TEKNISKA HÖGSKOLAN

HÖGSKOLAN I JÖNKÖPING

A comparative study on Traditional Software Development Methods and Agile Software Development Methods

Gulshan Aslam

Faisal Farooq

MASTER THESIS 2011
INFORMATICS

TEKNISKA HÖGSKOLAN

HÖGSKOLAN I JÖNKÖPING

En jämförande studie om Traditionell mjukvaruutveckling Metoder och Agile Software Development Metoder

Gulshan Aslam

Faisal Farooq

Detta examensarbete är utfört vid Tekniska Högskolan i Jönköping inom ämnesområdet informatik. Arbetet är ett led i masterutbildningen med inriktning informationsteknik och management. Författarna svarar själva för framförda åsikter, slutsatser och resultat.

Handledare: Christer Thörn
Examinator: Vladimir Tarasov

Omfattning: 30 hp (D-nivå)

Datum: June 9, 2011

Arkiveringsnummer:

Postadress:	Besöksadress:	Telefon:
Box 1026	Gjuterigatan 5	036-10 10 00 (vx)
551 11 Jönköping		

Abstract

Everyone is talking about the software development methods but these methods are categorised into the different parts and the most important are two categories, one is agile software development methods and second is using the traditional software development methods. Agile software methods are relatively considered to be quick and for the small teams. Our main mission is to check which method is better from each other, so for that purpose we go out in the software development market to meet the professional to ask about their satisfaction on these software development methods. Our research is based on to see the suitable method for the professionals; see the challenges on the adoptability of methods and which method is quicker. To perform this study we have gone through a survey questionnaire, and results are analysed by using mixed method approach. Results shows that professionals from both types of methods are satisfied but professionals with traditional methods are more satisfy with their methods with respect to development of quality software, whereas agile professionals are more satisfied with their methods with respect of better communication with their customers. With agility point of view, our study says that both methods have characteristics which support agility but not fully support, so in such case we need to customize features from both types of methodologies.

Sammanfattning

Alla talar om metoderna mjukvaruutveckling men dessa metoder är kategoriseras i de olika delarna och de viktigaste är två kategorier, är en agile mjukvaruutveckling metoder och andra är att använda traditionella metoder mjukvaruutveckling. Agile programvara metoder är relativt anses vara snabb och för små grupper. Vår huvudsakliga uppgift är att kontrollera vilken metod är bättre från varandra, så för det ändamålet vi gå ut i mjukvaruutveckling marknaden och uppfyller de professionella för att fråga om de är nöjda med dessa metoder mjukvaruutveckling. Vår undersökning är baserad på att se lämplig metod för de yrkesverksamma, se utmaningarna på tillgänglighet för adoption av metoder och vilken metod är snabbare. För att utföra denna studie har vi gått igenom en enkät, och resultaten analyseras med hjälp av blandad metod tillvägagångssätt. Resultat visar att professionella från båda typerna av metoderna är nöjda men professionella med traditionella metoder är mer nöjda med sina metoder när det gäller utveckling av kvalitetsprogram, medan smidig proffs är mer nöjda med sina metoder med avseende på bättre kommunikation med sina kunder. Med agility synvinkel, säger vår studie att båda metoderna har egenskaper som stöd agility men inte fullt stöd, så i så fall måste vi anpassa drag från båda typerna av metoder.

Acknowledgements

I am very thankful to Almighty Allah who gave us courage to do this thesis, then Prayers of my Father, my wife, my brothers and sisters. I am very thankful to my coordinator Vladimir Tarasov and supervisor Christer Thörn, who guided me in a best professional way and always with me throughout the program. I would like to thank Jönköping University, who gave me opportunity to study in a world class environment without tuition fee. I would like to dedicate this work to my mother and my beautiful little son Muhammad Ibrahim.

Gulshan Aslam
School of Engineering,
Jonkoping University, Sweden.

I would like to thank to all people who helping and guiding me throughout my master thesis. I am heartily thankful to my supervisor, Christer Thörn, whose valuable guidance and motivation to choose the topic and start my work. His perpetual energy and passion in research inspired me to carry out work. In addition, he is always accessible and willing to help his students. I will show my gratitude to faculty of Information Engineering at Jonkoping University, who guided me throughout my master program. Especially, I am grateful to Vladimir Tarasov, for guiding me during my master program.

I am indebted to many of my colleagues to support me to improve the writing of thesis and my classmates as they made it a wonderful place to learn. It is pleasure to thank to my friends whose valuable suggestions have encouraged me to have great stay in Sweden during my period of study.

Faisal Farooq
School of Engineering,
Jonkoping University, Sweden.

Key words

Methods, Agile, Traditional, suitability, challenges, agility, rapid, developers, quality, risk, standards, communication, requirements

Contents

List of Figures

List of Abbreviations

SDM	Software development methodology
DSDM	Dynamic system development method
FDD	Feature driven development
OOS	Object oriented system
XP	Extreem programming
RUP	Rational unified process
JSD	Jackson system development
ISAC	Information system works and analysis
SDLC	System development life cycle
SIS	Swedish standards institute

List of Tables

1 Introduction

In thesis introduction, we discussed about concepts of the Agile/Traditional Methods and different challenges for their adoptability at both organizational and professional level. In this chapter, the purpose of research on the methods adoptability, developer's satisfaction with the methods and the way professionals who uses these methods at different level is discussed. After background some research questions are formulated in purpose and objective phase of this chapter, and finally it includes the limitations for the research with thesis outline.

1.1 Background

In this era, software industry is growing rapidly and adoptability for the software development methods becomes very hot topic in software industry. "*The current software situation is less than ideal*" [Pg.3- 26]. Since beginning of software engineering and software development, the professionals are using these methods to develop the software products in order to ensure the quality of the software product. But in some way, customers are not satisfied with the software product i.e. information system, sometime it is over budget or too late. Because of long time consumption, it becomes obsolete for the future use.

For the purpose of understanding and to differentiate methods, we divided software development methods in two categories like "Agile Methods" and those methods which are not following agile way development are named as "Tradition Methods".

In 90's, many developers decided to move on the *light weight techniques* [1], so on that time they developed the methods like *DSDM* in 1993 by 16 different organizations from both industry and academics [4], *XP* "*extreme programming*" by 20 professionals in 1998 [1] [4], "*new development rhythm"* developed by three managers of IBM in 1989, *crystal* methods 1991 also by IBM, and the most widely used methods now a day's *scrum* by Jeff Sutherland of easel corporation in 1993 [4] and many more agile or light weight methods have been developed on that time such as *FDD feature driven development* in 1997, *synch-n-stabilize* by two MIT professionals in 1995 [4]. As mentioned above, initially these methods are called as lightweight methods because of their working. The name agile introduced in the market in 2001 by the 17 methodologists in a meeting, which held to discuss about the future in software development [3] [26]. Because they thought that there are a lot of common features in *lightweight* methods but the name light weight does not cover the whole structure so the name *Agile* was decided .In agile methods, people play a driving role in the success of the project, and lot of short time meetings are conducted for knowledge sharing and for the random change in the project if required [2].

Methodologists say that working software without documentation is better than non-working software with a huge amount of documentation [2]. In agile, software product can be produced more early and if documentation is needed then it can be produced at any later time, but not too much documentation needed [2].

History of the traditional methods some time called conventional development methods are much older such as five to six decades and they are product development processes based on sequential building software goods and services theory. In traditional methods, very many hard and fast (rigid) policies, rules, processes, procedures, documentations and tools, are required [4] [5]. Traditional methods are everywhere and Professionals, who says that they are the only old and the truly tested development methods to fulfilment the needs for majority of the projects and large number of organizations [7].Traditional methods make it easier to identify the potential project risks in the start of development process [6]. Because, it explores the project in more details and depths, and due to early alert for the risks makes it easier to plan to stop these identified risks [6].

Traditional methods are considered *heavy* or *monumental* and spiral methods or waterfall methods are some examples of traditional methods. The development style has a very strong effect on the project process, significant upfront planning and documentation [6].In traditional methods, a large number of groups are made, so with these large number of groups more quality oriented work can be achieved [6].

As per "karlsson 2006" Besides all the discussion above some professional considers that it is very much time consuming to use traditional development methods for project, because of spending a lot of time and resources on the documentations, long planning phases and also developers attention should be on the development code for what they have expertise should not give a long time on planning and documentation [6].

1.2 Purpose/Objectives

The purpose of our research is to conduct comparative study on software development methods (Agile/Traditional) and their adoptability that how these methods help professionals in order to fulfil needs and requirements. During method adoptability, what kind of challenges, they face, and which method is more near to agility. In such case, these issues have so far not been adequately addressed. The main focus and objective of this research is to study adaptability of methods and compare them in order to achieve professional satisfaction.

Bearing this in mind, there are many differences in software development methods that has to be identified according to software develops and customer needs. Since, main focus of our thesis, we decided to conduct a comparative study on Traditional Software Development Methods and Agile Software Development Methods. Because, there are many developers who have adopted the agile methods, but on other hand, there are many developers who have satisfactions with Traditional Methods according to their needs. However, we are interested in finding empirical data and come up with tabular guidelines that would be helpful for software community in order to provide strong motivation to have selection of precise and suitable method.

To achieve our goals, there are some following specific research questions, which need to be answered.

- Which software development method (agile/traditional) is more suitable for software development professionals?(with respect to communication, quality standards, quality software, risk analysis)
- What are challenges, an organisation has to face while adopting a software development method (agile/traditional)?
- Which software development Method have higher positive attitude towards rapid development/agile values?

1.3 Limitations

The scope of this thesis is to represent the comparative study of traditional and agile methods. To verify the quality attribute of results, testing and adoptability in company will not be treated in this thesis. To cover the domain of our interest, we are strongly motivated to achieve the results on the basis of data received from the professionals and software companies by using surveys and questionnaires. During research, interviews will not be conducted with companies. Furthermore, we will publish survey to software companies from different countries, such as Sweden, Denmark, Pakistan, and in addition to, hopefully one company from each country like Germany, UK, and China.

To cover the domain to this thesis, we have plan to attract the professional as well as computer students, and their meaningful suggestion would be considered. Moreover, literature and early research has been contributed to this area, which is described within scope of this thesis.

1.4 Thesis outline

This thesis consists of five chapters, the chapter 2 contents based on literature reviews and theoretical background, which describes definitions and important concepts of software development methods. Furthermore, chapter 3 aims to elaborate the research method for this thesis, and chapter 4 represents the empirical data, analysis and results that required achieving the goals and objectives of this thesis. At last, chapter 5 concludes the research finding and further work that would be carried out.

2 Theoretical Background

This chapter will include the basic knowledge regarding the software development methods, how they work and why they are important in the software industry, so that a reader of this research could understand basics of the methodologies. In This chapter, we mainly focus on the most commonly used methods in Sweden, Denmark, Germany, Italy, United Kingdom, China and Pakistan, Also some other methods in the same category will be explained for a reliable historical view. I will first discuss the traditional software development methods which follow the agile software development methods, and then I will try to compare them so that we can write their merits and demerits.

2.1 Method

Since, the field of software engineering is not shy to introduce new methodology. In area of software development, in last 25 years, many software development approaches have been introduced but some to them have survived to be used today [29]. In term of software development, method is known as follows.

A systems development methodology (SDM), defined as a documented collection of policies, processes, and procedures, is commonly used by software development teams to improve the software development process in terms of increased productivity of information technology (IT) personnel and higher quality of the final IT solutions. [5]

2.2 Traditional Software Development Methods

In this part of study, we discuss some knowledge about those software methods which are known as traditional software development methods or Conventional software development methods. Discussion of their usage in the software industry and the advancement will give a big exposure to the readers of this thesis. The important traditional methods which that we are going to discuss is as follows.

2.2.1 Common Features of Traditional Software Development Methods

In the overall view of the traditional software development methods there are lot of common characteristics among them, the most important one is that these all are based on, more or less on the classical waterfall model [47].

Traditional methods are very easy to use and most of them are very well suitable for the all size of projects but especially large size projects. Traditional methods work very well where requirements are very much understood [18].

All traditional software development methods produce requirement specification and full plan of the project, which are then fixed and not changed during the project.

Traditional methods also provide a large documentation during the development of project. Traditional software development methods have very high skills and rigidity due to that it becomes very easy to manage. Each phase of the traditional software development method is very comprehensive and need to be completed before the start of next phase [18].

Traditional software development methods are capable of risk identification earlier in development cycle, which helps to plan for the roadblocks in the project or can be proved that the specific project is not feasible. Helpful to change the scope of project before it starts, which again remove the risk factor. These methods can easily adopt new team member and new comer can understand the project with help of on-going project documentation [6], [47].

Besides following the sequential structure of the project, traditional software development methods also work with iteration of the phases. But still the inner structure same like waterfall model [47].

One more new way of working is introduced like prototyping in spiral model, RUP and object oriented system OOS, which helps view on how software will work and with the involvement of the end user which never done before. But the inner structure is same like the waterfall model with the sport of heavy documentation. There are different traditional software development methods which work at different levels like Direct Model covers only the initial phase/part of waterfall model whereas RUP covers almost everything in different phase [18], [47], [21].

Later on, in this section we discussed several traditional software development methods like Waterfall Model, Spiral Model, JSD, ISAC, RUP, Direct Model, SIS-Reference Model and V-Shaped model which will support all above mentioned features and are built on one another.

2.2.2 Waterfall Model

In 1960's scientists and developers starts working on Life Cycle but that was not a well-designed life cycle, so working was like coding and fixing the problems but in 1970's, first well defined SDLC model introduced with the name Waterfall Model by Winston Royce and then after Barry Boehm in 1976 done a hard work to refine this Waterfall Model [16]. Its name is waterfall because it works like fall of water or flow of the water [15], [18], [20]. Scientists use different names for that model like *"Classic Life Cycle or linear Sequential Model"* [20]. Most of the organizations use this methodology with a customize form or way to fulfill their needs regarding the specific project [16].

Because of the late delivery and consumption of a lot of time some project managers made some big changes in development life cycle and some managers bypass some of the phases considering a short cut but this makes the project/product more unreliable and sometime more expensive in form of maintenance cost[16].

In Waterfall Model Seven to Eight Phases are considered very important and each step follows the new step until last step completed [17].

- Feasibility
- Requirement
- Product designing
- Detailed designing
- Code
- Integration
- Implementation
- Maintenance

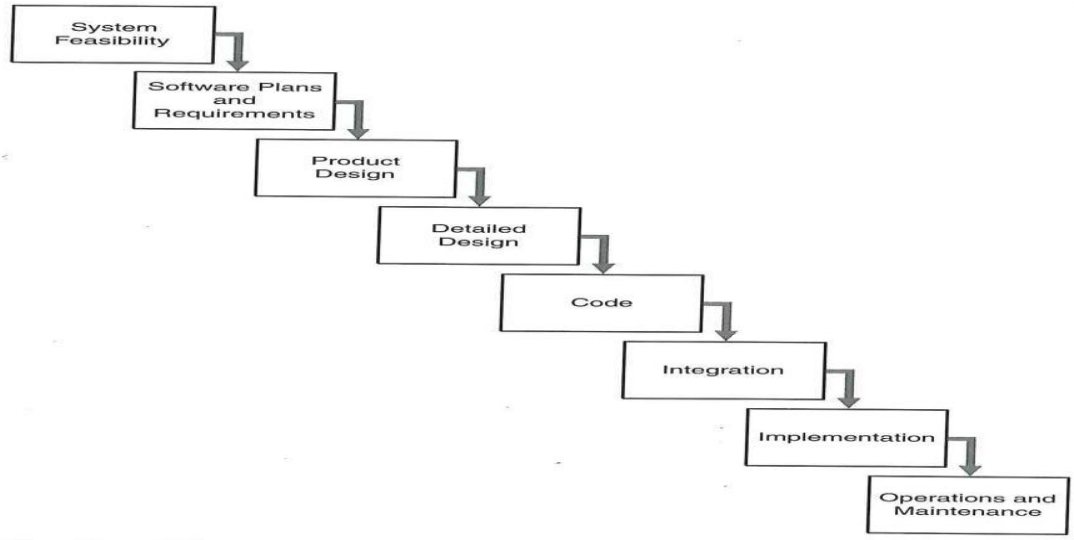


Figure 2-1 Water fall model [15]

One thing is considered very important that each step is predecessor for the next step and assumed to be complete and not to be updated in same project, because the as above we mentioned that it works like the flow of water and water never climbs upwards. Assumption play a very important role in the waterfall model and once we assumed that the phase is completed, so it is done [15], [20].

In 70's and 80's software developer and Scientists done a huge amount of research work on the waterfall model to enhance its capabilities and worth. So modified version of waterfall model is introduced which can be named as Back Flow of waterfall model

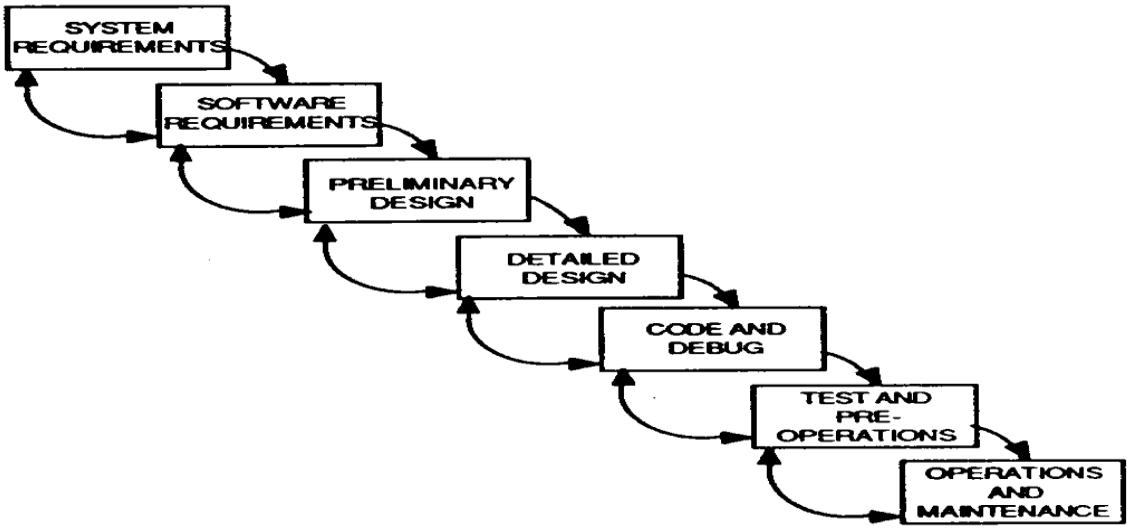


Figure 2-2 Waterfall model with the back flow [16]

2.2.3 Spiral Model

The method which is supporting the incremental delivery is introduced by the Barry Boehm in his article published in a famous journal IEEE computer in 1988 named as Spiral model [15], [21]. The evolution of the Spiral model is based on the long working experience on the refinement of Waterfall model [22]. Spiral model having four high-level phases [15], [21], [20] like first objective is identified as defining the product, business objective and constraints, Second it perform prototyping and risk analysis, Third phase is the product development which include coding, designing, testing and integration and the fourth one is planning for the further iteration which includes implementation, design planning, customer evaluation and delivery to the customers [15], [21].

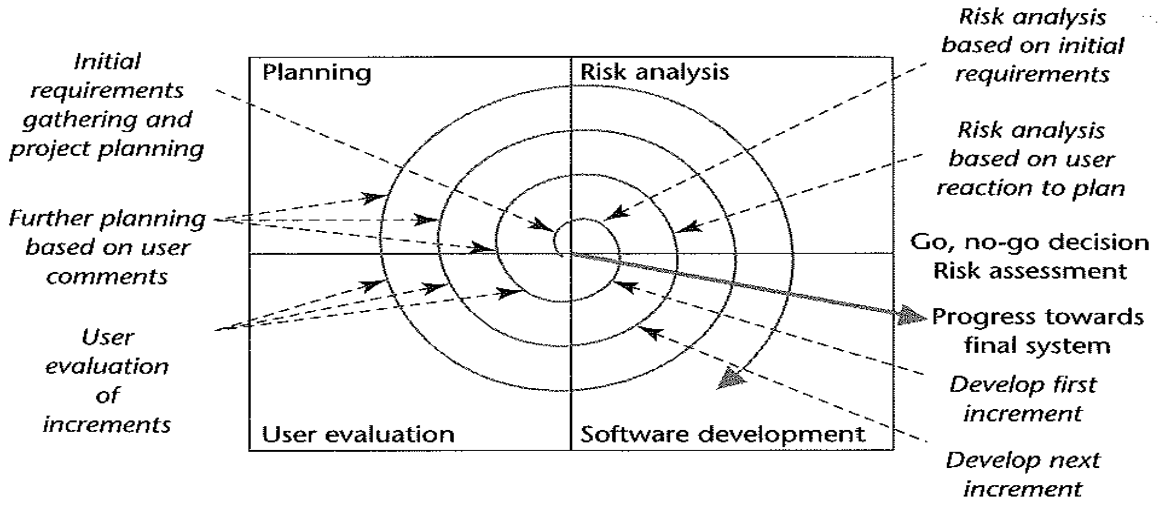


Figure 2-3 The Spiral model from [21]

The main purpose for the designing of Spiral model is to improve the risk analysis and management and to increase the ability of a system to survive [20]. The risk analysis may include cost check, prototyping is also incorporated as minimizing risk, and both reuse risk analysis and prototyping are often used between design and code section [22]. It also helps to flow back and redesign the work, so new features can be added and new risk issues can be solved [22]. Developers can use the new spiral model in case if some kind of maintenance is required [22].

2.2.4 Jackson System Development (JSD)

For every system developer it is very important goal to produce a well-designed system [23]. JSD (Jackson system development) developed by the Michael Jackson in 1983 [23]"*in one article Jackson said the very first JSD in start of 1980"* [24]. At this stage there are many systems development methods are available but the strong point of the JSD is that it focuses on the both phases like design and development, which is implementation phase [23].

JSD is the extension project of JSP to specify and implement the information system [24], Jackson said that an information system can be model or simulation of real world having some added functions to have information output [24]. It seemed very useful for simulation and command and control system e.g. Helicopter Fly-By-Wire and sub marine system [24], and after 1984 Jackson shifted towards software engineering for the software development. Still many organizations in Europe and US, JSD used successfully but on a small scale [24].

There are six independent steps in this effective method so that developer can learn more and more [23], five steps are used design and analysis and last step is used in implementation [23], [27], initially stage in JSD method is to draw a real world model once you have done, so you can enter into next phase/step [23], mentioned as follows.

- Entity Action Step
- Entity Structure Step
- Initial Model Step
- Function Step
- System Timing Step
- Implementation Step

2.2.5 ISAC (Information System Work and Analysis Change)

ISAC (information system works and analysis change) is a method which is been developed by Lundeberg in late 60's and early 70's by working with his

one colleague [19]. ISAC is further divided into two dimensions which are the change analysis and activity studies and ISAC is based on integrated set of graphical notations [19] and also the first integrated methodology that addresses the "*whole ISD process systematically starting from organizational design to technical design and implementation"*[19]. A Number of action research projects ISAC was developed [19]. ISAC was initially developed for the in-house development for single applications [19], and in 1980 ISAC was considered mostly used methodology in Scandinavian countries and also in Holland [19].

2.2.6 RUP (Rational Unified Model)

Rational unified model RUP developed by a software developer company named rational software [25] owned by IBM, and aiming that to guide the software development process [25], it is the approach which use case driven, iterative and architecture centric [28]. In short, RUP is a well-defined and structured process because of clearly defining who is responsible for what and when [28], by clearly defining the milestones and decision points. It is a process framework [25], [28] to provide customized process framework in software engineering [28]. It also contains many out of the box process configurations and process views to guide software professionals [28]. CMM (capability maturity model) development by SEI made it more important to have a well-defined and well documented project, so that the companies can achieve the CMM levels and also the project will be successful [25].

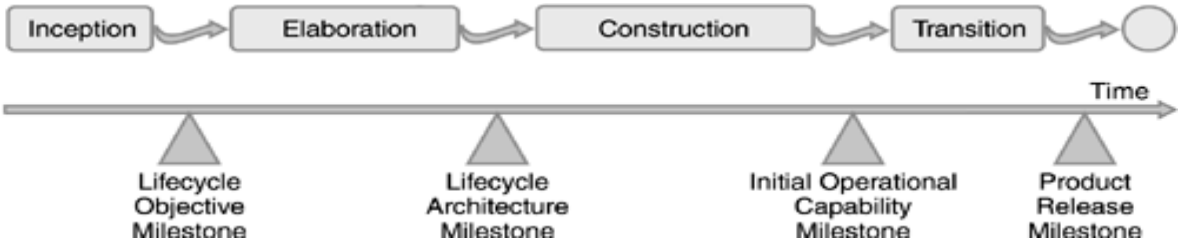


Figure 2-4 Milestones for the RUP Lifecycle Phases [28]

RUP has a very close connection with UML in underlying Object Oriented Models. It has two structures or dimensions, horizontal and vertical directions explained in figure mentioned below [25].

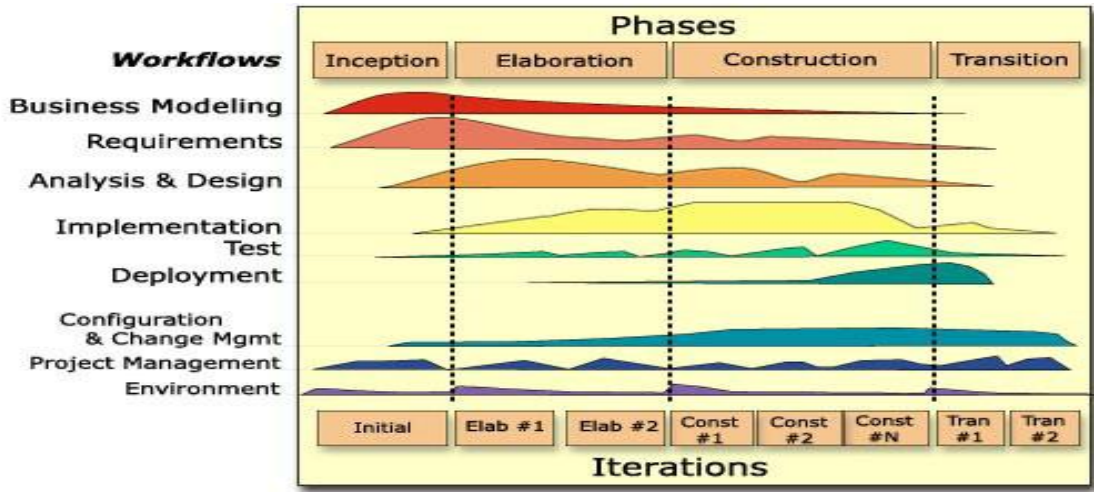


Figure 2-5 Two Dimensions of RUP [25]

Time and processes life cycle aspects are represented in the horizontal dimension and in vertical dimension shows the workflows. Dynamic aspects are also represented in horizontal dimension [25].

RUP has many software development practices which are suitable for the large projects and organizations [25].

- *Develop software iteratively.*
- *Manage requirements.*
- *Use component-based architectures.*
- *Visually model software.*
- *Continuously verify software quality.*
- *Control changes to software.[25]*

A lot of organizations, such as, telecommunication, manufacturing, system integrators, defense and financial services are using RUP for a better and quality product in both large and small size projects, which shows the versatility of the process (RUP) [25].

2.2.7 Direct Model

In 1985 SVEA-model was introduced by Lars Axelsson and Leif Ortman to make a process speedy in software development with simplicity and was very popular in the Swedish companies. Further work on the SVEA-model leads to the new model named as direct model, which helps the developers in continuing the work and clarify where SVEA model is unclear, so it provides more perfection while working on Direct model [15], [47].

It deals with direct coupling in business development, business users direct engagement, direct derivation of the results and fast work in development and direct production for system solution. It covers the initial part of waterfall model [47], [15].

