Project Suitability for Agile methodologies
Acknowledgements

The authors of this thesis would like to thank all the people supporting us during the process of this research and helped us with suggestions along the way.

Our supervisor Ralf Müller, for his guidance and constant support which helped us to stay focused from the beginning to the end of the research process.

Our parents who has always believed in us and had always supported us at the moments when would feel down

Richard Grove a Senior Project Manager at Webcor Builders (USA) who has helped us with his invaluable advices throughout the research process.

All the interview and questionnaire participants who had found time to take part in our research and provided us with necessary data.

The thesis would not be possible without people mentioned above

We have enjoyed conducting this research field and exploring the topics which were new for both of us.

Besiana Alite and Nikolay Spasibenko
Umea University, December 2008
Abstract

Software projects are known for their failure rate, where many are being delivered late, over budget or being canceled while in development. The reason to it is changing requirements and intangibility of the software. Being so abstract it is difficult to imaging all the aspects of the software at the requirements stage. Also technology is playing a major role since processing power, storage space, and data transfer speeds are improving from year to year.

Agile methodologies are addressing projects with unclear requirements making process of implementing new specifications along the project much easier and less costly. However the success rate of the software projects did not improve much since the introduction of Agile methodologies. This thesis is looking at what type of projects fit different methodologies and what are factors which practitioners should take into account when selecting methodology for a particular project,

The thesis opens up with introduction which sets the research question and provides a brief background to the research topic. In subsequent chapter literature review is conducted to find out what does literature and other researchers have said on the same topic. Third chapter discusses underlying research philosophy and discusses the data collection tools. Next chapter discusses the findings of the research. Interviews has been conducted with project management professionals from Sweden, US, UK and Canada. It was identified through the analysis of patters that Agile methodologies are not well suited for projects involving databases, embedded development and computationally complex projects. Through the analysis of the questionnaire several project characteristics were identified which suit Agile methodologies better than traditional ones: unclear requirements, high risk of failure etc… In the last chapter the thesis concludes the findings and its theoretical and practical implications.
# Table of Contents

Acknowledgements ........................................................................................................................................ 1  
Abstract .................................................................................................................................................... 2  
List of Figures and Tables ........................................................................................................................... 5  
Chapter 1: Introduction ................................................................................................................................. 6  
  1.1 Background ........................................................................................................................................ 6  
  1.2 Purpose of study ................................................................................................................................. 7  
  1.3 Research Objective ............................................................................................................................. 7  
  1.4 Organization of the thesis ................................................................................................................... 7  
Chapter 2: Literature Review ......................................................................................................................... 9  
  2.1 Information Sources ............................................................................................................................ 9  
  2.2 Literature categorization ..................................................................................................................... 9  
  2.3 Software Project Management ............................................................................................................ 9  
    2.3.1 Success Factors .......................................................................................................................... 10  
  2.4 Software Development Methodologies ............................................................................................... 11  
    2.4.1 Category 1: Traditional methodologies ..................................................................................... 11  
    2.4.2 Category 2: Agile methodologies ............................................................................................... 13  
  2.5 Agile Suitability Filters ......................................................................................................................... 17  
  2.6 Agile vs. Traditional Methodologies .................................................................................................... 20  
  2.6 Research Gap .................................................................................................................................... 21  
Chapter 3: Methodology ............................................................................................................................... 22  
  3.1 Underlying philosophy ....................................................................................................................... 22  
  3.2 Mixed Methodology ............................................................................................................................ 23  
  3.3 Inductive Approach ............................................................................................................................ 23  
  3.4 Data collection through interviews ................................................................................................... 24  
  3.5 Data Collection through questionnaires ........................................................................................... 25  
  3.6 Validity and Reliability ....................................................................................................................... 28  
  3.7 Ethical issues ..................................................................................................................................... 28  
  3.8 Analysis and presentation of results ................................................................................................. 29  
  3.9 Research process .............................................................................................................................. 29  
Chapter 4: Findings and discussion ............................................................................................................. 32
List of Figures and Tables

Figure 1: Waterfall Model (Adopted from Cantor (1998)) ................................................................. 11
Figure 2: Spiral Development Model (Adopted from: Bechtold (1999)) .............................................. 13
Figure 3: Comparison of the methodologies. (Adopted from: Baird (2002)) ...................................... 14
Figure 4: Scrum Process (Adopted from: Koch (2005)) ................................................................. 16
Figure 5: DSDM process diagram (Adopted from: Koch (2005)) ...................................................... 17
Figure 6: Cockburn’s Criticality and Team Size Factors (Adopted from: Cockburn (2002)) .......... 19
Figure 7: Radar Chart (Adopted from: Boehm and Turner (2003)) .................................................... 19
Figure 8: Slider Tool (Adopted from: Leadinganswers.typepad.com (Online)) ............................... 20
Figure 9: Research onion (Adopted from Saunders et al (2003)) ...................................................... 22
Figure 10: Inductive reasoning (Adopted from Socialresearchmethods.net (Online)) ...................... 23
Figure 11: Demographic data for interview participants ................................................................. 30
Figure 12: Agile methodologies demographic data ........................................................................... 30
Figure 13: Traditional methodologies demographic data ............................................................... 30
Figure 14: Questionnaire Scale ........................................................................................................... 37
Figure 15: Requirements ..................................................................................................................... 38
Figure 16: Problem Complexity ......................................................................................................... 39
Figure 17: Risk of failure ..................................................................................................................... 39
Figure 18: Quality required .................................................................................................................. 41
Figure 19: Commitment to testing ....................................................................................................... 43
Figure 20: Results ............................................................................................................................... 45

Table 1: XP Life Cycle. (Adopted from: Larman (2003)) ................................................................. 15
Table 2: Agile applicability based on project type and organizational stakeholders. (Adopted from: Chin (2004)) ......................................................................................................................... 18
Table 3: Interview participants ....................................................................................................... 24
Table 4: Questionnaire respondent matrix ........................................................................................ 27
Table 5: Requirements Phase ........................................................................................................... 38
Table 6: Design Phase ...................................................................................................................... 40
Table 7: Development Phase ........................................................................................................... 42
Table 8: Deployment Phase .............................................................................................................. 44
Chapter 1: Introduction

1.1 Background

Steinmueller (1995) discusses that in past 50 years information technology has been revolutionized by innovations in data storage devices, semiconductors etc… This allowed for the creation of new industries and transformation of the old industries. Use of technologically advanced devices is being an important part of everyone’s life. Behind each of this device is a piece of code (software), which allows for these devices to function properly. Jurisson (1999) mentions that almost all of the software development efforts can be undertaken as projects. “A project is a temporary endeavor undertaken to create a unique product, service, or result” (PMI, 2004, page 5). The projects are being run in the variety of organizations and industries. They range in sizes and their visibility. Software projects are complex with their development taking place in dynamic environments where technology and business conditions can change any moment. Project Management is an organized approach which allows managing teamwork complexities and complex interrelations between different tasks. (Jurisson, 1999) Project Management as defined by APM (2006, page 2) is “the process projects are defined, planned, monitored, controlled and delivered such that the agreed benefits are realized”.

As mentioned by Gardiner (2005) the field of project management research is best approached from the systems theory. The system is defined by PMI (2004, page 377) as “an integrated set of regularly interacting or interdependent components created to accomplish a defined objective”. Thus system could be viewed as means of translating inputs, using different process into outputs. Similarly project management is a tool of translating inputs such as business need and requirements into outputs such as new product or service. The processes within the system can be very simple or extremely complicated. These processes would depend on the nature of the project, its complexity, size, etc…Within software project management it is possible to identify different sets of processes and guidelines for managing these processes which are referred to as project methodologies.

There are many different software development methodologies in existence; they will be explored in greater detail in Chapter 2, however it is possible to differentiate between its major groups,

Boehm (2002) differentiates between plan-driven and Agile (light weight) methodologies. Plan driven methodologies are following a particular well documented plan and progress in successive phases with well defined milestones, from product requirement definition all the way to the software maintenance phase.

Agile uses iterations where a major deliverable goes all the way from requirements stage into production in short, defined period of time. Planning in Agile projects takes place for every iteration, rather than for the whole project upfront. In plan driven methodologies all the
specifications are done upfront thus making it very costly in case a last minute change would come. Agile methodologies in their turn do welcome change at any time, thus making it very efficient for the projects with unclear requirements or where the requirements are subject to change at any moment. Both types of methodologies have its advantages and disadvantages.

As mentioned by Boehm (2002) each of the methodologies works well within its home ground of project characteristics, however it will not do well if those characteristics do not match. Thus there is no generic (use anytime methodology), so it is up to project manager to choose the most suitable methodology for the project in hand. The choice of methodology will either make or break the project. It is crucial for the project success to use right methodology for the right situation.

1.2 Purpose of study
The initial idea that some projects are more appropriate to be performed with Agile methodologies rather than with traditional ones has come as a result of an increasing interest in Agile methodologies. More and more companies are adopting it successfully nowadays. There are specific interest groups (SIG) focused on Agile development present on LinkedIn and GanttHead networks, one of this SIGs is named as a “Right Standard for the Right Project” acknowledging the fact that there is no fits-all methodology, every project is unique therefore it should be approached from different position.

The purpose of this thesis is to investigate a knowledge gap on which software project types fit different approaches to software development. Such research would provide managers with the knowledge which can help to choose one methodology over another for the particular project type. It will also create a basis for the subsequent academic research into the topic. The unit of analysis for this research is type of projects.

1.3 Research Objective
A research objective of this thesis is to identify which software projects are better suited for Agile methodologies and which for traditional ones. Therefore the research question is:

For which projects it is appropriate to use Agile methodologies?

Additional Questions in order to define the research area:

- What are the differences between Agile and traditional methodologies?
- What tools are available to choose the methodology depending on the project?
- Why software project management is different from a traditional one?

