International Journal of Engineering & Technology, 7 (4.28) (2018) 510 -525

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

The Quality of Teamwork on Methodology in Software Development Workflow

Zairina Ibrahim[1*], Md Gapar Md Johar[2], Normy Rafida Abdul Rahman[3]

[1,2]Information Technology and Innovation Centre, MSU, University Drive Seksyen 13, Shah Alam 40100, Malaysia
[3]Faculty Business Management and Professional Studies, Management and Science University, Selangor, Malaysia.
**Corresponding author E-mail: zairina@msu.edu.my*

Abstract

This paper aims to validate a workflow for software development in process phases to enhance the quality of teamwork. Therefore, dependent on process of software development for projects teams in various phases. However, challenges such as teamwork and information system impede an impact of efficiency development among the teamwork. This paper presents a continuation of an ongoing theoretical framework developed, to further investigate the sustainable and collaborative in teamwork process. To develop the application framework, an approached of user requirement specification (URS), system requirement specification (SRS), layout Module, layout Database, user acceptance testing (UAT) and final acceptance testing (FAT) to completed of phases during the development. Current research on software development on application system is limited to readiness and awareness. This research extends the need for empirical findings from system analyst, top management, developer perception of software development. The paper was validate a process of teamwork in every phases to measure the impact methodology to the workflow and system application. This research found that significantly support the teamwork skill in better performance of application system.

Keywords: *Software Development; Information System; Teamwork.*

1. Introduction

An organizations are increasingly utilizing team-based structures for coordinating work and completing projects. Thus it is imperative for those creating, and performing in, teams to understand and utilize effective processes which lead to high performance. The ways team members interact and work with one another for reaching goals are referred to as processes. The need for, and usefulness of, different processes depends on which stage of work/project the team is at. According to Ingram (1), teamwork is a strategy that has a potential to improve the performance of individuals and organizations, but it needs to be nurtured over time. Therefore, organizations need to constantly look at ways to improve performance, especially in a rapid changing working environment. Management needs to inculcate teamwork activities within the organizations, be flexible to promote it and be willing to allow the teams to be part of decision making. In fact, Conti & Kleiner (2003) reported that teams offer greater participation, challenges, and feelings of accomplishment.

For the previous of findings research, according to Langfred (2), Hoegl et al.(3), Marks et al.(4) and Salas et al.(5), the issue of what processes and components comprise teamwork and how teamwork contributes to team effectiveness and team performance has been much studied, but there is no consensus concerning its conceptual structure. Therefore, using recent research and previous reviews, (6), identified and defined seven core components of teamwork. In fact, using these components and their relationships as a basis, they proposed the teamwork model that is used in this work. The model consists of a learning loop of the following basic teamwork components: communication, team orientation, team leadership, monitoring, feedback, backup, and coordination.

According to Kraut and Streeter (7), in the Dickinson and McIntyre model, the components of monitoring, feedback, and backup are the intermediate processes for ensuring effective teamwork. Finally, the 'output' component is coordination because it defines the performance of the team. Communication is a transversal component of particular importance, because it links the other components. To build software effectively, there is a need for tight coordination among the various efforts involved so that the work is completed and fits together.

The team that worked on Alpha organized the project according to generally recommended Scrum practices. Plans were made at the beginning of each sprint, after the team had reviewed what was produced in the previous sprint. Features were recorded in the sprint backlog. The team that worked on Alpha held three project retrospectives to identify and discuss problems and opportunities that arose during the development process. Daily meetings were organized throughout the project, though these were less frequent in the last two sprints. These meetings were usually about updating the others on progress, development issues, and the project in general. The daily meetings we observed lasted from 10 to 35 min, but were usually shorter than 15 min. The product owner, who was situated in another city, often participated in these meetings by telephone. He participated because both he and the Scrum master thought that it was important to share information constantly and participate in the decision-making process.

Thus, for the implication of previous research according to Langfred (2) , the agile software development emphasizes that teams should be self-managed. However, Scrum and agile methods offer no advice on how shared leadership should be implemented. A

practical implication of Langfred (2) findings is that, if an organization believes in letting teams be more self-managing, great care must be taken in the implementation. This is especially important when the team members have high individual autonomy. The Alpha project was the first big project for most developers. Even though they had worked together for years, they should probably have spent more time together focusing on improving teamwork in the initial phase of the project. The successful teams that (8) observed all gave themselves the time to learn to be a team. If developers who work together have problems becoming a team, they will also have problems becoming a self managing team.

2. Workflow Software in Teamwork

According to Rodd (9), there are five stages of developing a team. The first stage is a stage in which a new group of people comes together to start working as a team. Stage two deals with conflict resolution in a team which aims to bring the team members towards greater acceptance, increased trust and commitment to the task. In stage three, team members are encouraged to participate and contribute to the tasks assigned. This is followed by stage four in which the team members contribute equally and collaborate with each other in achieving common goals. Finally, stage five is the stage in which team members will reflect and celebrate their achievements and later moving on to the next level.