This model can be divided into four different steps that depends on each other and follow the parallel working process with a bit of documentation helpful for the upcoming step/level, constituting a milestone. Four levels of direct model are:

The starting run: It includes to see the alternatives, need of change is measured, development of plan on which whole is depends on. Mapping and description of idea are also included. [47]

Modeling and formation detail: It includes different work steps such as table specification, construction steps, input specification and output specification, routine outlining, prototyping, data modeling and model summary. [47]

Testing and Realization: Here at this level Information System is developed and tested. Organizations development, programming, technical adaptation and database development/construction are part of this step in direct model. [47]

Implementation and tuning: Completion of documentation, final changes are made if required, IS delivered. Program/database tuning is also included. [47]

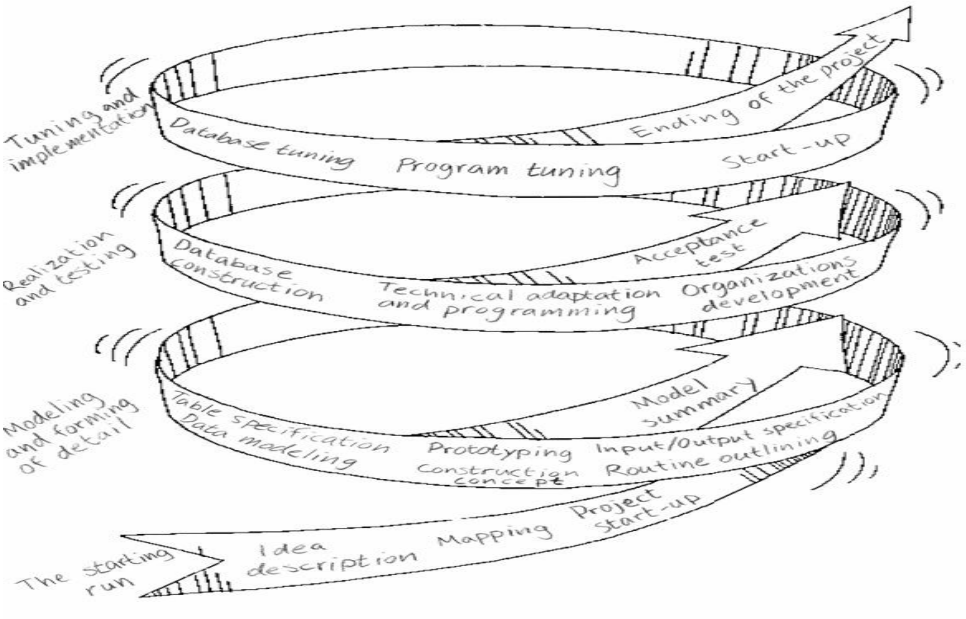


Figure 2-6 Direct Model [47]

2.2.8 SIS-Reference Model

In 1989, a revised version of SIS-RAS model was developed, known as SIS-Reference model, it has different stages performed in iterative manner instead of sequential [46][47].

Further all these stages are sub-divided into small piece of steps. When these steps are finished / finalized, decision is made for further iteration or to move on the next step/stage if functionality and quality is sufficient. At the end of system development process, the above mentioned structure of stages assures that nothing is missed or forgotten. A very important feature of SIS-Reference model is its customizability, where user influence can be emphasized with suitable parts of SIS-Reference model [46]. By using SIS-Reference model user can achieve, Prototyping technique, Maximum user influence, Fourth generation tools, and Incremental expansion.

In customization different structures with stages/steps are available. If we desire, we can chose either MAXI model with lot of stages but less number of steps in each stage. Whereas if we choose MINI model, which is totally opposite by having less number of stages with lot of steps in each stage. By implementing the above mentioned activities we can have a better plan which is more suitable for our software development project [46], [47].

2.2.9 V-Shaped Model

V-shaped model introduced in Germany, internationally accepted standard ISO/IEC 12207 or ISO 9001 and highly used in civil, military and large size federal IT projects [20]. Its execution process is sequential like old waterfall model. It has different phases and one phase must be finished before new phase is started [18].

Project management, quality assurance, software development and configuration management are the main focuses of v-shaped model [20]. Testing is considered very important in v-shaped model, so a test plan is developed before the start of development. Main function of the test plan is to meet the functional specification mentioned in the requirement gathering [18].

It helps to have a better communication between the customer and the developer and also it provides the low cost projects with better quality [20]. It simple and easy to use and have more success chances because of early testing phase. It works very well on those projects where requirements are easily understandable. Besides the above mentioned, it is also very rigid as like the previous waterfall model and very less flexible [18]. A figure 2-7 shows how v-shaped model works.

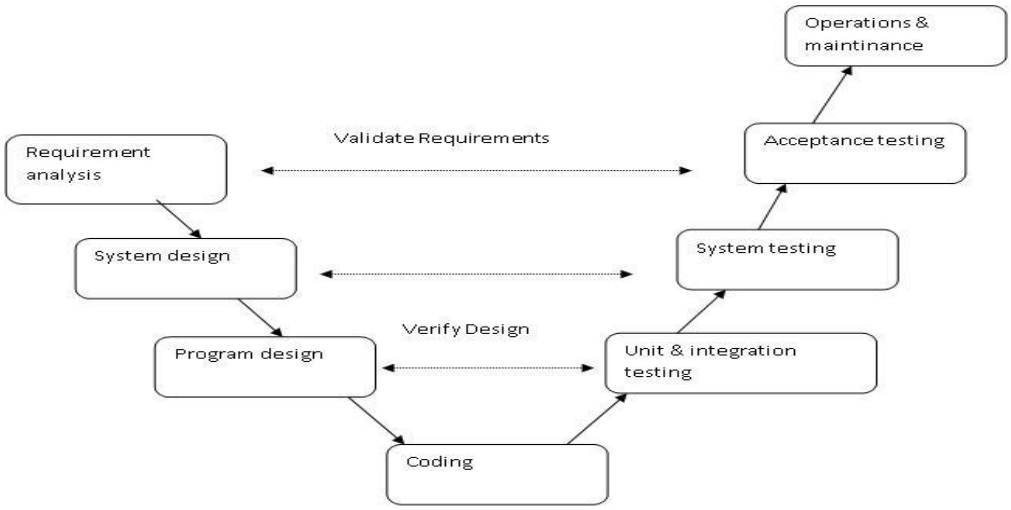


Figure 2-7 V-Shaped Model Diagrams [20]

2.3 Agile Software Development Methods

As oxford paper back dictionary defines "Agile" in term of quick-moving and possibly surprising term for software engineering that reflects the primary goals of software development method under term of agile [31].

In February 2001, the of group of agile software development methodologies was named as "Agile", when group of practitioners met and formed Agile Alliance that is association aimed to formalize the agile methodologies [32]. Furthermore, main purpose of introducing agile methodologies that could be employed to merge to new software engineering discipline, which shifts value of software development process from mechanistic (i.e. driven by process or rule of science) to organic (i.e. driven by software issue of people and their interaction) [32].

In report of *Scott W. Ambler* et al [33], he clearly stated that there is no official definition of agile software development methodology, but in working perspective agile methodology defined as follows.

"Agile software development is an evolutionary (iterative and incremental) approach which regularly produces high quality software in a cost effective and timely manner via a value driven lifecycle. It is performed in a highly collaborative, disciplined, and self-organizing manner with active stakeholder participation to ensure that the team understands and addresses the changing needs of its stakeholders. Agile software development teams provide repeatable results by adopting just the right amount of ceremony for the situation they face" [33] .

After first workshop on agile methodology was held in June 2002, Lindvall describes another working definition of agile software development methodologies as a group of software development processes, which are iterative, incremental, self-organize, and emergent [48]. Furthermore, figure 2-8 depicts the agile methodology.

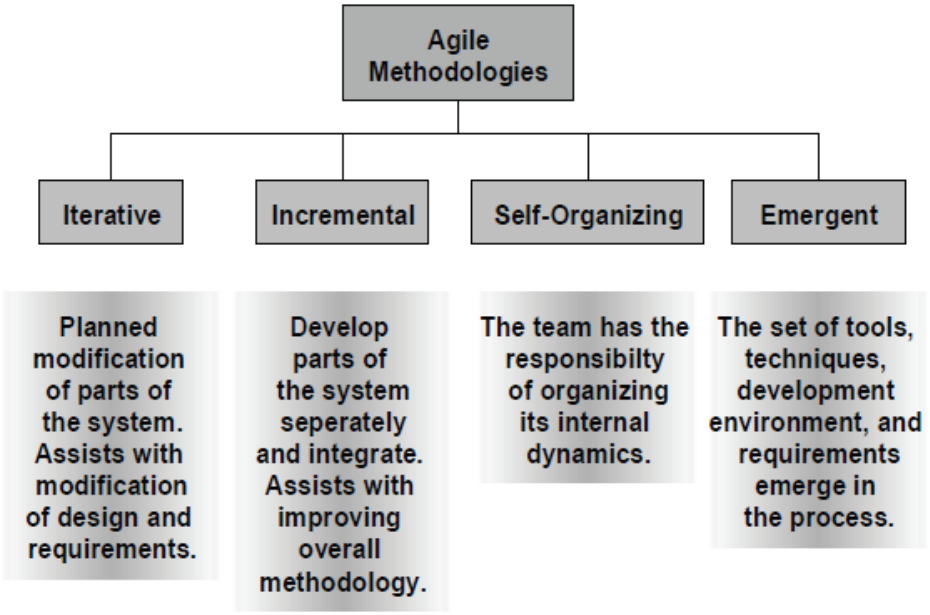


Figure 2-8 Agile methodology [32]

With respect to return on investment on agile methods that are generally characterized by using lightweight, informal, and highly adopted to new software development processes [4]. These characteristics enabled the agile methodology to develop software product in much short time as compared to traditional waterfall model [34]. Furthermore, Favaro suggested that agile software development represents iterative development in better way, and better control of changing requirements on early stages of projects [35]. However, according the fast delivery point of view, agile processes enabled short life cycle of projects.

2.3.1 Agile Manifesto

In order to improve the development processes, there was growing movement by Agile Alliance in 2002, when they introduced and promoted the agile manifesto to enable the professional in supporting for better software development [31]. There are following key points of agile manifesto [31]:

"Individual and interaction over process and tools"

"Working software over comprehensive documentation"

"Customer collaboration over contract negotiation"

"Respond to change over following plan"

Aiming at presenting agile manifesto, the first key point is to have better communication in team members that affects the success and failure of delivered software. On other side, tools and methodologies help the developers but agile encourages the developer to work in groups with better collaboration. Another key point is to deliver the working software rather than comprehensive documentation. Of course, documentation is not end product that could have importance to have better understand ability of software and executes plans in successful way. But it's much important to come forward to fix the changing requirements rather than documenting and planning. [31]

To be able to validate requirements, agile manifesto optionally encourages the developer team that end customer has to be involved with team in order to enable rapid development and early delivery to customer. However, development of required changes in response to customer feedback rapidly are more important rather than to have discussion on fixed plan. In such case, software planning also has importance to put developer on right track according scheduled time. [31]

2.3.2 Common Features of Agile Software Development Methods

With respect to fast delivery point of view, Miller [42] describes the following features of agile software development processes.

1. Modularity of development process level.
2. During development, iteration with short cycles enabling fast verification of requirements and correction of issues.
3. Iteration cycles has time schedule from 1 to 6 weeks.
4. Remove all unnecessary activities in development process.
5. Incremental development approach that enables the developer to work in steps in supporting software development.
6. Incremental approach and discussion on open issues minimize the risk.
7. People oriented that makes important of people rather than tool and technology.
8. Provides collaboration and communication working style.

2.3.3 Dynamic Systems Development Method

DSDM (Dynamic Systems Development Method) was invented in 1993 [29], and major objective of DSDM is to provide the richer framework for rapid application development. To be able to make more flexible control on RAD, DSDM enables the software professional with guidelines on how to use framework efficiently. Figure 2-9 depicts the DSDM process that consists of five phases.

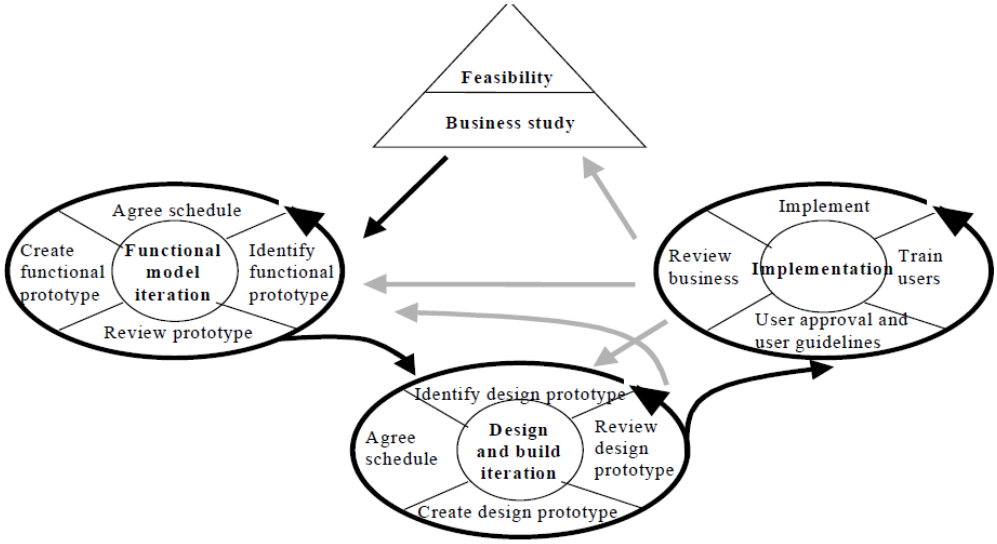


Figure 2-9 DSDM Process [29]

As stated above, DSDM contains five phases where first two phases are sequential that has to be completed once. To proceed with actual development, remaining three phases are considered as iterative and incremental that is important concern in order to develop system in rapid way. The feasibility study aimed to assess the DSDM that might be adopted in organization where project type, people issues, tool and technologies are taken into consideration to identify the risk level. Moreover, major objective of business study phase is to elaborate the features and characteristics that are to be analysed before starting project development. To identify software requirements, more precise way to organise the workshop in supporting to gather requirements, and agreed upon on requirement prioritization. However, major deliverables of this phase are architecture definition of system and outline of prototype plan. [29]

During functional model iteration, requirements content and approaches are planned to build functional prototype where analysis and coding are done to examine the result for next iteration. This phase aims to deliver functional model, which contains code of prototype and analysis models. The main objective of build and design phase is to build the system that delivers set of agreed requirements and features that has to be tested on every iteration.

To carry out further development, design and functional prototypes are reviewed by user who can put comments and suggestions for further development. In final implementation phase, final production of system is delivered to working environment where professionals are responsible to provide user manual and project review report, in addition to; user training and guideline are held to implement the system. [29]

2.3.4 XP (Extreme Programming)

XP (Extreme Programming) is most widely used method in agile methodologies and it was introduced by K. Beck [37]. There were a lot of reasons to promote and influence to XP, such as, problems caused by long development cycle of traditional software development methods [29]. Indeed, the main focus of XP is to get job done [29]. The term "Extreme" involves the common practices and principles at extreme level, see more detail in [37].

In addition to, key process can be organized by short development cycle, incremental planning, evolutionary design and ability to response the customer feedback [36]. In order to deliver immediate business value, XP has been designed for small team (less than 10 developers) with collaboration of customer, who been involved to have discussion on valid requirements and early feedback on incremental delivery software system [36].

According to Beck in [37], XP consists of five phases such as exploration, planning to release, productionizing, maintenance and death. See figure below to depict XP process.

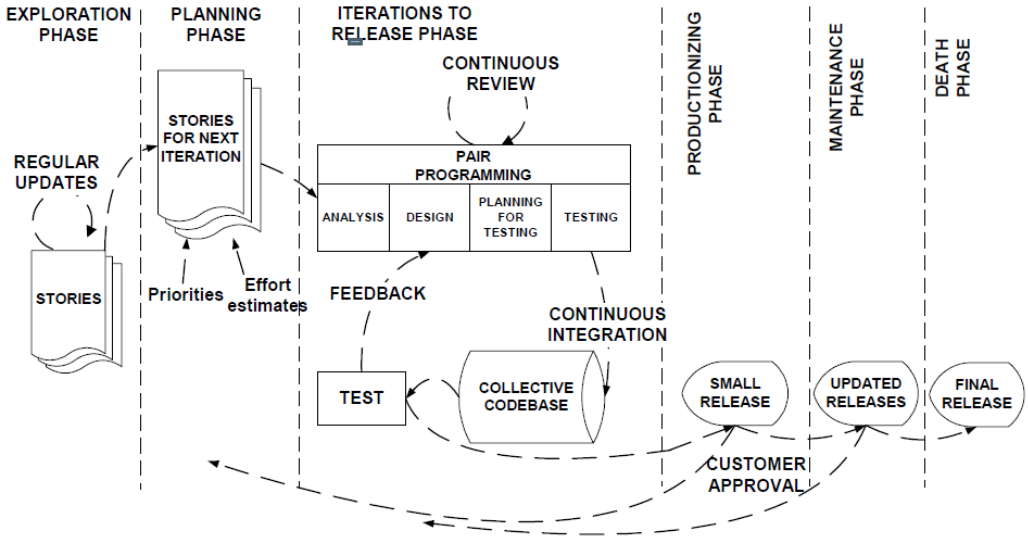


Figure 2-10 XP Process [29]

In report of Beck et al [37], exploration phase elaborates the start-up of system where customer writes the story cards, which are primarily requirements of first release. In order to make skill set; developers make exploration and familiarity with tool and technology that has to be tested for system. In particular, on the bases of initial story cards, system architecture possibly discussed and developed by using prototypes. Next phase aims to plan the first release of system that required the prioritization of story cards, which are to be included for first release. In order to make schedule of first release, how much efforts and estimated time required on each stories where customer and developer are agreed upon.

In next phase, first release might have several iterations to build up system architecture. One of the key elements is to put developer together by using concept of pair programming in order to analyse and design the story cards. During continues integration on every release, customer generates the function tests, which can be applied to test and verify every iteration. At the end, after all iterations customer has end product.

In productionizing phase, extra testing and checking required before final release in order to make sure reliability and productivity of system. Moreover, death phase is near when no more stories or requirements are left. By using XP, this is suitable time to document the system, when no changes required in architecture and code. The concept death that might be occurred in case system does not fulfil the desired functionality or being much costly for end customer.

During practices of XP, roles and responsibilities have been determined by Beck [37]. In order to make process useful, programmers are responsible to implement well defined code, and perform unit testing. Customer is an important role, who makes story cards and verifies the deliverables, and he has been involved to write functional tests by collaboration of tester. It also important to observer and track all stories are being developed according to estimated efforts and times that might have effects on other deliverables schedule.

To be able to adopt the XP process in precise way, team coach who has good knowledge of XP to determine deficiencies of method, in case team does not have enough knowledge on how they can change in process. Moreover, consultant who has been involved to provide the consultancy on tool and technology, and manager has eye on whole process to distinguish difficulties in the process. [29]

Since, the focus of our research mainly on adoptability of software development methods, as Beck reported in [38], XP should be adopted gradually as stated:

"*If you want to try XP, for goodness sake do not try to swallow it all at once. Pick he worst problem in your current process and try to solve it in XP way*"

There is no such experience of reports in which all practices of XP have been accepted to all size of projects [29]. Bearing this in mind, individual practices can be employed partially or that should be tolerated by stretching practices. XP practices are more suitable for small to medium size projects as Beck suggested in [37] the size to be limited between 3 or maximum 20 members. According to [29], it could be difficult to estimate the time of problem in hand. In addition to, Maurer and Mortal in [39], they include the concrete numbers on using XP and stated the questions on how XP would be productive in web projects.

To develop rapidly, XP influence the developers to involve with customer to write the stories cards in supporting the validity of requirements and to have rapid feedback. By using XP practices, in small to medium size team, rapid development can be enhanced at least once release after 2 to 3 month, and possibly new version can be released daily for short release. [29]

2.3.5 Scrum

The term 'Scrum' was pointed out to article of Takeuchi and Nonaka, in which quick, self-organizing and adoptable product development process creating from Japan is described [29]. Additionally, the term 'scrum' was introduced from strategy in game of rugby where it denotes "getting an out-of-play ball back into the game" with team work [41]. In particular, scrum is iterative, incremental process that is more suitable in environment of constantly changing requirements [40]. Figure 2-11 below describes the scrum processes and practices.

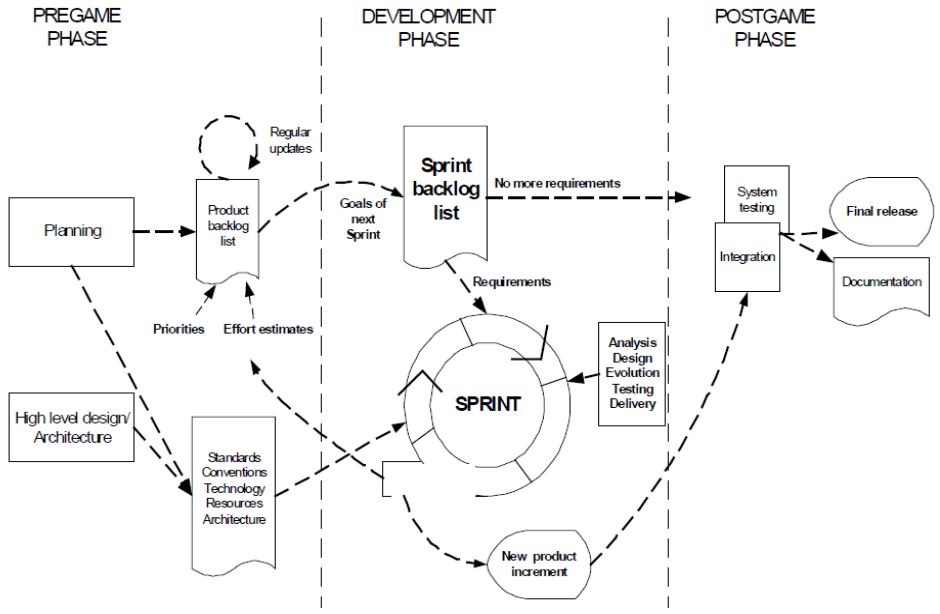


Figure 2-11 Scrum Process [29]

According to above figure, scrum process includes three phases: pre-game, development and postgame.

According to [29], first pregame phase consists of two phases such as planning & architecture or high level design. In requirements engineering, product back list phase aimed to generate the requirements from customer, sales, marking division, and customer support or software developer in order to prioritize the list of all system features. The main intuition of this phase is to plan fixable mostly 30 days sprint, and involved major dependent variables such as requirements, time, resources, knowledge and tools are discussed during sprint planning. To create system architecture, design review meeting is held to make decisions on proposed solutions according to current items in product catalogue.

Next development phase also called game phase that provided well-defined strategies to develop the sprints in way of iterative cycles. In addition to, this phase treated as black box where unpredictable is expected, when they may have change in environmental variable as stated before. Each sprint phase includes the way of traditional phase of software development, in particular, these are requirements, analysis, design, and evolution and deliver phase.

To verify the system architecture, first cycle of sprint considered most important where architecture of system could be examined in case any changes required on early stage of development. In last phase, post-game elaborates the practices including integration, testing of system and documentation. However, after many iteration of sprint cycle, system is ready for final release, when all requirements have been completed as all stakeholders were agreed upon.

2.3.6 Feature Driven Development

FDD (Feature Driven development) was introduced by Coad [43], and first time, it was adopted in area of bank application project development in 90's [44]. Same like other methodologies, FDD does not fit to all activities of entire software development but provides much inspiration to concentrate on design and building phases [44]. During monitoring of project, FDD focuses on quality aspects of process and contributes extensive deliverables. See figure 2-12 below depicts the FDD phases.

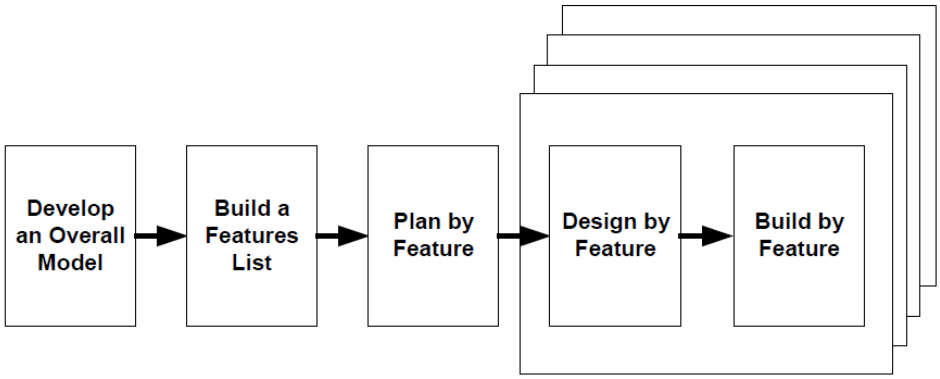


Figure 2-12 FDD Processes [29]

As above figure depicts the FDD that consists of five processes, which described by Palmer, see more detail in [44].

At early stage of system modelling, stakeholders such as domain experts, they define the context and scope of system to be built. FDD is not extensively involved to manage and gathering requirements but, somehow uses cases and functional specification are discussed to document requirements. Furthermore, domain of system is divided into sub domains areas where domain members and chief architecture have detail discussion to create high level description of the system that's called "walkthrough". However, teams are formed in groups, and after each group works in order to design object model of every domain in hand.

In next process, walkthrough, requirements and object model are considered to list down features of system that includes client valued functions. In order to make requirements validity, feature list is examined and reviewed by user and system sponsors. Aiming at planning, next process focuses to present high level plan, in which set of features are organized according to their dependency, and might have mile stones in supporting to review system progress. Last process aimed to select the group of features from feature sets and assigned to different group teams where each team involved in tasks such as design, coding, unit testing, system integration and code inspection by using iteration process. After effective iteration, the completed features are integrated to main build of system while next group of features are assigned for designing and building. [29]

2.3.7 Crystal

Crystal software development method is combination of different methodologies, which can be applied with respect of project size. As figure 2-12 indicates below, crystal family is indicated with different colours and darkness that determines darker colour seems to be used as heavier methodology [29]. Furthermore, figure 2-12 depicts the character symbols; where potential loss is reason of system failure that leads to different dimensions of crystal family such as critical level, comfort (C), discretionary money (D), essential money (E), and life (L), see more detail in [45].

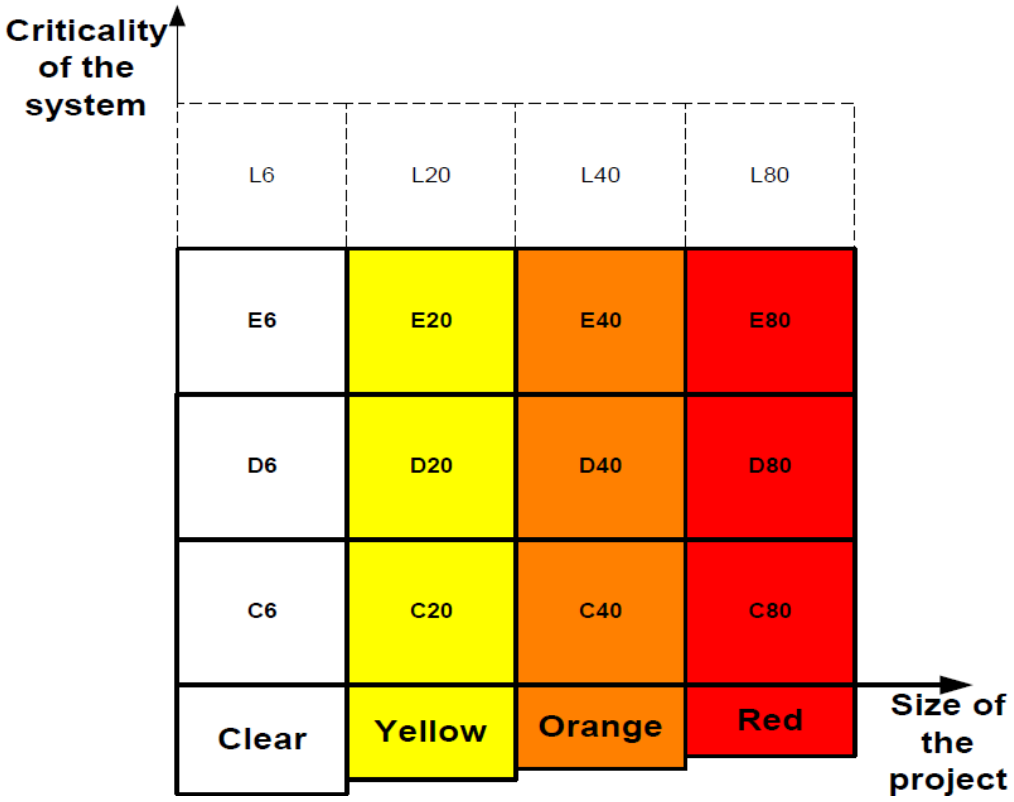


Figure 2-13 Crystal family dimensions [45]

Crystal family main focus is to use certain rules and common features for included software methodologies, and major concern is to adopt incremental development approach for projects, which have length of four months, but in report of Cockburn et al [45], he suggested the project length between one to three months. In particular, Crystal methodology is further constructed into three methodologies: crystal clear, crystal orange and crystal orange web. See more discussion in [45], [29], there is hot debate on differences and similarities, and process activities of these methodologies.

2.4 Comparison of Software Development Methods

As discussed earlier, software development methods are categorized into two categories, such as Traditional and Agile software development methods. Therefore, this section examined the previous study on comparison with respect to our research focus.

2.4.1 Differences between Agile and Traditional Software Development Methods

Throughout literature study of traditional and agile (section 2.2 and section 2.3), a clear difference can be made on different key features of software development methods. Underlying principle of tradition methodology is to put the developers on several phases, which are convinced by plan according to customer requirements [47]. But in area of agile methodology, development methods allow the developers to involve the customers in order to improve flexibility in changing requirements. Furthermore, developers of agile being involved to deliver the prototyping or beta version of system, and getting rapid feedback from customer in response to changing requirements. However, project size is limitation of agile of methodology, it makes more difficult to manage team, if team size more 40 developers, then most preferable approach is to choose the traditional methodology [40].

