk lastiger.”

“Je bent constant betrokken bij het verwerken van de wensen van de klant. Meestal, komt er een klant binnen die ongeveer weet wat hij wil en hoe hij dit ongeveer wil. En dan ga je kijken wat de klant precies wilt. Hoe ga je dit inpassen, welke partijen wil je erbij betrokken hebben. Aan de hand daarvan maak je ondertussen een design die je constant aanpast. En dat is wel redelijk iteratief.”

“Een business partner geeft aan; “ik wil dit”. Het “hoe” bepaal je dan met RUP door iteratief en incrementeel te werken. Maar de “vraagkant” wordt nooit mee geRUPt, dat risico lopen ze daar niet graag. Dat is een risico van RUP binnen het bedrijf. Een voorwaarde van RUP is dat de klant met je in zee gaat, zonder te weten wat hij precies kan verwachten.”

“Mijn ervaring is dat het van belang is te weten in hoeverre andere partijen van invloed zijn en in hoeverre zij openstaan voor veranderingen. Vooral met het testen van releases is het heel moeilijk als je afhankelijk bent van een andere afdeling.”

“Je hebt de eeuwige driehoek tussen tijd, geld en requirements. Je moet dan bepalen welke van deze drie variabel is. Welke variabel is, bepaalt welk proces je moet kiezen. Als je requirements vast staan, kan je beter het Waterval gebruiken. Als je budget vast staat, kan je beter RUP gebruiken.”

“Het komt voor dat bij het ene project alles soepel loopt en bij een ander alles stroef. Sommige werknemers liggen elkaar beter dan andere. Daar kan je wel iets over zeggen. De vraag is alleen of iemand dat ook wel echt wilt. Het lastige is ook dat er niet echt in een team gewerkt wordt. De meeste werken parttime aan een project terwijl zij ook met andere trajecten en projecten bezig zijn. In de kleine projecten zijn er 6-7 man die parttime werken.”

“RUP is veel sterker in het omgaan met risico’s. Vooral met vernieuwende projecten heb je veel meer aan RUP. Met wat simpelere, traditionele projecten, past een fabrieksmatige methode zoals het Waterval model wel weer veel beter.”

“Je blijft een bank. Dan moet je gedegen, goed, duidelijk, betrouwbaar, voorspelbaar zijn.”

“Aan de ene kant wil je vernieuwend zijn, maar aan de andere kant wordt je gelimiteerd door de wet en regelgeving. De enige mogelijkheid die ik zie is dat je een omgeving creëert, een soort van pilot omgeving, waarin je zulke dingen uitprobeert.”

“De requirements veranderen vaak. Je ziet ook veel vertraging oplopen bij het testen. Dit door de afhankelijkheid van andere departementen. De afstemming tussen alle partijen binnen de ING en de integratie daartussen, is erg lastig.”

“Het ene proces kan je meer helpen in het voorspelbaar zijn dan andere processen. Hoe zekerder je situatie is, hoe minder flexibel je proces hoeft te zijn. Hoe onzekerder en innovatiever je situatie is, hoe flexibeler je proces moet zijn.”

III.G Interview met Peter van den Hoven

Position: Team manager binnen Solution Delivery Team 2, Releases and Specials
Exploratory interview
Operations and IT Banking ING, March 24, 2009

“Er zijn veel verschillende projecten binnen onze afdeling. Zo hebben we ook veel kleinere applicaties die we ontwikkelen. Dit zijn projecten voor teams van ongeveer 8-10 mensen en binnen een bedrag van 0 tot 100.000 euro.”

“Er wordt bij ons zeer zeker een keuze gemaakt in het soort proces dat we toepassen. Wij kiezen tussen XDF, Waterval of een Package. De Package die we op dit moment toe passen is Hyp3.0.”

“Wij hebben als standaard, of default, XDF als proces. Maar aan de hand van het project zouden we kunnen kiezen om het Waterval model toe te passen. Dit kan bijvoorbeeld wanneer het project zeer complex is waarbij weinig interactie plaats vindt met de gebruiker, of wanneer er al heel veel legacy (oude templates, documenten) aanwezig is voor het betreffende project. Dan ga je niet alles veranderen om XDF toe te kunnen passen.”

“Voordelen van RUP vind ik dat risicomanagement ontzettend goed uitgewerkt is. Daarnaast is het proces en het gebruik daarvan erg goed gedocumenteerd. Ook de communicatie met de leverancier is soepeler omdat het een marktstandaard is. Nadelen vind ik dat het een vrij complex proces is. Ook dat wij een grote “in productie straat” hebben. Dit zorgt ervoor dat bij elke kleine release ontzettend grote teststappen genomen moeten worden, waardoor men al snel iteraties wil gaan combineren. Als laatste hebben we hier dat elke release 6 weken van te voren vast gezet moet worden. Voor kleine releases is het vaak een stuk lastiger om 6 weken van te voren vast te zetten (aangezien een release vaak maar 8 weken is). Maar al met al prefereer ik toch dat we RUP meer gaan toepassen voor projecten.”

“Een bank wil het liefst alles zo snel mogelijk vast timmeren. Hierdoor bereik je stabiliteit en betrouwbaarheid.”

“Voor hele kleine applicaties is het een klein pakket dat uitgevoerd wordt. Voor de wat grotere wordt in principe standaard XDF toegepast. Maar ook hiervan kan er afgeweken worden. De kleine projecten kunnen veranderingen zijn binnen een oudere applicatie. Als deze oude, gehele, applicatie met het Waterval model uitgevoerd is, doen wij dit over het algemeen ook voor de “change”.”