Agile software development methods have led to a number of changes in the way software is developed (10). One of the principles of the agile manifesto states that "the best architectures, requirements, and designs emerge from self organizing teams". Despite reports on the major improvement of agile development methods over traditional development methods (11), team performance is still a challenge. Team performance has been studied in a number of research fields, such as management science and psychology, which led to the development of teamwork effectiveness models. Research by Gulliksen Stray (12), summarized the following challenges of teamwork: wrong tasks being solved by team members as they are working on low priority items, critical decisions are made without team commitment due to a lack of communication, and very minimal time is spent by many agile teams to reflect their work process, thus, not releasing the learning potential.

2.1 Teamwork Factors

Effectiveness and efficiency are two factors that can be used to assess team performance. According to Hackman (13), effectiveness refers to the degree to which a team meets the expectations of the quality of the outcome. For example, it refers to the degree to which goals and quality of the project were met. Efficiency, on the other hand, refers to the degree to which the team meets time and budget objectives (3). Therefore, effectiveness is focusing on the comparison between actual and intended outputs while efficiency compares the actual and the intended inputs.

2.2 Communication

Communication has been widely recognized as an important component of teamwork. In fact, other studies recognize the importance of communication for project success (14-15). It provides the avenue to exchange information, share ideas among team members, coordinate efforts and provide feedback (16). It is not only the act of exchanging important information but also to ensure that the information is delivered to the right party and conveyed the sender's intended meaning accordingly Pinto and Pinto (16). In fact, Xiang et al.(17) for example, found that a project failure may be due to a lack of communication or a misunderstanding that occurs between team members and stakeholders of the project.

According to Hoegl et al.(3) described the quality of communication within a team in terms of the frequency, formalization, structure, and openness of the information exchange. Frequency refers to the extensiveness of communication among team members that is the amount of time spent by team members to communicate with each other. The degree of formalization refers to the spontaneity of communication. A formal communication such as scheduled meetings requires a large amount of preparation and planning prior to its occurrence. Informal communication, on the other hand, includes spontaneous contacts such as talks during a coffee break, short phone calls, and brief chat conversations. Software development teams may benefit from informal communication as it allows team members to share ideas and discuss problems more quickly and efficiently. Furthermore, the structure of communication is also important for software development teams. As the communication through a mediator is prone for miscommunication and may be time-consuming, team members should be able to communicate directly with each other. Moreover, the openness of information exchange is also important (16, 18). According to Hoegl et al. (3), it is one of the most important functions of teamwork. It can harm the integration of team members' knowledge and experience if team members are not open to each other and hold back any important information.

Moreover, communication also provides a basis for other factors that determine team performance. For example, communication is needed to coordinate and integrate team member's efforts and knowledge 25. In addition, communication is also required for a team to comprehend the collective missions, to ensure the team continuously shares the same mental model (19), and to facilitate trust within a team (20). Therefore, communication can be considered as a primary tool that is required in creating a high-performing team that leads to it be part of the TWQ model.

2.3 Coordination of Expertise

According to Hoegl et al. (3) argued that coordination is an important aspect of teamwork. Coordination refers to the development and agreement of a team of a common task-related goal structure, with well-defined sub-goals for each member, without any gap or overlap. Software development work is based on knowledge, thus, expertise is considered as an elementary resource which is not considered in the study. According to Faraj and Sproull (21), coordination of expertise involves managing the knowledge and skill dependencies. It includes identifying the expertise within a team, recognizing the requirement for expertise, and utilizing expertise to good use.

Identifying the expertise refers to knowing who has the skills or knowledge to perform certain tasks as well as to provide the answer or solution to a problem. This helps in assigning tasks to the right team members (22). It is proven that the team performance increases when team members know how expertise is distributed among the team members (23). It also enables team members to anticipate rather than react on other's behaviours (24). However, in the case of software development teams, knowing where the expertise is cannot bring it to good use if they have no idea when and where certain expertise is required in a timely manner. The need for expertise might be higher at different times in the project, thus, it is crucial that all team members know who has which expertise and who requires what expertise at any particular time. Software development team members must be able to recognize the need for certain expertise to enable them to support each other with their expertise or to get the expertise from outside if necessary. This will ensure processes can be carried out effectively and efficiently. Team members should share their knowledge and expertise through ongoing informal interactions (25). A collective mind within a team can be developed if the team members aware how a task done by a team member contributes to the task of another member of the team (26). This, in turn, will result in a better coordination (27).

According to Faraj and Sproull (21) found a significant positive relationship between coordination of expertise and team performance. This relationship is stronger than the relationship between

the presence of expertise and performance. Therefore Hoegl et al. (3) included coordination and balance of member contributions in their TWQ model. Coordination refers to the degree to which individual efforts are well structured and synchronized within the team while the balance of member contributions considers the degree to which team members are able to bring their expertise to their full potential. Therefore, the measure of coordination of expertise (21), to a certain extent, is a combination of measures of coordination and balance of member contributions (3), supplemented with the link to expertise. Since software development is knowledge work, we consider the link between coordination and expertise to be important.

2.4 Cohesion

According to Bollen and Hoyle (28), cohesion is defined as an individual's sense of belonging to a particular group and his or her feeling of morale associated with membership in a group. The definition indicates that cohesion has two dimensions that are a sense of belonging and the feeling of morale. However, team members may not like to associate themselves with the rest of their team if they are lacked a sense of belonging. Similarly, without a sense of morale, the team members may not be motivated to achieve organizational goals and objectives. On the other hand, Mullen and Copper (29) distinguished cohesion using three forces that are the interpersonal attraction of team members, commitment to the team task, and group pride-team spirit.