As reported in [40], there were some drawbacks in traditional approach, such as linearity, inflexibility in changing requirement and high formal processes irrespective with size of project. Bearing this in mind, Beck [37] took these drawbacks and introduced XP that was first agile method. Additionally, comparison study in [40] argued that traditional methods are considered as time wasting including activities like documentations, writing analysis and design documents, when deliverables of project are tightly scheduled with low time efforts. In such case, when time is limited, most desired approach would be agile methodology.

A survey has been conducted on comparison study of software development methods [47]; they suggested that practisers of traditional software development methods have more satisfaction level than practisers of agile software development methods. Moreover, it clearly stated that practisers of agile software development methods are more satisfied with their method in order to fulfil the customer desires and requirements as compare to processes of traditional methods are not fully recommended in support for customer communication.

2.4.2 Project Characteristic

With respect to project characteristic, table [30] below identified the key points in order to make choice of software development methodology.

Table 2-1 Agile and traditional methodology discriminator

Project Characteristic	**Agile Discriminator**	**Heavyweight Discriminator**
Primarily objective	*Rapid Value*	*High Assurance*
Requirements	*Largely emergent, rapidly changed , unknown*	*Known early and largely stable*
Size	*Smaller team and projects*	*Larger teams and projects*
Architecture	*Designed for current requirements*	*Designed for current and foreseeable requirements*
Planning and Control	*Internationalized plan, qualitative control*	*Documented plan, quantitative control*
Customers	*Dedicated, knowledgeable, collaborated, collected onsite customer.*	*As customer needed for contract provision.*
Developers	*Agile, knowledgeable, collected, and collaborated*	*Plan-oriented, adequate skills access to external knowledge*
Refactoring	*Inexpensive*	*Expensive*
Risk	*Unknown risk, major impact*	*Well understood risk, minor impact*

2.4.3 Agile Manifesto with Agile VS Traditional

In report [49], a hot debate was held between agile and traditional practitioners in supporting to discuss agile manifesto in term of traditional practices, and how traditionalists react and reject agile principles. With first agile manifesto value "Individuals and Interaction over Process and Tools", author strongly motivated on skills programmers and software engineers who required being involved in order to construct highly valuable software. In this particular, tools and processes are considered behind this aspect that traditional practitioners won't have much trouble to accept this. Even, SEI [49] has formulated People Maturity Model to support the CMM, and mostly practitioners accept that people matter.

With second agile value "Working Software over Comprehensive Documentations", again author prefers first option by stating that traditional approach based on document-driven life cycle, and increased number of documents over the years rather than focus should be on end product. Traditionalists likely to measure progress of software by adding extensive documents to management in order to show up development activities, even, they considered this as least activities and avoid the code units. Author argued that RS document and user manual are matter but delivered product is most important concern of agile practitioners. [49]

However, as stated before, RS document matters but agile community activists that requirements should be expressed in automated acceptance tests than whole RS document once. In this sense, they want to get requirements incrementally in order to make sure exactly what client needs. [50]

So far, 3rd agile principle "Customer Collaboration over Contract Negotiation", author agreed on both sides by concluding remarks that contract would never be need if customer collaboration would be sufficient [49]. On other hand, author also deeply believes in contract, it would not take significant efforts of customer collaboration but some time degree of laws makes people lose sight what is important [49]. Very often, business people try to protect them on what they might have missed out in specification [50]. Additionally, anti-agile users reasoned that hiring onsite customer leads to increase the overall budget, and in some cases, it makes rework and project scope creep [50].

According to [49], again author recommends both sides on "Responding to Change over following Plan" in case traditionalist designed flexible and rigid plan. Bearing this in mind, some customers required changes, which might not be valid requirements, even, some time they do not know exactly what they desire in system. Furthermore, late changes in project affect the cost and schedule of project.

2.4.4 Challenges

In this section, our objective is to discuss challenges that CIOs and project manager must be aware off, when they are going to adopt agile software development process in their organization. The changing in development environment such as, tools, technologies, and programming languages may cause of development problems.

2.4.4.1 Management and Organization

To adopt agile methodology, organizations need to reconfigure team structures, managerial strategies and technology components in order to implement methodology successfully [51]. In traditional approach, project manager seems to be planner and controller that role must be shifted to facilitator who responsible to be part of development team in order to final decisions on team member suggestions. In this case, biggest challenge is to treat the project manager to resign authority he/she previously designed. Hence, in case of agile methodology, organizations depend on development teams that potentially shit the power from management to development teams [51]. As reported in [53], agile method, such as XP does not address communication issues between multiple teams that create additional overhead on developers and decrease agility of each project.

As stated before in section 2.3.4, in agile methods, customer has to be involved with software development teams for decision making. In such case, decision making environment is quite different from tradition approach where project manager is responsible for decision making. Therefore, organization may need to build culture of trust between employees and people in order make collaboration decision making that may take organizations time, efforts and patience. Furthermore, it would not be easy task to find out such customer who is "Collaborative, Knowledgeable, Committed, Authorized and Representative" [51].

2.4.4.2 People

In report of P Boem & R Turner et al [52], he clearly stated that organizations must think about synchronization of teams, when they planning to adopt new methodology. Some organization have experienced successfully with medium-sized teams-of-teams [52]. It is quite challenging to find out capable team leader who have mix of technical, people and agility skills. On other hand, there are some issues on roles and responsibility using agile methodology. In addition to, very often, agile development team members cross boundaries on standard description of designation that might be required more knowledge and skills to perform sufficiently.

2.4.4.3 Process

According to [51], there is biggest challenge in migration of methodology is to change in process model from life cycle model in order to achieve feature based development using evolutionary and iterative development. In this sense, such changes in process model might have impact on roles of people in organization, tool and technologies, communication ways and work procedure strategies. Moreover, people with traditional longer life cycle required adjustment with agile process [52]. Thus, agile makes software development life cycle like maintenance phase by adding short and focused iterations [52].

To apply the agile process to legacy system, it creates numerous issues whether maintenance or new development of legacy system is required [52]. In requirement engineering, professionals argued that agile tends to focus on functional requirements using story cards. In this case, software developer might need to add additional information to requirement statement or to story cards for non-functional requirements, such as security and reliability [52]. With agile approaches, the level of uncertainty and ambiguity are much higher as compared to traditional approaches, when resource loading, estimation and calculation are important concern [52].

2.4.4.4 Tools and Techniques

In this area, tools and techniques are very important in supporting to facilitate development teams for rapid development, and have significant role in successful implementation of methodology. In such case, those organizations making plan to adopt agile methodologies, they must invest on tools that support iterative rapid development and configuration management [51]. However, people must be trained in order to use tools and techniques correctly that may lead to put more investment to adopt methodology in success way.

3 Methods

This chapter is about the research method that is an important component, when we do research. To ensure the validity and quality of work, we need to follow appropriate research method to be able to carry out research and to make our work reliable and manage in precise way. During research, method plays important role in order to achieve the goals of the specific problem.

3.1 Research Methods

To be able to execute research, there are many research methods and data collection techniques available to follow in order to achieve the research finding. However, the selection of research method depends on problem in hand. There are two major types of research methods / research techniques available for data gathering and analysis.

Qualitative Research

Quantitative Research

3.1.1 Qualitative Research

In qualitative research main emphasis is on understanding from both respondent and informants, and qualitative research skills and experience plays an important role for the analysis of the data [8]. Qualitative research method is appropriate, when research problem form of unstructured nature [8], and to study the complexity of the problem rather than abstract it away [11]. Furthermore, qualitative research helps to study a topic to provide rich findings better than achieved by using other methods. Qualitative research is a scientific research and it consists of components that to be investigated in such way [10].

- *Seeks answers to a question.*
- *Systematically uses a predefined set of procedures to answer the question.*
- *Collects evidence.*
- *Produces findings that were not determined in advance.*
- *Produces findings that are applicable beyond the immediate boundaries of the study.*

In such case, example from [13], research group extensively interested to investigate the needs of telecommunication users and how new technologies can enhance their lives in some ways. The underlying believe is that research in this area, has been carried out by using technological possibilities rather than the needs of users that could be analysed by using quantitative approach to obtain abstract view from large sample. However, there is focus on context by stating question on how telecommunication technologies are integrated into lives of users. Moreover, any situation, which requires knowledge in depth is well suited to qualitative approach, such as questions that involve "how" and "why" lead to qualitative exploration [13], thus qualitative research was favoured with respect to their research problem.

3.1.2 Quantitative Research

Quantitative research is about "*a formal, objective, systematic process in which numerical data are utilized to obtain information about the world*" [12]. In this type of research technique, researchers adopt quantitative approach to make generalization of population [13], and when they are interested to test hypothesis and quantify them. Moreover, quantitative method enables the researchers to analyse the data in logical and consistent manner by using sampling [8]. In some case, quantitative and qualitative methods can be applied to achieve the results, see example below.

By using mixed approach, where qualitative and quantitative has been applied on study of the use of resources at university of Melbourne [13]. In that particular case, survey has been designed for students who considered under Australian studies. In order to get border view of picture and quantifiable data, quantitative approach was adopted in supporting to identify the types of resources, which students were using in class. On other hand, using qualitative approach, mostly in case studies, interview were conducted with students, teachers and coordinator to have detail understanding on how the use of resources was integrated in precise way.

In our case, the scope of this thesis is to conduct a comparative study of traditional method and agile method. Bearing this mind, we are interested to collect data by using survey that consists of structured and unstructured questions. In such case, unstructured questions seem to be opened questions, or that can be changed, translated and reformatted [8]. However, we are strongly motivated to use qualitative approach on variable or data, which is not easily to quantify, see more detail in section 3.2.2. Furthermore, the underlying principle of using qualitative approach is to get results richer and informative, where professional can put their experience, suggestions and motivations.

With respect to our problem, there are questionnaires and data that need to be quantify in both software development methods. To assist in solving such problems, our survey consists of close ended questions, which are used to quantify the data and analyse them in order to proceed further to achieve research goals. In short, we are interested to use both qualitative and quantitate methods (mixed approach) to collect and analyse the data.

In order to manage research in more precise way, below figure 3.1 depicts the research activities that will be taken into consideration to execute research plan.

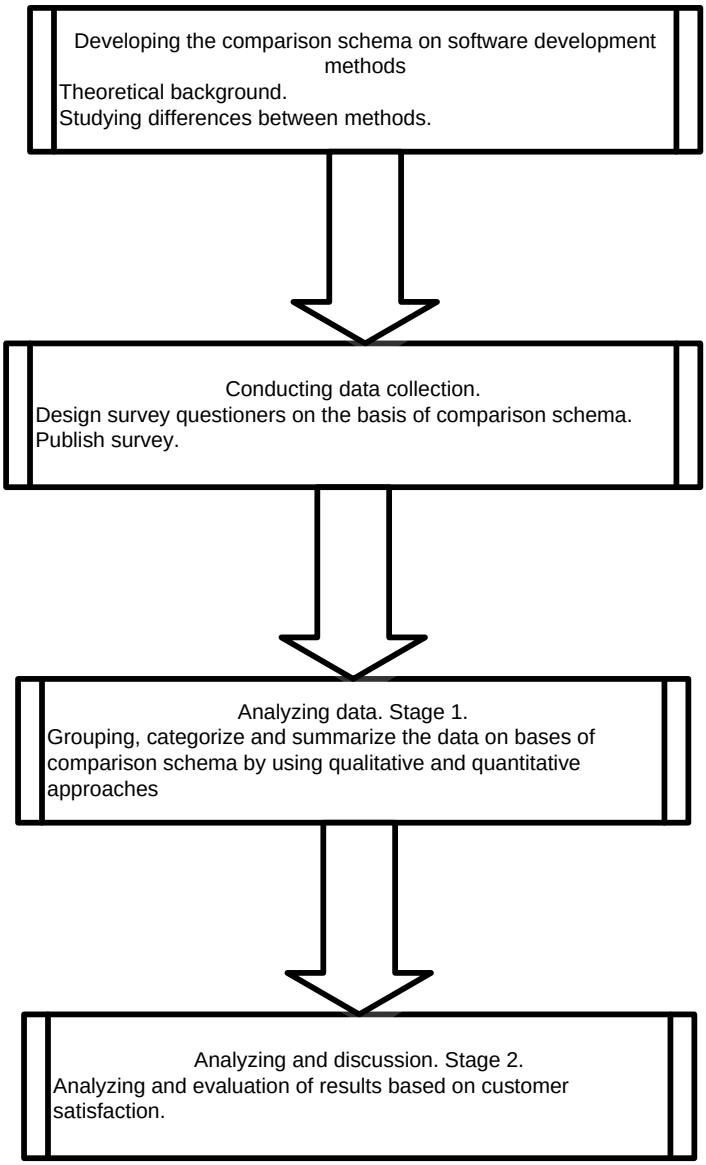


Figure 3-1 Research Activities

3.2 Data Collection Techniques

Software engineering is the involvement of professionals working in a real situation and environment. These professionals develop, maintain and evolve the software. To understand these professionals working, we should study software engineering as it works. To do that, different data collection techniques are available with variety of methodologies and theories. In such case, we go for literature review/theoretical background and survey questionnaires (inquisitive technique), called direct technique for collecting primary data from the participants [11].

3.2.1 Literature Review

The main purpose of this phase is to establish a background which consists of knowledge regarding the thesis to motivate our work. It includes the methods and their implementation, what are the main differences in these methods, and under what circumstances these methods are helpful, and how developers use methods to satisfy the customer's wants/requirements.

3.2.2 Survey Questionnaires

Before the discussion on questionnaires, first we should know what actually survey is? It not just a bundle of questions or check lists to get information.

"*It is a comprehensive research method for collecting information to describe, compare or explain knowledge, attitude and behaviour [pg.58-11]".*

For our study as mentioned above, we go for questionnaires, which can be open ended and close ended. Both types of questions are used in the survey questionnaire. Responses from the questions are measured by ranking with semantic scaling technique as mentioned in the table, the most left represents the good and most right represents the bad [14].

Table 3-1 Semantic grading scale

1	*Extremely agreed*	*Quite agreed*	*Slightly agreed*	*Neither agreed nor disagreed*	*Slightly disagreed*	*Quite disagreed*	*Extremely disagreed*
2	*Extremely satisfied*	*Quite satisfied*	*Slightly satisfied*	*Neither satisfied nor dissatisfied*	*Slightly dissatisfied*	*Quite dissatisfied*	*Extremely dissatisfied*

Another very important issue is to publish and administrate the survey. There are different options available like

- Self-administered questionnaires (using internet)
- Telephone survey
- One-to-one interviews

From the above mentioned list, we go for the first one i.e. self-administered questionnaire by using http://www.kwiksurveys.com free survey distribution website. One of the major benefits for using that platform is that we get data from different geographical areas. To achieve quality work, we must identify, what kind of study on the topic yet have been done and determine how other researchers have collected data on this topic i.e. what kind of questions they used? [11]. we try to make sure that all the questions are answered properly, by structuring the survey questionnaire with easily understandable questions.

3.2.3 Survey questions formulations

It is very important to describe the purpose of every question. So we have divided our survey in four different sections that contains sixteen questions. First section presents general view and gathers basic information from the respondent, and results from these questions will help to categorize participants and their organizations with respect to size, application and the methods which they follow, and rest of three sections reflect our research questions one by one.

In section one; we have four questions, which are very helpful in analysing the answers with respect to the data received on these questions. Such as,

Question 1: which software development method "SDM", do you use?

In this question, we have separated/categorized the participants with respect to the methods which they use or if they don't follow any method. There are three options like *agile software development methods*, *traditional software development methods* and *no method"*.

Question 2: Which option describes your position in your organization?

In this question we have list of eight different designations/positions which can be viewed in the survey questionnaire available in (appendix), which are most commonly used in the software development organizations. However survey participants can be categorised with regard to their current position in their organization.

Question 3: What type of applications, does your organization work on?

This question helps us to categorize the organizations with respect to projects and application on which they are working, which can be viewed in the survey questionnaire available in (appendix). We can analyse the organization with respect of their method, application and what level of satisfaction they have.

Question 4: How many software development professionals are working in your organization?

Purpose of this question was to see the size of organization with respect to the number of professionals working in that organization. We used EU standard for the SME, i.e. *less than 10, 10 to 49, 50 to 250* and *above 250*.

In section 2, we have six questions and main emphasis was to analyse that how software development methods help the software development professional to satisfy their customers and how it helps to fulfil the client's/customers' requirements and demands.

Question 5: How satisfied you are with your software development method?

In this question, we tried to find out an overall response from the developer/professionals that what level of satisfaction they have with their software development methods. To categorize the level of satisfaction, we used seven levels of scaling (semantic scaling) [14].

Question 6: To what extent, do you agree that your software development method helps you to have understandable communication with your customers?

In this question we tried to investigate the professional's point of view on method/methods helps them to have a better and understandable communication with their client/customer. To better understand the level of agreement we used seven levels of scaling (semantic scaling), see table 3.1 row one.

Question 7: To what extent, do you agree that your software development method help you to fulfill your customer's requirements / demands?

Purpose of this question is to investigate that whether the professional are agreed on this point that their method helps them to fulfill the customer's wants and needs (requirements). After receiving the data from this question we can analyze better that which method is more help full and if the method helps in better communication.

Question 8: Do you follow any software quality standards?

This question is used to categorize the developers and organization that whether they are using the quality standards or not e.g. CMMI and ISO 9001 etc. Here we used two options *Yes* and *No.*

Question 9: To what extent, do you agree that your software development method helps you to develop high quality software?

This question is used to gather opinion from developers that whether their software development method is helpful for them to produce high quality software. To better understand the level of agreement we used seven levels of scaling (semantic scaling), see table 3.1 row one.

Question 10: To what extent, do you agree that your software development method helps you in risk analysis?

Risk analysis is considered very important in project management and software development. Focus of this question get the opinion from developers that how they feel or find their method in risk analysis. Level of agreement is same as mentioned in question 9.

In section 3, our focus of study is to gather data from the professional with respect to the challenges on the adoptability of software development method. So for we put questions in that would be easy to categorize the data in different direction.

Question 11: Which factor is most important for you as a professional when adopting a method?

Purpose of this question is to gather information from the professionals, who represent different level of organizations, which factor is most important for them to adopt a SDM. Such as *low cost, easy to handle, great productivity, great reliability* or if they can't explain they can say that they *do not know*.

Question 12: Which method do you find more suitable with respect to project size? (Small / medium / large)

This question is very important and helpful to get an opinion from developers/professionals using SDM, which software development method is suitable in a specific size of project i.e. *small / medium / large*. Different methods are used in this question which can be viewed in (appendix).

Question 13: Which option is best suitable for you, when adopting new methodologies?

Here in this question, our main purpose is to see the strategy of software development organizations for method adoptability. Such as *Market leader by adopting the new methods, Follows the market leaders, only adopt proven methods* or *do not like to adopt new methods*.

Question 14: Briefly specify your opinion about the different challenges during the adoptability of a software development method.

This is an open ended question, where our purpose is to get a brief answers on different challenges during the adoptability of software development methods from the professionals/developers using SDM according to their experiences.

Question 15: To what extent, you agree that your software development method support you in rapid development.

Purpose of this question is to analyze that how software development method helps professionals in rapid development. We used seven levels of scaling (semantic scaling) [14].

Question 16: To understand the agility in software development project, which one is most important to focus on?

In this question, we tried to find the preferences of developers/professionals, by putting choices that whether the characteristics of traditional methods help in rapid development or agile manifesto characteristics. In such case, we used seven scaling techniques to have better understandability on preferences. See the question and its result in (appendix).

3.3 Finding Population

Finding the population is a very important phase of our thesis study, we use different ways for searching the professionals and software organizations with good reputation such as using the personal relations with the professionals working in different organizations at different geographical areas, second we access local software houses in Jönköping city, third way is to go for search of professionals on http://www.linkedIn.com web, where we can find a lot of professionals who can be our population for the thesis primary data gathering. Furthermore, the last one is some PhD students in Jönköping University and also from some other universities working with in the same area of study. If we have some participants who are student so they must be enrolled in PhD or Master's degree. We would like to make it sure that we distribute the survey questionnaire to the professionals using agile methods and professionals using traditional methods.

3.4 Evaluation Methods

Those methods which are used to validate the results are evaluation methods. Evaluation is done on the basis of the results received from the different professionals from different geographical areas, which helps us to compare these methods. Because in survey we used both type of questions i.e. open ended questions and close ended questions. Response from respondent for each question is calculated for the identification of most important part in close ended questions.

From open ended questions, comments/responses from the participants are thoroughly studied and categories for responses were developed. After detailed study of all the data, methods adoptability challenges were identified and summary of results represent the participants opinion. At the end external validity, internal validity and reliability will be used for the analysis of the results.

4 Results and Analysis

Purpose of this chapter is to represent results in general form, and analyse the survey results with respect to our research focus. First of all we discuss empirical findings in section 4.1 results. Afterward we analyse these results by comparing with each other.

4.1 Results

To cover domain of our interest, survey was conducted on comparison study of agile and traditional software development methods. An important aspect of this survey questions was to collect the data that could be interpreted for analysis and conclusion with respect to our research focus. Therefore, we were interested to target software companies and IT professionals by emails and phone. In order to get valuable response, we requested to professionals who have been involved in method adoptability, and have experience on software development processes. Moreover, our mainly objective was to target Swedish IT companies.

In our survey results, population sampling size is not enough to perform analysis statistically, and ratio of respondents in both software development methods is not much differing to each other. In such case, we are likely to analyse and conclude empirical findings on basis of respondent frequency, and need to investigate reasons behind empirical findings. However, it would be hard to make strong conclusion on small sampling but we are concerned to generalize, grouping and categorize responses in order to contribute our research findings.

We have received 37 responses from different type of software development professionals. Only one professional responded that he does not use any software development method while all other 36 developers are using software development methods. Therefore, after review the results, it was evident that 56% population using agile software process while 40% population using tradition software development. Furthermore, refer to appendix for detail population numbers every survey question. (See appendix)

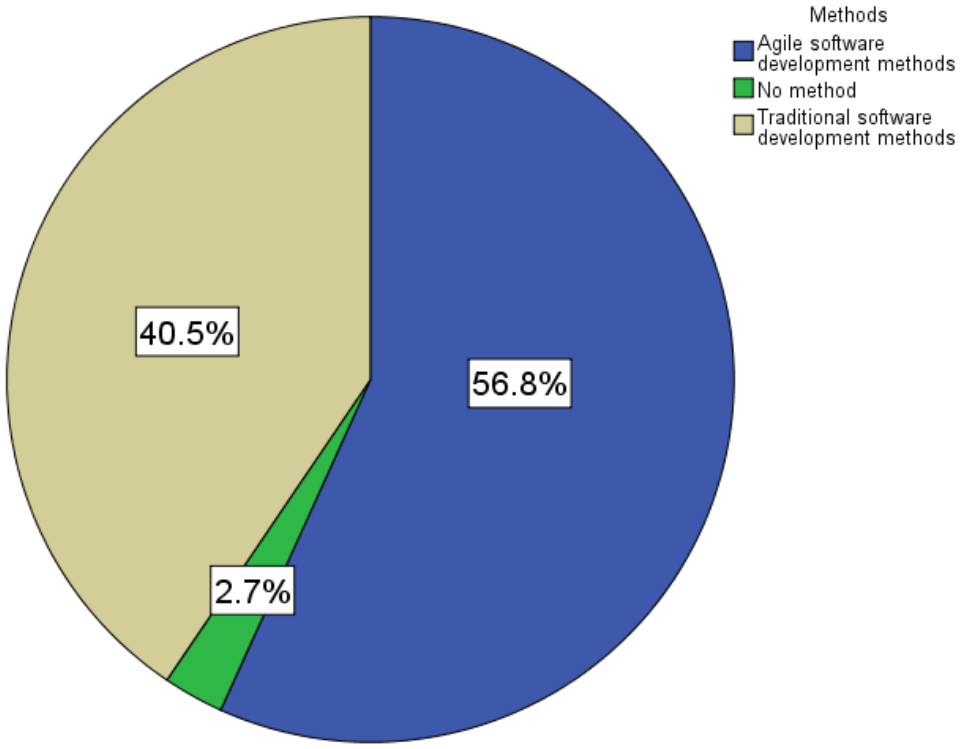


Figure 4-1Graph representation percentage of professionals with respect to their methods

An additionally, figure below depicts the participants of agile and traditional methods with respect to different size of organizations.

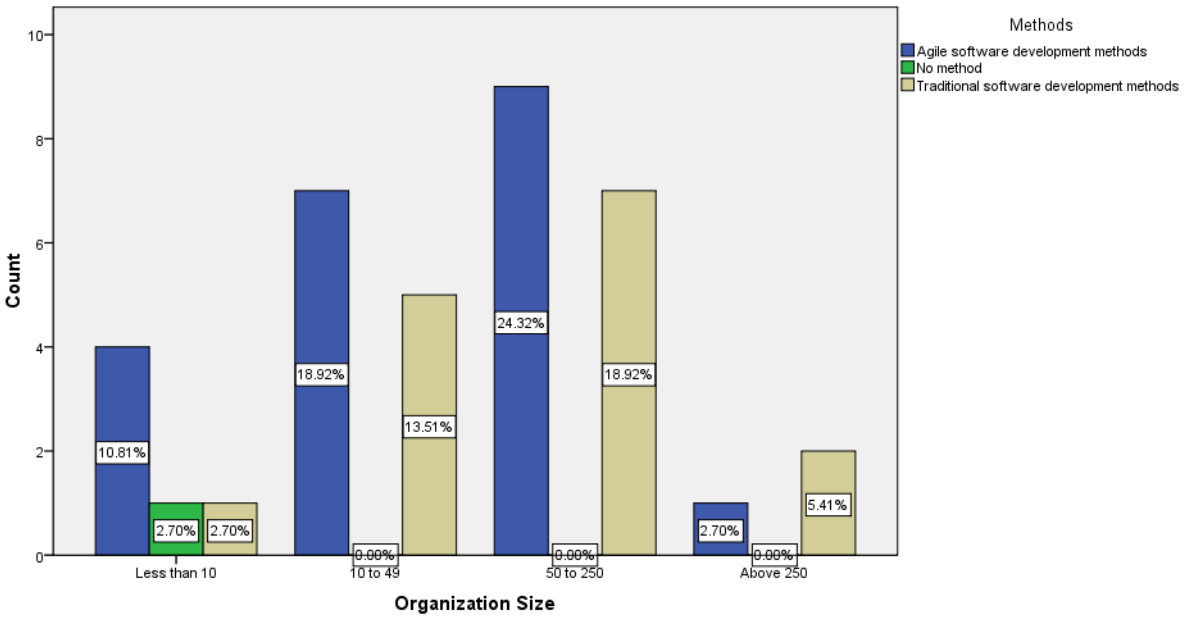


Figure 4-2 Graph representation number of professionals in different size of organization

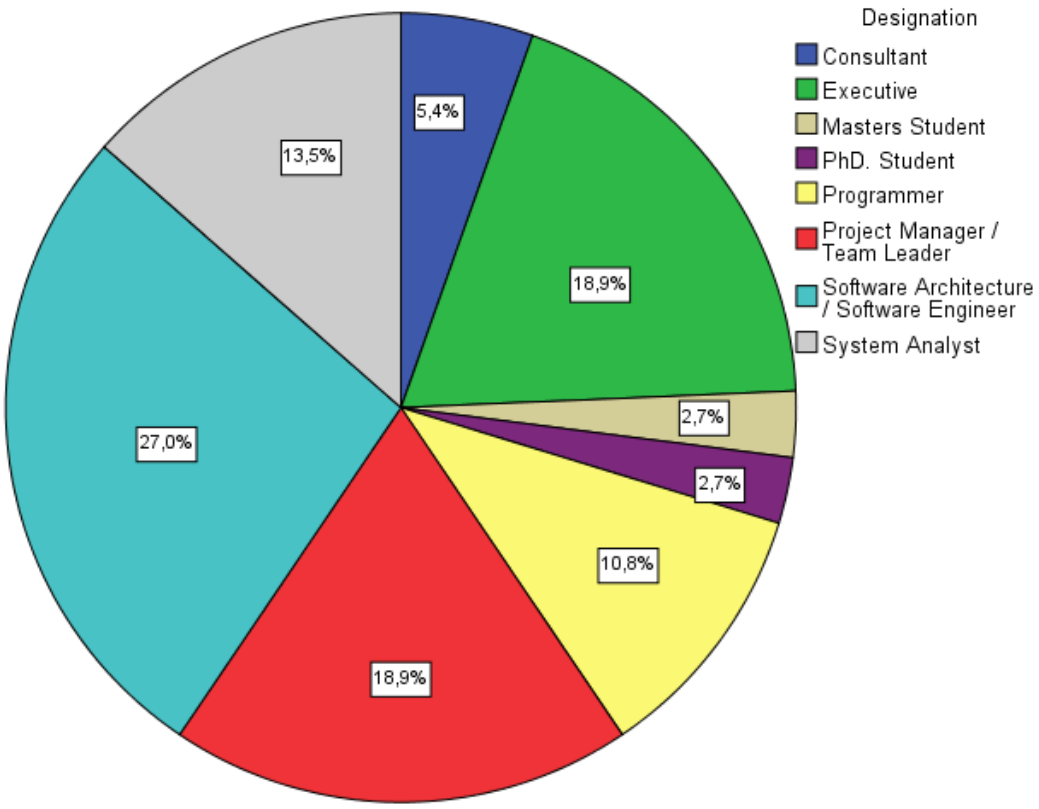


Figure 4-3 Graph representation number of professionals with their designations

4.2 Survey Analysis

Analysis section consists of three parts which describes analysis for our research questions one by one, based on comparison of survey results on different angels.