“Het zou voor deze kleinere applicaties mogelijk zijn om een Agile proces toe te passen. Alleen zit je ten eerste heel erg met de veranderingen in de markt. ING wil op dit moment kosten besparen. Dan ga je dus niet experimenteren met een nieuw proces. Ten tweede is de uitbestedingmaatregel lastig. Je hebt simpelweg geen ontwikkelaars op de vloer.”

“Een proces als XP zou mogelijk zijn als je de manier van samenwerken veranderd met de leverancier. Als je bijvoorbeeld zou zeggen; “als jullie 700 functiepunten leveren betalen wij jullie zoveel, maar als jullie er 100 meer halen, krijgen jullie een bonus.” Hierdoor zou er meer aanzet kunnen zijn om een proces als XP toe te passen in samenwerking met de ING.”

III.H Interview met Ron Stal

Position: Delivery Manager binnen Cap Gemini. Vooral betrokken bij financial services projecten
Exploratory interview
Cap Gemini, April 2, 2009

“We hebben over al onze afdelingen RUP als standaard proces. Heel soms wordt er voor bepaalde projecten een agile proces toegepast.”

“Als je een standaard proces toepast zorg je voor voorspelbaarheid en betrouwbaarheid. Dat is voor vele klanten essentieel. Doordat je je methode standaardiseert weet de klant wat hij kan verwachten, wanneer hij iets kan verwachten en wat de rol is die hij zelf zal moeten spelen.“

“Het allerbelangrijkste aan RUP is dat het de risico’s naar voren haalt. Het wordt veel sneller duidelijk wat de problemen zijn en of je hierop kan inspelen. Daarnaast is het een proces die veelal toepasbaar is.”

“RUP wordt zo toegepast; in de eerste fase is het van belang duidelijk te communiceren met de klant. Welke rollen moeten er beschikbaar zijn, hoe gaan we samenwerken etc. Daarna moeten de requirements vastgesteld worden. Deze twee fasen gebeuren iteratief. Als de requirements dan helemaal vastgesteld zijn komen we in de constructie fase. We leveren dan elke 6 tot 8 weken functies op.”

“Het probleem van heel iteratief en incrementeel werken is dat de markt daar niet echt geschikt voor is. Je merkt nu veel dat bedrijven “factory matig” willen werken. Voor elk onderdeel een apart blok waar iets gebouwd wordt. Eerst worden de requirements bepaald. Dan wordt de architectuur gemaakt, dan wordt het gebouwd en daarna getest. Voor sommige projecten is het bijvoorbeeld zo dat wij, Cap Gemini, de applicatie bouwen voor ING, en dat dit getest wordt door Logica. Aparte fabrieken dus.”

“Het voordeel van het door 1 partij laten doen, is dat de testers veel beter weten wat er aan de hand is en hierop inspelen. Daarnaast ontwikkel je een langdurige relatie en is het mogelijk meer voorspelbaarheid te creëren. Maar er zijn ook mensen die zeggen dat je het bouwen en testen nooit door dezelfde partij moet laten doen. Doordat je twee partijen gebruikt zal de ene partij gecontroleerd worden door de ander.”

“Je kan niet voor alle projecten RUP toepassen. Als je een heel klein project hebt, ga je niet een heel proces gebruiken. Maar ik denk wel dat een standaard proces in deze markt veel meer gewild is. Nogmaals, je probeert voorspelbaarheid en betrouwbaarheid te creëren.”

“Mijn mening is dat je RUP aan bijna alles kan aanpassen. Maar inderdaad, ik zou niet zo snel RUP toepassen voor een heel klein project, maar misschien dat ik dan geen 1 proces zou toepassen.”

“Een heel groot langdurig project nodigt problemen uit. Dat werkt niet. De wereld kan zoveel veranderen in een jaar, dan moet je niet halverwege je project zijn. Je moet dan zorgen dat het project opgedeeld kan worden in kleinere projecten. Waarin elk kleiner project een toevoeging is voor je bedrijf maar dus ook eventueel uitgebreid kan worden. Voor deze opgesplitste projecten kan je dan weer RUP toepassen. Bij langdurige trajecten is het handiger om een programma op te starten (waar dan weer kortlopende projecten onder hangen (< 9 maanden)). Het einddoel van een programma is meestal helder, de weg naar het einddoel wordt meestal in de loop van het programma bepaald / aangepast.”

III.I Interview met Danny Wijnand

Position: Team manager binnen Solution Delivery Team 2, Releases and Specials
Discussion regarding the characteristics of projects
Operations and IT Banking ING, April 8, 2009

“Aspecten als budget, tijdsduur, scope zijn geen karakteristieken waarop ik mijn methodologie kies. Die aspecten zijn gebaseerd op project management. Je wilt met je methodologie zorgen dat die aspecten succesvol behaald worden. Dus de methodologie is niet afhankelijk van die aspecten maar heeft invloed daarop. De andere factoren die je noemt, die bepalen welke methodologie je kiest. Dit zijn bijvoorbeeld; volwassenheid van de requirements, relaties binnen het team, commitment van de business partner."

"Extra karakteristieken zijn in mijn mening:

- Zijn de stakeholders traditioneel?
- Hoe innovatief (complex) is het gene dat je moet bouwen
- Waar zit je op je lifecycle van de applicatie. Is het oud? Of Ontzettend nieuw?
- Scope duidelijkheid is ook erg van belang
- Commitment is ook erg van belang
- Ik zou relationship within the team anders beschouwen. Ik vind dat ook de relaties met de supplier en andere departementen erin zit"

"Karakteristieken die van mij weg mogen zijn:

- Budget weg
- Tijd weg
- Team size weg"

"Ik vind eigenlijk niet dat het V-model een echte methodologie is. Heb ik ook nooit geweten. Ik vind het eerder een manier om te testen, die erg goed is trouwens. Maar deze manier van testen kan je ook toepassen als je RUP gebruikt of als je het Waterval model toepast."