Cohesion was found to be an important antecedent for team performance (30). According to Hoegl et al. (3), a high-quality teamwork is unlikely to be achieved without the existence of a sense of belonging and a desire to stay on the team and keep it going. In a case of a software development team, cohesion seems to be important as the tasks of the team require high coordination and communication (31).

However, according to Mullen and Copper (29), in their meta-analysis, it revealed disagreements in the literature about the relationship between group cohesion and performance. Therefore, they concluded that the relationship between cohesion and performance is significant but small. In fact, Chang and Bordia (32) pointed out that the disagreements in literature are more towards the inconsistency in measurements and definitions of cohesion and performance. They suggested that the consistency in the measurements and definitions of cohesion and performance is highly required in order to give a more conclusive stand. Based on the multidimensional model, Carron et al.(30) studied the relationship between group cohesion and performance. They found that there is a direct relationship between specific dimensions of group cohesiveness and performance in which cohesion is indicated to be an antecedent of performance rather than a consequence of it.

Although there are different views by various authors, cohesion is still considered and expected to be an important teamwork quality factor for software development teams. An adequate feeling of togetherness and belonging seems to be required in order to achieve high-quality collaboration.

2.5 Trust

According to Friedlander (33) found that trust is a key predictor of team performance. There are many different definitions of trust. Trust is seen as the degree to which someone believes his team members are dependable (34), are willing to be vulnerable to the action of others (35), care about the group's interest (36), or are competent (37). According to Mayer et al.(35), trust can be defined as the willingness of a team member to be vulnerable to the actions of another team member based on the expectation that the other will perform a particular action important to the trust or, irrespective of the ability to monitor or control the other team member.

Trust influences many team processes such as the willingness to share information, give substantial feedback and manage time correctly (15). Trust and the feeling of being trusted encourage team members to communicate more openly and share information more freely with each other. When team members feel that their contribution is not being appreciated accordingly, it is likely that they will not share information (38-41). Additionally, trust has positive effects on the satisfaction of communication that are the perceived accuracy of information was given (26), and satisfaction of working with the group.

Furthermore, trust promotes the way team members interpret others' behaviours such as performance monitoring (5). When a team is lacked trust, instead of working together as a team to produce value-adding ideas, team members will spend more time checking upon others and inspecting each other (42). Moreover, there is a tendency that they will interpret ambiguous behaviours or actions of others, such as missing deadlines, as intentionally damaging actions against a team member or the team (43). According to Handy (44), the risk of a self-fulfilling prophecy is likely to occur when the trust within the team is low, team members might feel that there is no need to be trustworthy if they have the feeling that they are not being trusted by others anyway. Thus, trust is highly required within a team in order for team members to accept and understand that other members are keeping an eye on them, serving the greater good of the team (5). Due to the importance of trust as a supporting mechanism for teamwork, this study argues that this factor cannot be eliminated.

2.6 Mutual Support

According to Tjosvold and Tjosvold (45), mutual support is an essential element of teamwork quality. The idea of teamwork is based on the idea of mutual support of the team members rather than the competition among them (3). Competition between people can exert a positive influence on the motivation and performance of individual tasks. However, cooperation or mutual support among team members is more important for interdependent tasks such as software development. Instead of trying to outdo each other, team members working on a shared goal should try to support each other. They should show respect, provide assistance and support when required, and inspire ideas of other team members and develop them further. On the other hand, if team members demonstrate competitive behaviours, this can lead to distrust and frustration within the team (45). Therefore, both quality and acceptance of ideas generated by team members will increase when team members cooperate (46). Mutual support, therefore, is also an important teamwork element and highly required to enable the team to reach their goals. The better team members support each other, the more effective and efficient these goals can be reached.

2.7 Value Diversity

There are different types of team diversity such as informational, social category and value diversity. Each type of diversity involves different types of challenges and opportunities. In addition, each has a different influence on team performance (47). In most situations, when people refer to team diversity, they are referring to social category diversity (48). It concerns differences in social differences such as age, gender, race, and ethnicity. This study is focusing on team collaboration rather than team composition, thus, the focus will be on value diversity. Value diversity is the result of different perspectives by team members on the team's task, goal, or mission. Such differences can lead to a relationship, task, or process conflicts (47). For example, a conflict is likely to occur between members who value quality and members who value efficiency concerning the allocation of resources or mission of the team – either producing high-quality products or a high amount of products. Software development is a complex process in which team members are interdependent, thus, good interaction such as communication, coordination, and mutual support among team members is important. High-value diversity, therefore, can have a

negative influence on team interaction. According to Jehn et al.(47), relationship conflicts among team members are likely to decrease when team members share the same values. Thus, Hackman (49), on the other hand, suggested that when team members share the same values and goals, interpersonal relationships are enhanced. Low-value diversity, therefore, enhances teamwork. Not only it influences teamwork, value diversity also influences team performance. In addition, low-value diversity is required in order to be efficient, effective and to sustain a high moral within the team (47).