4.2.1 Suitable method for professionals

Professional/developers satisfaction with their methods/methodologies is very important. We analyse with respect to different directions, such as what sort of organization developer/professional is working, size of organisation, which method being used and what position the respondent have, to meet the better quality software, better communication with customers, fulfil the customer's requirements and needs, analysing the risk and importance of methods with production of quality software. We have categorized the analysis in different sections which are as mentioned as follows.

4.2.1.1 Overall Satisfaction with methods

As we categorized the professionals with respect to their methods, when we compared different survey results with each other, we find that approximately 80% of professionals with traditional software development methods are satisfied, where as 13% are dissatisfied, and approximately 72% of professionals with agile software development method are satisfied, whereas 19% of professionals are dissatisfied with agile methods, whereas approximately 9% answered neutral.

We analysed that approximately 13% of professionals using traditional software development methods belongs to large scale of organization, approximately 47% belongs to medium level of organizations, approximately 33% belongs to small and approximately 7% of professionals belongs to micro level organizations. Whereas approximately 5% of professionals using agile software development methods belong to large scale of organization, approximately 43% belongs to medium level of organizations, approximately 33% belongs to small and approximately 7% of professionals belong to micro level organizations.

With the help of above mentioned analysis we assume that micro level organizations prefer to use agile methods but large scale organizations prefer to have traditional methods.

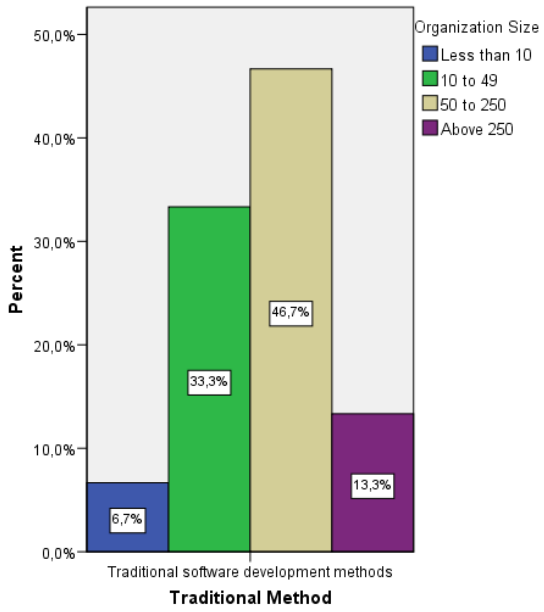


Figure 4-4 Represents professionals from traditional practices with respect to organization size

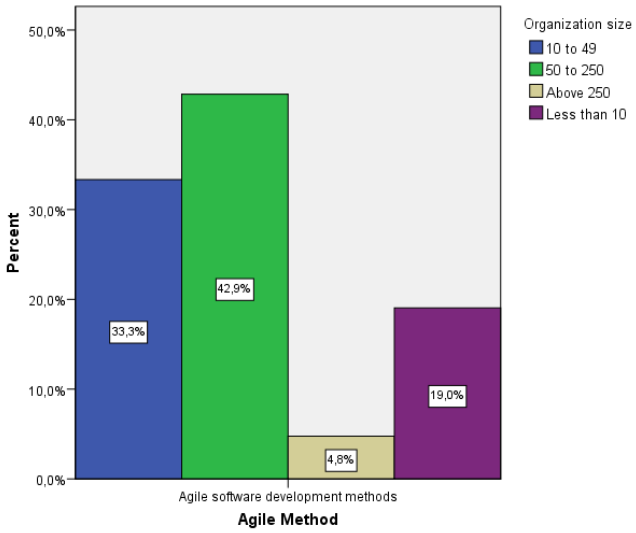


Figure 4-5 Represents professionals from agile practices with respect to organization size

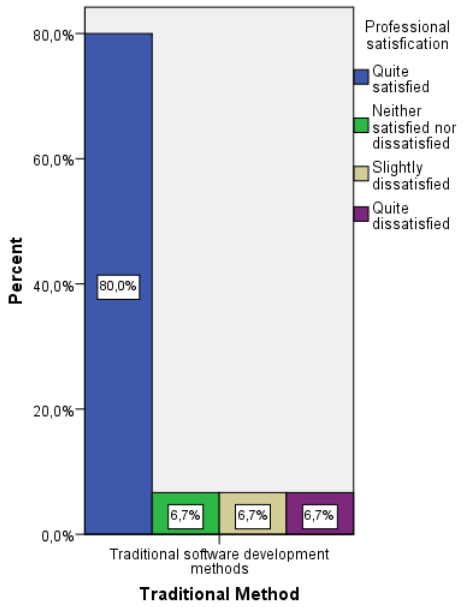


Figure 4-6 Represents ratio of satisfaction for professionals from traditional practices

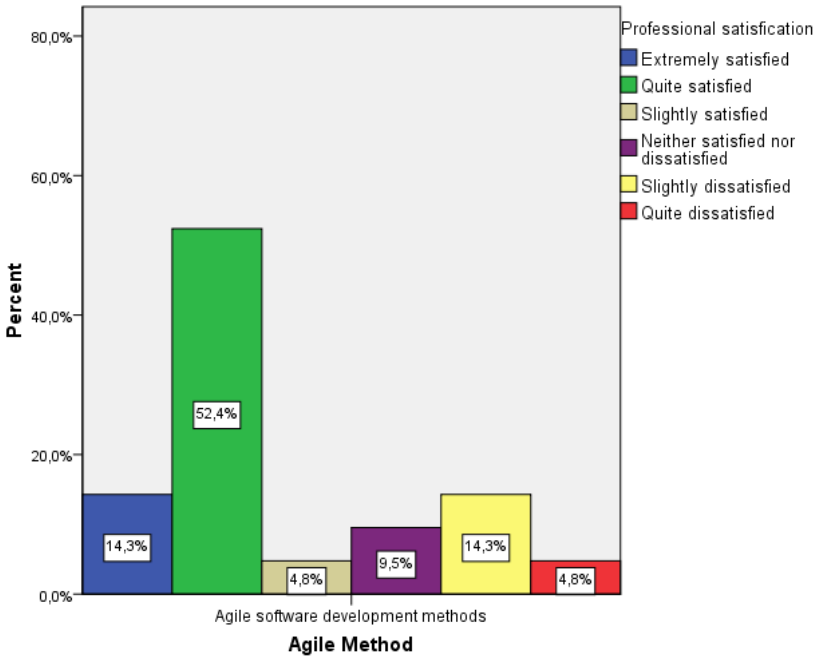


Figure 4-7 Represents ratio of satisfaction for professionals from agile practices

Comparison shows that from professionals using traditional methods, approx. 27% are software architecture/software engineer from which 20% are quite satisfied, approx. 27% are executives from which approx. 13% are quite satisfied, approx. 20% are system analysts and all system analysts using traditional methods are quite satisfied. Whereas from professionals using agile methods, approx. 29% are software architecture/software engineer from which approx. 19% are quite satisfied, approx. 24% are managers/team leaders from which approx. 19% are satisfied (quite & slightly), approx. 14% are executives and programmers are approx. 14%.

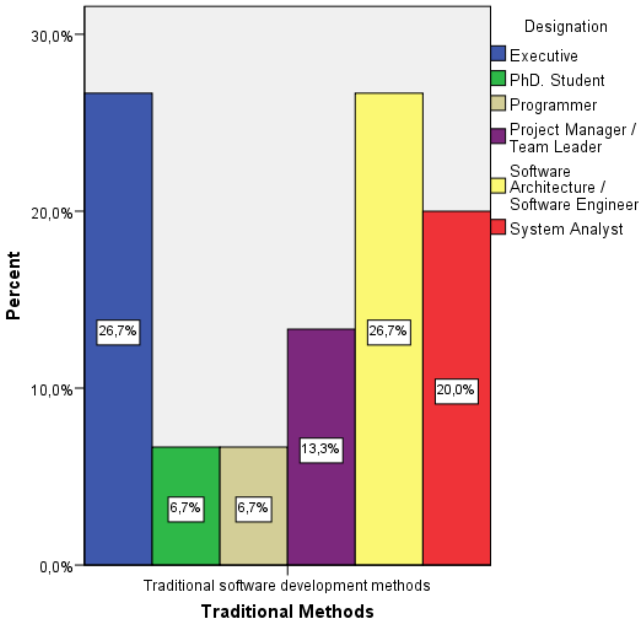


Figure 4-8 Represents professionals from traditional practices with respect to their position in organization

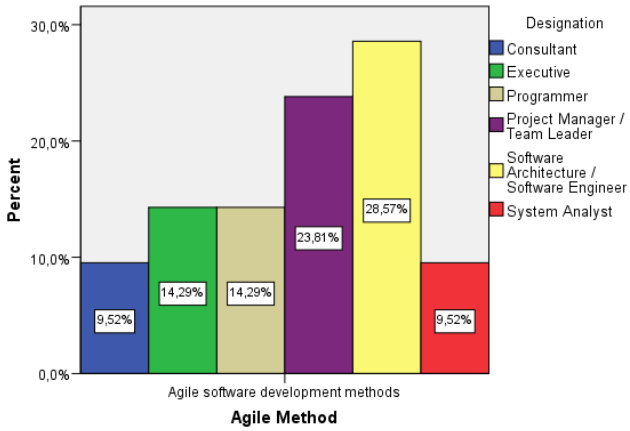


Figure 4-9 Represents professionals from agile practices with respect to their position in organization

We analysed that approx. 43% of professionals with agile methods are developing web based applications from which approx. 29% satisfied (quite & extremely) and approx. 19% of agile professionals are developing desktop applications from which approx. 15% are satisfied (slightly & quite).Whereas approx. 20% of professionals with traditional methods are developing web based applications and all of them are quite satisfied, 20% of professionals from traditional methods develops desktop applications and all these 20% professionals are quite satisfied and 20% of professionals from traditional methods develops distributed applications and all 20% are quite satisfied.

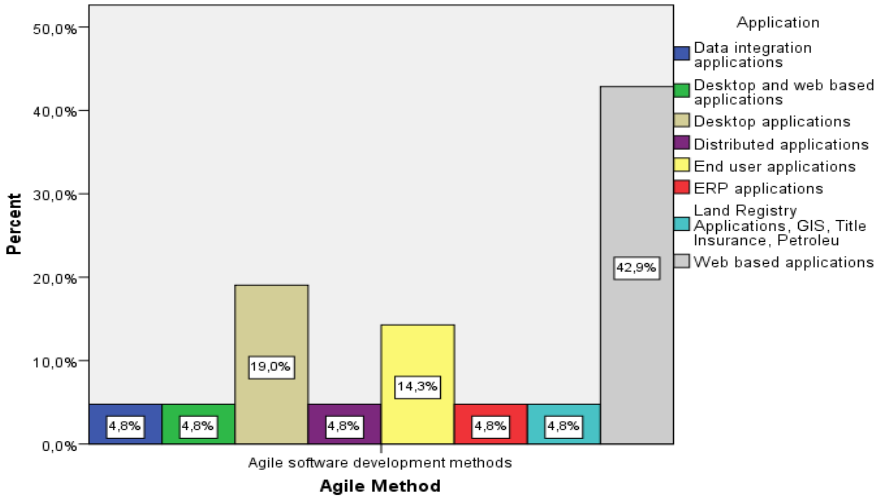


Figure 4-10 Agile software development method professionals with respect to software application

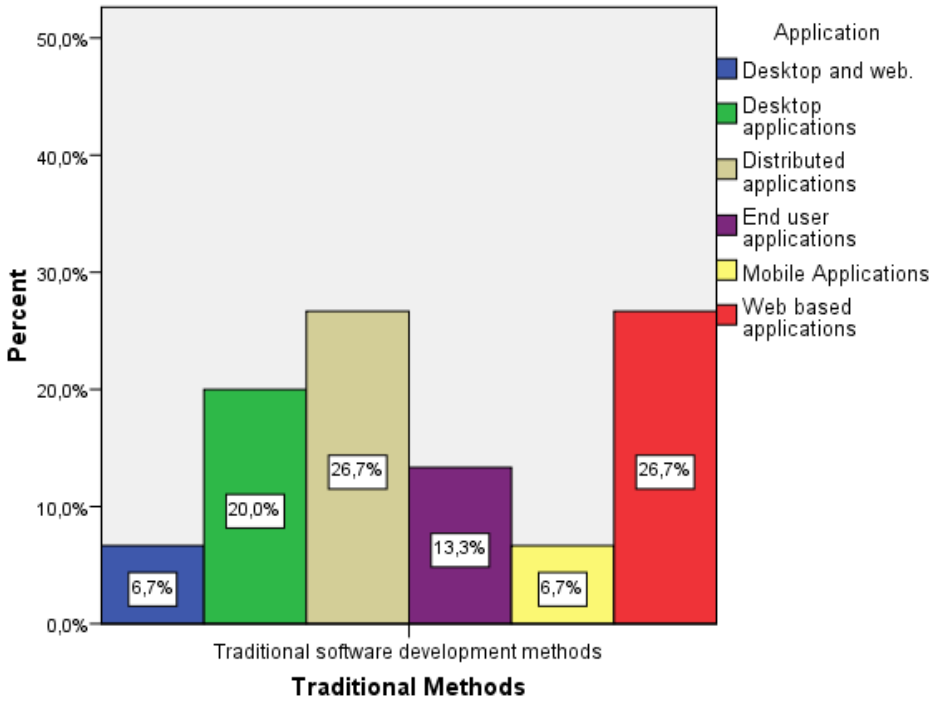


Figure 4-11 Traditional software development method professionals with respect to software applications

4.2.1.2 Following software quality standards

We analysed that 40% of professionals with traditional software development methods follow software quality standards and 60% do not follow software quality standards. Whereas approx. 38% of professionals with agile methods follow software quality standards and approx. 62% do not software quality standards.

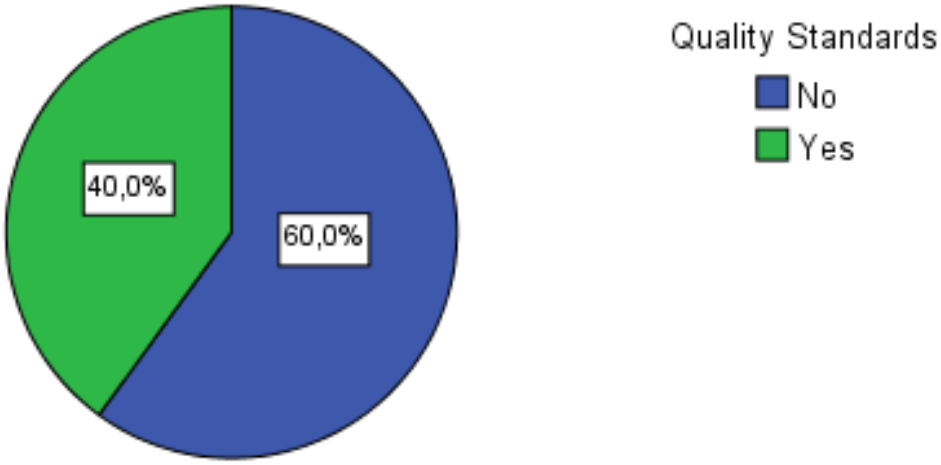


Figure 4-12 Ratio of traditional professional with respect to following the software quality standards

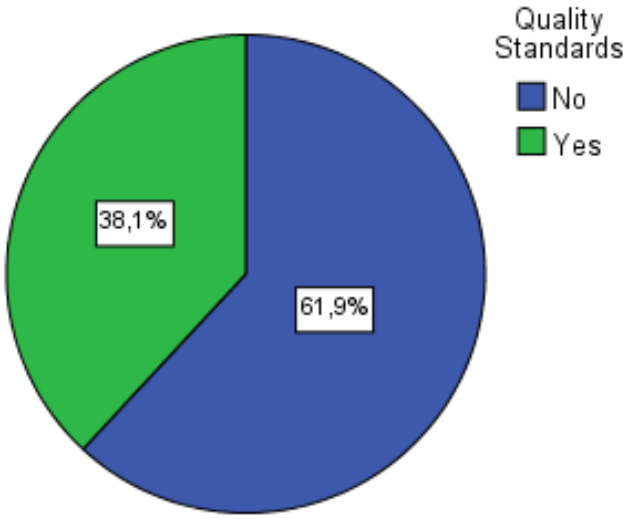


Figure 4-13 Ratio of agile professional with respect to following the software quality standards

4.2.1.3 Importance of methods in risk analysis

Importance After comparing the results, we find that approx. 81% of traditional software development methods user agreed that their method helps them in risk analysis, and approx. 19% of traditional methods users answered in neutral. Whereas approx. 62% agile software development methods users are agreed that their method helps them in risk analysis, and 19% of agile users are disagreed. This also justify the old study mentioned in frame of reference, traditional software development methods section.

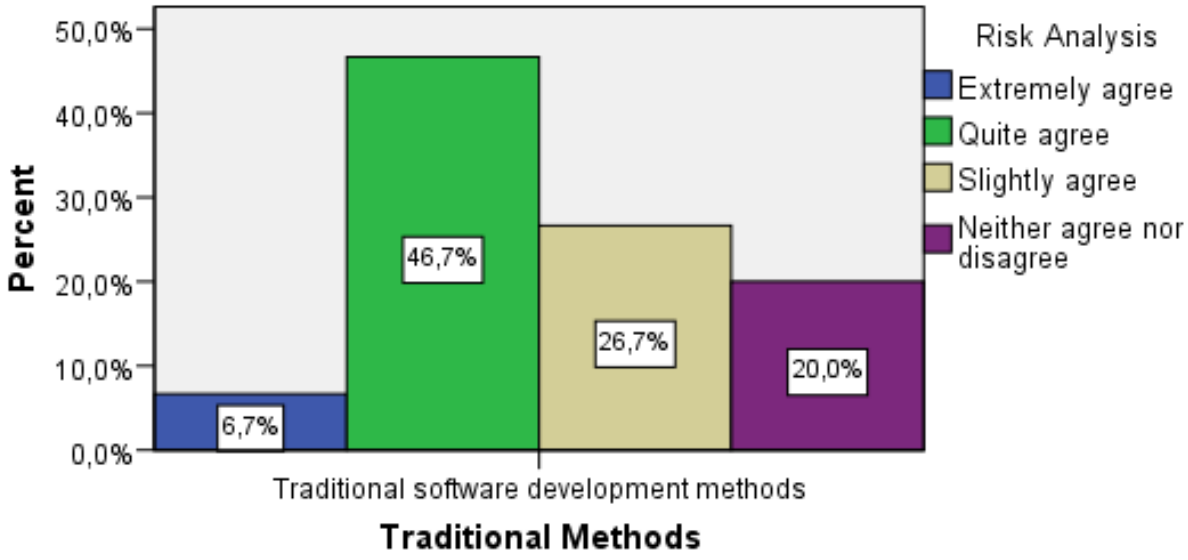


Figure 4-14 Represents ratio of the traditional method users with respect to methods ability to analyze the risk

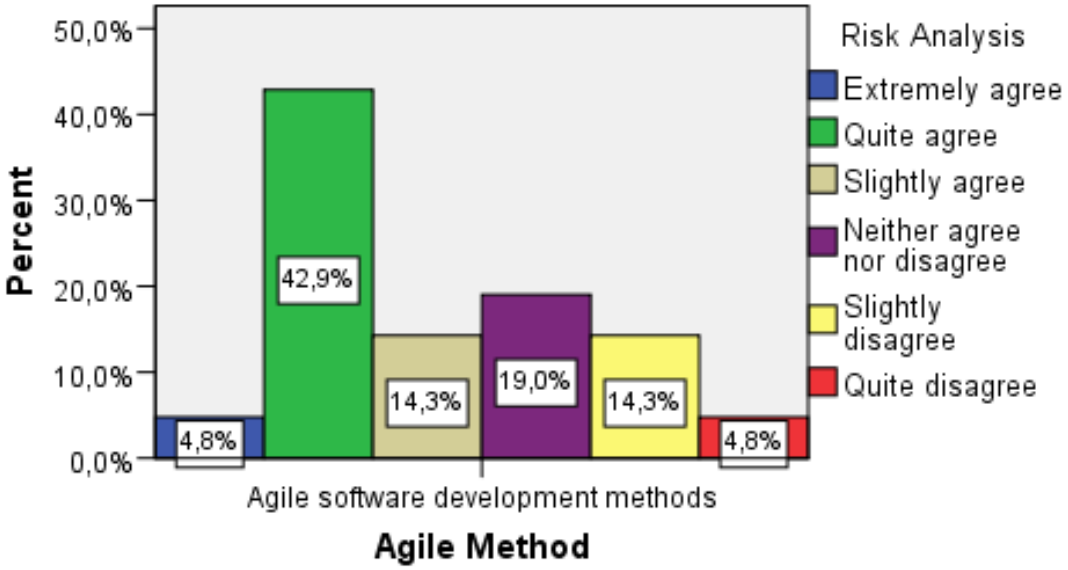


Figure 4-15 Represents ratio of the agile method users with respect to methods ability to analyze the risk

4.2.1.4 Importance of methods in better communication with customer

Analysis shows that approx. 86% of professionals with traditional software development methods are agreed that their practices with traditional methods helps them to have better communication with their customers and approx. 14% of professionals are disagreed. While 91% of participants of agile methods are agreed that their practices with agile methods helps them to have better communication with their customers and approx. 9% of professionals are disagreed. Moreover, figure below depicts respondent ratio on how they are agreed with practices of their method to help them to have a better communication with customers.

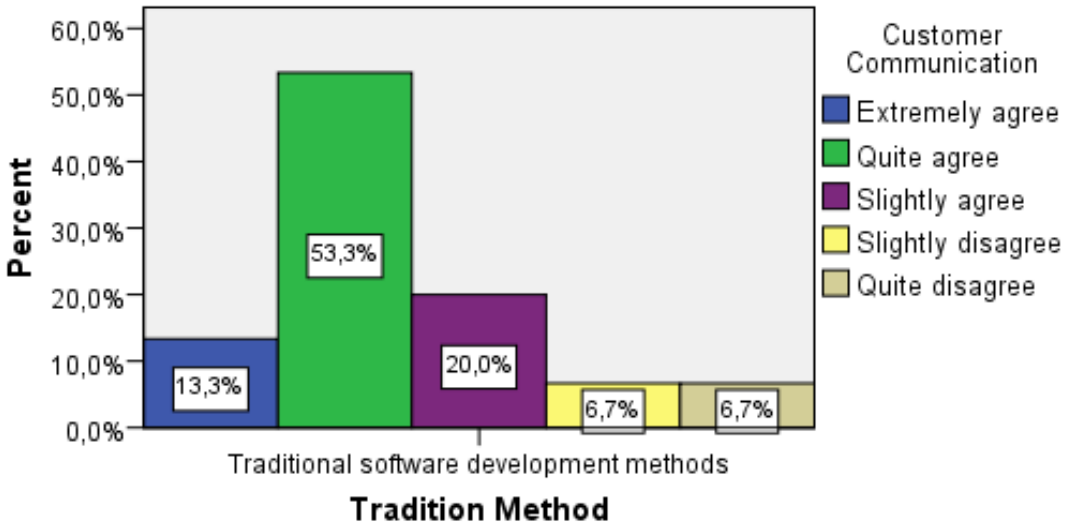


Figure 4-16 Represents the ratio of traditional methods professionals satisfaction with respect to ability of method to have a good communication with customers

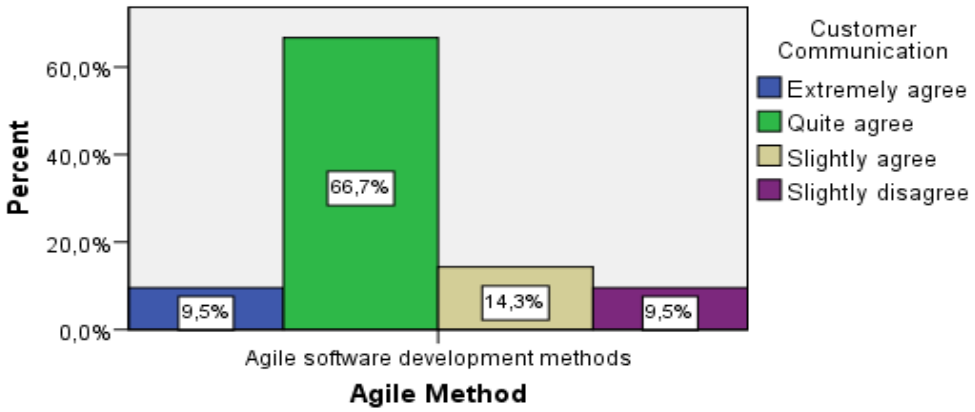


Figure 4-17 Represents the ratio of agile methods professionals satisfaction with respect to ability of method to have a good communication with customers.

Results of survey shows, professionals from agile methodology (23.8% Project Managers/Team leads and 19% Software Engineers) recommended agile software development methods to have better communication with customers. On other hand respondent from traditional approach, (26.6%) software engineers and (20%) system analysts are likely to work with tradition software development method that would have better approach during requirements engineering in order to make feasibility report (see section). In short, we assume that agile practices are considered better in term to have understandable communication with customer. See figures below describe the participant of agile and traditional methodology.

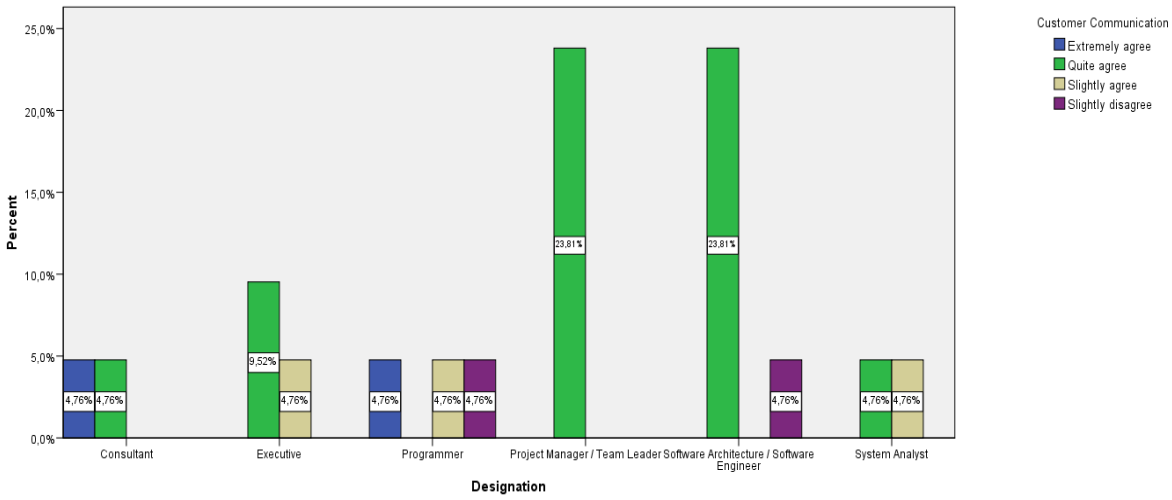


Figure 4-18 Agile graph representation percentage of different designation

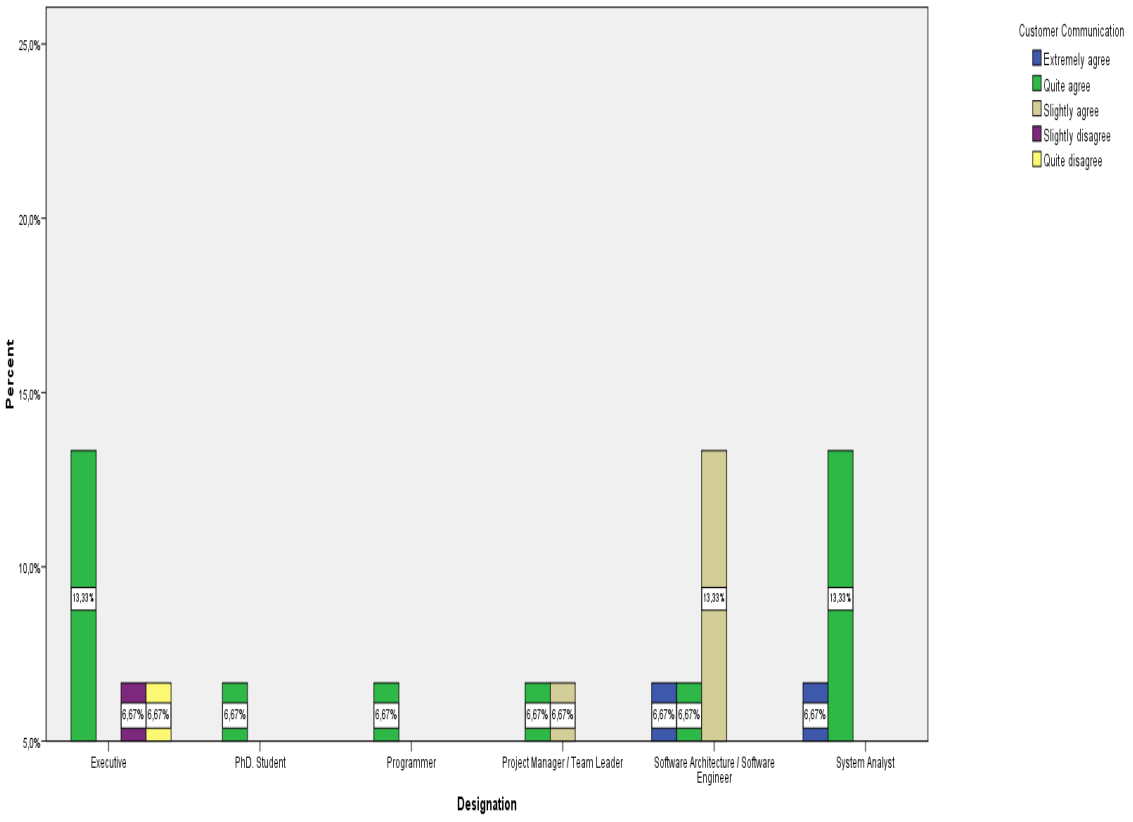


Figure 4-19 Traditional graph representation percentage of different designations

In such case, another observation can be made on organization size where participants of micro organization (19.1%), they are quite agreed to use agile practices. Evaluation of survey shows that productivity of agile processes in term of communication with customers decreased when they have large team size. In this sense, traditional software methods are more precise in large organization. See figure below.