1.4 Organization of the thesis
The thesis starts with introduction which states the research objective and purpose of the study, Second chapter represent literature review, which discusses what other authors have, wrote on the topic of this study. Third chapter discussed methodology, research philosophy and research
methods employed in this thesis. Fourth chapter discusses findings of the study. And then chapter five contains summary of the results and discussed managerial and theoretical implication of the research as well future areas of investigation.
Chapter 2: Literature Review

The following chapter presents literature review findings on Agile and traditional methodologies; it discusses the field of software project management and provides information on the most common traditional and Agile methodologies. Towards the end of the chapter the existing filters through which projects should pass in order to qualify for use of Agile methodologies are discussed as well as pros and cons of traditional and Agile methodologies.

2.1 Information Sources
The information used for this literature review has been gathered from publications on software project management in journals such as IEEE Software and IEEE Computer, conference proceedings, books and Internet databases in particular Emerald, Ebscohost, ACM digital library and Google Scholar. The keywords used for search are: Agile versus Traditional (Plan–driven) methodologies, Agile, Waterfall, Agile filters, software project management, software development process, software project methodologies, Agile advantages and disadvantages etc…

2.2 Literature categorization
Literature used in this review falls into three categories: 1) General literature on software project management, 2) Literature on Agile methodologies 2) Literature on traditional (plan-driven methodologies).

2.3 Software Project Management
PMI (2004, page 8) defines Project Management as “the application of knowledge, skills, tools, and techniques to project activities to meet project requirements”. Simply speaking project is a planned non-routine activity with known time span. The software projects are not exclusion in this case, they adhere to the similar constraints as the other projects: cost, quality and time. However as mentioned by Hughes and Cotterell (1999) software project have some certain characteristics, which make them different from all the other projects. First of all its invisibility, software is not tangible, therefore the progress of work is not usually visible. Software is a complex product; it contains more complexity per unit of money spent, then any other engineering projects. The flexibility with which software can be changed is usually seen as an advantage; however it is being a disadvantage at the same time, since it is expected to be changed or adjusted to the system it interfaces. Stepanek (2005) adds to the list that in software project management requirements are usually incomplete and as mentioned previously changes in scope are inevitable. Also technology is playing a significant role; Intel.Com (2008, Online) describes Moore’s law which states that number of transistors on chip will double every two years, with increase in number of transistors comes the increase in processing power, thus creating more opportunities for software developers, who are less and less confined by platform limitations. The discipline is also very broad i.e. an operating system requires many different technologies in order for it to be developed; a single individual cannot simply know and
understand all of them. The best practices in software development are not mature yet, presenting additional challenges.

For the purposes of this thesis, it is important to give definition of project success: the traditional definition that project is successful, when it is delivered within budget, on time and with quality required, is simply not enough as mentioned by Brandon (2006). Procaccino et al. (2006) have conducted a study, which identified that project is perceived successful by development team, when the requirements of customers/users were met by completed system. Gardiner (2005) also adds that projects need to be aligned with organization’s strategy. The project, which does not align with the strategy, can be deemed as a waste of time and resources, which could have been assigned to the project with greater priorities. Therefore the definition of the success for the software project as perceived by the authors of this literature review can be extended to – project delivery on time, within budget and quality required, meeting all the requirements of client and creating business value, by aligning it with organizational strategy.

2.3.1 Success Factors

There have been numerous studies identifying success factors for projects; Pinto and Slevin (1987) list some of them: Project objectives, Top management support, project planning, communication with client, human relations, technical tasks, client acceptance, project control, communication and problem handling. The factors mentioned are not specific for software project management Therefore it is important to look at what are the problems and challenges faced by software projects managers, and how can the risks of running into these problems be avoided or their impact reduced.

Software projects are known for their failures to meet one or more of three constraints mentioned previously. Brandon (2006) mentions with reference to Standish Group CHAOS report that in 1994 only 16% of all IT projects were delivered on time and within budget, this numbers have changed in 2004 when the success rate was 28%, however it was lower than in 2003 when it was 34%. The failure rate increases together with the size of the project, as its complexity increases. Lientz and Larssen (2006) identify some of the reasons for failure of IT efforts: issues are detected too late; issues are not managed well, the same mistakes done over and over again. Brandon (2006, page 14) also lists the major causes of IT project failures over the last years – “lack of end-user involvement; lack of executive support; poor project management and/or planning; unclear business justification; and problems with requirements, scope methodology and estimation”. In order for IT project to have a better chance for success an organization and project manager should address the factors mentioned above. It is evident that the choice of correct methodology and its proper application can address some of the issues mentioned previously.
2.4 Software Development Methodologies

The literature on Software Methodologies can be divided into two categories:

- Category 1: Traditional methodologies
- Category 2: Agile methodologies

The common thing between these two categories is that literature in both of them mostly prescribes a chosen methodology as the way forward and being the best solution available. However, Brooks (1987) back in 1987 had warned that there is no silver bullet available to slay the “werewolf”, who software projects turn into when missing schedules and exceeding the budgets. Even though those words were written more than twenty years ago, they are still applicable even today and have been quoted by Shore and Warden (2008) in the introduction to their book on Agile development. Methodologies are not able to offer a solution to all problems; they just increase the probability of success if applied correctly.

2.4.1 Category 1: Traditional methodologies

According to McConnell (1998) the traditional plan based methodologies in their majority are based on the Waterfall model. Thomset (2002, pages 136, 137) describes waterfall as the model where the application is being developed as a “whole with each phase as a stand-alone activity, and subsequent phases are not commenced until preceding phases are complete”. He also mentions that its strengths are the simplicity and ease of scheduling, however he point out that the main weakness of the waterfall model is that “it is poorly suited to chaotic and client-driven business environment of the 21st century”.

![Waterfall Model](image)

The other weakness of the waterfall model is that it is not suitable for changing requirements; a change would mean coming back to the requirements phase and starting all over. Waterfall model comes in many implementations and as mentioned by Cantor (1998) most of these implementations place milestones and formal review after each phase.
Bechtold (1999) states that waterfall methodology is highly effective for well understood projects with a short time span and fixed requirements. However, it should be pointed that in today’s fast paced business environment project requirements are unlikely to stay stable; therefore it is a very limited number of projects which may benefit by using waterfall model. Sodhi and Sodhi (2001, page 196) discuss that waterfall model is not popular between object-oriented software developers, due to the fact that “more interactions are needed between successive steps and identification for object-oriented software is not an easy job”.

McConnell (1996) discusses the whole range of modified waterfall methodologies: sashimi (waterfall with overlapping phases); waterfall with subprojects; waterfall with risk reduction. All these methodologies are addressing weaknesses of the traditional waterfall model.

Another traditional software development model is Spiral as mentioned by Sodhi and Sodhi 2001 it is another variation of traditional waterfall model; however it does address deficiencies of its predecessor. The idea of the model is that development team goes through all the stages over and over again until finally delivering the final product. The development team start in the middle of the model, does throughout assessment of the situation and then proceeds through the stage. At the end of each stage development team presents customer a Build (an intermediate version of software), on which customer reflects and it is these reflections which are being fed into the next stage. The plans are developed in detail only for the current phase of the spiral, thus avoiding developing unrealistic plans far into the future. Bechtold (1999) states that Spiral methodology is highly effective when the customer doesn’t know what the final version should look like and what features it should have. The other advantage is that if the project is going to fail it will usually fail early through development, thus saving budget and time, which could have been invested into it. The reduction of risk on each cycle also serves as an additional advantage. Sodhi and Sodhi (2001) add to it that model is being favored by object-oriented programmers due to its flexibility in managing software life cycle.
Cantor (1998) mentions that spiral model is overly elaborate and therefore difficult to adopt. It can be very expensive to adopt and requires a careful content management. McConnell (1996) adds to it – whenever the project is simple enough and risks are low it should be performed with traditional waterfall model; it will not yield any advantages by using Spiral, as it does not require all the flexibility offered by the Spiral model. Sodhi and Sodhi (2001) state that the weakness the model is its lack of matching with existing standards, it also needs to be more uniform and consistent.

2.4.2 Category 2: Agile methodologies

Boehm (2006) discusses that Agile methods have emerged in the late 1990; and include methodologies like: eXtreme Programming (XP); Scrum; Dynamic Systems Development (DSDM); Feature Driven Development; Crystal and Evo. Highsmith (2004, page 5) mention that Agile methods target development of new products as well as enhancing products in the domains of: Software products (PC applications, OS, ERP); Industrial products with embedded software; and “internally developed IT products that fit the speed, mobility, and exploration factor criteria”. This thesis will look primarily into XP, Scrum and DSDM methodologies; the primary reason for it, is the availability of data, since there is an extensive amount of literature available on these methods and they are used by many organizations worldwide as mentioned by Boehm (2006), Larman (2003), Barnett and Schwaber (2004). Agile methodologies instead of avoiding risks mitigate though close interaction with the customer. (Koch, 2005) Agile also has very different approach to quality as mentioned by Koch (2005, page 134) – “The Agile methods change the ground rules on which the traditional approach to QA is based by making the requirements specification a moving target”, thus defining quality as a fitness for business purpose. In 2001 an Agile manifesto was issued by the proprietors of the above given methodologies. The four main values preferences of Agile are:

- **Individuals and Interaction** over processes and tools
- **Working software** over comprehensive documentation
• **Customer collaboration** over contract negotiation  
• **Responding to change** over following a plan

*Adopted from: Boehm B. (2006)*

Baird S. (2002) has compared different methodologies in terms of their flexibility (how they accept change) and quality (defects and accuracy of the product).