According to Hoegl et al. (3) argued that the effort of team members are important for teamwork quality. However, the effort is only one of the multiple facets team members might have shared expectations about. Value diversity regards the team goal and mission, which is of a higher order than effort. Hence, it is likely that team members will prioritize the task of the team and share the same ideas concerning work norms when they share the same mission or vision. Therefore, this study proposes to use a more encompassing measure of value diversity instead of the most specific measure of effort used by Hoegl et al. (3).

2.8 Project Performance

In order to identify the value of production and teamwork, it requires a measure of performance. An objective measure of team performance is being used in this study which makes it difficult to define success. The perception of project success may differ due to the different interests of the parties involved in the software development projects. Thus, what is perceived as a project success or failure to a person may not be same to others. For example, software development team members normally take the completion of the scope of the project as a success which differs from external stakeholders who normally assess the project performance in terms of time and costs measures (50). It is difficult to objectively measure team performance as the evaluation criteria differ across team members, team leaders, and stakeholders. Nevertheless, various performance measures can be found in the literature despite their subjectivity remains debatable. In fact, Hoegl et al. (3) emphasized the importance of various team performance ratings which may come from both internal and external sources in order to improve objectivity.

Software development has been measured in multiple ways. Nevertheless, most of the time the emphasis is on the team performance which is the ability of a software development team to accomplish the desired level of costs, time and product quality. Team performance is also being associated with the effectiveness and efficiency which determine a project success. Effectiveness refers to the degree to which a team meets the expectations of the quality of the outcome Hackman (13) while efficiency reflects the degree to which a team meets time and budget objectives (3). In software development, the efficiency which may be represented by project management influences the effectiveness which may be represented by the product quality. Nevertheless, a good project management may result in a poor product quality and vice versa.

It is crucial in any software development project to determine upfront the project requirements in terms of quality, time and costs, and assessment criteria in order to precisely define the success. Therefore, when a product in a software development project meets the anticipated level of quality and being released to the client within the agreed timeframe and cost estimates, it can be described as successful (50). Furthermore, it is also important that all parties involved are well informed of the goals and assessment criteria (51). Therefore, the project performance can be measured and assessed by the different parties involved.

2.9 Software Quality and its Risks

The primary goal of software developers is developing and producing quality systems that meet the user's requirements. "Software quality" is defined in terms of customer satisfaction. "Risk", on the other hand, refers to any potential threat to the delivery of a quality product. In order to meet the goal of quality software, software developers focus on particular risks including project and schedule slips, cost increases, technical and quality risks, the timeliness of the product, risks that the final product will not fit the business for which it was designed.

Projects are managed to focus on and to mitigate these risks. Risk tables, lists of risks categorized by type, probability and impact are some of the tools used to help identify and manage these risks. The checklist process is more or less similar to the process pilots will normally go through prior to taking off. People, as airline passengers, are given the comfort level knowing the fact that the pilots are going through such procedure. However, unlike pilots, developers may choose to ignore some risks. This is due to the fact that risk levels are determined based on the anticipated impact of the risk and its probability of occurring. Developers will only address risks that categorized above the limited specified levels.

2.10 Software Development Life Cycle (SDLC) and Risk

There are several models in software development that reduce planning risks. All of these models contain similar phases: develop a statement of the customer's desires - the requirements phase; design how to achieve those desires - the design phase; code and test what was designed - the implementation phase; and determine that the requirements are satisfied by the system developed - system testing phase. The system development life cycle (SDLC) refers to the ordering and content of the steps through these phases. Many SDLCs are linear and require the documented completion of a phase before going on to the next phase and are directed at the satisfaction of the customer's explicit requirements.

2.11 Lifecycle Measurement (LIME) and Stakeholders

The LIME (52) introduces a multidimensional analysis of quality and performance during software development. This model defines quality in terms of an economic dimension from the managers' viewpoint, the social dimension from users' viewpoint and technical dimension from developers' viewpoint. The managers are focusing particularly on the cost and schedule drivers while the developers are focusing on technical quality that has a different impact during each SDLC phase. This model also examines the role of a stakeholder in all phases of the project development. Although this method includes a consideration of a stakeholder in all phases of project development, the stakeholder is the user for whom quality is achieved by the satisfaction of the specified requirements.

Table 2.1: Comparison of Different Software Development Models.

Model / Features	Agile Model	Spiral Model	Waterfall Model	Iterative Model	Prototype Model
Requirement	Frequently Changed	Beginning	Beginning	Beginning	Frequently Changed
Understanding	Well Understood	Well Understood	Well Understood	Not Well Understood	Not Well Understood
Cost	Very High	Intermediate	Low	Low	High
Guarantee of Success	Very High	High	Low	High	Good
Resource Control	No	Yes	Yes	Yes	No
Cost Control	Yes	Yes	Yes	No	No
Simplicity	Complex	Intermediate	Simple	Intermediate	Simple
Risk Involvement	Null	Low	High	Null	Null
Expertise Required	Very High	High	High	High	Medium
Changes	Difficult	Easy	Difficult	Easy	Easy
Risk Analysis	Yes	Yes	Only at beginning	No Risk Analysis	No Risk Analysis
User Involvement	High	High	Only at beginning	Intermediate	High
Overlapping Phases	Yes	Yes	No	No	Yes
Flexibility	Highly Flexible	Flexible	Rigid	Less Flexible	High Flexible

Source: Prof. Dinesh ch. Jain & Ms. Shikha Maheshwari, 2012.