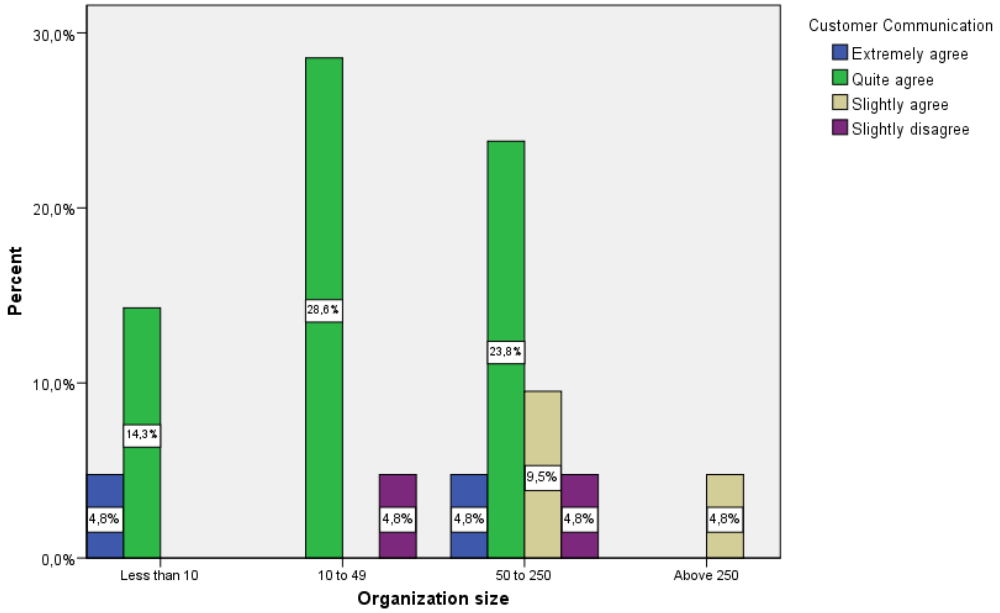


Figure 4-20 Agile graph representation percentage of organization size

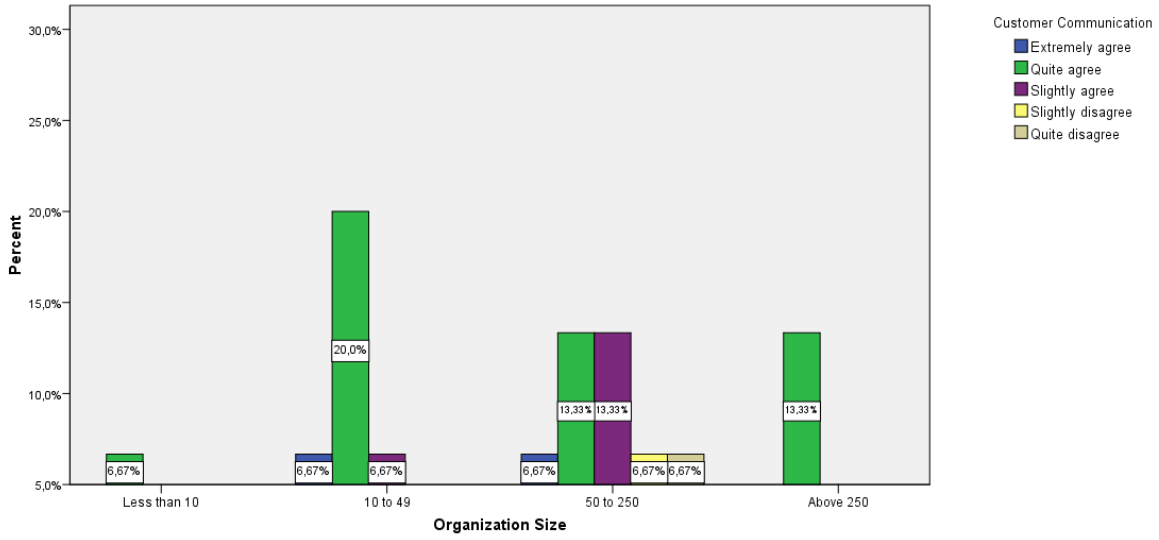


Figure 4-21 Traditional graph representation percentage of organization size

4.2.1.5 Importance of methods to fulfil customer demands/requirements

We find that approx. (95%) professionals using agile software development methods are satisfied with their practices in order to fulfil customer demands/Requirements and approx. (5%) of professionals from agile methods are dissatisfied, but on other side approx. (94%) professionals using tradition software development methods are satisfied with their methodology in order to fulfil customer demands/Requirements and approx. (6%) of professionals are dissatisfied. See figure below to depict percentage of respondents.

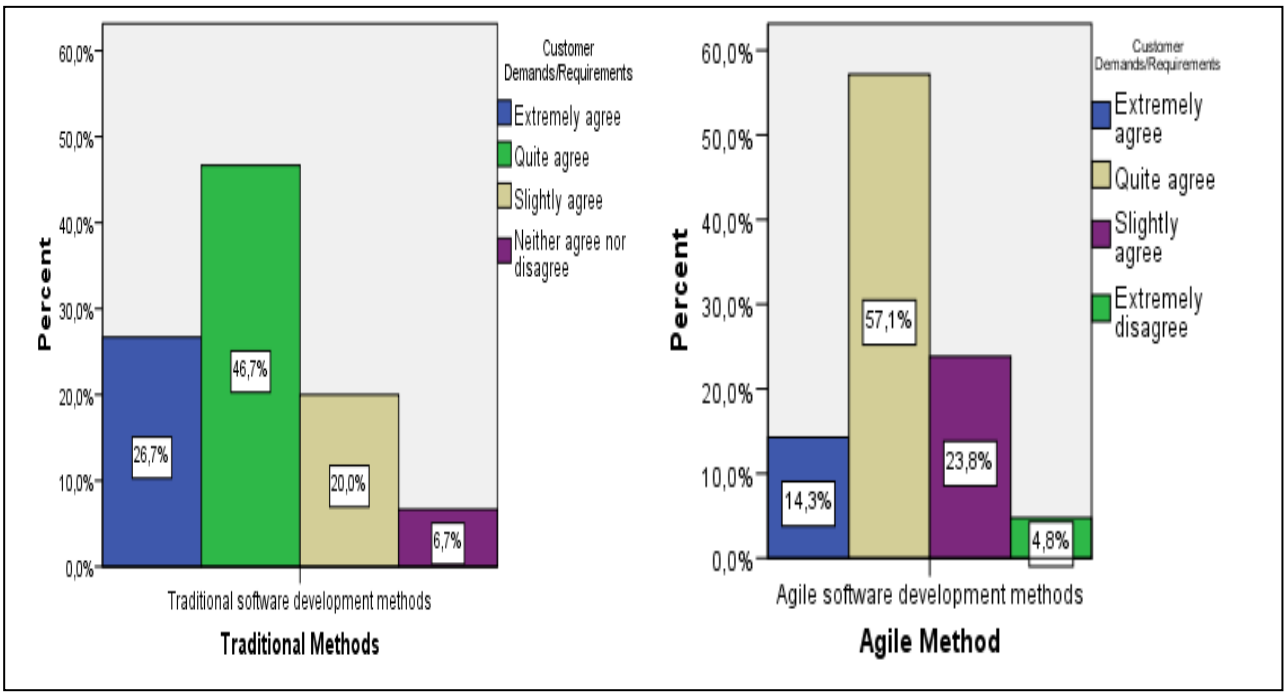


Figure 4-22 represents the ratio of traditional and agile methods professionals with respect to customer demand and requirements

Agile software development methods are highly adopted to fulfil the customer demands and requirements (section). From agile participant, as our survey results show, Software Engineer/Software architecture (28.57%) are agree on agile methods in order to fulfil customer demands/Requirements. On the other hand software engineers/software architecture (26.67%) and System analysts (20%) are satisfied with traditional methods. See figure below describes percentage of agile and traditional participant.

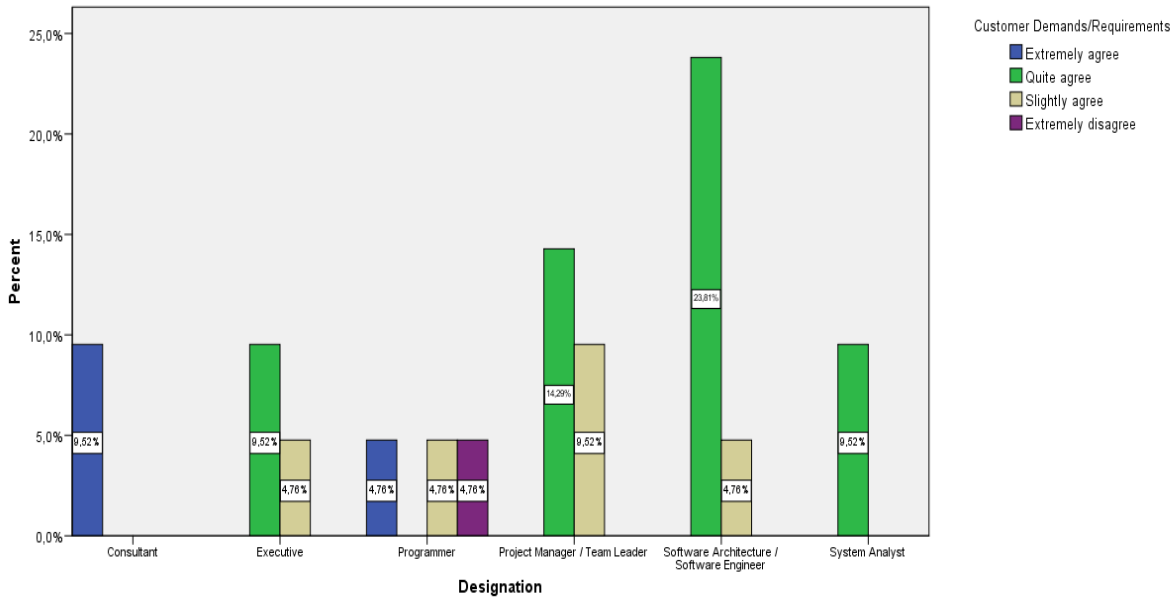


Figure 4-23 Represents satisfaction of agile method users on customer demand and requirements with respect to their designations

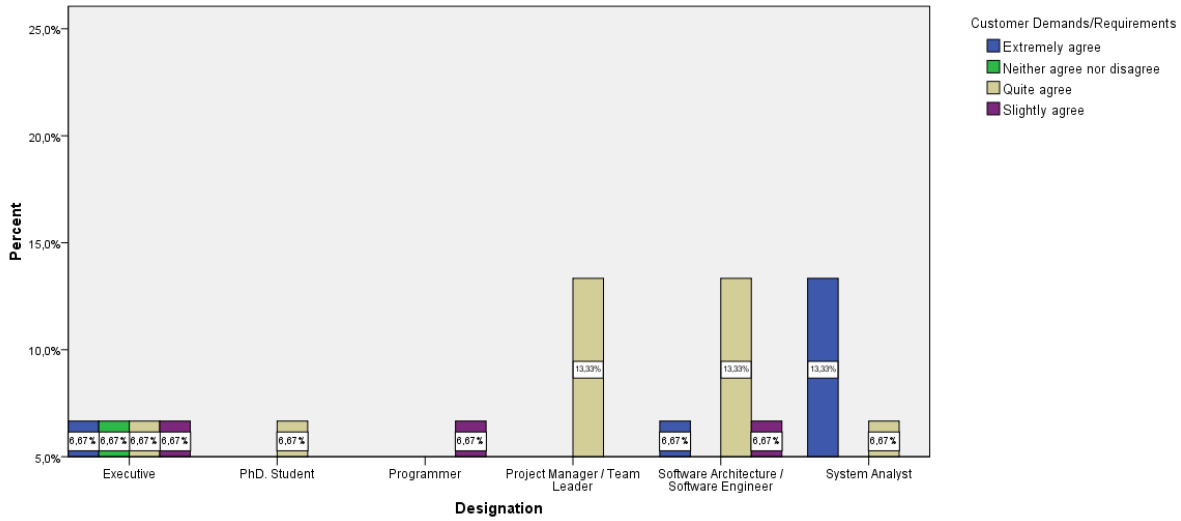


Figure 4-24 Represents satisfaction of traditional method users on customer demand and requirements with respect to their position

In this particular, we analysed that in small organization, it's quite easy to fulfil customer demands using agile and traditional but in medium sized organization where we assumed that agile is suitable approach. See figure below depicts the agile and traditional participant, survey result reflects that traditional method approaches are considered for large organization where risk level can be identified on customer demands and requirements before in response to change system. We concluded that participants are more satisfied with functionalities of agile methods in response to customer demands and requirements.

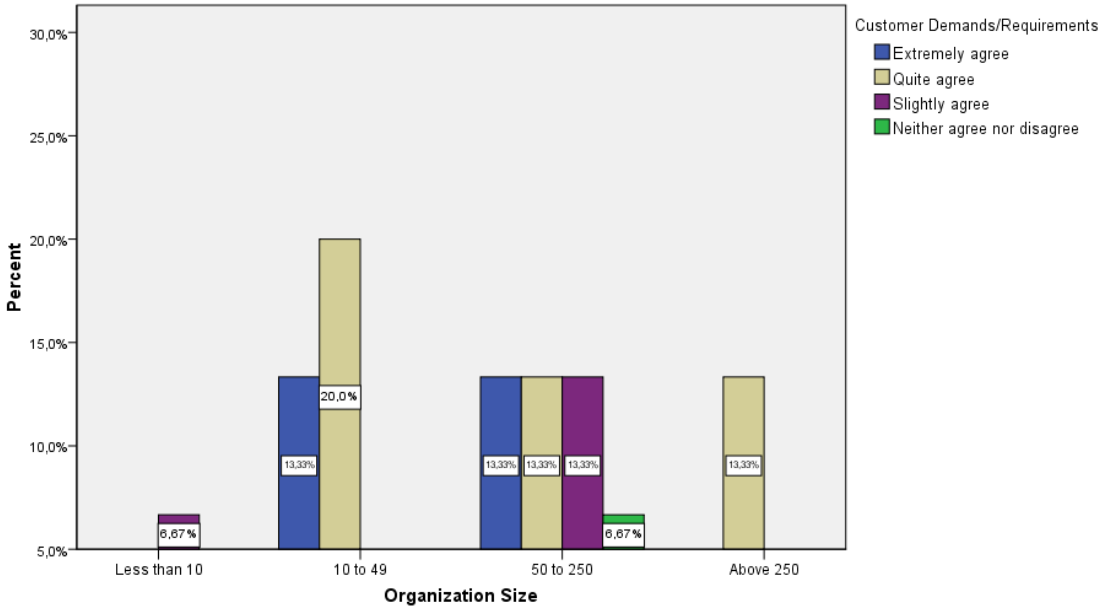


Figure 4-25 Represents ratio of professionals from traditional practices with respect to importance of method to produce quality software

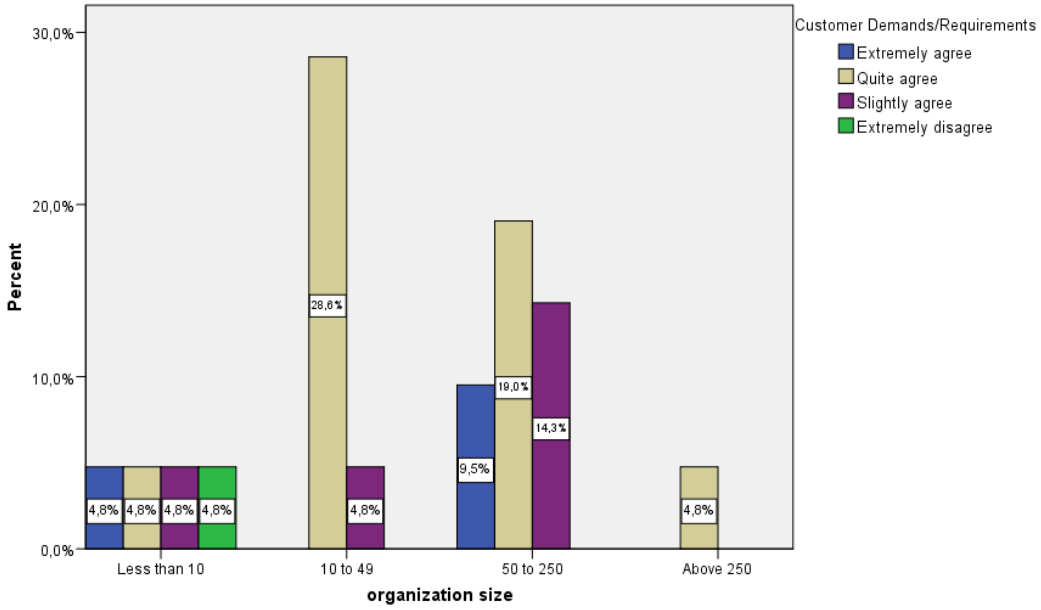


Figure 4-26 Represents ratio of professionals from agile practices with respect to importance of method to produce quality software

4.2.1.6 Importance of methods to produce quality software

A software development method plays an important role, In order to produce quality software. Comparison shows that professionals using traditional software development methods are agreed (extremely agreed 20%, slightly agreed 13.33%, quite agreed 66.77%) with their practices in order to produce a quality software. On the other side approx. 81% professionals using agile software development methods are agreed (extremely agreed 19%, slightly agreed 23.8%, quite agreed 38.1%) with their practices in order to produce a quality software, 4.8% professionals of agile software development methods are disagreed with their practices, and 14.3% kept them neutral.

With the help that analysis we can assume that professionals with traditional software development methods are more satisfied with their method as compared to professionals using agile software development methods to produce quality software. Moreover, figures below depicts respondent ratio on how they are agreed with practices of their method to help them to produce quality software.

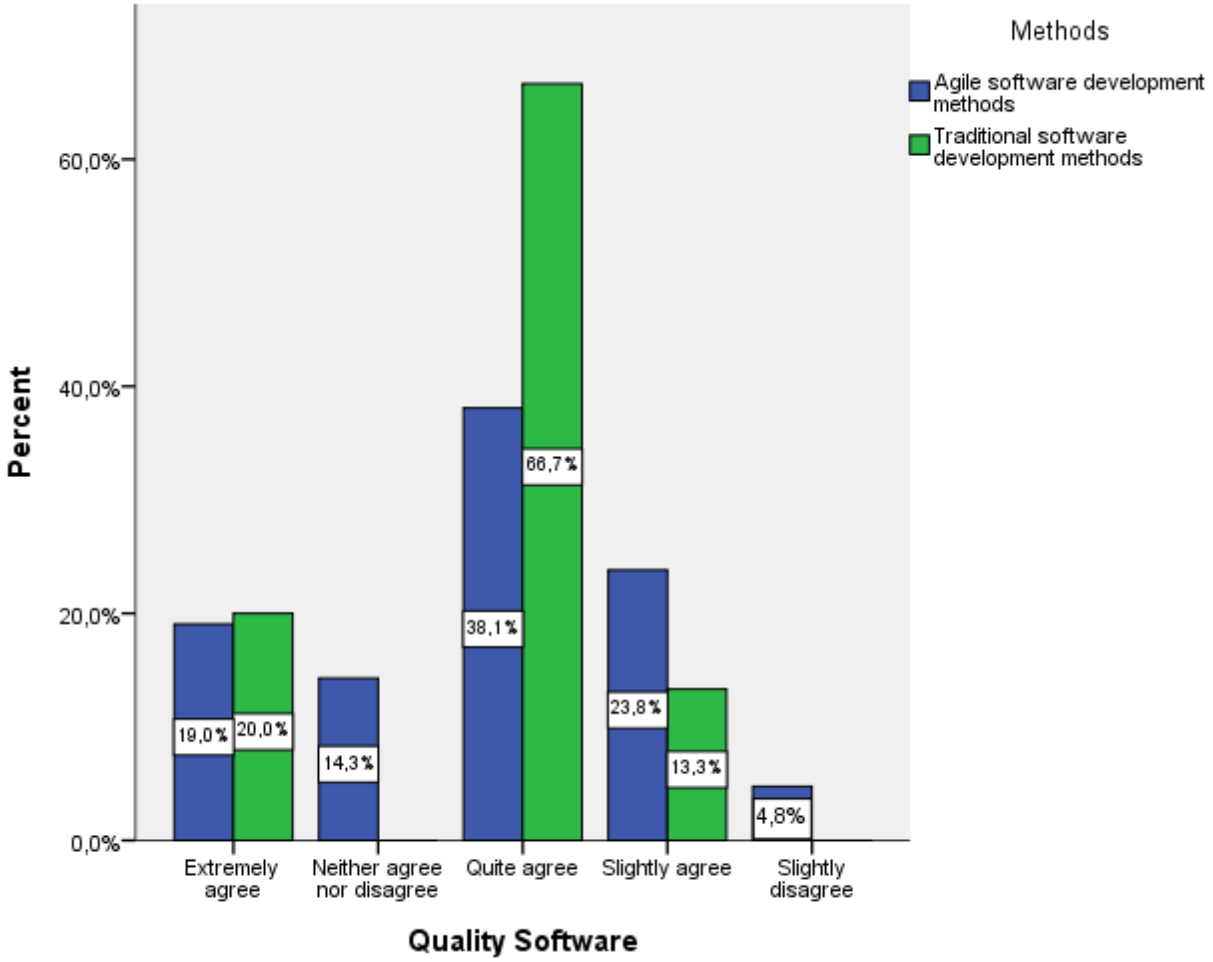


Figure 4-27 Represents ratio of professionals from traditional and agile practices with respect to importance of method to produce quality software

4.2.1 Challenges on method adoptability

Another objective of this research was to elaborate the challenges/limitations while adopting software development methods. Agile and traditional users answered in response to open question that can be generalized in order to include the challenges what they face during adoptability of method. Moreover, this section specifies software development method in area of agile and tradition with respect to project size. However, organization strategies are discussed on how they follow market standards to have better choice of methodology.

4.2.1.1 Adoptability Factors

As survey results depict the picture on adoptability factors that agile participants (52.4%) are agreed on agile methodology, which has great productivity while traditional participants are 20%. With great reliability perspective, participants from tradition (33.3%) are satisfied than agile approach. Additionally, 20% traditional community recommended that traditional methodology is more suitable when they have low cost factor. Indeed, as survey result reflects, we can conclude that tradition methods have low cost with great reliability, but on another hand to adopt an agile methodology is cost effective with higher productivity of processes. See figure below.

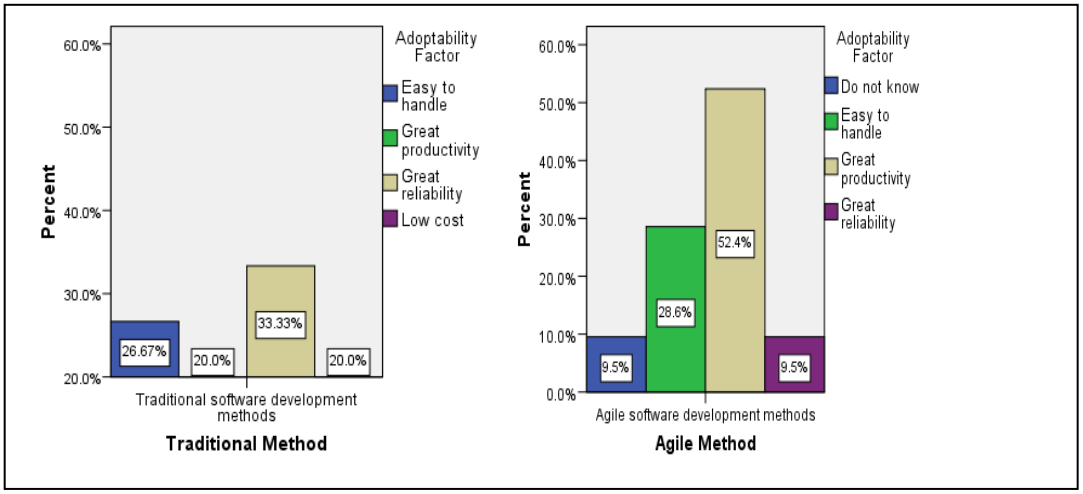


Figure 4-28 Represents ratio of traditional and agile participant on adoptability factor

With productivity point of view, another experiment can be made on different size of organization where participant's ratio indicates that productivity of methodology is most important in small (10.81%) and medium organization (21.62%) weather other participants (5.41%) suggested that reliability factor most important for large organization. In such case, some respondents (8.11%) are likely to recommend that methodology could be easily handled in micro size organization. From survey results, we assumed that great productivity is most important factor for small and medium size organization while adopting a methodology, in addition to, large organization would prefer great reliability of software development method. See figure below.

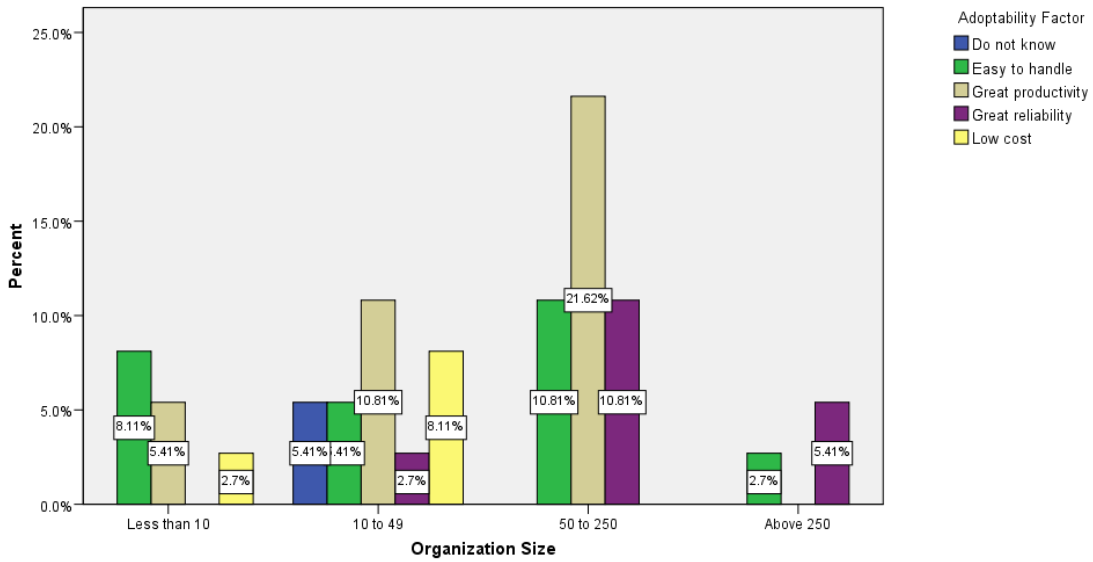


Figure 4-29 Represents percentage of participants with respect to organization size

4.2.1.2 Suggestions from respondents on challenges of method adoptability

Here, we identified some major challenges that IT professionals face during methods adoptability. In this particular, participants contributed their suggestions from three different areas, such as Project Management, Software Development, and Client Satisfaction. Moreover, we intended to generalize the challenges in these areas, and discussed impact on software development, when organizations adopt software development methodology.