"Ikzelf gebruik liever het Spiral model dan het Waterval model. Het Spiral model zorgt ook voor die iteraties die ik zo belangrijk vind. Daarnaast is de risk assesment erg goed.

"Ik weet niet zo goed of de factor "Role of other departments" echt een rol speelt. Als de andere afdelingen het Waterval model toepassen mis je misschien de integratie voordelen. Maar je hebt nog wel de voordelen voor jou eigen onderdeel"

III.J Interview met Frank Hollander

Position: Senior Manager OIB
General discussion regarding the content of Senior Manager and his opinion regarding processes
Operations and IT Banking ING, April 29, 2009

"Senior Manager houdt in dat ik de eindverantwoordelijke ben voor het project. Ik probeer de organisatie te houden tussen de verschillende afdeling."

Wij als managers stellen welk proces er voor grote projecten toegepast zal worden (XDF of Waterval)."

"RUP of XDF is erg goed als je risicofactoren van grote onzekerheid zijn. Die factoren ontdek je sneller met RUP doordat deze naar voren gehaald worden in het proces."

"XDF is in principe het standaard proces dat we toepassen voor projecten. Mocht de klant heel traditioneel ingesteld zijn, en zijn de wensen onduidelijk, dan is Waterval een beter proces. Doordat je Waterval gebruikt zorg je er voor dat de klant zijn wensen duidelijk maakt en dat je later geen problemen tegen komt betreffende je requirements."

“Door Fixed Pricing en het gebruik van uitbesteding is puur RUP erg moeilijk. Vandaar dat XDF is ontstaan, die een combinatie maakt van RUP en Waterval elementen.”

“Voor kleine projecten is het zeer goed mogelijk lichtere processen toe te passen zoals Extreme Programming.”

III.K Interview met Jannes Smit

Position: Service Delivery Team I SoDC Call Face NL
Interview by email regarding the usage of processes in his team
Operations and IT Banking ING, May 26, 2009

“Wij hanteren twee processen naast elkaar:
1) een proces wat we gebruiken in onze Siebel omgeving (Selly systeem) wat gebaseerd is op een agile DSDM achtige aanpak. Deze aanpak is ontstaan in de loop der jaren waarbij we steeds de best practices uit eerdere siebel onderhoudsprojecten hebben vervolmaakt, waarbij we ons zo nu en dan wat lieten inspireren door een RUP, DSDM of Agile boekje.
2) een proces wat we gebruiken in ons grote nieuwbouw project (Ringo systeem) wat gebaseerd is op RUP (de XDF aanpak). Deze aanpak is als gevolg van ondere uitbestedingsrichtlijnen en contractrichtlijnen veel meer waterval dan we zouden wensen. Hier zit dus een heleboel ruimte voor verbetering. Onze uitdaging is de succesvolle aanpak die we hanteren voor kleine onderhoudstrajecten toepasbaar te maken voor grootschalige nieuwbouw projecten. Wij zijn hierover in gesprek met ontwikkelproces goeroes binnen IBM. Zie het als een soort van “agile voor large systems” aanpak.”

“Voor de RUP/XDF hebben we gekozen omdat we grote nieuwbouwtrajecten moesten gaan doen met dusdanig grote volumes dat uitbesteding wel noodzakelijk was. RUP/XDF was een methode waar een heleboel best practices in zaten waar we ons goed in herkenden en bovendien was RUP een methode die zo’n beetje de standaard was bij onze software leveranciers Cap en IBM. Omdat we van mening waren dat het slim was om dezelfde taal te spreken als onze leveranciers hebben we ook voor RUP gekozen.”

“Agile voor large systems achtige aanpak heeft mijn voorkeur. Dit is een aanpak waarin we de voordelen van onze kleinschalige onderhoudsaanpak combineren met de voordelen van onze grootschalige nieuwbouw aanpak waarin uitbesteding een grote rol speelt. We denken met goede tooling (JAZZ bv) een aantal van de nadelen die aan uitbesteding kleven te ondervangen.”

“Onderhoudsreleasessiebel zijn kleine projecten met een core team van 3 configurators met daarom heen een paar testers, projectmanager gebruikersvertegenwoordiger en beheervertegenwoordiger.
nieuwbouwreleases Siebel zijn projecten van circa 300 tot 450 functiepunten per release, circa 3 releases per jaar. Nieuwreleases javawebsphere zijn grote projecten van circa 900 functiepunten per release, circa 3 releases per jaar. Vorig jaar hebben we zelfs een release gedaan van 2300 functiepunten maar dat project ging dan ook faliekant mis omdat het veel te groot was. We doen daarnaast ook nog kleinere projecten, de zogenaamde specials in de cobol hoek dan wel .NET hoek. daarvoor hebben we niet echt een standaard aanpak. In de .net hoek is het een RUP achtige aanpak, in de cobolhoek vaak een klassiek DSM dan wel VSOM (ja dat is nog een uit de oude doos, oorspronkelijk Volmac methode) achtige aanpak.”