![Figure 3: Comparison of the methodologies. (Adopted from: Baird (2002))](image)

As it can be seen from the figure above XP approach is able to meet customer expectations by accepting changes as well as deliver quality product. As mentioned by Barnett and Schwaber (2004) XP methodology addresses processes required for a software development project, it does not address project management activities. The XP methodology lifecycle starts with exploration which is followed by cycle of planning and iterations until the final version is ready. During the iteration there are several build being released, which developer update with their code. The number of these iterations is not fixed, and they are being performed until the final version is suitable for production phase. Beck (2000) a creator of eXtreme Programming described its practices:

• The Planning Game – quick determination of the scope for next release and its update  
• Small releases – the system appears in production as soon as possible and then updated using short cycles  
• Metaphor – all development is guided with simple shared story of how the system works  
• Simple design – the system should be as simple as possible, without extra complexity  
• Testing – tests are continually being developed, and require to be passed flawlessly in order for development to continue  
• Refactoring – developers continually improve the system in order to remove duplication, improve communication and add flexibility  
• Pair programming – all code is written by 2 people using one machine  
• Collective ownership – anyone on the team can change code at any time  
• Continuous integration – system is being integrated and build every time the task is completed  
• 40 hour week – it is a rule not to work more than 40 hours per week  
• On-site customer – the customer is available to answer questions at any time
- Coding standards – The code is written in accordance with the rules emphasizing communication through the code.

The are many organizations, which have adopted eXtreme programming as their methodology, however implementing it as it was set out by Beck (2000) creates many challenges and it is unlikely to find organization, which follows these practices to the point.

<table>
<thead>
<tr>
<th>Lifecycle</th>
<th>Purpose</th>
<th>Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exploration</td>
<td>Enough well estimated story cards for first release, feasibility ensured</td>
<td>Prototypes, exploratory proof of technology programming, story card writing and estimating</td>
</tr>
<tr>
<td>Planning</td>
<td>Agree on date and stories for first release</td>
<td>Release planning game, story card writing and estimating</td>
</tr>
<tr>
<td>Iterations to first release</td>
<td>Implement a tested system ready for release</td>
<td>Testing and programming, iteration planning game, task writing and estimating</td>
</tr>
<tr>
<td>Productionizing</td>
<td>Operational Deployment</td>
<td>Documentation, training, marketing</td>
</tr>
<tr>
<td>Maintenance</td>
<td>Enhance, fix. Build major releases</td>
<td>May include these phases again for incremental releases</td>
</tr>
</tbody>
</table>

*Table 1: XP Life Cycle. (Adopted from: Larman (2003))*

Table above represents life cycle of XP methodology, it should be noted that the XP cycle of defining and building value happens very quickly unlike in waterfall methodology, which was discussed previously.

Scrum is another popular Agile methodology, as mentioned by Larman (2003) Scrum life cycle consist of four phases: planning, staging, developing and release. In the planning phase – the vision, expectations are being established as well as funding is being secured. Staging phase follows with identification of more requirements and their prioritization for first release. In the development phase a system ready for release is being implemented in a series of 30-day iterations. Release phase is operational when documentation and training for the product is being developed. Scrum is very precise on the length of iterations it is fixed to 30 days, thus providing a customer with the release every month.

Larman (2003, page 134) lists strengths of scrum methodology: “simple practices and management work products; individual and team problem solving and self management; evolutionary and incremental requirements and development, and adaptive behavior; customer participation and steering; focus; openness and visibility; easily combined with other methods; team communication, learning and value building; teambuilding via daily scrum”.

15
Schwaber (2004) discusses that the iteration in Scrum is known as Sprint, this sprint as it has been mentioned previously has a duration of 30 days. Every day starts with a Daily Scrum meeting where the team members report on the estimate of number of hours needed to finish the task. According to Subramaniam and Hunt (2006) a Sprint backlog contains tasks scheduled for current iteration, if at some point the estimate of hours to finish the tasks on the backlog will exceed the amount of hours left, these tasks will be moved to the next iteration.

As mentioned by Koch (2005, page 257) – “Scrum is defined not so much, by its process as by the practices that comprise it”. The practices are listed below:

- The Scrum Master – a new role which replaces traditional role of team leader or project manager. A scrum master would ensure that the scrum practices are being followed.
- Product Backlog – according to Schwaber (2004, 153) is “a prioritized list with project requirements with estimated times to turn them into completed product”.
- Scrum Teams – team which is committed to scrum goals and has full authority to do “whatever is necessary” to achieve the goal.
- Daily Scrum Meetings – as mentioned previously it is a short daily meeting used to synchronize the work and to report progress
- Sprint Planning Meeting – a meeting in which customers, users, management and the scrum team determine the goals for the next sprint
- Sprint – a 30 day iteration
- Sprint Review – a meeting at the end of the sprint where the product increment is being presented to the customers

DSDM or Dynamic Systems Development method as it was mentioned by Boehm and Turner (2003) is more a framework for developing software than a particular method. As it can be seen from Figure 5 below the DSDM life cycle consists of five phases: Feasibility, Business study, Functional model iteration, Design and build iteration and Implementation. Koch 2005 discusses that customer involvement in DSDM process is imperative due to DSDM’s strong focus on business purpose of the development; it is being an essential element for the acceptance of the deliverable. He also mentions that similar to other Agile methods DSDM uses iterations and
frequent releases. DSDM employs verify and validate approach to testing in each phase of the project, rather than at the end of the project as it is done in traditional methodologies.

Boehm and Turner (2003) state that the DSDM methodology feels a lot like traditional methodologies therefore it is easier to adopt for process-based organizations.

2.5 Agile Suitability Filters

To this point the literature review simply discussed different methodologies and basic principles behind them. There is plenty of literature on these topics; however there is barely any literature which would discuss how organization should choose the methodology to apply to its projects. Nowadays organizations have large project portfolios and it is unlikely that one methodology would suit every single project. Several authors have proposed Agile suitability filters, which help to determine whether Agile methodology is correct to apply to a project or not.

Chin (2004) mentioned that an organization can benefit from Agile methods when it operates in an environment of high uncertainty. He had also outlined two criterions for methodology selection.

1st Criterion

Chin (2004) discusses that there are different types of project environments exist within technology and science area. In an operational project environment the projects are being performed with a regular frequency. An example of such environment is service provisioning, projects in this case are very similar therefore organization knows what needs to be done and how will it be done. In such environment the level of uncertainty is low therefore traditional methodologies would suit best.

Another type of environment is technology development environment, for example technology platform development. In such environment there no much space for plan, which team could follow, therefore it would benefit most from Agile methods.
For the last type of project environment – the product/process development environment Chin (2004) suggests a mix of traditional and Agile approaches. There is high level of uncertainty present, however there is also an opportunity to apply traditional methods in order to get different teams work in cohesion with each other.

2nd Criterion

The second criterion outlined by Chin (2004) is looking at organizational stakeholders. The first type is a Single organization, an example of it is an R&D project undertaken within an organization where the customer is internal and there is no need to work with suppliers and other external parties, such project would benefit more from Agile methods due to its creative nature and greater freedom for Agile methodology.

Project can also span multiple, distinct organizations in these case according to Chin (2004) Agile methodologies are not applicable and the project would benefit greatly from traditional practices.

The third type of organizational stakeholder structure is the Single Company, Multiple Organizations; for example a project operating in a single company, but answering to the interests of the different departments. The introduction of Agile methods would depend on the motivation of the leaders of those departments. Such projects therefore could use any of the methodologies or a mix.

<table>
<thead>
<tr>
<th>Operational Projects</th>
<th>Multiple, External Stakeholders</th>
<th>Multiple, Internal Stakeholders</th>
<th>Single Organization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product/Process Development Projects</td>
<td>Classic/Agile</td>
<td>Classic/Agile</td>
<td>Agile</td>
</tr>
<tr>
<td>Technology/Platform Development Projects</td>
<td>Classic/Agile</td>
<td>Agile</td>
<td>Agile</td>
</tr>
</tbody>
</table>

Table 2: Agile applicability based on project type and organizational stakeholders. (Adopted from: Chin (2004))

Another way to access project suitability to Agile methods is by using Criticality and team Size factors scale introduced by Cockburn (2002). As it can be seen from the Figure 6 below, the X axis represent the number of people involved in the project it varies from 1-6 up to a 1000. The Y axis represents project criticality in terms of what defect may cause – would it be a mild discomfort or can it result in the loss of life. Different methodologies will span over different cells in the scale. Therefore the choice of methodology according to Cockburn (2002) depends on criticality and team size.
Boehm and Turner (2003) created a radar chart, which is used to assess project characteristics and its suitability for Agile methodologies. As it can be seen from Figure 7, the radar chart has 5 axes: Criticality, Personnel, Dynamism, Culture, and Size. It is difficult not to notice similarity with Cockburn (2002) in terms of Team Size and Criticality axis, however Boehm and Turner (2003) expand it by introducing such factors as Dynamism (amount of change requests), Culture (How ordered is organization? Can it adapt to Agile process?), Personnel (How is skilled is personnel in developing software). The idea of the chart is that the larger values on the axis are more suitable for traditional formal methodologies, while lower value, such as small team, high number of changes, low criticality is better accommodated by Agile methodologies.
Other tools which can be used in conjunction with each other is DSDM Project Suitability Filter (PSF) and Organization Suitability Filter (OSF) (Calgarypln.Org, Online). These tools take a form of questionnaire which determines whether the project is suitable for an Agile development approach and whether the whole organization is capable of adapting to it.

An OSF questionnaire is divided into 8 areas of evaluation: Users, User Management, Organization, Culture, IT Staff, IT Management, Management Organization and Techniques. The questions are asking to assign risk which threatening iterative development in one of the 8 areas listed above, later on all those risks are being quantified to prepare a spiders web graph, which would plot those risks, and provide a team with an overview of the risk map.

A Project Suitability filter consists of YES or NO question and has a purpose of providing the project team with an overview of the potential problem areas. Answering NO on any of the questions highlights a potential problem, when implementing Agile methodology.

The Slider tool was created by Mike Griffith and described at his blog Leadinganswers.Typepad.com, (Online), it is a simple tool which shows the applicability of Agile or traditional methods.

![Slider Tool](Adopted from: Leadinganswers.typepad.com (Online))

As it is possible to see from the Figure 8 factors like – project uncertainty, customer responsiveness, Innovative Cultures, Co-location possibility are best suited for Agile methodologies. While traditional methodologies would best suit project with high stability, low technical risk, inert to change and of low criticality.