As far as Project Management concern, IT professionals specify time and cost, which are critical factors that increases the budget of the project. While adopting methodology, change in management that leads to restructure teams and organization according to their project size and situation. Therefore, project manager needs to be more persistent and follow up closely in order to make sure all developers must have understanding of new methodology processes. In this case, communication gap between team members might have impact on software development practices. In addition, mangers from traditional approach, they feel challenges in role, accountability, leadership with new methodology. Therefore, in some cases, organization does not want to change or precisely open his polices in response to strictly follow new software development methodology practices.

In supporting to end client, there are some issues to be considered when client preferences involved as they feel better with specific method. Bearing this in mind, some participant argued that some clients do not like scrum when clients do not have much skills and knowledge of scrum method. In this case, there is need to held workshops in order to teach methodology practices and how to proceed with defined processes.

To be able to shift to another methodology, IT professionals claim with concluding remarks that we are in between of shifting methodology. In this scenario, methodology experts debate on agile scrum that says nothing about code, and only time box (sprint) with fixed resources to manage the people. In such case, when organizations adopted new methodology and seems interested to replace with software development process like, introducing scrum and stop working with software development process improvement that indicates quality goes down. Furthermore, in case of agile adoptability, agile is treated in same way as traditional that caused to create frustration and internal friction, at the end top executives says “agile is not working” lack of understand ability of method.

In order to encapsulate above discussion, we strongly motivated, key challenges are cost and time for adopting a new methodology, how much time develops take to adjust with new methodology that have major on change management. In addition to, these factors caused to increase overall budget of project.

To depict the picture on proven methodologies, mostly participants (54.1%) likely to work with proven methods, somehow (21%) follow market leading approach. In case of micro organization, 5% professionals are perfectly fine with their software development processes, and they do not want to adopt new methodology. However, participants from small (24%) and medium organization (24%) recommended using proven methods. See figure below. From survey results, we examined that proven methods are more suitable approach rather than applying new methodology or mixed practices without analysing impact of new methodology on organization and project.

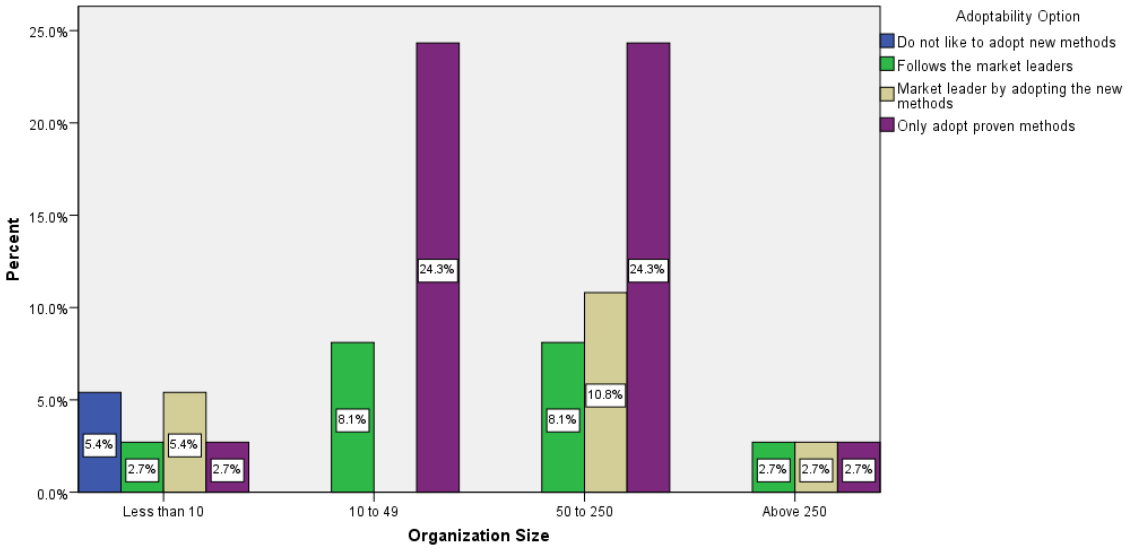


Figure 4-30 Represents percentage of participants on adoptability options with respect to organization size

4.2.1.3 Adoptability with respect to project size

As stated before, project size depends on budget, scope, duration of project and team size. Here, we tended to analyse specific software development method from agile and tradition approach with respect to project size. See figure below.

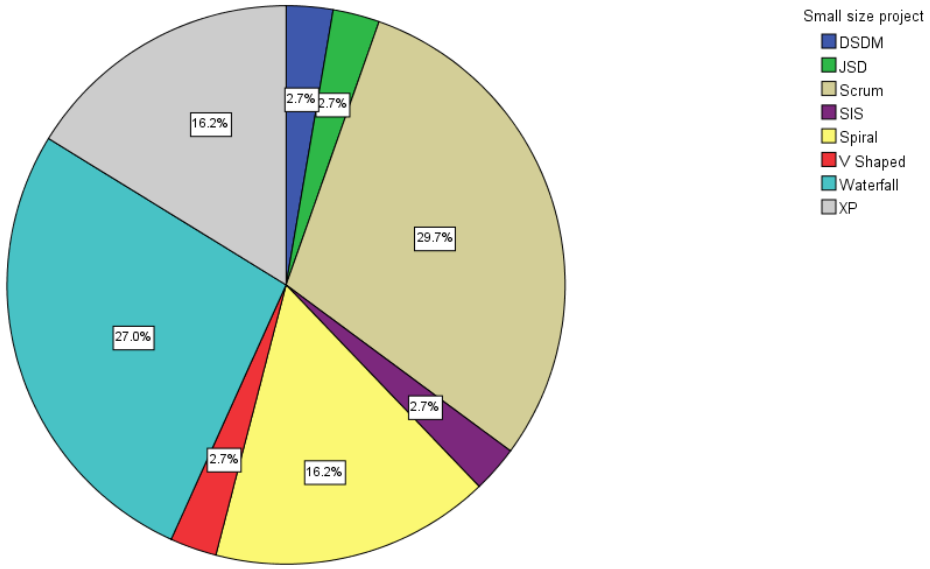


Figure 4-31 Represents percentage of selection of software development method for small size project

As above figure shown, mostly professionals (29%) are agreed on Scrum method that highly adtoped for small size projects weather Waterfall could be better approach. Sprial and XP respondents have same level. When we talk about medium project, Scrum still desirable method of professionals while Sprial method being used as well. To assist in working with large projects, Waterfall and Spiral are most suitable methods. See figure below.

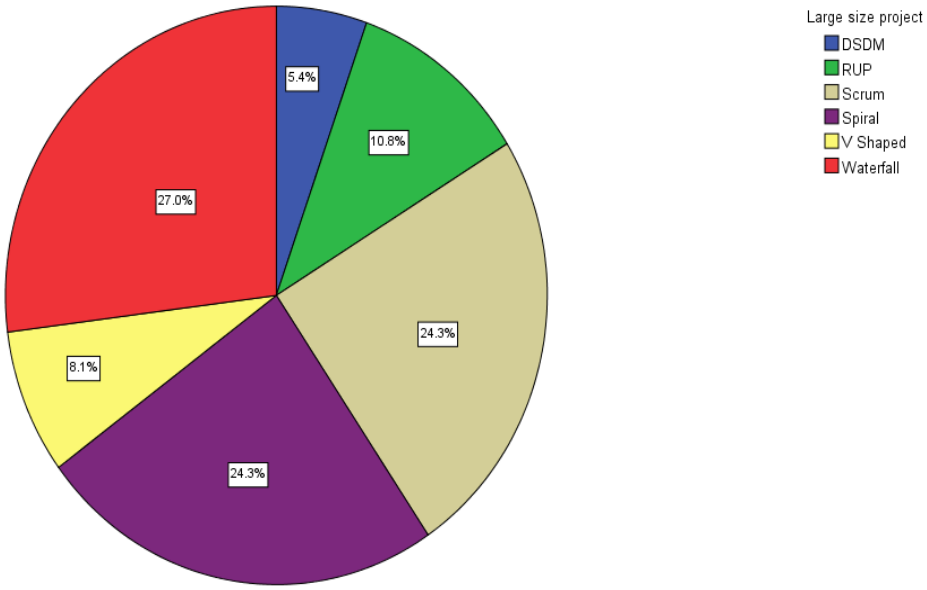


Figure 4-32 Represents percentage of selection of software development method for large size project

More important for the scope of this thesis however, we summerized that IT professionals likely to adopt methodology, which has great productivity and having proved well defined software development processes. With large team and large projects, traditional (Waterfall, Sprial) is considered in order to achive great realability, and agile (Scrum) method is appropriate for small and medium project. Therefore, key challenges are time, cost and lack of knowledge of methodology that influences to create unknown issues in team managment, and have major impact on budget of project.

4.2.2 Method attitude towards agility

This section comprised of analysis on methods support with rapid development and agile values. After comparison we found that participants from both type of methods i.e. “traditional & agile” are agreed that there methods help them in rapid development. Such as approx.87% professionals from traditional practices are agreed and approx. 91% professionals from agile practices are agreed and importantly there is no disagreement from any professional.

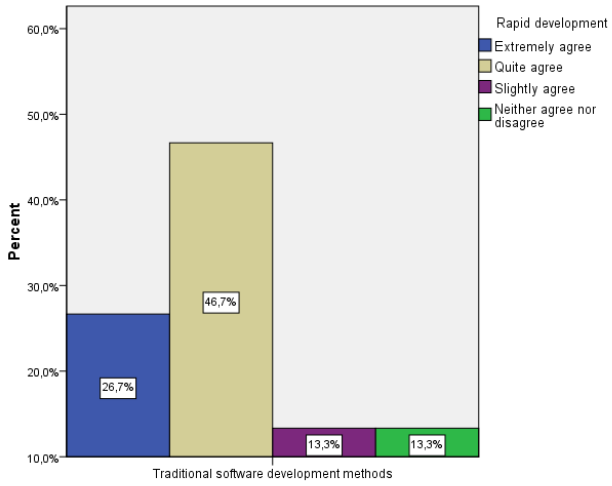


Figure 4-33 Traditional methods users with respect to rapid development

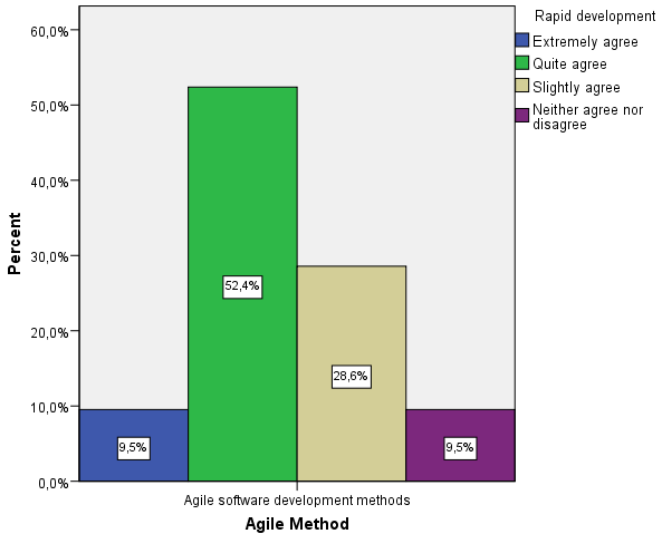


Figure 4-34 Agile methods users with respect to rapid development

We also analysed that professionals from traditional methods with web based application 26.67% are agreed, distributed applications approx. 27% are agreed, desktop applications approx. 20% are agreed that their methods help them in rapid development. On the other hand professionals form agile methods with web based applications approx. 38% are agreed, desktop applications approx. 19%, End user application approx. 13% are agreed that their methods help them in rapid development. For details see the graphs mentioned below.

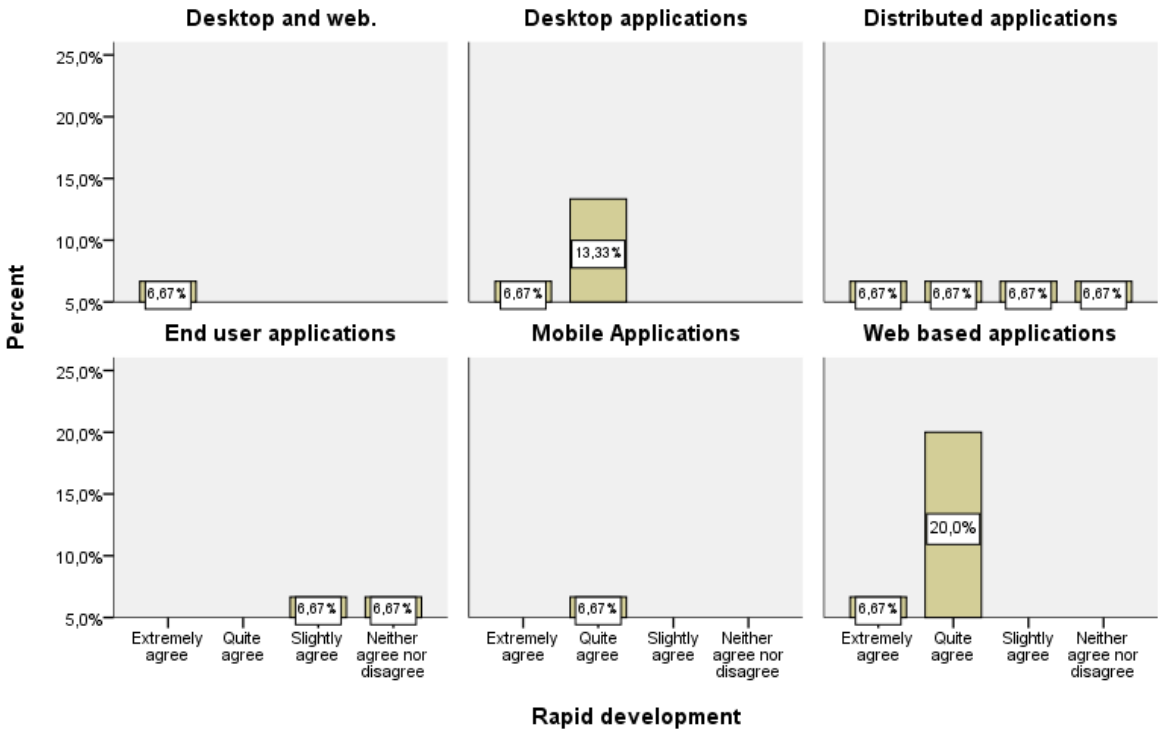


Figure 4-35 Traditional methods professionals with respect to rapid development and application development

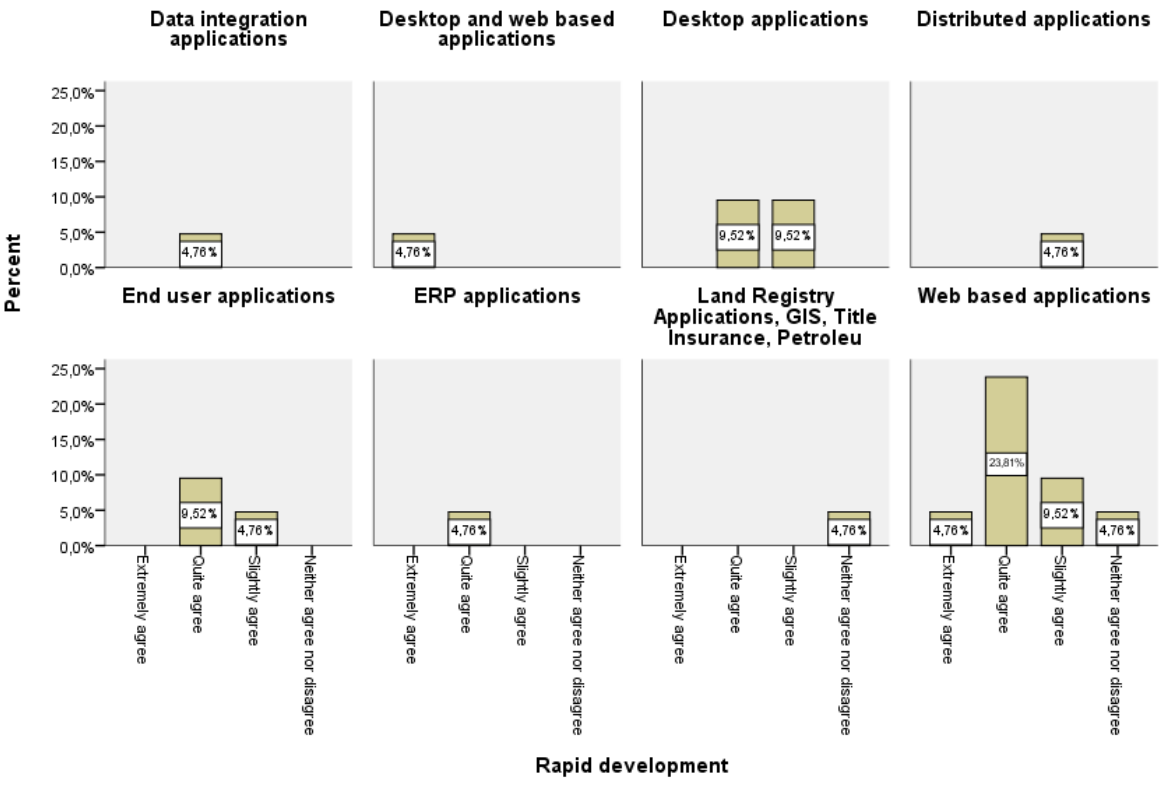


Figure 4-36 Agile methods professionals with respect to rapid development and application development

We also analysed that professionals using traditional methods approx. 7% from micro organizations, approx. 33% from small size organization, approx. 33% from medium size organization, and 13% from large size organization are agree with their method help in rapid development. Whereas professionals with agile methods approx. 19% from micro organizations, approx. 28% from small organizations, approx. 39% from medium size organization and approx. 5% from large size organization are agree with their method help in rapid development.

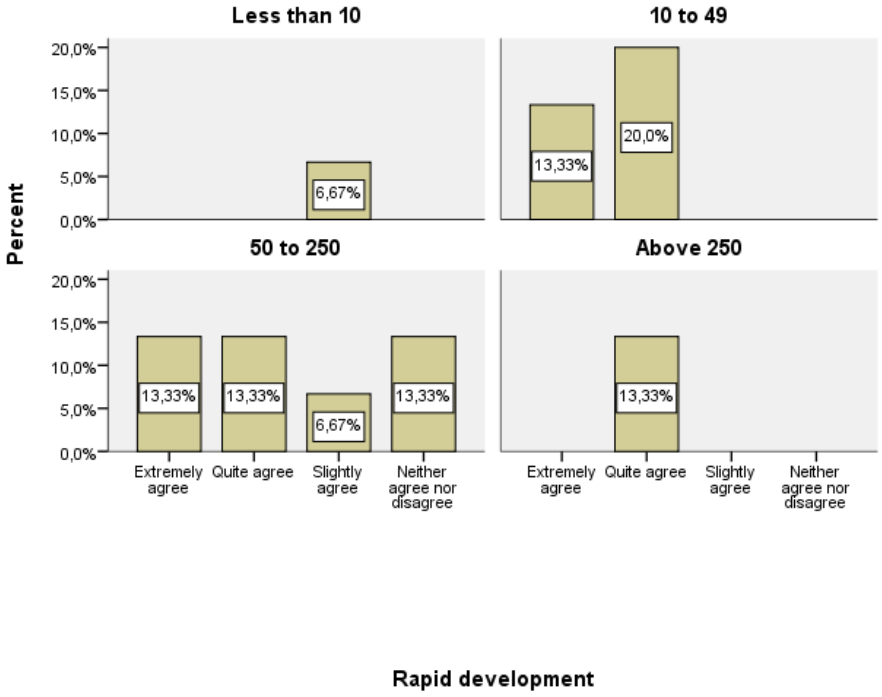


Figure 4-37 Traditional methods and rapid development

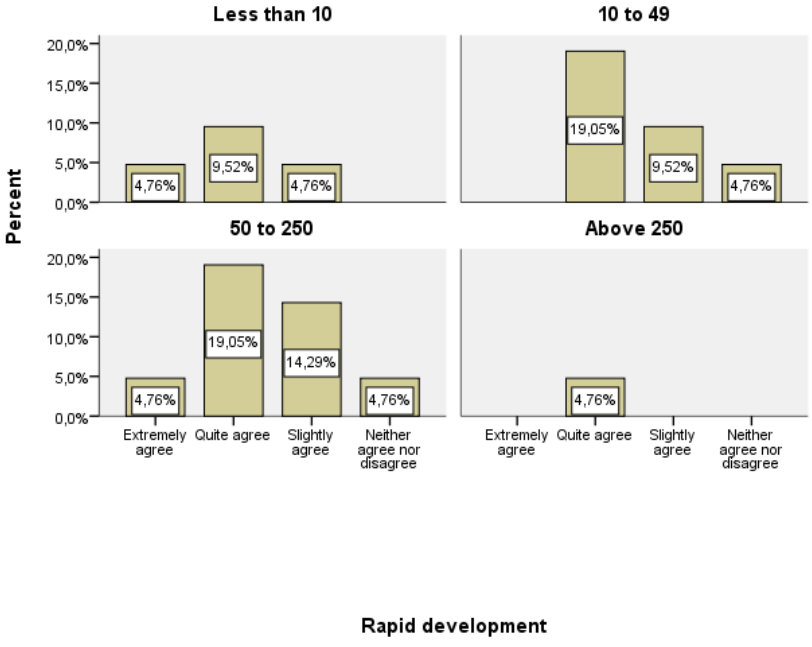


Figure 4-38 Agile methods and rapid development

4.2.2.1 "Good software developers" or "Good development tools"

We analysed which is very interesting, after comparison that from professionals using traditional methods approx. 86% professionals are agreed with first option i.e. "good software developers", whereas approx. 14% are agreed with second option i.e. "good development tools". With the help of above mentioned analysis we can assume that professionals from traditional methods contradict with previous study and supports contents of agile manifesto. See chapter 2 section software development method.

When we analysed this scenario with respect to size of organization, we find that professionals from traditional methods approx. 7% from micro organization, approx. 33% each from small & medium sized organizations and approx. 14% from large sized organizations are agree with first option i.e. *"good software developers*", whereas approx. 13% from medium sized organizations are agreed with second option i.e. *"good development tools"*.

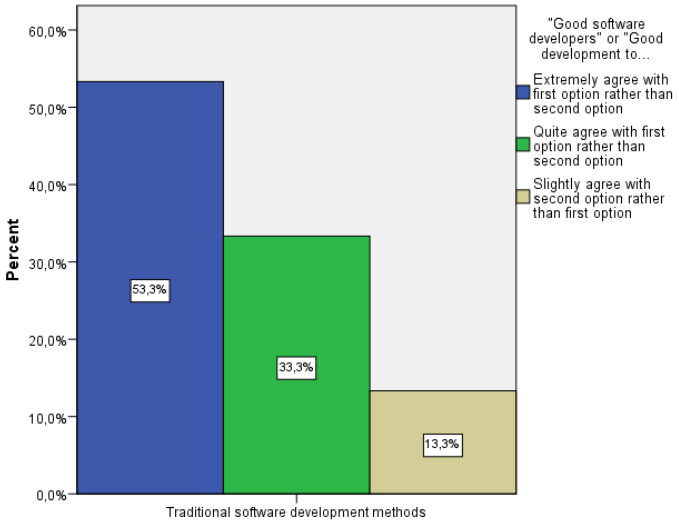


Figure 4-39 Traditional methods level of agreement with "Good software developers" or "Good development tools"

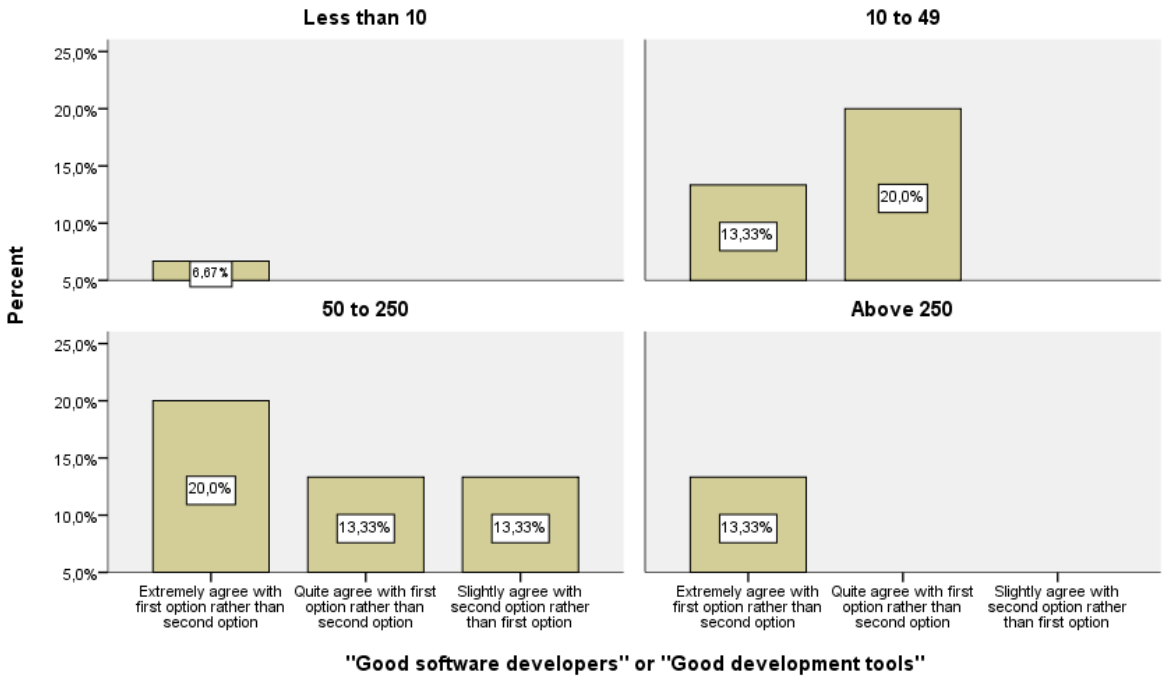


Figure 4-40: Traditional methods level of agreement with "Good software developers" or "Good development tools" regarding size of organization

Whereas from professionals using agile methods approx. 95% professionals are agreed with the first option i.e. *"good software developers"* and approx. 5% are neutral, which also justify previous research mentioned in chapter 2 theoretical background, agile software development methods section.
Whereas with respect to size of organization from professionals with agile methods approx. 19% from micro sized organization, approx. 28% from small sized organization, approx. 43% medium sized organization and approx. 5% belongs to large sized organization are agree with the first option i.e. *"good software developers"*, whereas approx. 5% are neutral.

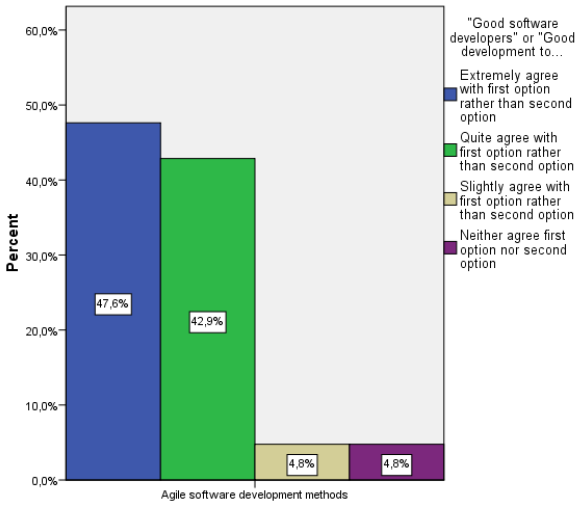


Figure 4-41: Agile methods level of agreement with "Good software developers" or "Good development tools"

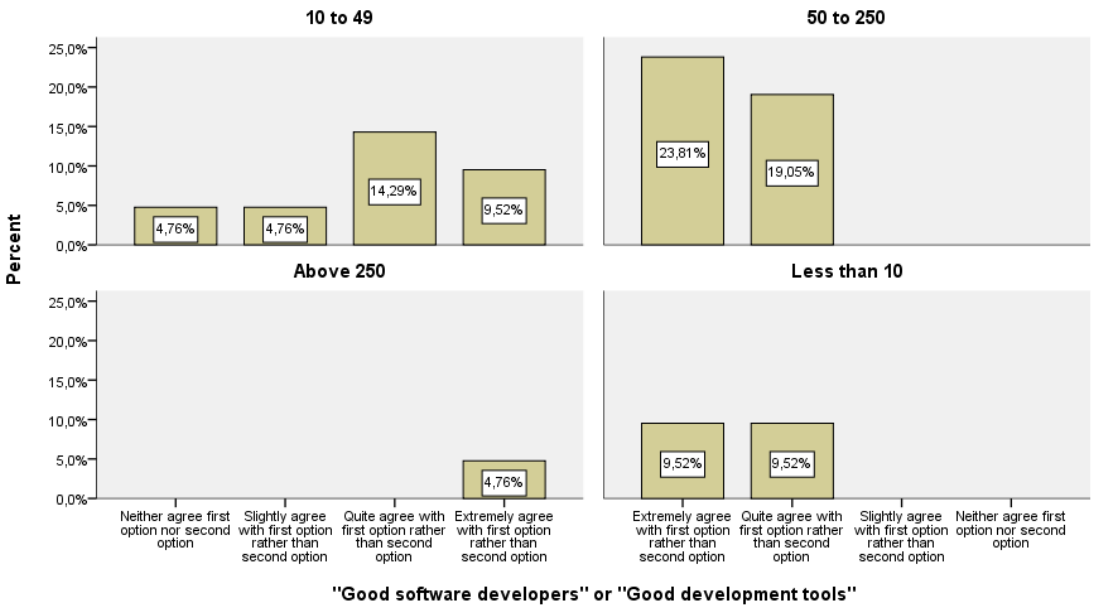


Figure 4-42: Agile methods level of agreement with "Good software developers" or "Good development tools" regarding size of organization

4.2.2.2 "Requirement specifications" or "Customer coordination"

After comparison we analysed that from professionals using traditional methods approx. 87% professionals are agreed with first option i.e. *"Requirement specifications",* whereas approx. 6% are agreed with second option i.e. "*customers coordination"* and approx. 7% professionals are neutral. This also justify our literature study see chapter 2 section traditional software development methods. we analysed with respect to size of organization, we find that professionals from traditional methods approx. 7% from micro organization, approx. 26% each from small sized organizations, approx. 40% from medium sized organizations and approx. 13% from large sized organizations are agree with first option i.e. *"Requirement specifications"*, whereas approx. 7% professionals from small sized organizations are agreed with second option i.e. "*customers' coordination".* Whereas approx. 7% from medium sized organizations are neutral.