APPENDIX IV – SURVEY

Help mee aan het ontwerpen van een beslissingsraamwerk betreffende software ontwikkel processen

Delft University of Technology
Faculty of Technology, Policy and Management

Operations and IT Banking ING (OIB ING)
Solution and Delivery Centre Call Face Retail NL

April, 2009

Deze enquête betreft het afstudeeronderzoek van Itamar Sharon. In dit onderzoek zal onderzocht worden in hoeverre de eigenschappen van een project binnen OIB ING invloed hebben op effectiviteit van een software ontwikkel proces. Voor dit onderzoek is het noodzakelijk informatie te verkrijgen van praktische aard. Ik stel het daarom zeer op prijs als u mij wilt helpen door deze enquête in te vullen. Deze enquête is bedoeld om een overzicht te krijgen van de factoren die invloed zouden kunnen hebben op het selecteren van een proces. Daarnaast zal er gevraagd worden in hoeverre u deze factoren van belang vindt. Aan de hand van uw informatie is het voor mij mogelijk een beslissingsraamwerk te maken die kan helpen in het bepalen welk proces geschikt is voor het betreffende project. Ik wil u alvast vriendelijk bedanken voor uw tijd. U kunt de ingevulde enquête digitaal opsturen naar mijn email, of, als u hem uitgeprint heeft, in mijn postvakje leggen op OL verdieping 5. Zou u dit alstublieft voor 30 april willen inleveren, in verband met afstudeer planning!
Met vriendelijke groet,

Itamar Sharon
E. itamar_sharon@hotmail.com
E. itamar.sharon@ing.nl
T. 0624770593

Delft University of Technology

De enquête begint met vragen betreffende uw werk binnen OIB ING en in hoeverre u ervaring heeft met software ontwikkel processen. Daarnaast wordt ook uw mening gevraagd betreffende deze processen. Na deze vragen zal de enquête zich vooral richten op de eigenschappen van projecten die een rol kunnen spelen in het selecteren van een proces.

Wat is uw naam en uw functie binnen OIB ING?

Antwoord:

Hoe lang werkt u al binnen OIB ING? Heeft u ook werkervaring buiten de ING? Zo ja, waar was dit en wat was uw functie?

Antwoord:

In welk opzicht bent u betrokken bij het uitvoeren van projecten?

Antwoord:

Bent u bekend met software ontwikkel processen (Waterval model, RUP, eXtreme Programming, DSDM etc.)? En zo ja, met welke processen heeft u dan ervaring en hoe lang al?

Antwoord:

Heeft u voorkeur voor een bepaald software proces in het algemeen en waarom?

Antwoord:

Heeft u voorkeur voor een bepaald software proces om toe te passen binnen OIB ING en waarom?

Antwoord:

Vindt u dat het toepassen van software ontwikkel processen op dit moment goed verloopt binnen OIB ING en waarom?

Antwoord:

Vindt u dat het mogelijk moet zijn om de keuze te hebben uit meerdere software ontwikkel processen voor een project of wilt u liever 1 standaard proces die voor alle projecten toegepast wordt? Zou u uw antwoord kort willen toelichten?

Antwoord:

In mijn onderzoek stel ik dat de keuze van een software ontwikkel proces afhankelijk is van het betreffende project. Voor dit onderzoek is het van belang te weten welke factoren dan van invloed zijn op het selecteren van een software ontwikkel proces voor een bepaald project. Ik wil een project opsplitsen in een aantal belangrijke factoren die het project karakteriseren. Het is dus van belang om te weten welke factoren van een project een rol kunnen spelen.

Aan de hand van het analyseren van gedane projecten en aan de hand van interviews met medewerkers binnen Channels, ben ik tot een aantal factoren gekomen. Ik zal kort toelichten wat het idee is achter deze factoren. Elke factor speelt een rol in een project. In elk project is deze factor anders. Bijvoorbeeld, het budget is voor het ene project vele malen hoger dan voor het andere. Dit heeft invloed op de toepasbaarheid van het software ontwikkel proces dat toegepast wordt voor het project.
Hieronder staat de lijst aangegeven met factoren die een rol zouden kunnen spelen in het selectieproces. Wilt u de lijst doornemen en aangeven of u het eens bent met de factoren door mij geselecteerd. U kunt de factoren die u niet van belang vindt doorstrepen in de tabel (of, als u het digitaal doet, rood maken). Mocht u nog factoren bedenken die niet in de tabel aangegeven staan, kunt u die aangeven in de lege vlakken.

Budget voor het project	Tijdsduur voor het project	Team grootte	Volwassenheid van de requirements
Verwachtte veranderingen in requirements	De invloed van andere afdelingen (Payments, Infrastructure etc.)	Relatie met het hele team (incl. leverancier, klant en andere afdelingen)	Bekendheid met de betreffende applicatie
Flexibiliteit van de klant (Hoe traditioneel is deze ingesteld)	De betrokkenheid van de klant	Duidelijkheid van de scope van het project	Duidelijkheid van de risico's
Stabiliteit van de omgeving	Methode van contracten afsluiten (Fixed price of Time/Material)		

Hieronder staan dezelfde factoren aangegeven. Het is noodzakelijk om te weten in hoeverre u de factoren van belang vindt. U kunt daarvoor een schaal van 1 tot 5 toepassen. Mocht u een factor zeer onbelangrijk vinden, dan kunt u een 1 neerzetten in de kolom. Als u de factor zeer belangrijk mocht vinden dan kunt u dit logischerwijs aangeven met een 5. Bijvoorbeeld, als u vindt dat het budget een belangrijke rol speelt in de toepasbaarheid van een ontwikkel proces, zoals RUP, geeft u bijvoorbeeld een 4. En vind u dat de invloed die andere afdelingen hebben in het project een onbelangrijke factor, dan kunt dit aangeven met bijvoorbeeld een 1. Op de stippellijn kunt u de factoren aangeven die u genoteerd heeft in de tabel hierboven. Ook aan deze factoren kan dan een gewicht gegeven worden.