### 2.6 Agile vs. Traditional Methodologies

As it has been mentioned before literature in its majority prescribes either one or another approach to software development, there are many proponents that Agile is the way forward; however they are also they are also being opposed by authors like Harrison (2003) and Rakitin (2001). It is difficult to say who is right or who is wrong as it was mentioned previously there is no Silver Bullet exist, which could serve as an answer for everything. One thing that is evident though is that there is a methodology bias in existence. This bias was described by Mike Griffith (Leadinganswers.Typepad.Com, 2008, (Online)) as a bias hammer which makes projects fit into different methods, even those that are not suitable for it. For example an Agile practitioner may be biased towards Agile methods and will try to use them with the project, which is best performed with traditional methods. This bias has been also mentioned by Jiang and Eberlein (2008), who argued for the convergence of two categories of methods in order to benefit from both. Some principles of Agile such as a dedicated customer available at any time to answer questions would also benefit waterfall methodology.
2.6 Research Gap

The review so far has described two categories of literature on software development methodologies; it has also described several approaches for determining appropriateness of Agile methodology to given project, however this field is under researched so the literature does not provide a clear indication on which projects are better suited for Agile.

The chapter has reviewed the literature on software project management, Agile and traditional methodologies. Existing filters for Agile methodologies have been identified and reviewed in details, as well as specific common methodologies. All this have led to the identification of the research gap in existing literature. Next stage in the research process is to identify research methodology and data collection tools.
Chapter 3: Methodology

After conducting a literature review which has involved extensive search on the databases, library and internet in general, the research gap has been identified. According to Dawson (2002) the next logical step in this sequence is to define the research methodology. It is essential in order to understand different issues, dilemmas and constraints within the research.

Following chapter will discuss underlying philosophy for the research, data collection tools used and validity and reliability issues. It will also present information regarding research participants and discuss ethical issues associated with this research.

The selection of research philosophy and approach was done using research process “onion model” described by Saunders et al. (2003).

3.1 Underlying philosophy

The research philosophy adopted for this study uses realist ontology and epistemology. The idea behind realism as described by Walliman (2001, page 154) is - “Science is an attempt to find out about a one real world, the truth about the world are true regardless of what people think, and there is a unique best description of any chosen aspect of the world”. At this point it is necessary to differentiate between two branches of realism: direct realism and critical realism. As identified by Saunders (2003) direct realism states that what you see and feel is how the world is, while critical realism argues that what we see and feel are sensations and they can be deceiving. While for direct realist the study at one level in organization is enough to get to some conclusion, critical realist would require multilevel study of organization before the conclusion could be
reached (implying that “our knowledge is result of social conditioning” (Saunders, 2003, page 105) ). For this thesis the direct realism perspective has been chosen. Software Project Management has emerged fairly recently, with the advances in computer technology. The failure rate for projects is high without regard to methodologies applied. Therefore it is essential to develop an understanding of what type of project would benefit the most from which methodology. Study, at the different levels and perspectives, is likely to deliver results, which would add to the confusion when facing a choice of methodology. It is project manager responsibility to decide what is best for the project; therefore the research should focus on his/her understanding of the issue.

3.2 Mixed Methodology
Bryman and Bell (2003) differentiate between qualitative and quantitative approaches to research. Qualitative research looks into attitudes, behaviors and experiences of the participants, trying to develop in depth understanding of the issue. Quantitative in its turn generates statistics through the use of questionnaires and structured interviews. Each of these methods has its advantages and disadvantages; therefore for this thesis a mixed methodology is selected. Selection of the methodology for the project involves many factors which need to be taken into account by project manager. Therefore it is necessary to study the process of thinking involved in methodology selection, thus it requires qualitative approach; however since some of this factors could be identified beforehand this thesis also uses quantitative approach to determine which generic factors need to be considered when selecting methodology.

In order to get the best of both approaches this thesis uses an idea of triangulation. The use of triangulation allows offsetting weaknesses of either approach (Bryman and Bell, 2003), Dawson (2002)). This thesis employs both standardized interviews and questionnaires, which allows increasing validity of the research

3.3 Inductive Approach
Realism research philosophy allows for both inductive and deductive approaches to the research. For this study an inductive approach has been chosen. Inductive approach uses observations in order to develop a conclusion about observed events, or even generalized conclusion about all of the events of the same type. (Jupp (2006). Thus the hypothesis follows only after conducting observations.

![Figure 10: Inductive reasoning (Adopted from Socialresearchmethods.net (Online))](image)
As it can be seen from the figure 10 – first specific observations take place, which are then followed by the pattern and regularity detection. This leads to the tentative hypothesis that can be explored and as a final stage a conclusion or general hypothesis can be developed. Compared to the deduction, induction allows for different solutions to emerge, while deductive process is focused on one solution.

3.4 Data collection through interviews

For the data collection as mentioned previously interviews and questionnaires have been used. The interviews in their majority have been conducted over the phone, with respondents identified through professional networks on ganttthead.com and linkedin.com. The reason for conducting interviews over the phone is the geographical location of the respondents and limitations of time allocated for the thesis research. According to Bryman and Bell telephone interviews have certain advantage that respondent is not being affected by personal characteristics of the interviewer. Also as mentioned by Cohen et al (2000) it allows respondents to do interview in more comfortable time for them, which is more difficult in case with face to face interviews. For this thesis interviewees were selected on the basis of:

- Significant experience in the management of software projects
- An experience in the use of Agile and traditional project methodologies

These constraints are necessary in order to get comparable information from the respondents. The interviews has been conducted with respondents who are directly involved in the process of managing software projects.

The table below contains some basic information about interviews and the interview type:

<table>
<thead>
<tr>
<th>Interviewee</th>
<th>Position</th>
<th>Country</th>
<th>Interview Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interviewee 1</td>
<td>Project Manager</td>
<td>Sweden</td>
<td>Face to face</td>
</tr>
<tr>
<td>Interviewee 2</td>
<td>Project Manager (Markus)</td>
<td>UK</td>
<td>Telephone</td>
</tr>
<tr>
<td>Interviewee 3</td>
<td>Senior Project Manager</td>
<td>US</td>
<td>Telephone</td>
</tr>
<tr>
<td>Interviewee 4</td>
<td>Development Project Manager</td>
<td>UK</td>
<td>Telephone</td>
</tr>
<tr>
<td>Interviewee 5</td>
<td>Project Manager and PM Consultant</td>
<td>Canada</td>
<td>Telephone</td>
</tr>
<tr>
<td>Interviewee 6</td>
<td>Project Manager</td>
<td>US</td>
<td>Telephone</td>
</tr>
<tr>
<td>Interviewee 7</td>
<td>Senior Project Manager</td>
<td>Sweden</td>
<td>Telephone</td>
</tr>
</tbody>
</table>

Table 3: Interview participants

For this thesis structured interviews were used. In a structured interview each respondent receives the same questions as every other. The questions are being read out in the same order to all respondents in order to achieve reliability. (Bryman and Bell, 2003)

In order to eliminate some of the sources of error, which structured interviews do suffer from, the respondents were given a set of questions prior to interview in order to familiarize themselves and to remove other sources of error described by Bryman and Bell (2003):
1. The way the questions is asked by interviewer
2. Misunderstanding on the part of the interviewee
3. Memory problems on the part of interviewee

In order to address principle source of error, which is - the way the information is recorder by the interviewer, the respondents prior to interview were asked for permission to record interviews into audio files. This has also facilitated process of interview, since the respondents could answer in their own preferable pace and the interview was conducted in a more efficient manner, without disturbances needed to record information. Using mp3 recording has helped in the analysis of the interviews, since it allowed for detailed transcripts to be developed.

As it was mentioned previously the respondents have been identified through gantthead and linkedin professional networks. They have been contacted through the internal messaging system with the proposal for the interview; with the following communication a specific time slot has been arranged for each respondent. After first interviews it was estimated that average telephone interview would last approximately 15-20 minutes, this time was then communicated to other respondents in order to help them to plan their day more effectively.

The interview consisted of ten questions, which were aiming to find out interviewee background with software project methodologies, preferences for particular methodology, application of methodologies and their performance.

3.5 Data Collection through questionnaires
The questionnaires used for this thesis have been proposed by one of the respondents for the telephone interview. Use of questionnaires aims to strengthen some of the points, which were mentioned by respondents during interviews as well as test an assumption that a choice of methodology is heavily influenced by project environment.

Gray (2004) identifies several reasons for the use of questionnaires:

- Questionnaires can be conducted quick and cost less.
- It is possible to collect data from many more people than in case with interviews
- Questions can be answered at the most comfortable time for the respondent and at the most convenient pace
- Analysis of the closed questions is simple and takes less time
- Questionnaires are not influenced by personal qualities of the researcher

However questionnaires can suffer from low response rate Respondents may consider it being too long; therefore the amount of questions is very restricted. Unlike interview in using questionnaire researcher cannot explain questions in greater detail or detect confusion from the respondent’s side, therefore the answers provided could be inaccurate. Another disadvantage is that it is impossible to know who exactly have answered questionnaire, since the person to whom it was sent initially could have delegated a task of answering it to somebody else. Also unlike
interview it is impossible to collect additional data. Some data could be missing, due to the respondent’s decision to skip “boring or hard” questions. (Gray, 2004); (Bryman and Bell, 2003)

The questionnaire was developed with suggestion of Richard Grove a senior project manager at Webcor Builders (USA). The idea behind the questionnaire is that choice of methodology is influenced by different factors which affect the project during its different phases.

To assure the validity of the questionnaires the questions were built in a way which cover the research issues both content and detail wise. The questionnaires were tested with two potential respondents who have suggested additional factors and commented on the presentation style and potential respondents. Additional factors included: resource/programmer count, in-house/outsourced development, billing rates and fixed rates. An attempt was made to organize the questions in a clear structure by arranging factors for grading into project phases where they could be encountered. These are characteristics which may compromise the validity of the questionnaires according to Grey (2004).

The questionnaire used for this thesis has been uploaded online on surveyconsole.com; this has simplified access to questionnaire for the respondents. An online questionnaire proves much easier to distribute and is much faster to fill in. Personal invitations were sent to selected members of gantthead.com, linkedin.com (who has listed Agile methodologies as their interests and primary work domain in their public profiles) and companies who employ Agile methods for their projects. Several invitations were also posted on the specialized discussion boards in linkedin.com, Agilerussia.ru.