2.12 Requirement Specification

According to Molokken and Magne (53), requirement specifications are required just at the beginning of the waterfall, spiral, and iterative models. However, the requirement specifications are frequently changed during the development process for prototype and agile models.

2.13 Understanding Requirement

According to Boehm (54), pointed out that waterfall, spiral, and agile models require a good understanding of the requirements, unlike prototype and iterative models which do not require a good understanding of the requirements. Table 2.1 is a comparison of different software development models.

2.14 Cost

According to Abrahamsson et al.(55), data was obtained for a cost driver value of 'very high quality' (expressed as' very low number of errors' or 'errors threaten human life'). The data shows that waterfall and iterative models are used for projects that have low-cost requirements for software development. On the other hand, spiral model is suitable for projects with the intermediate cost while the prototype model is suitable for projects with cost more than waterfall and spiral models. Agile model, however, leads to very high cost.

2.15 Guarantee of Success

According to Deepshikha (56), as per the research work, the guarantee of success for software projects adopting the waterfall model is very low compared to the prototype model in which its guarantee of success is good. On the other hand, if the projects adopt spiral and iterative models, both have the intermediate guarantee of success between good and high. Nevertheless, out of all the five models covered in the research work, the agile model have a very high guarantee of success.

2.16 Resources Control

Based on the research work by Deepshikha (56), it was concluded that prototype and agile models do not have their control over resources, however, waterfall, spiral, and iterative models have control over resources.

2.17 Cost Control

Data was obtained for a cost driver value of 'range of development experience'. According to Deepshikha (56), the prototype and iterative models are the only models that do not have their cost control features, which make them inappropriate. The data values for the waterfall, spiral, and agile models are also supported by this research because they have cost control feature, which makes them better as compared to others since cost control factor is important for all software projects.

2.18 Simplicity

Data was obtained for a cost driver value of 'multi-skilled and experienced'. Therefore, the data indicated that the waterfall and prototype models are most suitable for projects in which simplicity is the main factor (55). The spiral and iterative models have limited impact because their outcomes were intermediate with regards to the simplicity factor while the agile model is unsuitable due to its complex nature. In addition, due to the complexity of the agile model, it requires more time and money to complete a software project.

2.19 Risk Involved

According to Deepshikha (56), the data indicated that the spiral model is the most suitable for projects because software projects adopting this model involve low risk, whereas waterfall model is unsuitable because the high risk involved in software projects.

2.20 Expertise Required

Data was obtained for a cost driver value of 'range of development experience'. Thus, the prototype model is the most appropriate in which only developers with a range of experience are available (55). The waterfall, spiral, and iterative models are slightly less suitable because they require developers with a high level of expertise, whereas the agile model is inappropriate because it requires personnel with very high expertise and experience. Due to the strong positive value of the prototype model, it may suggest that the developers, instead of managers, are performing objective setting and evaluation.

2.21 Changes Incorporated

Based on the research by Mayer et al. (35), it is observed that the prototype, spiral and iterative models are most suitable of all as they require fewer changes to be incorporated after the project is complete. Software projects may require more cost and time to update the changes should the model requires more changes during usage. On the other hand, the waterfall and agile models are totally inappropriate because if changes are required to be incorporated, they face many difficulties while incorporating them into the software project.

2.22 Risk Analysis

According to Deepshikha (56), the data was obtained for a cost driver value of 'risk involvement (expressed as 'complex, difficult or challenging to implement' or 'very complex or novel algorithm'). Based on the data obtained, it showed that the risk was only involved at the beginning for waterfall model while the prototype and iterative models did not involve any risk analysis when being used in any software project. The spiral and agile models, on the other hand, did have the risk analysis when being used in any software project.

2.23 User Involvement

According to Chan and Leung (57), it was observed from the data obtained that the waterfall model has a very less user involvement because user involvement was required only at the beginning of the project. The iterative model needed an intermediate user involvement, whereas spiral and agile models required a high user involvement as part of their requirements.

2.24 Overlapping Phases

According to Deepshikha (56), it was observed that the waterfall and iterative models had no overlapping phases while the prototype, spiral, and agile models did have overlapping phases.

2.25 Flexibility

According to Deepshikha (56), the data was obtained for a cost driver value of 'range of flexibility'. The data showed that the agile and prototype models were highly flexible and were most appropriate while the spiral model was less flexible than the agile and prototype models. The waterfall model was the most rigid compared to the other models.

3. Methodology

Intellectual Notes Lecture System (INOLES) is being developed for the purpose of research sampling in implementing a new system development approach. The aim is to introduce a new system that is more effective and efficient in completing the tasks throughout the system development cycle. In addition, this study intends to analyze the quality of skills in a teamwork in system development projects.

The quality of skills in a teamwork has a significant impact throughout the system development Sommerville (58).In ASDF method, it provides a different perspective of teamwork that is easy to be practiced and understood in the system development environment especially in the third and fourth stages of ASDF method that are the module integration and database implementation. This will help to improve the quality of skills in teamwork.