Factor	**Gewicht 1-5**
Budget voor het project	
Tijdsduur voor het project	
Team grootte	
Volwassenheid van de requirements	
Verwachtte veranderingen in requirements	
De invloed van andere afdelingen	
Relatie met het hele team	
Bekendheid met de betreffende applicatie	
Flexibiliteit van de klant	
De betrokkenheid van de klant	
Duidelijkheid van de scope van het project	
Duidelijkheid van de risico's	
Stabiliteit van de omgeving	
Methode van contracten afsluiten	
…	
…	
…	
…	
…	

Ik wil u van harte bedanken voor het invullen van deze enquête. Deze informatie zal mij ontzettend helpen in het ontwikkelen van een beslissingsraamwerk. U kunt de ingevulde enquête digitaal opsturen naar mijn email, of, als u hem uitgeprint heeft, in mijn postvakje leggen op OL verdieping 5.Mocht u nog aanvullende reacties of vragen hebben kunt u mij altijd bereiken op:

T. 0624770593
E. itamar_sharon@hotmail.com
itamar.sharon@ing.nl

APPENDIX V – SURVEY RESULTS

From the survey, presented in Appendix IV, certain conclusions can be drawn. Below, four tables are presented. In the first table an overview is provided of the selected characteristics and the weights, given by personnel. Each respondent has expressed its opinion by giving each characteristic a number from 1 to 5. When the respondent finds the importance of a characteristic to be very low, the number 1 is given. When the characteristic is found very important, the number 5 is given. From these weights an average can be calculated. This is presented in the final column. When analyzing the results presented in table V-1 it is possible to state that some characteristics are found to be of no importance.

TABLE V-1 Weights given by personnel to the initial characteristics

Karakteristieken	**Gewichten**													**Som**	**Gem.**
Budget voor het project	2	0	5	1	2	1	5	5	1	2	-	1	5	30	2.5
Tijdsduur voor het project	4	5	4	1	2	3	4	4	3	2	-	3	5	40	3.34
Team Grootte	4	4	3	3	2	1	3	3	4	3	-	3	2	35	2.92
Volwassenheid van de requirements	5	5	4	3	4	4	5	5	5	4	-	4	4	53	4.42
Verwachtte veranderingen in requirements	4	5	5	4	4	4	3	4	5	4	-	4	4	51	4.25
De invloed van andere afdelingen	3	3	0	1	2	3	2	1	3	1	-	2	5	26	2.17
Relatie met het hele team	4	4	0	3	2	5	5	2	3	1	-	2	5	36	3
Bekendheid met de betreffende applicatie	2	3	0	2	4	3	2	4	3	1	-	1	4	29	2.42
Flexibiliteit van de klant	3	3	2	4	2	4	2	3	4	2	-	4	3	36	3
De betrokkenheid van de klant	2	5	0	4	0	5	4	4	4	3	-	4	5	40	3.34
Duidelijkheid van de scope van het project	3	5	5	1	3	4	5	5	3	1	-	5	5	45	3.75
Duidelijkheid van de risico's	4	5	5	5	5	5	3	4	3	3	-	5	5	52	4.34
Stabiliteit van de omgeving	4	3	4	1	0	2	5	3	2	1	-	3	5	33	2.75
Methode van contracten afsluiten	2	4	0	4	4	3	2	2	4	3	-	1	5	34	2.84

The survey gave the respondents the possibility to answer whether or not he or she found that certain characteristics should be deleted. This is presented in table V-2. Combining the results from tables V.1 and V.2, it is possible to conclude that four characteristics can be deleted (or changed). The characteristic "The influence of other departments" is seen as not important, and will be deleted from the framework. Furthermore, the characteristic "Familiarity with the application" will also be deleted. The characteristics "Budget for the project" and "Time Schedule" will be combined into the characteristic "Project Size". Other characteristics mentioned by the respondents are chosen not to be deleted. For example "Commitment of the client" was by some seen as not important. However, I decided to still implement this characteristic because I found it of great importance. In literature as well, it is stated that this characteristic influences the suitability of processes (Appendix VI).

TABLE V-2 Characteristics which should be deleted according to personnel

De invloed van andere afdelingen	Frank	Wouter	Wim	Jeroen V
Bekendheid met de betreffende applicatie	Frank	Wouter	Wim	
De betrokkenheid van de klant	Frank	Floris	Jeroen V	
Methode van contracten afsluiten	Frank	Wim	Jeroen V	
Budget voor het project	Wouter	Danny	Jeroen van B	
Relatie met het hele team	Frank	Jeroen V		
Stabiliteit van de omgeving	Wouter	Floris		
Team grootte	Danny	Jeroen V		
Flexibiliteit van de klant	Wim	Jeroen V		
Tijdsduur voor het project	Wouter			
Duidelijkheid van de scope van het project	Wouter			

As was mentioned before, the characteristic “Project size” was added to replace the characteristics “Budget for the project” and “Time schedule”. This is based on the opinions given by the respondents. Another characteristic that will be added to the framework is “Outsourcing yes or no”. This was mentioned by two respondents (see table V-3). Furthermore, in literature as well it is described to be a big influence on the suitability of software processes (Appendix VI). Other characteristics mentioned by respondents are chosen not to be implemented.

TABLE V-3 Characteristics recommended by personnel

Wel of geen outsourcing	Ed	Floris
Omvang van het project	Ed	Jeroen B
Omvang van de risico’s	Ed	
Targetplatform voor de implementatie	Ed	
Standaard die de leverancier gebruikt	Frank	
Ervaring team met het te hanteren proces	Wouter	
Externe economische situatie	Christien	
Nieuwheid gebruikte technieken	Christien	

In the survey, respondents were also asked to give answers to open questions. Two questions provided interesting information. These are presented in table V-4. The first interesting question is “Do you think that the application of software development processes within OIB ING is going well?”. Nine out of the eleven respondents answered this to be not the case. Therefore, this subject is definitely an important issue within OIB ING and confirms the statements made in the research definition. The other interesting question was “Do you want to have multiple software development processes to be applicable within the organization for different projects, or do you rather have one process which is tailored to fit different projects?”. This question concerns the main statement made in this research (which is that multiple processes should be available for different projects). Five out of ten respondents agreed with this statement. However, the other five found that a standardized process which can be changed to fit different projects is more favourable.