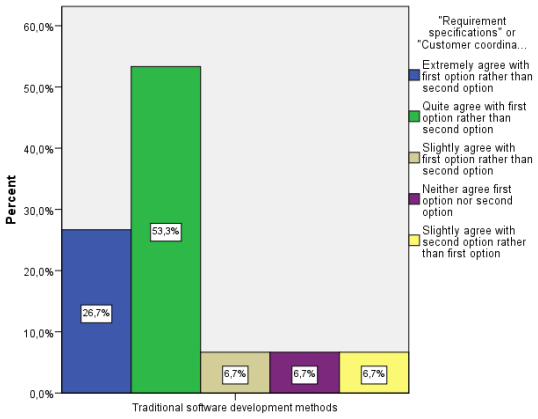


Figure 4-43 Traditional methods level of agreement with "requirement specifications or customer coordination"

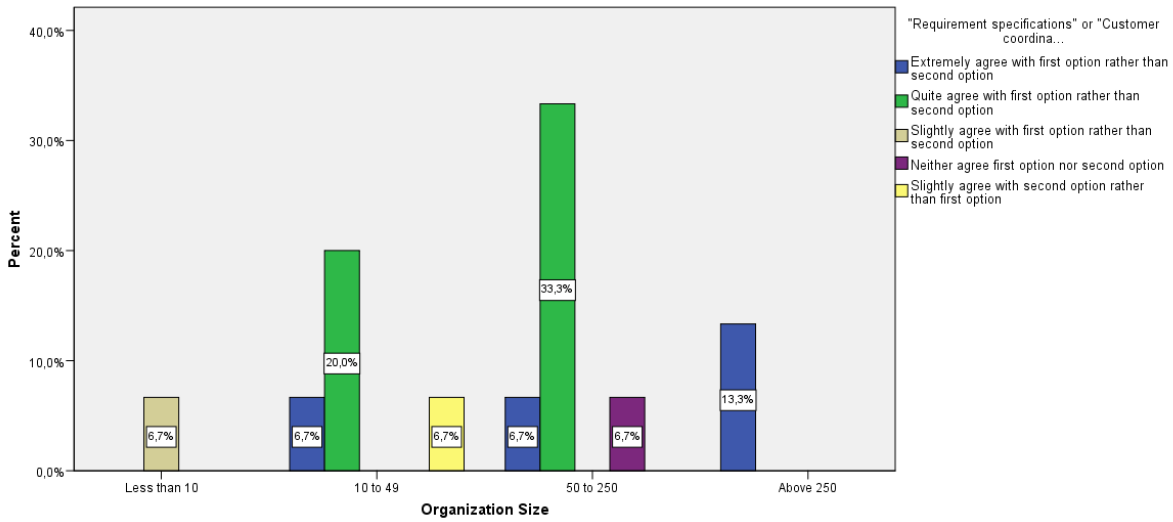


Figure 4-44 Traditional methods level of agreement with "requirement specifications or customer coordination" with respect to size of organization

On the other hand from professionals using agile methods approx. 71% professionals are agreed with the first option i.e. *"Requirement specifications"*, whereas approx. 29% are agreed with second option i.e. "*customers' coordination".*

Furthermore, we analysed with respect to size of organization that from professionals with agile methods approx. 14% from micro organization, approx. 15% from small organizations, approx. 38% from medium organizations and approx. 5% from large sized organization are agree with first option i.e. *"Requirement specifications"*, whereas approx. 5% from micro organization, approx. 14% from small organizations, approx. 4% from medium organizations are agree with option second i.e. "*customers' coordination"* whereas approx. 5% professionals from small organizations are neutral. The analysis helped us to assume that professionals with agile methods prefers to go for requirement specification rather than customers coordination, which contradicts with agile manifesto (see chapter 2, section agile manifesto) and supports traditional methods (see chapter 2, section traditional software development methods).

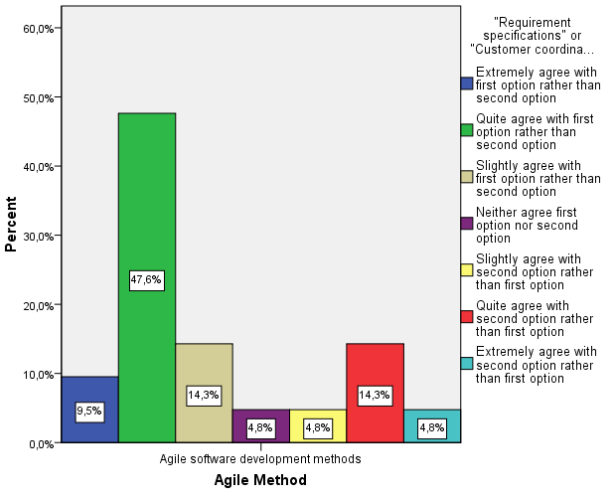


Figure 4-45 Agile methods level of agreement with "requirement specifications or customer coordination"

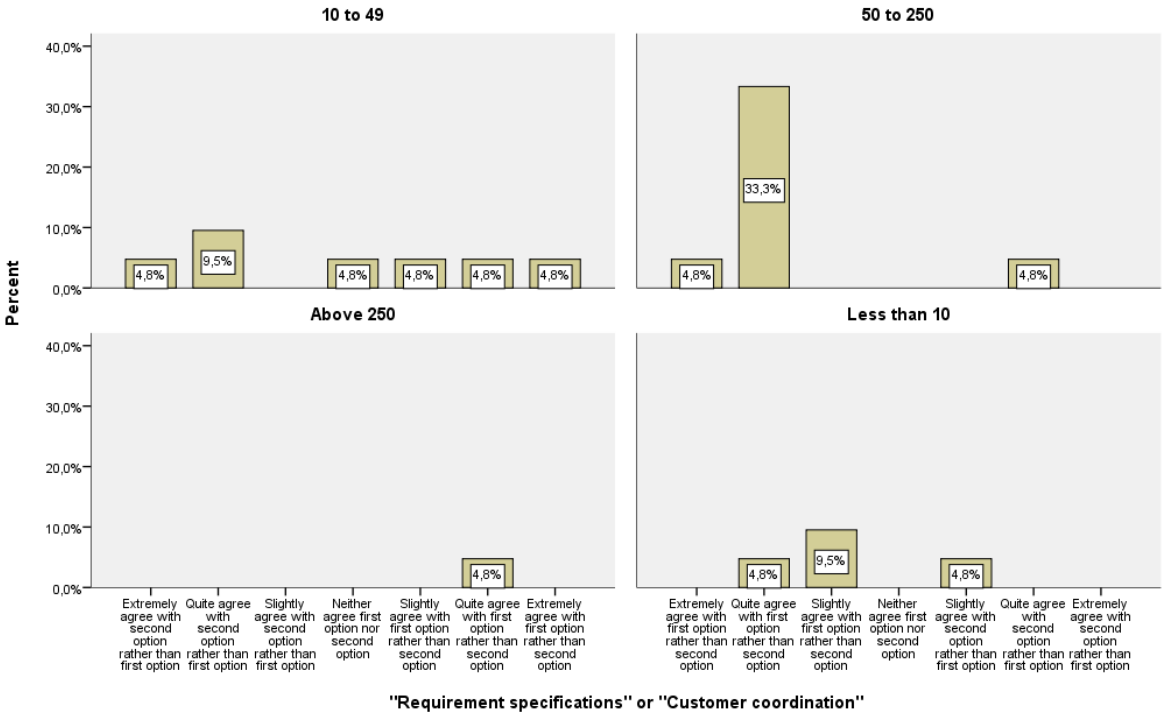


Figure 4-46 Agile methods level of agreement with "requirement specifications or customer coordination" with respect to size of organization

4.2.2.3 "Follow the Project plan" or "Flexibility"

We analysed that from professional using traditional methods approx. 80% agree with first option i.e. *"follow the project plan"*, whereas approx. 20% agree with second option i.e. *"flexibility"*. So we can assume that it support the concept of traditional methods but contradicts with agile manifesto (see chapter 2, section traditional software development methods, agile software development methods).

we analysed with respect to size of organization, we find that professionals from traditional methods approx. 7% from micro organization, approx. 33% from small sized organizations, approx. 32% from medium sized organizations and approx. 7% from large sized organizations are agree with first option i.e. *"follow the project plan*", whereas approx. 14% professionals from medium sized organizations, approx. 7% from large sized organizations are agreed with second option i.e. *"flexibility".*

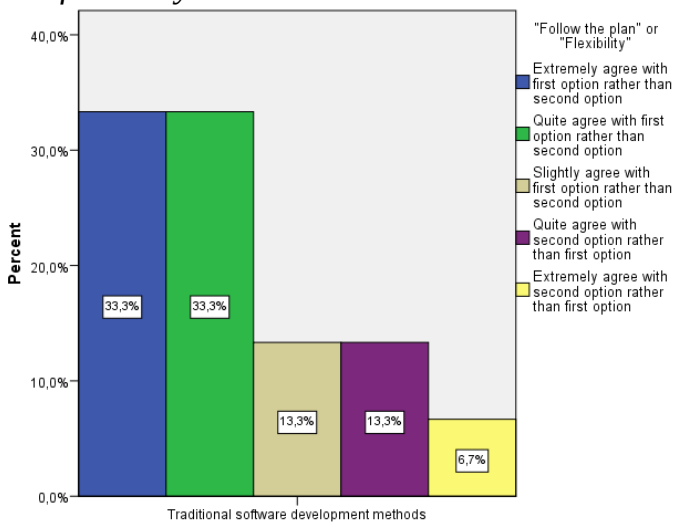


Figure 4-47 Traditional methods level of agreement with "follow the plan or flexibility"

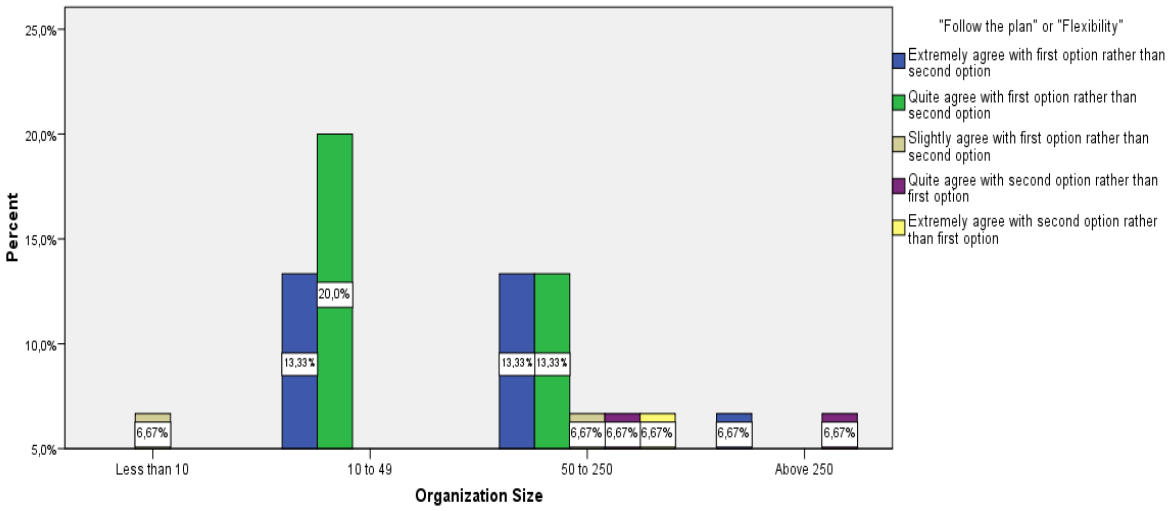


Figure 4-48 Traditional methods level of agreement with "follow the plan or flexibility" with respect to size of the organization

On the other hand from professionals using agile methods approx. 62% agree with first option i.e. "*follow the project plan"*, whereas approx. 34% agree with second option i.e. *flexibility*, so more percentage of professionals in agile methods agreed on following the project plan which support the concept of traditional methods but contradicts with agile manifesto (see chapter 2, section traditional software development methods, agile software development methods).

we analysed with respect to size of organization, we find that professionals from agile methods approx. 14% from micro organization, approx. 9% from small sized organizations, approx. 34% from medium sized organizations and approx. 5% from large sized organizations are agree with first option i.e. *"follow the project plan*", whereas approx. 5% professionals from micro organizations, approx. 19% professionals from small size organizations, approx. 9% from medium sized organizations are agreed with second option i.e. *"flexibility"* and approx. 5% from small sized organizations are neutral.

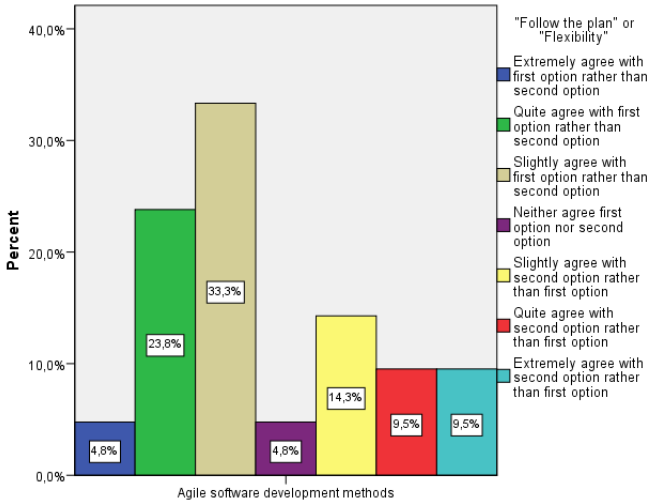


Figure 4-49 agile methods level of agreement with "follow the plan or flexibility"

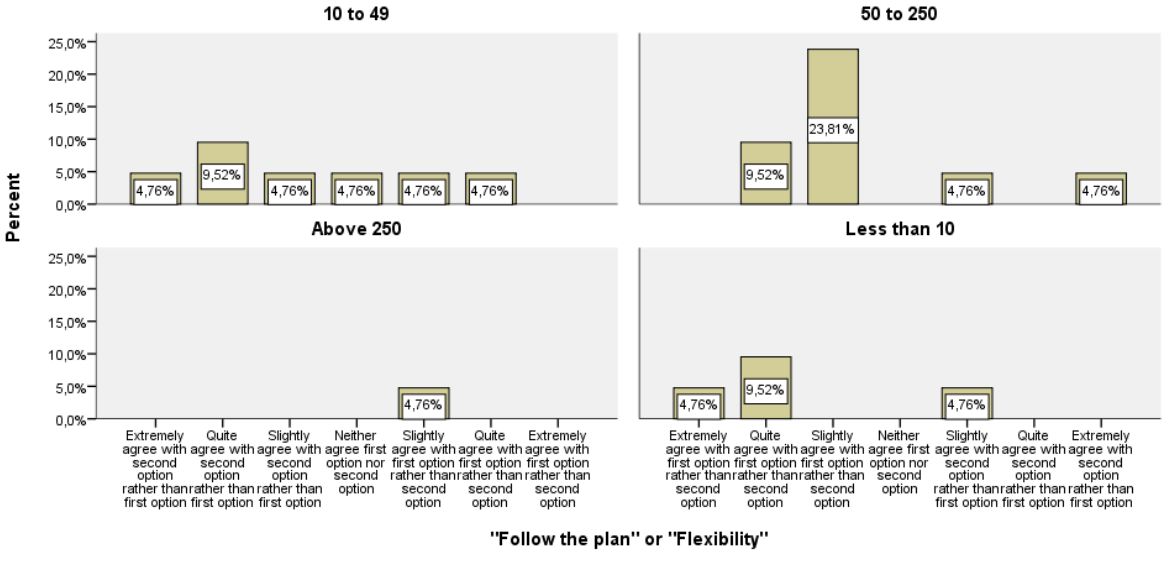


Figure 4-50 Agile methods level of agreement with "follow the plan or flexibility" with respect to size of the organization

4.2.2.4 "Focus on software" or "Large Documentation"

Approx. 73% of professionals with traditional methods are agreed on option one i.e. *"focus on software",* whereas approx. 20% agree on second option i.e.

"*large documentation"* and approx. 7% are neutral. We can assume that high ratio of support with focus on software shows that this supports agile manifesto (see chapter 2, agile software development methods).
we also analysed with respect to size of organization, we find that professionals from traditional methods approx. 7% from micro organization, approx. 20% each from small sized organizations, approx. 40% from medium sized organizations and approx. 7% from large sized organizations are agree with first option i.e. "*focus on software"*, whereas approx. 13% professionals from small sized organizations, approx. 6% from large sized organizations are agree with second option i.e. "*large documentation"* and approx. 7% from medium sized organizations are neutral.

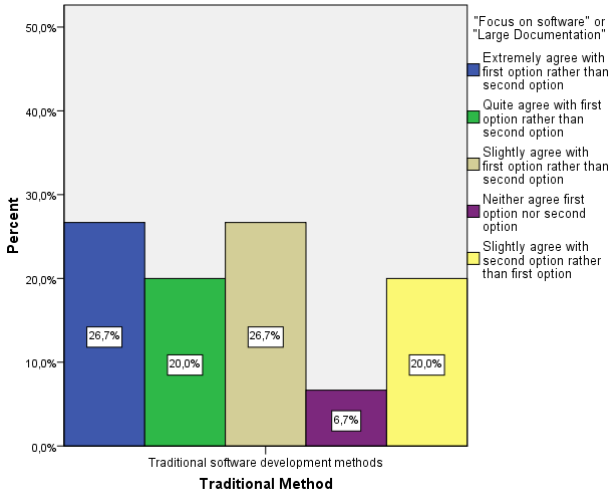


Figure 4-51 Traditional methods level of agreement with "focus on software or large documentation"

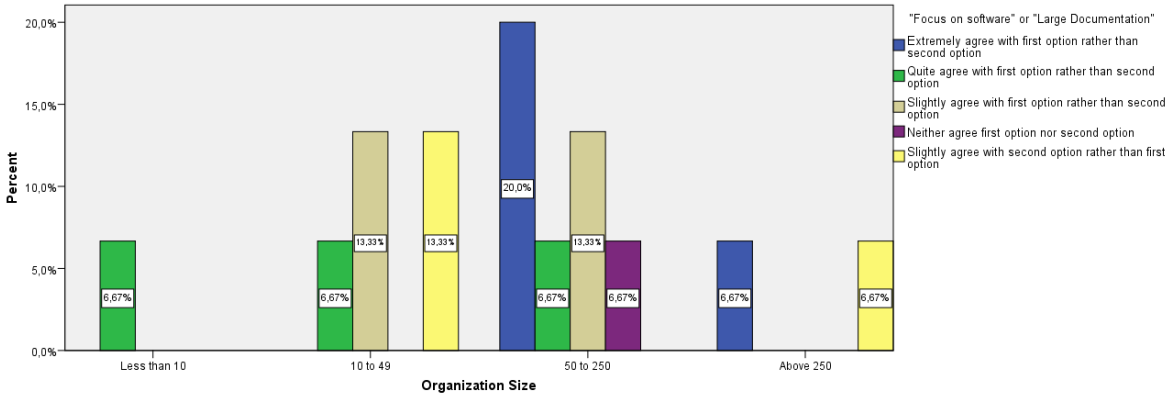


Figure 4-52 Traditional methods level of agreement with "focus on software or large documentation" with respect to size of the organization

On the other hand approx. 81% of professionals from agile methods are agree on option one i.e. "*focus on software",* whereas approx. 14% agree on second option i.e. "*large documentation"* and approx. 10% are neutral, so we can assume that, it completely supports agile manifesto (see chapter 2, agile software development methods).

On the other hand we analysed that from agile professionals approx. 14% from micro organization, approx. 34% from small organizations, approx. 28% from medium sized organizations and approx. 5% from large size organizations are agree with first option i.e. "*focus on software",* and approx. 5% professionals from micro organization, approx. 4% from medium sized organization are agree with second option i.e. "*large documentation"* whereas approx. 10% professional are neutral.

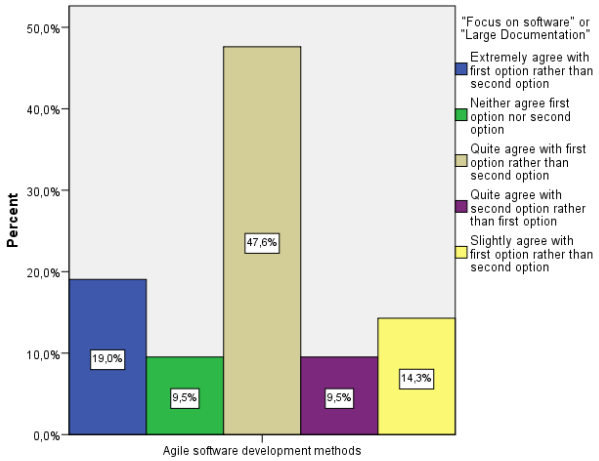


Figure 4-53 Agile methods level of agreement with "focus on software or large documentation"

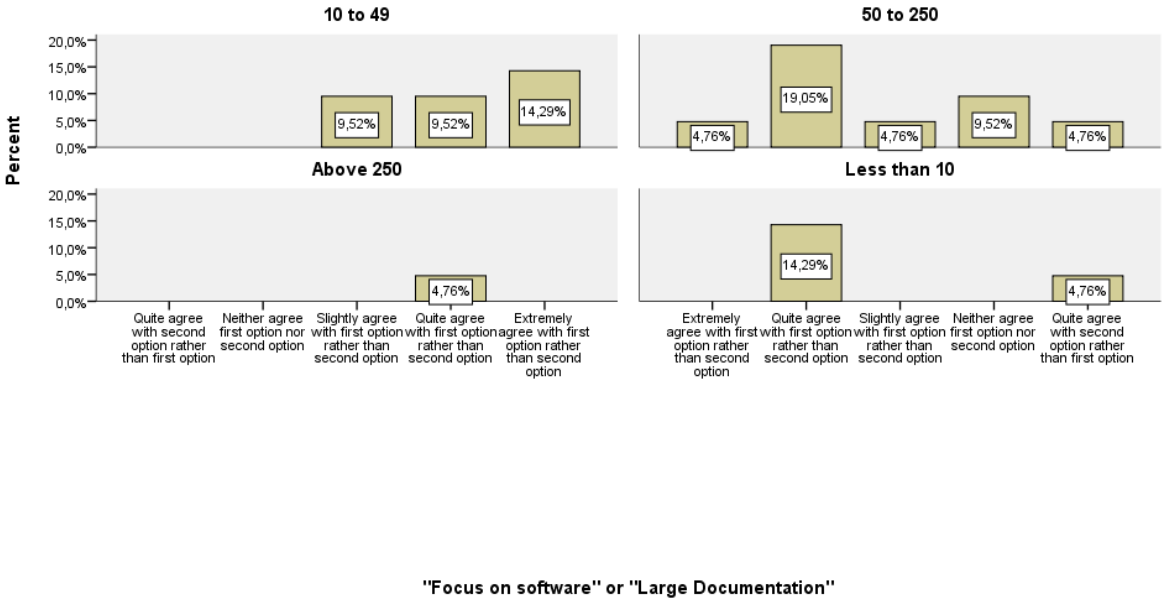


Figure 4-54 Agile methods level of agreement with "focus on software or large documentation" with respect to size of the organization

5 Conclusion and Discussion

Since, our main focus of research was to identify comparison between traditional and agile methods. In last chapter, we presented survey results and analysed data to proceed further for research findings. Here, this chapter structured as follows, we elaborated discussion in term of suitable method for professionals, and challenges during method adoptability and method attitude towards agility in order to find precise results with respect to our research questions. At the end, research work concluded and further studies are specified.

5.1 Discussion

5.1.1 Suitable Method for Professionals

In first research question, we aimed to find out satisfaction level of professionals on agile and traditional approaches, which may support them in risk analysis, communication with customers and better quality of software. In general, as resulted before, most professionals recommended traditional approach that is more reliable as compare to agile. In this sense, we assumed that professionals put consideration on reliability than productivity. As reported before in [47], our study supports hypothesis where professionals are much satisfied with traditional practices.

As we examined before (section 4.2.1), micro level and small organizations like to work with agile practices. The reason behind that agile focuses on collaboration between team members with better communication channels [31], and provides flexibility in changing requirements. In case, large organizations apply agile practices to their environment that may influence of lack of communication between large groups and teams. Therefore, in case of agile, everything is uncertain [53], and even could be possible to change project later on that could be handled in small organization. On other side, in large organization, these issues are considered carefully, and they prefer traditional practices, even agile does not address communication issues between multiple teams. However, we summarised that as agile practices reflect, it is perfectly adoptable for micro and small organizations.

As we resulted before, traditional practices support better in risk analysis where every development phase carefully examined before moving to next phase [17]. As we find out in our results, professionals do not agree on agile practices that could help in risk analysis. Thus, as mentioned above, agile approach is considered for small organizations that argument assists previous comparison study on methodology [30], and here we argued that micro and small organization may not bother risk analysis. Moreover, they start immediately coding on initial requirements, and focus on functional software delivery iteratively with onsite customer.

In order to support communication with customers, we assist that agile practices are considered better in supporting developers to communicate with their customers. In this particular, if we identified the reasons behind that onsite customer has to be part of development team by providing valid requirements, and early feedback on system verification with test cases [36]. In addition to, within small team size (10 members) as reported in [36], developers feel confident to have well understood requirements for immediate business value that supports to our result findings.

Therefore, if we apply agile approaches to large organization where they may need to hire customer for every team. In this scenario, multiple customers could be reflected to produce contradictory requirements. However, we capsulate above discussion by stating that agile practices entirely fine for customer collaboration within small team, even in case of large teams, it would be hard for one customer to communicate with multiple teams. In such case, we assumed that professionals may not have same results on requirements validation and verification by their customers.

As far as software quality concern, our study suggests traditional approaches where every development phase inspected carefully that have impact on overall quality of software. In case of agile, everything uncertain [53], even some professionals claimed that development phases of agile seem like maintenance phase of traditional approach [52]. Moreover, agile professionals keen to change rapidly in system in response to customer desires, very often, story cards by customer are not enough for non-functional requirements, such as security and reliability [52]. Bearing this in mind, as above mentioned reasons influence to low quality software, we interpret that agile professional may have domain experts for non-functional requirements or might be they need to identify requirements validation carefully with short development life cycle.

In including remarks, it would be hard to specify which method is most suitable for organizations. In such case, different parameter can be taken into consideration in order to answer first research question. As we inferred before, agile professionals may not require much risk analysis with short development life cycle in small organizations. Moreover, our study does not encourage professionals to work with agile approaches within large teams that influence communication issues. However, we investigated traditional approaches better to achieve high quality software whereas agile aims on software delivery immediately with customer desires.

5.1.2 Challenge on Method Adoptability

However, main focus of this research was to identify challenges while adopting methodology. Since, organization using traditional approaches, and when they try to replace them with agile processes, in this particular, there are some challenges/limitations and unknown issues may need to look deeply. As resulted before, professionals like to work with proven methods, and agile is considered most productive in all stages of software development life cycle, but there is need to identify how it reflects in organization.

Project Managers feel challenges in their role, in some cases; he/she has to be part of development team in order to make decision on requirements identification. In this scenario, team lead or project assistant should be more consistent and need to follow up developer more precisely. With low budget projects, it would not be possible to add up more resources. Even, when large organizations adopt agile methodology, they need to restructure and reconfigure teams or groups, which tend to create communication problems within groups. As agile focuses and improves the collaboration within team [31]. On other hand, in small organization it would not be issue.

In some case, organizations do not want to change their policies, if they are bound to follow the methodology practices. In such case, if organization going to adopt agile methodology, they must have agile expert or coach, in case they want to change in process to avoid changing in policies. Very often, some professionals have not enough knowledge of agile methodology; they try to replace development process with managerial process that indicates unknown issues and decreases productivity of software development method. In certain cases, developers suggested that organizations might have changes in their method or adopt new method according to end customer desires; it does not support old study. Even very often, it may not possible to find such a customer who is much skilled in method adoptability [51].