Therefore, this research is proposing a new method called ASDF model that is created to solve problems in system development. One may take the system approach as an organized way of dealing with problems. In this dynamic world, the system life cycle is an organizational process of developing and maintaining systems. It helps in establishing a system project plan because it gives an overall list of processes and sub-processes required for system development.

ASDF model is a new proposed approach for this research. In fact, means a combination of various activities. It is intended for solving problems in an extended project timeline teamwork and a decrease of a high team collaboration and understanding in system development. There are six steps in implementing the system development process. The ASDF model can solve the problem in estimating the timeline and increasing a high team collaboration and understanding in system development.

This chapter will describe the qualitative methods used for obtaining the research data. The data gathered for this case study are based on 10 thematic questionnaires on a new proposed approach of system development life cycle (SDLC) model known as Analysis System Development Framework (ASDF). ASDF is being compared to an agile method through informal discussions and interviews with vendors, clients and stakeholders of system development projects as well as INOLES sampling. First, it will describe the case study through the use of the questionnaire in conducting a research. The aim of this study is to focus on developers in a public sector, private sector, statutory body as well as students to measure the quality of skills in teamwork. Second, it will present the methodology used to gather primary data that is thematic interviews through the use of a questionnaire. Third, it will describe the samples and respondents participating in this study as well as the interview recruitment techniques used. Lastly, the validity and credibility of thematic interviews through the use of a questionnaire as a qualitative research methodology are evaluated.

Being a new proposed method, ASDF allows projects to be developed more efficiently during the requirement process by having work tracks running in parallel rather than waiting for fully developed requirements that may have little interaction with the industry and little opportunities for mid-course correction. There are heaps of benefits compared to the agile way of requirement elicitation and elaboration. This new proposed ASDF method definitely marks a big change in the way solutions are designed and delivered in shorter time frames by various organizations. The comparison is shown in Table 3.1 below:

Table 3.1 Comparison of ASDF and Agile Models

Agile Model	ASDF Model
Fewer of Predictability In some cases, software deliverables, especially for a short time frame project of over three (3) months, the developer cannot quantify and extent it is difficult to assess the effort required at the beginning of the software development	**Precise of step in predictability** Every step in development, they can use for backward and forward to review to predict precisely. An adaptive teamwork which is able to respond to the changing requirements.
Deficiency of necessary documentation Lack of emphasis on necessary in user and system requirement, layout designing of system and documentation.	**Complete of necessary documentation** The teamwork does not have to invest time and effort, by the time they deliver the product, it is found that the client's requirement has changed. The documentation is crisp which result in time saving.
Plan timeframe an easily falls off track The project has the tendency to be off track if the client representative is not clear with what is expected to be the final outcome at the end of the project.	**Plan timeframe in track** Tracking the project in first action can help the teamwork in track for finished the development. Face to face communication and continuous inputs from the client representative leave no space for guesswork.
Grander demand on developer and clients Only senior programmers are capable of making the required decision during the development process. Hence there is no place for newbie programmers unless with the existence of experienced resources.	**Grander developer and client's effort** The end result is high-quality skills of teamwork in the least possible timeframe and satisfied client. Therefore, they get the great result and development to produce for user.
Extended time and commitment If the project needs the extended of timeframe, mean is the project is failed to finished on time. Therefore, the tester, developer, and committee of project needs to more commitment and extra time to teams commit for the project. This is can be effected to the time and energy to whom involved.	**Time of punctuality** The project development can handle the punctuality time to make sure the project on time deliver to the end user. This is to produce the time for effectively of quality in teamwork.

3.1 Structure of ASDF Method

A methodology is an important element of a system development life cycle. Thus, in developing INOLES sampling, the study is adopting ASDF method. ASDF method has six steps to be completed throughout the development cycle as illustrated in Figure 3.1.

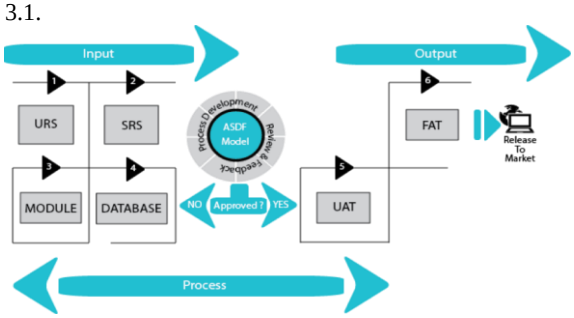

Figure 3.1: Diagram of Proposed ASDF Method

3.2 ASDF Method Workflow

In this research, the principles of such a framework are proposed, based on extensions to a structure known as the Unified Modeling Language (UML). Despite the exact scope of the roles and responsibilities may differ from one system development to another, System Analysis (SA) has been perceived by those involved in system developments as a specific set of responsibilities within a more general process of system transformation and change. Nevertheless, if SA is not appropriately positioned within a defined framework that guides the overall change process in an integrated

design, the value of SA work in system development may be deteriorated.

Certain aspects of a typical analysis change process have associated methodologies. For example, a system development may be guided by Waterfall, RUP, Agile, DSDM, SSADM or a host of derivatives and for project management, one might resort to methodologies like PRINCE, among others.