TABLE V-4 Other interesting results from the survey

Gaat het toepassen van processen goed?	Nee	Nee	Nee	Nee	Nee	Nee	Nee	Nee	Nee	Ja	Ja
Wilt u meerdere processen beschikbaar hebben?	Ja	Ja	Ja	Ja	Ja						
Of wilt u een standaard proces hebben?	Ja	Ja	Ja	Ja	Ja						

Within the survey respondents had the possibility to state their opinions regarding software development within OIB ING and in general. Below, the most interesting statements made in the survey are presented:

Roel Noback (IT Architect): *“RUP heeft mijn grote voorkeur. Is te tunen richting beide kanten van het spectrum (waterval en eXtreme Programming), afhankelijk van de situatie. Ik heb afdoende slechte ervaringen met waterval methodes, vooral in uitbestedingstrajecten. Ik denk dat met name het risk mitigation aspect (tackle eerst de risicovolle delen) van RUP van grote waarde is voor dit soort trajecten.”*

Roel Noback (IT Architect): *“Daar waar RUP wordt toegepast is het vaak RUP in name only. Daar waar waterval methode wordt gebruikt is denk ik een foute keuze gemaakt.”*

Ed Schimmer (Senior Business Analyst): “*De keuze van proces is afhankelijk van het soort project (o.a. schaalgrootte) en de doelstellingen van het project. Er wordt binnen ING al snel gekozen voor* *watervalmethode, en te weinig voor een aanpak als bv RUP. Consequentie is vaak dat projecten te groot zijn en te lang duren, met een aanzienlijke kans op mislukking."*

Ed Schimmer (Senior Business Analyst): "*Er zouden meerder keuzes mogelijk moeten zijn. In veel gevallen zal de watervalmethode nog gebruikt worden, bv bij Betaalsystemen gericht op massale verwerking. De RUP-methode en XDF moeten meer gepromoot worden, zeker waar delen van de ontwikkeling worden geoutsourced. Bij kleinere projecten zou meer aandacht moeten zijn voor kortcyclische ontwikkeling (DSDM)."*

Frank Hollander (Senior Manager OIB): *"Liefst twee standaard methodieken – RUP/Waterval. Een derde alleen als het echt nodig is. Waarom:*
1) RUP is industrie standaard (voor zover je dat kan zeggen binnen IT)
2) Waterval "kent" iedereen."

Wouter Blokdijk (IT Architect): *"Ik heb gemerkt dat zodra er veel druk van hoger management (level 2 en hoger) op een opdacht komt dat er steeds minder proces in een project komt (meer focus op eindproduct dan proces). Zodra er minder strakke business aandacht komt zie ik meer proces en verminderde focus op het eindproduct (werkende software)."*

Floris Zwart (Senior Architect): *"Vanuit een Governance point of view heeft 1 proces de voorkeur. Dat vergroot de transparantie over budgetten, uitnutting, etc."*

Danny Wijnand (Delivery Unit Manager): *"Rational Unified Process aangezien het markstandaard is, Agile als er echt iets nieuws gemaakt moet worden en nog alle kanten op kan gaan (vereist flexibele houding opdrachtgever)."*

Christien Bergman (Manager Requirements Management): **"***Het liefst zou ik zien dat we meerdere processen in de vingers hebben (een beperkt aantal, 3 lijkt me maximaal) en de meest geschikte kiezen voor die specifieke situatie. Dat komt de resultaten volgens mij ten goede. Ik ben echter bang dat dit voor OIB als geheel een brug te ver gaat zijn. Nieuwe processen adopteren is wel heel lastig voor deze organisatie."*

Jeroen van Barneveld (Manager Systems Management): **"***Afhankelijk van de situatie. Als 1 proces afdoende is, heeft dat de voorkeur (vanwege uniformiteit, maar 1 procesbeschrijving nodig, etc). Als meerdere situaties bestaan waarvoor niet 1 proces in alle gevallen het beste aansluit, kan ook gekozen worden voor meerdere processen."*

Ed van Kooten (BPI Manager): "*Ik opteer voor 1 (markt) standaard; projecten zijn beter te vergelijken, project medewerkers zijn makkelijker uit te wisselen tussen projecten, minder opleidingskosten, uitbesteding wordt eenvoudiger.*"

Jeroen Busscher (Account Manager): *"Waterval en rup hebben mijn voorkeur. Indien de gevraagde change vrij helder is kan waterval aanpak zeer geschikt zijn. Indien vertrekpunt minder helder is en er zijn meer risico's gesignaleerd dan kan rup geschikt zijn. De 1^{e} iteraties(s) worden dan met name gebruikt om risico's te mitigeren. Indien de change (erg) groot is (>5000 manuren zou ik zeggen) dan heb ik voorkeur voor rup aangezien je dan het traject beheersbaarder kunt sturen."*

Daniel Busscher (Account Manager): "De bureacratie zou verminderd moeten worden in relatie tot het aantal partijen dat meedoet binnen het proces. Het proces an sich is best goed. Het succes staat of valt met de organisatie discipline binnen het ontwikkelproces en de bereidheid bij opdrachtgever/supplier om elkaar af en toe de hand te reiken."