Respondents were asked to rate Agile and traditional methodologies by answering the question – “How well does methodology X handle _____?”

Different answer options reflect issues faced by project managers during different phases of project life cycle. The life cycle used in this case has been divided into four phases: requirements, design, development and deployment. Each issue provided for grading had it polar opposite i.e. poor requirements, well defined requirements. The questionnaire employed closed type questions where respondents were asked to rate items on the scale of “bad, poor, well, and excellent”. The reason for using this four categories and exclusion of the “OK” option is that respondents are forced into taking a position on the issue, without being able to use ambiguous OK option.

The disadvantages of using closed questions in the questionnaire are: loss of spontaneous response and bias in answer categories. (Gray, 2004)

The questionnaire consisted of three parts: Agile methodologies, Traditional methodology and RUP methodologies (which are not discussed in this thesis since barely any respondents have answered that part. Table 4 demonstrates geographical information about respondents.
Table 4: Questionnaire respondent matrix

For the table above plus sign represents respondent answering questions in the relevant section of the questionnaire, while minus represents no answer received.

As it is possible to see from the table above the total number of respondents to the questionnaire is 23 from whom 21 had answered questions on Agile methodologies and 20 had answered questions on traditional methodologies. This sample size does not provide for generalizable analysis results, which would require at least 30 responses (Saunders et al, 2003); however it is possible to make some conclusions which are valid for this sample.
3.6 Validity and Reliability
According to Marczyk et al (2005) validity is concerned whether the measurement approach measures what is supposed to measure while reliability refers to the consistency of the measure.

To insure the validity of the structured interview it was attempted to assure that the questions focus would be directly related to the research objective. Also to strengthen the validity the research followed pattern discussed Arksey and Knight (1999):

- The interview questions were tested beforehand with Ralf Müller the supervisor for this thesis following the advice of Gillham (2000) who mentions that it is better to test the interview questions with people outside the targeted group. The clarity of the questions was tested prior to the interviewing main respondents, as well as feedback received on possible improvements such as Questions 6 and 10 in the Appendix 2.
- The interview technique was focused on building rapport and trust, giving the interviewers the opportunity to express themselves freely
- Sufficient time was given to each subject in order to explore topic in depth.

As mentioned by Grey (2004) interviews can be used when the available sample is of a small size, hence the external validity may be restricted. However to make the finding more plausible even though still not generalisable, questionnaires were used.

As many studies has shown, to achieve a higher validity and reliability of the interview one of the ways is to try to avoid the bias coming from the characteristics of the interviewer or the respondents. For the purpose of this thesis highly structured interview was used, which is one of the ways for controlling reliability as it is argued by Silverman (1993) and open-ended questions which allows respondents to bring their own views and enables for unpredicted but important issues to rise. Also the fact that the interviews were conducted over the phone strengthen their reliability as discussed by Miller and Cannell (1997) as the interviewee might disclose more information without being intimidated by the researcher’s presence.

3.7 Ethical issues
In terms of ethics this thesis adheres to the four rules identified by Diener and Crandal (1978):

- There is no harm to the participants
- There is no lack of informed consent
- There is no invasion of privacy
- There is no deception involved

For this thesis all interview participants were assured that their names and any information, which could identify their company, would not be published. All the data is presented in the thesis is aggregated from different participants, therefore it is impossible to pinpoint particular answer. In questionnaire the data such as name, position and email address has been collected,
however every respondent has been informed that this fields are optional and there is no need to fill them in. The reason for collecting data such as position and name is to eliminate some part of bias, which could arise if the questionnaire is completed not by an intended person. Similar to interviews all data is provided in aggregated form, which does not allow individual responses to be recognized.

All respondents have received a letter, which outlined the aims of the research, and some basic information about it. Furthermore the first page, which questionnaire link has been directing participants had contained same information. The research results have been promised to be communicated to the respondents, who chose to communicate their email address.

3.8 Analysis and presentation of results

Next chapter is organized based on the findings from questionnaires and interviews which were discussed in the previous section. The interviews which were conducted with industry professionals required a throughout analysis which would highlight pattern in the respondent answers. Patterns identified and their discussions form the first part of this chapter. It is unclear from the interview results on which basis software development methodology is selected for a particular project. The identified patterns represent broad categories into which software projects fall. These patterns represent Project characteristics in which Agile is not appropriate. Patterns such as “Agile is best used with unclear requirements are excluded from this discussion”. Also there have been patterns identified in interviewee views on Agile methodologies, these are important in order to create an understanding on how Agile methodologies are perceived by respondents.

Second part of the next chapter is dedicated to the questionnaire analysis and discussion of findings. The main idea of the questionnaire as it was mentioned previously is to investigate how methodology families perform against different factors faced in project lifecycle. One of the patterns identified through the interviews is that choice of project methodology depends on the environment. Therefore the questionnaire serves as an extension to the interview, trying to develop understanding of the issues faced by project manager when deciding on methodology for the project.

3.9 Research process

The research has been conducted for the duration of one month starting with the first interview on the 18th November 2008 and finishing with the closure of the online questionnaire on 20th of December 2008.

The analysis of the interviews was done by identification of the patterns in the answers of the respondents. These patterns are reported in the chapter 4 of this thesis. Obvious patterns such as: Agile are suitable for projects with unclear requirements have been left out, since there is enormous amount of publications, which mention it, and since it is one of the Agile principles. Figure 11 represents the graph which summarizes demographic data of interview participants.
Figure 11: Demographic data for interview participants

Analysis of the questionnaires was done by calculating means and standard deviations for each of the questions and groups of questions. The most significant results were then commented upon in Chapter 4 and compared with findings from interviews and literature review. The demographic spread of the respondents is reported in the figures 12 and 13.

Figure 12: Agile methodologies demographic data

Figure 13: Traditional methodologies demographic data
The chapter so far has discussed underlying philosophy for the research, data collection tools and issues associated with validity, reliability and research ethics. Next chapter will present and discuss results obtained through the research.
Chapter 4: Findings and discussion

Following chapter will present findings and discussion for the results of interviews and questionnaires.

4.1 Interview findings

This section starts with general statements about Agile methodologies, which were repeated by all the respondents repeatedly. The following statements were found to be true for all the respondents.

- Agile is not a silver bullet
- Each methodology got its advantages and disadvantages
- Methodologies can be integrated together in order to get the best of both worlds

The above statements go against the view which a lot of literature on Agile is trying to build - that Agile is the only way forward. From the interviews it became clear that it is not. Respondents have provided examples of the projects which were failures using Agile even though projects has possessed all the characteristics usually described as being suitable for Agile.

Further on this section continue with identifying several categories of project where Agile doesn’t work well. These categories do not exclude Agile methods, but do work better with traditional methodologies.

- Databases
- Embedded Software development
- Computationally complex projects
- Incident response teams

Other factors which were mentioned by respondents are: the need for the assessment of the client and client’s organization, and application maintenance projects.

At this point it is necessary to point out that all respondents have also mentioned a number of cases when Agile and traditional methodologies should be used, which are in line with literature, which was discussed in Chapter 2 of this thesis. These cases are presented below

- Agile is best used, when the specific requirements are vague, ambitious or unknown (i.e. projects where rapid prototyping is required to deliver to a conceptual expectation)
- Traditional methodologies are best used in large scale complex projects (i.e. military projects)
• Agile also benefits the projects, which require the delivery to semi-immediate timelines (i.e. where products and associated system support are required to meet with an immediate competitive disadvantage that can arise through a competitor's actions)

The number of factors project manager need to take into account when choosing methodology is enormous. Throughout interviews it was identified that these factors fall into four categories:

• Project team
• Customer
• Project Management constraints (i.e. Agile is good to develop working product for the release date, however it may not have all the functionality)
• Technical project characteristics

For the questionnaire these factors were arranged into the respective project phases where they are encountered by project managers during the course of the project.

4.1.1 There is no such thing as a silver bullet
The idea of silver bullet was mentioned previously in the Chapter 2, the interview respondents were asked what they think on this issue. All of the respondents have answered negatively to this question. According to one of the respondents that is a big problem, because people are looking to have one answer, but there is none. There is no such methodology which would suit every single project in the best way possible. Software technology is changing too fast, as well as its application methods, while it is true that you can achieve excellent results with Agile now, it may not necessary be true in a future.

“You can fail as quickly with Agile as quickly as with any other methodology” (Interviewee 5)

Agile is not a panacea for software project management, it is just another approach to software development which works great in some circumstances and fails in other. Project which has well defined requirements upfront, with a very little risk, and little complexity would do much better with traditional methodologies, thank with Agile.

“When the things are really easy you do not need a complex method” (Interviewee 3)

Every project can be performed with any methodology but it would not necessary be the best fit for it. Even large scale projects, which are traditionally attributed to be the best fit for traditional methodologies with extensive upfront planning, can be divided into small projects and be performed in Agile way. It may be as effective or as catastrophic as traditional approach.

“A lot of this is based on the client, to say that there is one way of solving every project is like saying there are no human factors in the equation.” (Interviewee 4)
4.1.2 Methodology advantages and disadvantages

Each project management methodology has its advantages and disadvantages, it is essential to understand them in order to be able to choose between them.

Agile methodologies ensure that quality is being built into the product straight from the beginning, such practices as code refactoring optimize existing code and ensure full functionality of the end product. Agile also able to deliver working product to the market much faster than waterfall approach, although it does not guarantee all the functionality time to market in many cases results in higher or lower profits. Agile can deliver higher business value for the client, since all the requirement changes are implemented throughout the project, which results in the client getting the product required. Requirements in Agile are handled in business priority order. It should be noted here that Agile delivers higher value for money, but does not save money. For instance releasing a product which does not meet customer expectations would mean less value created in terms of future sales etc… Lostechies.Com (Online) discusses hypothetical situations of traditional and Agile development, which result in a conclusion that “Agile provides tremendous value, but not necessary cost savings”. Agile development tends to work out more expensive than development do with traditional methodologies. The project can start even with very poor requirements when using Agile methods. Most difficult and risky parts of the project are tackled first thus ensuring earlier success.