However, it looks like an over-arching framework that covers business change and transformation in a holistic way, right from the expression of a change initiative within a strategic change programme and ending in a delivered change accepted as "Fundamental Process of IT" (FPIT), with the benefits of the change duly evaluated does not exist until now. Thus, SA may be able to be positioned appropriately together with other roles and disciplines if the said framework exists.

From the perspective of business transformation and change, SA is perceived as an increasingly useful set of responsibilities. Thus, a methodology to define the tools, techniques, and activities that SA may get involved is expected to be in place. Unfortunately, that is not the case. This may be due to the fact that the scope, as well as the roles and responsibilities of SA, is still vague which requires an effort to clearly define them.

It is quite common for many System Analysts to simply adopt any methodology practiced by IT Suppliers or Consultants. Most of the time the methodology is known as software development methodologies or proprietary change workflow model. Although it may seem adequate to fulfill the requirement in their own field, however, in actual their needs may not be addressed adequately and accordingly.

Nevertheless, it is a wise move if a general workflow model to be put in place in order to control the process of system change prior to developing a methodology specifically to cater SA. Once the general workflow is in place, it is easier to identify which part of the workflow is relevant and under the purview of SA together with other disciplines involved. This approach, in a way, is in-line with software application development, for example, a Solution Architect is responsible for completing certain tasks together with other experts in order to develop or modify a software product. In fact, there is one or more Analysis Systems and hence, the change and transformation workflow model described in this chapter is known as Analysis Systems Development Framework (ASDF).

3.3 User Requirement Specification of ASDF Method

A user requirement specification (URS) by Sommerville (59) is the main document of a system development life cycle. This document is important and highly required Sommerville (59) for both business (investment protection) and regulatory reasons (defining intended purpose). Thus, every project should allocate ample time to work on this document especially in defining and writing testable requirements.

Both client and vendor should agree with what is stated in the URS as both parties need to clearly understand what the system is intended to do according to the client's requirements. In addition, since the URS is a living document, a change control procedure should be in place as the URS should be kept updated at all times throughout the development cycle. In fact, even after the development is completed, should there be any software upgrade the URS should concurrently be updated to keep up with the latest version of the software. The URS should emphasize on "what" rather than "how". Therefore, the focus should be on the functions to be carried out, the data on which the system will operate, and the operating environment rather than the implementation methodology.

There are several advantages of a well-written URS by Sommerville (59). It can help the user community to prevent them from being tempted by the technology and deceived by a poor system. This is due to the fact that they will select a system based on well-defined requirements as well as according to what they actually require the system to perform. In addition, by having a specific requirement, they can straight away focus on their specific requirements rather than the hassle of having to go through all the marketing promises. Moreover, by going through the URS, it helps to evaluate which system can meet the requirements by providing the required functionality.

According to Sommerville (59), in reality, it is not as simple as it sounds when it comes to differentiating various types of the requirement. A requirement that emphasizes on security, such as only authorized users may access the system, may appear to be a non-functional requirement. Therefore, when it comes to formulating the URS, this requirement should be described in more detail in order to make it a functional requirement, such as the need to include user authentication facilities in the system. Moreover, in order to clearly identify and define user requirements, it is important to look at them from clients' perspective which will result in a precise classification of functional or non-functional requirements. This is part of the criteria of a well-written URS apart from providing clear descriptions of the content and quality required by the clients.

3.3.1 Functional Requirements

When developing a system, it is important to identify and understand what clients want the system to perform and what objectives they want to achieve. The key is to specify what clients want rather than how to deliver the system to clients. Thus, by having the functional requirements, it helps developers understand the reason behind the tasks involved especially in terms of why clients want the system to perform certain tasks and whether there is any limitation or preference of certain tasks. They key of system functional requirements is that it should be complete and consistent in the sense that every single functionality required by the clients should be defined and described accordingly. This is due to the fact that functional system requirements differ from general requirements as they cover the tasks that should be performed by the system to specific requirements covering what the system should do to very specific requirements representing ways of doing things internally or done by the existing system (59).

3.3.2 Non-Functional Requirements

Non-functional requirements identify user categories such as whether they are professional or personal users and describe user characteristics such as prior knowledge and experiences. The special needs and subjective preferences of professional users such as journalists and editors and personal users such as news audience are also clearly defined. In addition, the users' environment in which the system will be used is also being described. Non-functional requirements also specify issues such as legal issues, intellectual property rights, security and privacy requirements.

Non-functional requirements may be related to system properties such as reliability, response time, and store occupancy. They may also describe the system implementation limitations such as the input and output devices capabilities or the data representations used in interfaces with other systems. Therefore, according to Sommerville (59), non-functional requirements do not focus on specific services delivered by the system to its users.

3.3.3 User Requirement Specification Capturing Methods

Different tools may be used to carry out the requirements identification and analysis process that include the followings:

1. Scenarios / Use Cases is a well-described realistic example of how users may perform certain tasks in a specified context of the future platform. Users provide their assistance to developers to gather and refine these examples. This is for the purpose of ensuring the intended use of the system is clearly demonstrated. In addition, it is also to exhibit the precise way users perform complete specific tasks.