With concluding remarks, we summarised that it takes time and much costly to adopt methodology, in case professionals are not much skilled or do not have enough knowledge about methodology. On other hand, it depends on project size, for example, Waterfall is perfectly fine for large projects [18], if we apply Scrum to large projects that might be required refactoring and rework on design and implementation phase. However, we examined that communication is challenging issue between large teams in big organization while adopting agile practices.

5.1.3 Method Attitude towards Agility

As we analysed in previous chapter that professionals from both type of methods (agile software development methods & traditional software development methods) are agreed that their methods are helpful for rapid development, but the professionals from agile methods ratio of satisfaction is high. Opinion from professional regarding agility and rapid development is very interesting.

We examined that high level of satisfactory ratio from both type of professionals are agreed with good software developers rather than good development tools. This answer from traditional method user shows that may be they feel that although their method is very helpful but still good software developers are required, who can understand the tools and use the tool to implement the requirement. High ratio of respondents from both type of methods also shows that competency of a personal is very important in both form of methodologies.

In chapter 4 analysis section, we analysed that professionals from both agile and traditional methods prefers requirement specification over customer coordination. Because this analysis shows the experiences of professionals from different methods, so we assumed that may be agile practitioners feels that more involvement of customers during the project lead to lose the concentration with project and more risk factor is involved. Also we assume that good requirement specification leads to good and rapid output.

In analysis section we find that high ratio of professionals from traditional methods are attracted towards follow the project plan rather than flexibility in a project. Although the number of traditional professionals is high in ratio but the interesting thing is that from agile professionals high ratio of professionals also prefer to follow the project plan rather than flexibility in the project. We assume that may be agile professionals experience problems due to the flexibility or continuous change in the project plan and may be they cannot focus on the project, which can lead to some risks and poor quality of the project.

Analysis after comparison shows in chapter 4 that high ratio of survey respondents from both form of methods (agile and traditional) are agreed with first option focus on the software rather than large documentation of the project. We assume with respect to users of traditional methods that may be they have feeling of losing concentration on the project work and software development while spending a lot of time over writing bundle of pages in the form of project documentation.

In such case, we can assume from analysis and the above mentioned discussion that professionals from both type of methods prefer to use a customized method properties, like they can adopt some features from agile manifesto and some features from traditional method. This also mentioned in our analysis and discussion.

5.2 Conclusion and Future Study

With respect to our study, we conclude that professional from both of types of method (agile/traditional) are satisfied with their methodologies. Such as, with regards to professional communication with customers agile methods are assumed to be preferred. Whereas to produced good quality software and risk analysis traditional methods are considered appropriate. We encapsulate that both types of methods are suitable to fulfill customer requirements and demands.

For method adoptability, we capture that professionals apply proven methods where agile is most productive and traditional is considered reliable. During adoption of agile methodology, changes in roles and structure of organization are considered challenging issues. Moreover, we summarized that professionals need to investigate communications issues, time, and cost of implementation and impact of new methodology on project budget.

According to our study we conclude that professionals from both type of methodology are satisfied with their methods support in rapid development, but on the other hand no method fully support the agile values, but to gain the agility we can customize the features such as, from traditional methods follow the project plan and requirement specification whereas from agile methods good software developers and focus on software can be referred to higher positive attitude.

Although, professionals are satisfied with their methods but process needs to be purified and repeated with large size of respondents to focus on each part of the research by dividing it in small sections. Research finding can be implemented on organizations success factors for profitability. Furthermore, advanced study on methods required, especially on agile methods. Therefore, they can equally helpful for experienced and inexperienced professionals, also it should be purified, so that, large size teams can also work with agile methods.

6 References

[1] D. Cohen, M. Lindvall, and P. Costa, “An introduction to agile methods,” *Advances in Computers*, vol. 62, 2004, p. 1–66

[2] J. Koskela and V. teknillinen tutkimuskeskus, *Software configuration management in agile methods*, VTT Technical Research Centre of Finland, 2003.

[3] D.F. Rico, “Effects of agile methods on website quality for electronic commerce,” *hicss*, 2008, p. 464.

[4] D.F. Rico, “What is the Return on Investment (ROI) of Agile Methods?” http://davidfrico.com/biography-f.htm .

[5] F.K.Y. Chan and J.Y.L. Thong, “Acceptance of agile methodologies: A critical review and conceptual framework,” *Decision Support Systems*, vol. 46, 2009, p. 803–814.

[6] D.E. Krutz, “Conventional Programming Techniques vs. Agile Programming Techniques.”

[7] B. Wood, R. Pethia, L.R. Gold, R. Firth, and C.-M.U.P.P.S.E. INST, *A guide to the assessment of software development methods*, Citeseer, 1988.

[8] P.N. Ghauri and K. Gr\onhaug, *Research methods in business studies: a practical guide*, Prentice Hall, 2005.

[9] Dr Paul Ewings , Dr Roy Powell, Mr Andy Barton, Dr Colin Pritchard: HELP SHEET 9, Peninsula Research And Development Support Unit.

[10] F.H. International, N. Mack, C. Woodsong, and E.U.A. for I. Development, *Qualitative Research Methods: A data collector’s field guide*, FLI, 2005.

[11] F. Shull, J. Singer, and D.I.K. Sj\oberg, *Guide to advanced empirical software engineering*, Springer Verlag, 2007.

[12] Quantitative Research Method; http://www.fortunecity.com/greenfield/grizzly/432/rra2.htm (Acc. 2011-03-03)

[13] K. Williamson, *Research Methods for Students, Academics and Professionals: Information management and systems*, 2ed. Wagga

Wagga, N.S.W. : Centre for Information Studies, Charles Sturt University, 2002.

[14] http://www.fao.org/docrep/w3241e/w3241e04.htm (last accessed 27th April 2011)

[15] M. Cantor and I. Books24x7, *Object-oriented project management with UML*, Wiley, 1998.

[16] A.M. Davis, E.H. Bersoff, and E.R. Comer, "A strategy for comparing alternative software development life cycle models," *IEEE Transactions on Software Engineering*, 1988, p. 1453–1461.

[17] D. Aveson and G. Fitzgerald, "Methodologies for developing information systems: a historical perspective," *The Past and Future of Information Systems: 1976–2006 and Beyond*, 2006, p. 27–38.

[18] P.K. Ragunath, S. Velmourougan, P. Davachelvan, S. Kayalvizhi, and R. Ravimohan, "Evolving a New Model (SDLC Model-2010) for Software Development Life Cycle (SDLC)," *IJCSNS*, vol. 10, 2010, p. 112.

[19] J. Iivari and K. Lyytinen, "Research on Information Systems Development in Scandinavia," *Rethinking management information systems: an interdisciplinary perspective*, 1999, p. 57.

[20] R.A. Boggs, "The SDLC and Six Sigma: An Essay on Which is Which and Why," *Issues in Information Systems*, vol. 5, 2004.

[21] Simon Bennett, Steve McRobb, Ray Farmer (2003); *Object-Oriented System Analysis And Design Using UML*, ISBN 0 201 59620 2, England.

[22] B.W. Boehm, "A spiral model of software development and enhancement," *Computer*, vol. 21, 1988.

[23] F. Rohde, "An ontological evaluation of JACKSON's system development model," *Australasian Journal of Information Systems*, vol. 2, 2007.

[24] M. Jackson, "The Origins of JSP and JSD: a Personal Recollection," *IEEE Annals of the History of Computing*, vol. 22, 2000, p. 61–63.

[25] Philippe Kruchten (2001); *What Is the Rational Unified Process?*, Rational Fellow, Rational Software Canada. Copy right 2001. http://www.ida.liu.se/~HKGC10/www-pub-old/vt05/art_rup/WhatIstheRationalUnifiedProcessJan01.pdf

[26] S.W. Ambler and R. Jeffries, *Agile modeling: effective practices for extreme programming and the unified process*, John Wiley & Sons, Inc. New York, NY, USA, 2002.

[27] A. Finkelstein, J. Kramer, B. Nuseibeh, L. Finkelstein, and M. Goedicke, "Viewpoints: A framework for integrating multiple perspectives in system development," *International Journal of Software Engineering and Knowledge Engineering*, vol. 2, 1992.

[28] P. Kroll and P. Kruchten, *The rational unified process made easy: a practitioner's guide to the RUP*, Addison-Wesley Professional, 2003.

[29] P Abrahamsson, O Salo, J Ronkainen,*"Agile software development methods. Review and analysis",* ISBN 951–386010–8 (URL: http://www.inf.vtt.fi/pdf/)

[30] B. Boehm, "Get Ready for Agile Methods, with Care," *IEEE Software Development*, January 2002, pp.64-69

[31] J. Hunt, *Agile software construction*, Springer, 2006.

[32] E. Mnkandla and B. Dwolatzky, "Agile Software Methods: State-of-the-Art," *Agile software development quality assurance*, 2007, p. 1-22.

[33] S.W. Ambler, "The Agile Scaling Model (ASM): Adapting Agile Methods for Complex Environments," *Environments*, 2009.

[34] M. Huo, J. Verner, L. Zhu, and M.A. Babar, "Software quality and agile methods," 2004.

[35] Favaro, J. Managing Requirments for Business Value. IEEE Software, Vol. 19, 2002, pp. 15-17.

[36] P. Abrahamsson, "Extreme programming: First results from a controlled case study," 2003.

[37] K. Beck, *Extreme programming explained: embrace change*, Addison-Wesley Professional, 2000.

[38] K. Beck, “Embracing change with extreme programming,” *Computer*, vol. 32, 1999, p. 70–77.

[39] F. Maurer and S. Martel, “Extreme programming. Rapid development for Web-based applications,” *Internet Computing, IEEE*, vol. 6, 2002, p. 86–90.

[40] M.A. Awad, “A comparison between agile and traditional software development methodologies,” *University of Western Australia*, 2005.

[41] K. Schwaber and M. Beedle, “Agile software development with Scrum,” 2001.

[42] G.G. Miller, “The characteristics of agile software processes,” *tools*, 2001, p. 0385.

[43] P. Coad, E. Lefebvre, and J. De Luca, *Java modeling in color with UML: Enterprise components and process*, Prentice Hall, 1999.

[44] S.R. Palmer and J.M. Felsing, *A practical guide to feature-driven development*, Prentice Hall, 2002.

[45] A. Cockburn, *Agile software development*, Addison-Wesley Boston, 2002.

[46] Apelkrans, M et al. (2001). OOS/UML: En objektorienterad Systemutveck -lingsmodell för processorienterad affärsutveckling. Lund: Studentlitteratur.

[47] O. Fransson, “Agile Software Development in Sweden,” Jonkoping University, 2005.

[48] M. Lindvall, V. Basili, B. Boehm, P. Costa, K. Dangle, F. Shull, R. Tesoriero, L. Williams, and M. Zelkowitz, “Empirical findings in agile methods,” *Extreme Programming and Agile Methods—XP/Agile Universe 2002*, 2002, p. 81–92.

[49] R.L. Glass, “Agile versus traditional: Make love, not war!,” *Cutter IT Journal*, vol. 14, 2001, p. 12–18.

[50] N. Boateng, “Pro Agile and Anti-Agile Methods for Software Development.”

[51] S. Nerur, R.K. Mahapatra, and G. Mangalaraj, “Challenges of migrating to agile methodologies,” *Communications of the ACM*, vol. 48, 2005, p. 72–78.

[52] B. Boehm and R. Turner, “Management challenges to implementing agile processes in traditional development organizations,” *IEEE software*, 2005, p. 30–39.

[53] M. Lindvall, D. Muthig, A. Dagnino, C. Wallin, M. Stupperich, D. Kiefer, and J. May, others, “Agile software development in large organizations,” *Computer*, 2004, p. 26–34.

7 Appendix

7.1 Appendix I Survey Questionnaires

7.1.1 A comparative study on Traditional Software Development Methods and Agile Software Development Methods

7.1.2 Questionnaire Confidentiality

This Survey is used for a thesis titled “A comparative study on Traditional Software Development Methods and Agile Software Development Methods”, which is written for a master program at Jonkoping University. All information that you provide will be treated confidentially, and your identity and organization name will not be connected to published data. In short your information will be treated as confidential and secured.

Which software development method do you use?

- Agile software development methods
- Traditional software development methods
- No method

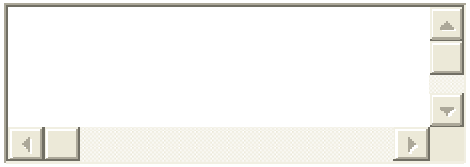
Which option describes your position in your organization?

- Programmer

- System Analyst
- Software Architecture / Software Engineer
- Consultant
- Project Manager / Team Leader
- Executive
- PhD. Student
- Masters Student

What type of applications, does your organization work on?

- End user applications
- Desktop applications
- Web based applications
- Safety critical applications
- Distributed applications
- Multimedia applications
- Others



How many software development professionals are working in your organization?

- Less than 10
- 10 to 49
- 50 to 250
- Above 250

How satisfied you are with your software development method?

- Extremely satisfied
- Quite satisfied
- Slightly satisfied
- Neither satisfied nor dissatisfied
- Slightly dissatisfied
- Quite dissatisfied

- Extremely dissatisfied

To what extent, do you agree that your software development method helps you to have understandable communication with your customers?

- Extremely agree
- Quite agree
- Slightly agree
- Neither agree nor disagree
- Slightly disagree
- Quite disagree
- Extremely disagree

To what extent, do you agree that your software development method help you to fulfill your customer's requirements / demands?

- Extremely agree
- Quite agree
- Slightly agree
- Neither agree nor disagree

- Slightly disagree
- Quite disagree
- Extremely disagree

Do you follow any software quality standards? E.g. CMMI and ISO 9000 etc.

- Yes
- No

To what extent, do you agree that your software development method helps you to develop high quality software?

- Extremely agree
- Quite agree
- Slightly agree
- Neither agree nor disagree
- Slightly disagree
- Quite disagree
- Extremely disagree

To what extent, do you agree that your software development method helps you in risk analysis?

- Extremely agree
- Quite agree
- Slightly agree
- Neither agree nor disagree
- Slightly disagree
- Quite disagree
- Extremely disagree

Which factor is most important for you as a professional when adopting a method?

- Low cost
- Easy to handle
- Great productivity
- Great reliability
- Do not know

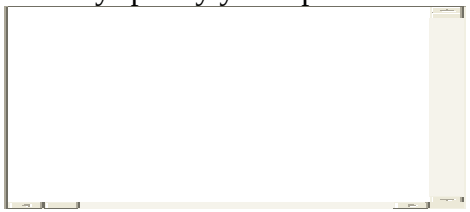
Which method do you find more suitable with respect to project size?
(Small / medium / large)

	Waterfall	Spiral	JSD	RUP	SIS	V Shaped	XP	FDD	DSDM	Scrum	Crystal
Small size project	○	○	○	○	○	○	○	○	○	○	○
Medium size project	○	○	○	○	○	○	○	○	○	○	○
Large size project	○	○	○	○	○	○	○	○	○	○	○

Which option is best suitable for you, when adopting new methodologies?

- ◉ Market leader by adopting the new methods
- ○ Follows the market leaders
- ○ Only adopt proven methods
- ○ Do not like to adopt new methods

*Briefly specify your opinion about the different challenges during the adoptability of a software development method.



To what extent, you agree that your software development method support you in rapid development.

- Extremely agree
- Quite agree
- Slightly agree
- Neither agree nor disagree
- Slightly disagree
- Quite disagree
- Extremely disagree

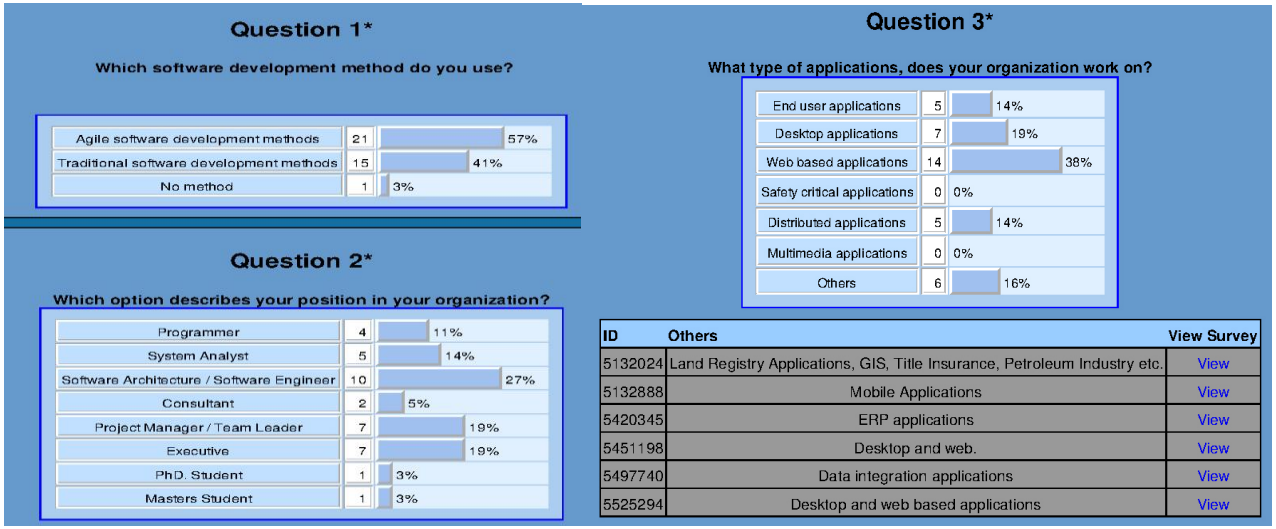
To understand the agility in software development project, which one is most important to focus on?

	Extremely agree with first option rather than second option	**Quite agree with first option rather than second option**	**Slightly agree with first option rather than second option**	**Neither agree first option nor second option**	**Slightly agree with second option rather than first option**	**Quite agree with second option rather than first option**	**Extremely agree with second option rather than first option**
"Good software developers" or "Good development tools"							
"Requirement specifications" or "Customer coordination"							

"Follow the Project plan" or "Flexibility"							
"Focus on software" or "Large Documentation"							

7.2 Appendix 2

7.2.1 Snapshots of survey results



Question 1*

Which software development method do you use?

Agile software development methods	21	57%
Traditional software development methods	15	41%
No method	1	3%

Question 2*

Which option describes your position in your organization?

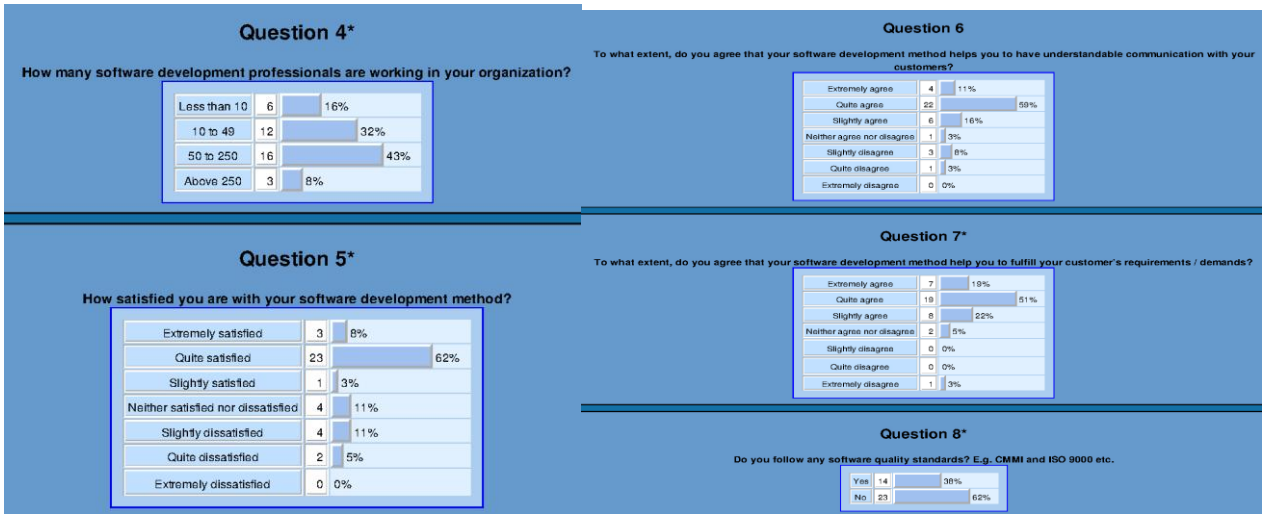
Programmer	4	11%
System Analyst	5	14%
Software Architecture / Software Engineer	10	27%
Consultant	2	5%
Project Manager / Team Leader	7	19%
Executive	7	19%
PhD. Student	1	3%
Masters Student	1	3%

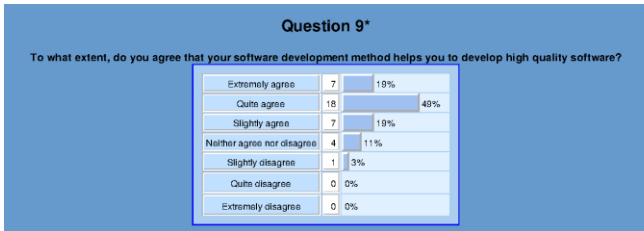
Question 3*

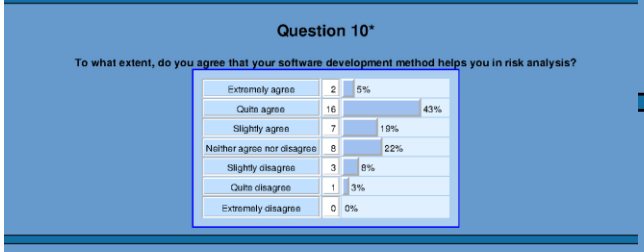
What type of applications, does your organization work on?

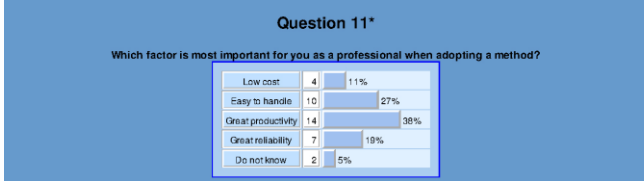
End user applications	5	14%
Desktop applications	7	19%
Web based applications	14	38%
Safety critical applications	0	0%
Distributed applications	5	14%
Multimedia applications	0	0%
Others	6	16%

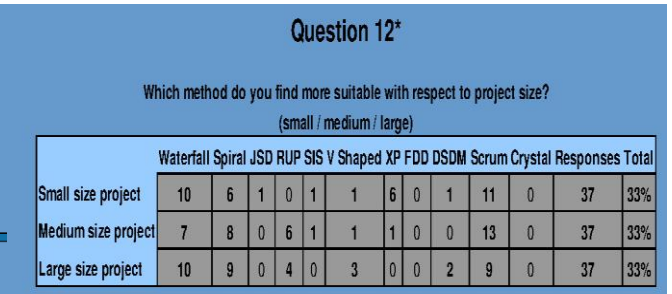
ID	Others	View Survey
5132024	Land Registry Applications, GIS, Title Insurance, Petroleum Industry etc.	View
5132888	Mobile Applications	View
5420345	ERP applications	View
5451198	Desktop and web.	View
5497740	Data integration applications	View
5525294	Desktop and web based applications	View


Question 4*
How many software development professionals are working in your organization?
Less than 10 6 16%
10 to 49 12 32%
50 to 250 16 43%
Above 250 3 8%
Question 5*
How satisfied you are with your software development method?
Extremely satisfied 3 8%
Quite satisfied 23 62%
Slightly satisfied 1 3%
Neither satisfied nor dissatisfied 4 11%
Slightly dissatisfied 4 11%
Quite dissatisfied 2 5%
Extremely dissatisfied 0 0%
Question 6
To what extent, do you agree that your software development method helps you to have understandable communication with your customers?
Extremely agree 4 11%
Quite agree 22 59%
Slightly agree 6 16%
Neither agree nor disagree 1 3%
Slightly disagree 3 8%
Quite disagree 1 3%
Extremely disagree 0 0%
Question 7*
To what extent, do you agree that your software development method help you to fulfill your customer's requirements / demands?
Extremely agree 7 19%
Quite agree 19 51%
Slightly agree 8 22%
Neither agree nor disagree 2 5%
Slightly disagree 0 0%
Quite disagree 0 0%
Extremely disagree 1 3%
Question 8*
Do you follow any software quality standards? E.g. CMMI and ISO 9000 etc.
Yes 14 38%
No 23 62%

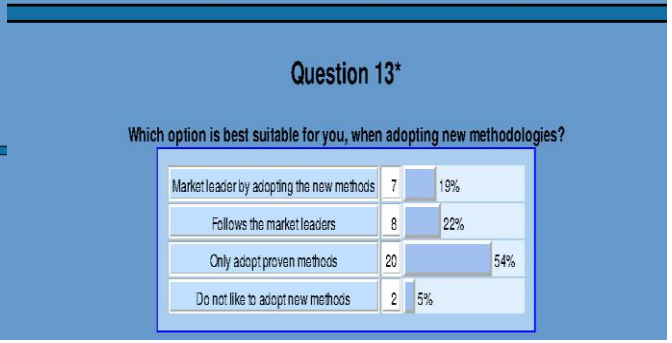








	Waterfall	Spiral	JSD	RUP	SIS	V Shaped	XP	FDD	DSDM	Scrum	Crystal	Responses	Total
Small size project	10	6	1	0	1	1	6	0	1	11	0	37	33%
Medium size project	7	8	0	6	1	1	1	0	0	13	0	37	33%
Large size project	10	9	0	4	0	3	0	0	2	9	0	37	33%



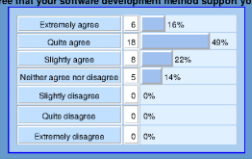
Question 14*

Briefly specify your opinion about the different challenges during the adoptability of a software development method.

Text Answers (37)	View
some they are much time consuming and they are a bit expensive, so the budget of the project rises. so due to that reason I use traditional methods for the medium and large size projects. they are not quite suitable for small projects	View
now a days, time is critical, it depends upon the with in time delivery.	View
N/A	View
Cost.	View
In reality, there is formal method of development. Client's requirements keep on changing so it's mostly spiral.	View
It totally depends on Organization structure and nature of Project.	View
Client usually prefers to adopt those method in which they have been working in past and they adopt those method as they have seen success in them so we have to follow the same. e.g often client doesn't like scrum as it is still not being used much in the market.	View
N/A	View
understanding its basic flow, tailoring it according to your situations and applying your adaptation on actual work.	View
	View
-Learning new ways/process in a short time period is not very easy -Reliability issues in the end product. As the new method does not carry a history of success with the organization	View
Reference material for new technologies.	View
a	View
In general we select the method which is helpful in all the phases of software development.	View
N/A	View
new software method should comply with need of current project, new software technique only used when demand of software development does not fall in current in used methodology, following should always be in consideration when adopting new method. 1: changes to current method 2: understanding time in new method. 3: impact of new method on current project.	View
Cost, Productivity and Reliability are core factors to consider	View
N/A	View
...	View
No Comments!	View
Our team faces a communication gap to strictly follow any SD method. As the communication gap causes lack of understanding. And another major challenge is organizational policies about restricted/confidential data and rules of their business process.	View
Need to educate whole team regarding adopting software development method. Mostly people donot like to change their conventional approach.	View
	View
Mgt not onboard creates quite some disturbance. Team not understanding the meaning of commitment. Hard get acceptance for test. Once the above has been handled it normally goes very well.	View
...	View
Get traditional managers (taylorism approach) to understand that their role, responsibilities, accountability, organization and leadership challenges changes with new methodology (e.g. scrum)	View
cost and risk are most important for me as a challenge.	View
As with any changes, it takes a while for all parties to adjust to new routines.	View
No comments.	View
Customer Satisfaction	View
Cost. Resources.	View
The major challange is education and onboarding of the employees. Handling change management in the organisation.	View
It takes time, you can't change ways-of-work over a night. As a manager you need to follow up closely and be persistant	View
1.there is alot of difficulties in analysis from the client side 2. there is problem to convince the client in different requirements to fulfill which is not possible.	View
None	View
N/A	View
Its depends on structure of team and project size. All team member should have skills and knowledge about method, how to proceed otherwise it will be much costly and have impact on software quality.	View

Question 15*

To what extent, you agree that your software development method support you in rapid development.



Question 16*

To understand the agility in software development project, which one is most important to focus on?

	Extremely agree with first option rather than second option	Quite agree with first option rather than second option	Slightly agree with first option rather than second option	Neither agree first option nor second option	Slightly agree with second option rather than first option	Quite agree with second option rather than first option	Extremely agree with second option rather than first option	Responses	Total
"Good software developers" or "Good development tools"	18	14	1	2	2	0	0	37	25%
"Requirement specifications" or "Customer coordination"	6	18	4	3	2	3	1	37	25%
"Follow the plan" or "Flexibility"	6	10	9	2	3	4	3	37	25%
"Focus on software" or "Large Documentation"	8	13	7	4	3	2	0	37	25%