Many organizations face difficulties when implementing Agile, since it requires a change of culture and some practices of the Agile methodologies can be considered by some too hard to adopt i.e. pair programming delivers higher quality but requires very close communication and relationship between programmers.

*It is easy to miss apply and abuse Agile methods since there is no guidance there so we give more freedom to people.* (Interviewee 5)

Another disadvantage of Agile methodologies mentioned by respondents was its lack of documentation, even though in Agile it is usually about things going forward, but it is also necessary to be able to look back at what has been done. The fact that there are very little requirements available at the beginning of the project makes it harder to negotiate fixed price contracts.

An obvious disadvantage of traditional methodologies is that its change management process is focused on preventing change rather than welcoming it. A change in requirements may result in significant cost overruns and delays.

4.1.3 Integrated methodology

All of the respondents have mentioned that Agile and traditional approaches are not mutually exclusive and can complement each other. These hybrid approaches take the best of both world and combine it for the benefit of the project.
Although the overall program or project may require a traditional model, in delivering the small components, Agile methods may be the best way of ensuring that the critical path is maintained. (Interviewee 2)

Traditional methods provide effective tools of monitoring and control in terms of resources and finances. Best practices such as pair programming can be utilized during traditional life cycle as well. It is essential to select the best for the project. As it was mentioned by one of the respondents: “I attempt to make things together so they would fit to each project so it is mostly scrum, maybe a little of traditional waterfall, a bit of lean or a little bit of whatever… It is all built around the project.”

4.1.4 Databases
Formalized databases and database systems which are used in virtually any business focused application are found to be anti-Agile. The reason for it is that databases in organizations are coupled with many other databases and applications meaning that changing a name of column would affect all the applications which are supported by this database. At any time there may be different teams working on different modules or applications, making database changes increases chances of having missed values and not working software. Although there have been some solutions suggested, it is recommended to approach such projects using traditional waterfall-like methodologies.

4.1.5 Embedded Software development
Embedded software is software which is present permanently in the device it controls; it is being built in into the devices it controls. It has a dedicated to the task function, which cannot be changed, for example a computer control chip in a washing machine. This software does not have much of user interface to be presented to the client; therefore it is difficult for client to understand what was actually done. For instance in the development of low level drivers, there are no visual means of demonstrating progress except running the device, but that will not reflect actual progress. As it was mentioned in interviews in traditional methodologies it is easier to demonstrate progress. According to one of the respondents – “It does not mean that you cannot use Agile approach but you got to be more creative in doing that.” (Interviewee 5)

4.1.6 Computationally complex projects
Computationally complex projects have been mentioned as requiring traditional methods as well. They are less adapted to frequent changes, prototypes and require a lot of planning. Agile development can also be used in this case, but it would create way too many challenges for the project team.

4.1.7 Incident response teams
Interviewees also mentioned that Agile approach does not seem to work well with incident response teams. For instance – “if it is an application support incident response team working using Agile methods and a high priority requirement comes in on day three it would throw things out of water” . (Interviewee 4)
4.1.8 Client assessment
All of the respondents mentioned the need to assess client in order to understand whether Agile would work. Agile requires very close involvement of the client or client representative, as it was mentioned previously it is essential for client to be responsive and to be comfortable with Agile methods.

4.1.9 Application maintenance projects
The respondents have mentioned that one of the project types which benefit significantly from using Agile is ongoing maintenance and management of existing products; the reason for it that there is no date insight for such projects. Following Agile principles project team is able to release patches and add-ons for applications on the regular basis.

The interview results did allow making several conclusions regarding the projects to which Agile methodologies and traditional methodologies are more applicable. The next part of the analysis looks at the factors which lead to the selection of methodology for a project.

4.2 Questionnaire Analysis
The following analysis will look at the four project life cycles phases one by one and examine some of the factors, which have been mentioned by interviewees as the factors, which need to be considered when selecting project methodology for a particular project. This questionnaire has been developed with the help of two Senior Project Managers with significant experience in software project management, traditional and Agile methodologies.

As it has been mentioned previously in the methodology section the questionnaire received 21 responses for Agile section and 20 responses for section on traditional methodologies, such sample size is too small in order to make conclusions which could be generalized, however for the purposes of this thesis we would like to argue that any conclusions or findings are relevant for the sample from which the data has been obtained.

For the analysis purposes the scale values (Bad, Poor, Well, Excellent) have been assigned numerical values where: Bad-1, Poor-2, Well-3, and Excellent-4.
As it is possible to see from the figure above – mean values above 2.5 fall into positive half of the scale, while values below 2.5 fall into negative half of the scale. However a mean of 2.51 is not a clear indicator for any conclusion, therefore for the purposes of this thesis the assumption for a condition is made that all the values from 1 until 2.25 represent negative values and all the values from 2.75 till 4 represent positive values. The values which fall into interval between 2.25 and 2.75 are considered to be uncertain.

**4.2.1 Requirements**

Every project starts with requirement definition. In software projects requirements are the basis for future design, coding and testing. (Hamilton, 1999) It is an essential part of any project and its success is heavily dependent on this stage. The factors discussed for these phase aimed to find out how do methodologies handle such factors as requirements, project manager business domain expertise, complexity etc…

Table 5 summarizes questions and average scores for the requirements part of questionnaire.
Overall judging by the average score Agile outperforms traditional methodologies during the requirements phase. During interview all interviewees have mentioned that Agile methodologies are best for the project when the requirements are unclear and that it is essential for the customer to work very closely with the project team. These views are also supported in literature review in Chapter 2 by Chin (2004), Boehm (2002) and confirmed by questionnaire results for the given sample. As it is possible to see traditional methodologies have received very low score in the relevant question, while Agile methodologies have received excellent score, thus providing support to the statement that “Agile methodologies are preferable on the projects where requirements are unclear”.

![Figure 15: Requirements](image-url)
Another factor which was mentioned by interviewees is that projects with high level of complexity benefit significantly from the use of Agile methodologies. As it can be seen from the figure below – Agile methodologies have received a high score on both projects with low and high complexity; however traditional methodologies have received a score of 2.5 which falls into undetermined are, but comparison of mean between both traditional and Agile methodologies allows making a conclusion for the given sample that Agile methodologies are superior to traditional methodologies when employed on the projects with high complexity.

![Figure 16: Problem Complexity](image)

According to the conclusion made in the previous two paragraphs, Agile methodologies allow achieving good results on the projects where requirements are unclear and level of complexity is high. As discussed in literature review with reference to McConnell (1996) and Koch (2005) such projects by their definition carry high level of risk attached to them, from the table above it is possible to see that Agile methodologies received significantly greater average score than traditional methodologies on the high risk of failure category.

![Figure 17: Risk of failure](image)

For this phase and for given sample it is possible to conclude that Agile methodologies are best used when the requirements are unclear, level of complexity is high and there is a high risk of failure.

Also another factor which Agile methodologies handle better than traditional ones is a high risk tolerance of the project champion. This factor has not been confirmed through interviews or
literature, however using the previous paragraph if Agile projects are high risk than it is essential for project champion to have high risk tolerance.

4.2.2 Design
Design stage is crucial stage in software development; the reason for it is that at this stage outputs from the previous requirements stage which define what software should do, are being translated into how it should do it. (Hamilton, 1999) In simple words during this stage a plan for the solution is being developed. There are many issues to take into account at this stage such as: software compatibility, robustness (quality), usability etc… The success of the project is highly dependent on this stage.

Bellow is the table which summarizes questions, averages and number of respondents for this phase.

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Scale</th>
<th>Agile Mean</th>
<th>SD δ</th>
<th>Agile N (Number of responses)</th>
<th>Traditional Mean</th>
<th>SD δ</th>
<th>Traditioinal N (Number of respondents)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quality required</td>
<td>Robust</td>
<td>3.14</td>
<td>0.79</td>
<td>21</td>
<td>2.80</td>
<td>0.77</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Adequate</td>
<td>3.38</td>
<td>0.59</td>
<td>21</td>
<td>2.85</td>
<td>0.59</td>
<td>20</td>
</tr>
<tr>
<td>End-user participation in design</td>
<td>High</td>
<td>3.86</td>
<td>0.36</td>
<td>21</td>
<td>2.75</td>
<td>0.79</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Low</td>
<td>2.71</td>
<td>1.01</td>
<td>21</td>
<td>2.25</td>
<td>0.91</td>
<td>20</td>
</tr>
<tr>
<td>Application architect experience</td>
<td>Broad</td>
<td>3.19</td>
<td>0.60</td>
<td>21</td>
<td>2.90</td>
<td>0.72</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Narrow</td>
<td>2.62</td>
<td>0.86</td>
<td>21</td>
<td>2.15</td>
<td>0.75</td>
<td>20</td>
</tr>
<tr>
<td>The programmers’ skill set</td>
<td>Broad</td>
<td>3.52</td>
<td>0.60</td>
<td>21</td>
<td>2.95</td>
<td>0.76</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Narrow</td>
<td>2.52</td>
<td>0.93</td>
<td>21</td>
<td>2.35</td>
<td>0.75</td>
<td>20</td>
</tr>
<tr>
<td>Platform vendor involvement required</td>
<td>High</td>
<td>2.86</td>
<td>0.65</td>
<td>21</td>
<td>2.80</td>
<td>0.83</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Low</td>
<td>2.57</td>
<td>0.87</td>
<td>21</td>
<td>2.40</td>
<td>0.88</td>
<td>20</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>3.04</td>
<td>0.85</td>
<td></td>
<td>2.62</td>
<td>0.81</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Design Phase

Agile methodologies have received better total score for the design attributes of the project, than traditional methodologies.