APPENDIX VI – CHARACTERISTICS FROM LITERATURE REVIEW

In chapter four characteristics were found which influence the suitability of software development processes. Some of these characteristics are found in literature. Below an overview of sources is given which describe (one way or another) the characteristic. In most of the sources no connection was made between a characteristic of software development processes and a characteristic of a software project. Therefore, some characteristics needed to be interpreted from the sources.

Characteristic	**Literature Reference**
Budget	(Ambler et al, 2005), (Boehm, 1988), (Dybå & Dingsøyr, 2008) (Pressman, 2005) (Reifer, 2001) (Runeson & Greberg, 2004) (Sommerville, 2007) (Vliet, 2008)
Time Schedule	(Boehm, 1988), (Dybå & Dingsøyr, 2008) (Pressman, 2005) (Runeson & Greberg, 2004) (Schwaber, 1995) (Sommerville, 2007) (Stapleton, 2003)
Team Size	(Beck & Fowler, 2001), (Dybå & Dingsøyr, 2008), (Hunt, 2006), (Leffingwell, 2007), (Nebulon Pty Ltd, 2009) (Palmer & Felsing, 2002) (Pressman, 2005) (Rising & Janoff, 2000) (Runeson & Greberg, 2004) (Schwaber, 1995) (Sommerville, 2007) (Stapleton, 2003) (Vliet, 2008)
Maturity of Requirements	(Ambler et al, 2005), (Dybå & Dingsøyr, 2008), (Leffingwell, 2007) (Reifer, 2001) (Sommerville, 2007) (Vliet, 2008)
The Number of Changes Expected in the Requirements	(Ambler et al, 2005), (Beck & Fowler, 2001), (Dybå & Dingsøyr, 2008), (Hunt, 2006), (Leffingwell, 2007) (Palmer & Felsing, 2002)

	(Pressman, 2005) (Royce, 1970) (Reifer, 2001) (Runeson & Greberg, 2004) (Schwaber, 1995) (Sommerville, 2007) (Vliet, 2008)
The Role of Other Departments	
Relationship Within the Entire Team	(Dybå & Dingsøyr, 2008), (Leffingwell, 2007) (Palmer & Felsing, 2002) (Rising & Janoff, 2000) (Runeson & Greberg, 2004) (Vliet, 2008)
Commitment of the Business Partner	(Beck & Fowler, 2001), (Boehm, 1988), (Dybå & Dingsøyr, 2008), (Leffingwell, 2007), (McBreen, 2003) (Palmer & Felsing, 2002) (Runeson & Greberg, 2004) (Sommerville, 2007)
Scope Clearness	
Risk Clearness	(Boehm, 1988), (Dybå & Dingsøyr, 2008) (Pressman, 2005) (Royce, 1970) (Schwaber, 1995) (Sommerville, 2007)
Familiarity with the Application	
Stability of the Environment	
Flexibility of the Stakeholder	
Method of Contracting	

APPENDIX VII – VALIDATION

Experimental Validation Questionnaire

Testing the framework by using actual executed projects
The Questionnaire:
Which scales did you give the characteristics? (Present the numbers below) **Christien**: 4,5,3,2,2,3,1,5,2,Fixed,Yes **Wouter**: 2,2,2,4,4,3,2,2,4,Fixed,Yes **Ed**: 3,3,2,4,3,2,2,2,2,Time/Material,Yes
Is the questionnaire clear for you as user? **Christien**: Yes, the questionnaire is clear. However, I would suggest adding words to the scales. This clarifies what the highest scale is and what the lowest. Furthermore, I really like the style in which the framework is made. It really feels like ING. **Wouter**: Yes. Environmental Stability was not clear. It would be good to add a explanation tab to the framework. **Ed**: Yes it is very clear.
Is the usage of scales clear? And what is your opinion regarding this method? **Christien**: Again, I would suggest adding words to the scales. I like the weights that are given. It makes the feeling of project managers disappear and gives a research based score. I would suggest implementing a final tab in which you explain the entire framework. It should be clear without reading the entire report. **Wouter**: The scales are not clear. I would suggest to include the value of the scales. For example project size and team size as an absolute number. **Ed**: No, the scales are not clear for me. I would suggest to use absolute numbers and the questions should be more concrete. Own interpretation of a question should not be possible.
The Conclusion:
After applying an actual project case, Is the conclusion clear for you? **Christien**: Yes the conclusion is clear for me. I suggest changing the setting by placing the calculation table above the table with the green and yellow score. Furthermore I think the 25% separation is very good. However, I would explain why you chose this number. **Wouter**: The conclusion is clear. However, in my result, FDD is most suitable. I do not know this process. I would suggest to add a linking page to let the user get information regarding these processes. **Ed**: Yes, clear.
Do you think the outcome is viable for your project? **Christien**: It is. I agree with the result. Waterfall and RUP are in my opinion as well the better processes for this kind of project. **Wouter**: I do not know FDD. But RUP and DSDM are also suitable which I think are indeed viable. **Ed**: I expected RUP and this was the case. This is correct.
Calculation:
Are you as user interested in the calculation tab? If so, do you understand the steps taken, or would you like more information? **Christien**: Yes and No. Yes because it gives insight which is very important. No, because managers might use it to alter the final result. There are some things I would like to see in this tab: the score given in the questionnaire should be presented here as well. And I would like to get an alert when a certain process is concludes as most suitable, but on a certain characteristic it is not suitable at all.

Wouter: I am very interested in this tab. It shows the basis for the framework. I would suggest to add the scales indicated in the questionnaire here. Furthermore, I would also add an explanation regarding this tab.
Ed: I am very interested in this tab. I would suggest to add the scales given in the questionnaire and include an extensive explanation of the entire framework.

Testing the framework by applying illusory projects

Questionnaire:

Which illusory projects did you apply? (For example; Simple & small or Big & complex)
Illusory Project1:
Christien: Small problem release project on existing application: 1,1,5,5,5,5,5,3,1,Fixed,Yes
Wouter: 5,5,2,2,5,3,4,3,5,Time/Material,Yes
Ed: 2,2,1,5,3,1,2,1,4,Time/Material,Yes

Illusory Project 2:
Christien: Very big project (15 million euro project): 5,5,1,2,3,1,4,1,4,Fixed,Yes
Wouter: 3,4,4,4,2,4,3,1,4,Time/Material,No
Ed: 4,4,4,4,2,4,4,4,2,Fixed,No

Illusory Project 3:
Christien: Same as above, changing the 7th char into scale 2 and 10th char into scale Time/Material
Wouter: Same as above, changing the 8th char into scale 4 and Fixed Pricing and no Outsourcing

Conclusion:

Do you agree with the outcomes for each illusory project?
Illusory Project 1:
Christien: Yes I really do
Wouter: I expected a squeeze to RUP and this is the case.
Ed: I agree with the final result. However, I thought that DSDM would be a bit less viable than stated in the framework.

Illusory Project 2:
Christien: I thought that RUP would be a bit stronger
Wouter: I expected RUP or Waterfall. The conclusion now is RUP which I agree on.
Ed: I expected the Waterfall model. The conclusion however was RUP. I understand this is the case and clearly agree with this. I understand that Waterfall is in fact a very limited process and should only be used in extremes.

Illusory Project 3:
Christien: Really Surprised! It is a real eye opener. It shows that this project should be executed with Scrum. However, the result actually shows that this project is impossible to succeed. It shows that the project is way to big with to high risks. So I am really surprised, but I do understand the result.
Wouter: I would expect Waterfall here. However, it is RUP. I see that the project should be in extremes for Waterfall to be the conclusion. I am not really sure about this, but I think it might be correct.

General Discussion:

All aspects:

What is your opinion regarding the functionality of the decision framework?
Positive:

Christien: The framework is simply correct. It works very pleasant and it is easy accessible.
Wouter: It forces you to take a good look at your project. It will be a confrontation for many users which I think is very good.
Ed: The possibility to change the weights in the framework is very good. Furthermore, the conclusion is very clear.

Negative:
Christien: When a certain process is concluded to be most suitable, while on some characteristics the score is very low, a certain notice should be stated. For example, Scrum is best. However, your team size is very big, so this might become a problem -> think about this.
Wouter: The framework suggests that we know all processes. However, this is not the case.
Ed: It is really lacking explanation. Furthermore, in the current framework own interpretation is very viable, which should not be the case.

Which improvements would you suggest?
Christien: Implement red flags for example.
Wouter: Links to process pages with explanation, and a general explanation of the entire framework. Maybe take a better look at the scales. I would suggest more absolute numbers, or changing the scales from 1 to 3 instead of 1 to 5.
Ed: One thing I would like to mention is that in my opinion some characteristics have relationships with each other. For example: If the risks are very low, it might be possible that other characteristics lose their importance.

Would you use this framework when starting up a project at OIB ING?
Christien: Yes, definitely! Especially for learning purposes. It can really show people that the way we are working now is less appropriate. Your framework can be a real eye opener. I think your framework will definitely have impact not only on our department but on entire OIB.
Wouter: Yes. It is a great tool to convince yourself and the client. Furthermore, it forces you to better analyze your project.
Ed: Yes. The framework triggers you to think about your project. Which is always very good.

Face Validation Questionnaire

General questions regarding the validation of the framework:

Design of the framework:

Is the concept used appropriate to assist project managers in selecting a suitable software development process?
Jeroen: Well yes. The concept is appropriate. I do think everything could be much smaller. Why use all the room for each question?
Jos: Yes it is. All relevant characteristics are included.

Are the steps made in the framework clear?
Jeroen: No. I need guidance which is not available. Furthermore I do not understand the questions. Your questions suggest that I already know what the exact scales are. This is usually not the case.
Jos: The steps are clear. However, I do need more elucidation. The characteristics could be more clear and the *Project Size* and *Team Size* should be noted in absolute numbers.

Results of the framework:

After experimenting with different possibilities (and thus different possible projects) do you agree with the outcomes?
Jeroen: Partially I agree.
Jos: Yes I do. However, I do think the Waterfall model is a bit too strong. I would never use this process for any project (A project of Rijkswaterstaat was tested).

Do you agree with the outcomes for each characteristic individually?
Jeroen: Yes I do.
Jos: Yes I do.

Usability of the framework:

Do you think it is useful to apply this framework in a large-scale organization?
Jeroen: I think there are many things missing in this framework (which I stated below). I do think this framework is useful. It makes you think about your project and its characteristics. This is always imperative in a large organization.
Jos: It is a very nice method.

Is the framework an appropriate tool to assist project managers in selecting a suitable software development process?
Jos: Yes.

What improvements do you suggest?
Jeroen: First of all, I suggest to add some explanation. Furthermore, the questions in the questionnaire are not that clear. The most important aspect is relationships. In this framework every characteristic is seen as an individual part of a project. However, I think that certain characteristics are dependent of each other. Try to take a good look at this.
Jos: It would be interesting to find out if it is possible to use this tool for individual elements of a project. So, can you use multiple processes for 1 single project. For example: Infrastructure can be done with Waterfall model and the application can be done with RUP.
Another improvement would be to include more information. A good option is to create “clouds” of information which appear when scrolling over an element.