During interviewees several respondents have mentioned that Agile methodologies are better in terms of delivering high quality product. As it can be seen from the Table 6 Agile methodologies has scored significantly better in terms of delivering quality required. This result is also supported by Koch (2004) whose statement on the Agile quality is quoted in the Chapter 2 of this thesis.
From the Table 6 it is possible to see Agile benefits significantly from end-user participation in design, thus proving previously mentioned statement about Agile methodologies working in close contact with end-users all the way throughout the project. So far it has been confirmed for a given sample in both requirements and design stages.

For the given sample Agile methodologies also benefit significantly more from broader skill sets in comparison with traditional methodologies. This is applicable for both application architect and for programmers; one of the interviews has mentioned that Agile methodologies work well when team members can assume different roles.

For the design group of questions it is possible to conclude that for the given sample Agile methodologies are able to deliver higher quality product and benefit significantly from team members with broad range of skills and close collaboration with the customer.
4.2.3 Development
During the development phase detailed designs are completed and code is being written. This phase includes software testing, which is important in order to deliver to the customer not faulty software.

The table below shows the score received by Agile and traditional methodologies in the development section of the questionnaire.

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Scale</th>
<th>Agile Mean</th>
<th>SD</th>
<th>Agile N (Number of responses)</th>
<th>Traditional Mean</th>
<th>SD</th>
<th>Traditional N (Number of respondents)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Testing policies</td>
<td>Rigorous</td>
<td>3.38</td>
<td>0.80</td>
<td>21</td>
<td>3.00</td>
<td>0.86</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Adequate</td>
<td>2.76</td>
<td>0.94</td>
<td>21</td>
<td>2.20</td>
<td>0.83</td>
<td>20</td>
</tr>
<tr>
<td>Commitment to testing by programmers</td>
<td>High</td>
<td>3.62</td>
<td>0.67</td>
<td>21</td>
<td>2.80</td>
<td>0.95</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Low</td>
<td>2.24</td>
<td>1.04</td>
<td>21</td>
<td>2.30</td>
<td>1.03</td>
<td>20</td>
</tr>
<tr>
<td>Commitment to testing by end users</td>
<td>High</td>
<td>3.57</td>
<td>0.68</td>
<td>21</td>
<td>2.80</td>
<td>0.89</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Low</td>
<td>2.10</td>
<td>1.00</td>
<td>21</td>
<td>2.20</td>
<td>0.95</td>
<td>20</td>
</tr>
<tr>
<td>Programmer familiarity with delivery method</td>
<td>High</td>
<td>3.48</td>
<td>0.51</td>
<td>21</td>
<td>3.05</td>
<td>0.76</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Low</td>
<td>2.57</td>
<td>0.87</td>
<td>21</td>
<td>1.95</td>
<td>0.76</td>
<td>20</td>
</tr>
<tr>
<td>Programmer experience with chosen development platform</td>
<td>High</td>
<td>3.29</td>
<td>0.56</td>
<td>21</td>
<td>2.80</td>
<td>0.70</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Low</td>
<td>2.67</td>
<td>0.86</td>
<td>21</td>
<td>2.05</td>
<td>0.89</td>
<td>20</td>
</tr>
<tr>
<td>Resource/programmer count</td>
<td>High</td>
<td>2.81</td>
<td>0.87</td>
<td>21</td>
<td>3.00</td>
<td>0.79</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Low</td>
<td>3.38</td>
<td>0.74</td>
<td>21</td>
<td>2.20</td>
<td>0.95</td>
<td>20</td>
</tr>
<tr>
<td>Billing Rates</td>
<td>High</td>
<td>2.95</td>
<td>0.80</td>
<td>21</td>
<td>3.00</td>
<td>0.79</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Low</td>
<td>3.05</td>
<td>0.74</td>
<td>21</td>
<td>2.70</td>
<td>0.80</td>
<td>20</td>
</tr>
<tr>
<td>Fixed Rates</td>
<td>High</td>
<td>2.81</td>
<td>0.75</td>
<td>21</td>
<td>2.90</td>
<td>0.72</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Low</td>
<td>2.76</td>
<td>0.89</td>
<td>21</td>
<td>2.55</td>
<td>0.76</td>
<td>20</td>
</tr>
<tr>
<td>Development</td>
<td>In-house</td>
<td>3.62</td>
<td>0.50</td>
<td>21</td>
<td>2.70</td>
<td>0.86</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Subcontract</td>
<td>2.81</td>
<td>0.75</td>
<td>21</td>
<td>2.80</td>
<td>0.83</td>
<td>20</td>
</tr>
<tr>
<td>Development platform flexibility</td>
<td>High</td>
<td>3.29</td>
<td>0.72</td>
<td>21</td>
<td>2.60</td>
<td>0.75</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Low</td>
<td>2.76</td>
<td>0.77</td>
<td>21</td>
<td>2.65</td>
<td>0.81</td>
<td>20</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>3.00</td>
<td>0.88</td>
<td>21</td>
<td>2.61</td>
<td>0.89</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 7: Development Phase

Agile methodologies have received better overall score of 3.00 for the factors associated with development phase in comparison with traditional methodologies which received the score of 2.61.

Testing policies define what approach should be used when deciding on system tests. As it is possible to see from the table above Agile methodologies have received better scores than traditional methodologies for handling testing policies. In the literature review it was discussed
that Agile methodologies such as DSDM adopt verify and validate approach throughout the whole project lifecycle, thus being very focused on testing.

High programmer experience with chosen development platform has been found to be handled by Agile methodologies better than by traditional ones. Also low level of experience has received better grade for Agile methodologies than for traditional ones. This is finding is applicable only for the given sample and is not confirmed by interviews or through the literature research.

Both Agile and traditional methodologies have received high average score for high commitment to testing by end users and programmers, while receiving low scores for the low level of commitment. Comparison of scores for Agile 3.62 and 3.57 versus traditional 2.80 for both shows that Agile handles high level of commitment much better than traditional methodologies. Therefore it is possible to conclude that for the given sample Agile methodologies handle high commitment to testing by end-users and programmers better than traditional methodologies.

![Figure 19: Commitment to testing](image)

Several respondents for the interviews have mentioned that Agile works better for small teams rather than for large ones. These statements are confirmed for the given sample by the questionnaire. As it can be seen from the table Agile methodologies have received an average mark of 3.38 versus 2.20 for traditional methodologies. Therefore it is possible to conclude for the given sample that Agile methodologies are best to be utilized by small teams.

Agile methodologies have received much higher score for the in-house development rather than subcontracted. This result allows making a conclusion for the given sample that Agile methodologies perform better when development is done in-house.

Another factor standing out in the questionnaire findings is ability of Agile methodologies to handle development platform flexibility. While both Agile and traditional methodologies received similar grade for handling low development platform flexibility, Agile received very high scores on its ability to handle high platform flexibility. While traditional methodologies
may not benefit significantly from high flexibility Agile projects with changing requirements benefit from higher flexibility.

4.2.4 Deployment
The table below shows the scores receive by Agile and traditional methodologies for deployment phase.

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Scale</th>
<th>Agile Mean</th>
<th>SD</th>
<th>Agile N (Number of responses)</th>
<th>Traditional Mean</th>
<th>SD</th>
<th>Traditional N (Number of respondents)</th>
</tr>
</thead>
<tbody>
<tr>
<td>End users’ ability to adapt to change</td>
<td>High</td>
<td>3.71</td>
<td>0.46</td>
<td>21</td>
<td>2.35</td>
<td>0.88</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Low</td>
<td>2.95</td>
<td>0.97</td>
<td>21</td>
<td>2.70</td>
<td>1.08</td>
<td>20</td>
</tr>
<tr>
<td>Budget elasticity</td>
<td>High</td>
<td>3.43</td>
<td>0.68</td>
<td>21</td>
<td>2.55</td>
<td>0.76</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Low</td>
<td>2.90</td>
<td>0.94</td>
<td>21</td>
<td>2.65</td>
<td>0.67</td>
<td>20</td>
</tr>
<tr>
<td>Project timeline</td>
<td>Fixed</td>
<td>3.29</td>
<td>0.64</td>
<td>21</td>
<td>2.50</td>
<td>0.83</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Free</td>
<td>3.38</td>
<td>0.86</td>
<td>21</td>
<td>2.70</td>
<td>0.80</td>
<td>20</td>
</tr>
<tr>
<td>Project manager workload</td>
<td>Heavy</td>
<td>2.71</td>
<td>0.56</td>
<td>21</td>
<td>2.50</td>
<td>0.83</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Light</td>
<td>3.29</td>
<td>0.56</td>
<td>21</td>
<td>2.85</td>
<td>0.75</td>
<td>20</td>
</tr>
<tr>
<td>Project progress reporting</td>
<td>Detailed</td>
<td>2.62</td>
<td>0.86</td>
<td>21</td>
<td>2.90</td>
<td>0.85</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Summary</td>
<td>3.24</td>
<td>0.54</td>
<td>21</td>
<td>2.95</td>
<td>0.76</td>
<td>20</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>3.15</td>
<td>0.79</td>
<td>21</td>
<td>2.67</td>
<td>0.83</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 8: Deployment Phase

Agile has received higher grades than traditional methodologies for the factors associated with deployment phase.

As it is possible to see from the Table 8 Agile methodologies has received much higher grades for handling fixed and free project timelines. During the interviews several respondents have mentioned that they believe that Agile methodologies a better than traditional methodologies in terms of meeting quality and time targets, although they haven’t been sure about the budget. From this data it is possible to draw a conclusion that for the given sample Agile methodologies do perform better than traditional methodologies in terms of handling project timeline.

Another two factors which are evident from the questionnaire, but could not be confirmed through interviews or through literature research are: End users ability to adapt to change and budget elasticity. Both high end users ability to adapt to change and high budget elasticity has been rated as being handled by Agile methodologies significantly better than by traditional ones. It should be noted though that these results are applicable only for the given sample and cannot be generalized over the whole population.

Overall results for methodologies in each section of the questionnaire are presented below.
Agile has received better marks comparing to traditional methodologies in all sections of the questionnaire. The factors which had the most significant differences in score between Agile and traditional methodologies have been discussed in details in this part of the thesis.

Next chapter will summarize the findings and discuss theoretical and managerial implications of the research as well as its strengths and weaknesses.
Chapter 5: Conclusions

This study aims for the identification of factors that help to decide for Agile or traditional project management methodologies in different types of projects. For that 7 interviews were conducted as well as a global web-based questionnaire with 20 responses. Analysis was done using pattern matching and comparison techniques.

5.1 Research Question
The research question stated at the beginning of this thesis is presented below:

For which projects it is appropriate to use Agile methodologies?

The answer for this question was obtained through the use of interviews and online questionnaire. The following are the characteristics of the project which would benefit significantly if done with Agile methodologies:

1) Poorly defined requirements
2) High level of complexity
3) Risk of failure is high
4) High development platform flexibility
5) High level of customer participation throughout all project phases
6) High level of quality required
7) Project team members have broad range of skills
8) Rigorous testing policies
9) Low resource/programmer count
10) Development is done in house
11) Client is comfortable with Agile methods

Also several other factors were identified solely through questionnaire:

1) High risk tolerance of the project champion
2) High programmer experience with chosen development platform
3) High end-user ability to adopt to change
4) High budget elasticity

In comparison with Agile it was found that projects with high resource count and sub-contract development benefit more from traditional methodologies.

In order to answer the research question it is also necessary to turn it around:

For which projects it is NOT appropriate to use Agile methodologies?

The results obtained for this question are listed below.
1) Databases
2) Embedded Software development
3) Computationally complex projects
4) Incident response teams

It is definitely possible to say when answering research question, that Agile methodologies are not applicable to every kind of a problem. So far this section has discussed characteristics of the project which would benefit from Agile methodologies as well as some specific project types, which are not appropriate for Agile methodologies.

5.2 Theoretical Implications
Existing theory does not cover in details which projects are appropriate for Agile methodologies and which project are better to be done with traditional methodologies. This research aimed at finding out when does Agile methodologies work best and when they do not work, hence it is better to do project with traditional methodologies. From the above it is possible to see that the following groups of attributes are influential for the selection Agile versus traditional methods: Requirements, project risk, team and development process characteristics. These groups include different factors within them which need to be accessed prior to selection of methodology.

An important theoretical implication of this thesis is an improved understanding that searching for the project types which would fit different methodologies would not deliver substantial results. During selection of methodology for the project, it is project characteristics which need to be assessed.

This research has also generated some results that haven’t been mentioned previously in the project management literature, although not generalisable, these results provide an interesting insight into the considerations of project manager when choosing project methodology. These factors are: risk tolerance of the project champion, programmer experience with chosen development platform, end-user ability to adopt to change and budget elasticity.

5.3 Managerial Implications
Main implication of this thesis is that it provides a foundation for choosing methodology for the software projects. As it has been mentioned previously a choice of methodology may make or break the project. Project characteristics identified for the Agile project are not complete and there are many more which need to be taken into account. However if project does not have many of those characteristics it may be better to do it using traditional methodologies. Agile methods are quite complicated in comparison with traditional ones, thus for some projects they may be too excessive.
When choosing a methodology for a project practitioners need to be aware of the following factors:

- Are requirement clear, can they change?
- Does client fully understand and comfortable with the method the project would employ?
- Is the project comfortable with the methodology and what is his/her risk attitude?
- How will development process take place? Is it done in-house, outsourced or distributed development?
- What are the development team characteristics?
- How fast does the product need to be operational? Is full functionality required in the first release?
- What are the quality, time and cost requirements?

5.4 Strengths and Limitations

The main research limitation of this thesis is its non random sampling for the questionnaire research this does not allow to generalize findings over whole population. Another limitation is small number of questionnaire responses (20); this also does not allow generalizing the results over the whole population. However it can be argued that the conclusions drawn from this research are valid for the sample they are taken from. Also the research requires further studies, which are proposed in Future Research section of this chapter.

The main strength of this research comes from knowledgeable respondents, each of whom has extensive experience and expertise with traditional and Agile methodologies. Its other strength is the broad geographic sample for the questionnaire. Also research has involved both interviews and questionnaires to gain greater insights into the problem are and attach more validity to the findings. The research creates a basis for future studies into the problem of methodology selection for a software project.

5.5 Future Research

The factors which have been identified through the questionnaire do not have relative weight attached to them. For instance no or low customer participation would mean that Agile would not work at all, while low complexity would still mean that Agile could be used with this project. Therefore the future research could focus on investigating which factors are more important for the selections of methodology. Such research could result in another Agile filter similar to once discussed in Chapter 2 of this thesis.

Another issue that needs further investigation is how do Agile methodologies compare to traditional ones, in terms of meeting cost, quality and time objectives. While do this research the authors have come across several studies which did not receive conclusive results. The interview questions have attempted to explore this issue as well. Several respondents have mentioned that Agile performs better in terms of quality and time while costing more. It was also mentioned that even though Agile works out more expensive for a client, in this situation actually gets more
value for money paid. The research into this issue would provide managers with some solid basis on which they can choose one or another methodology, when the project could be done either way.

It is necessary to assess the culture of client organization; several respondents have mentioned that working Agile would not work with client who has always worked with waterfall. Clients can be educated but it is a long process, without guarantees of success. A research into this issue would provide a stronger case for the applicability of the methodology to the particular project.

As it has been mentioned previously there is not much literature on the topic which this thesis attempts to explore. However there is a growing interest of project management community to this topic, therefore this thesis presents the contribution to the existing knowledge and provides project management practitioners with some of the information needed for choosing the RIGHT methodology for the RIGHT project.
References


Baird S. 2002 Sams Teach Yourself Extreme Programming in 24 Hours. Sams Publishing


Chin G. 2004 Agile Project Management: how to succeed in the face of changing project requirements. Amacom NY US p.21


Gillham, B. 2000 Developing a Questionnaire, Continuum, London, UK


Harrison W. 2003 Is Software Engineering as We know it over the Hill? IEEE Software May June 2003 pp 5-7

Highsmith J. 2004 Agile Project Management: Creating Innovative Products. Addison Wesley, p. 5


Jiang L., Eberlein A. 2008 Towards a framework for understanding the relationships between classical software engineering and Agile methodologies. APOS ’08: Proceedings of the 2008 international workshop on Scrutinizing Agile practices or shoot-out at the Agile corral. APM Publishing


Appendix 1: Online Questionnaire

Hello,
You are invited to participate in our survey, which is focused on identifying what are the project characteristics, which make it more suitable for traditional or agile software development methodologies.

Your participation in this study is completely voluntary. There are no foreseeable risks associated with this project. However, if you feel uncomfortable answering any questions, you can withdraw from the survey at any point. It is very important for us to learn your opinions.

Your survey responses will be strictly confidential and data from this research will be reported only in the aggregate. Your information will be coded and will remain confidential. If you have questions at any time about the survey or the procedures, you may contact Nikolas Sprinsken at 487.386.0963 or by email at the email address specified below.

Thank you very much for your time and support. Please start with the survey now by clicking on the Continue button below.

Please contact nikolaspr@emai.com if you have any questions regarding this survey.
How would you rate agile methodologies you have used if answering the question - "How well does methodology X handle ________?"

<table>
<thead>
<tr>
<th>Rating</th>
<th>Bad</th>
<th>Poor</th>
<th>Well</th>
<th>Excellent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Well defined requirements</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Poorly defined requirements</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High level of end-user participation in requirements identification</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low level of end-user participation in requirements identification</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High level of project manager business domain expertise</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low level of project manager business domain expertise</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High level of project champion risk tolerance</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low level of project champion risk tolerance</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High level of problem complexity</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low level of problem complexity</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High need for localization</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low need for localization</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High risk of failure</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low risk of failure</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Robust level of quality required</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adequate level of quality required</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High level of end-user participation in design</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low level of end-user participation in design</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Broad experience of application architect</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Narrow experience of application architect</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Programmers’ broad skill set</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Programmers’ narrow skill set</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Do you have experience with traditional methodologies?

- Yes (please specify) [ ]

Continue

Please contact [email provided] if you have any questions regarding the survey.
How would you rate traditional methodologies you have used if answering the question: “How well does methodology X handle _____?”

<table>
<thead>
<tr>
<th>Area</th>
<th>Bad</th>
<th>Poor</th>
<th>Well</th>
<th>Excellent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Well defined requirements</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Poorly defined requirements</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High level of end-user participation in requirements identification</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low level of end-user participation in requirements identification</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High level of project manager business domain expertise</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low level of project manager business domain expertise</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High level of project champion risk tolerance</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low level of project champion risk tolerance</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High level of problem complexity</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low level of problem complexity</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High need for localization</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low need for localization</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High risk of failure</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low risk of failure</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Robust level of quality required</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adequate level of quality required</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High level of end-user participation in design</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low level of end-user participation in design</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High level of project manager business domain expertise</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low level of project manager business domain expertise</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High level of project champion risk tolerance</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low level of project champion risk tolerance</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High level of problem complexity</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low level of problem complexity</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High need for localization</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low need for localization</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High risk of failure</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low risk of failure</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Robust level of quality required</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adequate level of quality required</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High level of end-user participation in design</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low level of end-user participation in design</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Broad experience of application architecture</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Narrow experience of application architecture</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Programmers' broad skill set</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix 2: Interview Questions

1) Can you briefly discuss your experience with traditional and Agile methodologies? Which ones have you worked with and for how many years?

2) How did you choose the methodology for the last project you had to do?

3) What are your views on Agile methodologies? Do you think it has got any disadvantage compared to the traditional ones? Do you have any preference?

4) Are there any projects which you think can benefit from the traditional methodologies? And how would describe them from organizational and technical point of view. i.e. resources, financing or databases etc...

5) Are there any projects which can benefit from Agile methodologies?

6) Which different projects fit different types of methodologies?

7) How does deliverables approval time affect Agile projects?

8) How does complexity affect choice of project methodology?

9) Do you think Agile is a “silver bullet”

10) How do you grade Agile projects in achieving time, cost, and quality objective in comparison with traditional ones?