2. User Survey is a questionnaire distributed to a sample population of users. The survey is normally used to gather data from a large sample of users. However, in certain circumstances, the data obtained may not be according to what they are anticipated. Nevertheless, surveys can be utilized to define needs, existing way of performing tasks, and perceptions towards the new system concepts.
3. Interviews are conducted based on questions that have been prepared prior to the interview sessions. Nevertheless, subject to responses from users, the interviewer may prompt users to further clarify their responses or relate to relevant issues that may not be addressed initially to the interview questions.

In this study, for the purpose of user requirement specification for INOLES sampling, a questionnaire to measure the quality of skills in teamwork in system development is being used. The questionnaire has the advantages of being more practical, enabling large amounts of information to be collected from a large number of people in a short period of time and in a relatively cost effective way, being able to be carried out by the researcher or by any number of people with limited effects on its validity and reliability. The results of the questionnaire can usually be quickly and easily quantified by either a researcher or through the use of a software package, and can be analysed more 'scientifically' and objectively than other forms of research.

3.4 System Requirement Specification of ASDF Method

The target audience for system requirements specification document is mainly system developers and users as the document defines the intended tasks the system is expected to perform and how the system will perform each function. The system developers treat this document as the authority to guide them in designing and developing system capabilities. On the other hand, the users will review the document to ensure it is comprehensive to accurately define and describe all the intended functionalities.

3.4.1 Hardware Requirements

Table 3.2: Summary of Hardware Requirement

Requirement	Description
1. Data Storage	Online Storage: The system should have the ability to store hundreds of megabytes of on-demand data. Nevertheless, the potential for requiring Gigabytes of data storage capacity should be taken into consideration. Further requirements gathering is required to get practical and reasonable estimates of such data storage requirements. Near-line Storage: All users and application data, as well as software installation and configuration files, must be fully backed up week-nightly. Furthermore, secure off-site storage will be performed on a weekly basis with 24 hour retrieval times.
2. Networking	Load Balancing: In order to ensure system stability and availability, application servers and database servers load balance exercise must be conducted at the application level. Specific scenarios, such as fail-over versus session-managed load balancing, should be addressed in the functional requirements specification. Private Network - The system requires a "back-end" private network environment in order to ensure that end-user operations (on the front-end) are not hindered by database backups, index propagations, or other large back-end data transfers. The system will use, where appropriate, the standard hardware and data communications resources provided by the ARS OCIO at the ARS George Washington Carver Centre in Beltsville, MD. This includes, but is not limited to, the general Ethernet network/T1 connection at the server/hosting site, network servers, and network management tools.

3.4.2 Software Requirements

Table 3.3: Summary of Software Requirement

Requirement	Description
1. Backup Software	Data and application backups will be managed through fully supported backup software solutions.
2. HTTP and GIS Server Applications	As the web will be the primary delivery protocol for the application, HTTP, and related GIS server applications will be required to support system functionality.
3. Web Browsers and Browser Plug-in	In support of External Interface requirements, commonly supported web browsers will be used to implement a thin-client architecture. The use of browser plug-ins will be judiciously restricted on an as-needed basis.
4. Email Services	As a secondary delivery protocol for alerts, data, and other information from the system, email server applications will be required to support system functionality.
5. Relational Database Management System	As the primary data storage mechanism for the corporate standard relational database management system, My SQL, and Firebase will be required to support system functionality.
6. Languages	The system will use, where appropriate, the standard software resources provided by the AngularJS. This includes, but is not limited to, MS ASP/Java Scripts, HTML, PHP, Firebase, and my SQL.
7. Database	The system will make data available in standardized data formats such as tab-delimited text for use on users' client systems including local installations of Firebase software or other off-line software applications.

3.4.3 Requirements Process

The purpose of the requirements elicitation process is to produce a requirements specification document that define and describe the formal requirements of the proposed system as specified by the stakeholders of the system. It is an engineering process to formulate a document that consists of the enterprise, software system functional, and software system non-functional requirements. The document is derived from an agreement achieved through constructive interactions among the various stakeholders of the proposed system. It is also to provide an adequate guidance to the developers to achieve a successful system in a time and resource effective manner.

This engineering process consists of "elicitation" of requirements through technical discussions, "specification" of requirements through textual and diagrammatic models, and "validation" of those requirements through confirmation of the models through discussions and presentations of those models. Broadly speaking, these requirements answer the why, what, and how of the proposed system across the community of stakeholders of the proposed system.

An adequate consensus is highly likely cannot be attained for all aspects of any non-trivial engineering effort due to the variety of interests and methods. In order to address any requirement conflict, it is expected that due diligence is put in place to explore the options available. Nevertheless, senior management needs to consult technical leadership on the remaining unresolved issues. This should be done prior to the completion of the requirements elicitation phase.

The management of various stakeholder organizations selects their respective representatives to participate in the requirements elicitation process. The representatives will focus on their respective organizational needs and wants in the requirements elicitation process. The representatives are categorized into four "world" that are subject, user, developer, and system representing. The subject world refers to the subject matter or domain experts of the scheduling and meeting organization. The product clients and ultimate users of the meeting scheduling system refer to the user world. On the other hand, the developer world refers to the software architects, designers, implementers, testers, and maintainers of the proposed system. Finally, the system world refers to the stated requirements of the proposed system.

3.4.4 Representatives

The selected representatives are assigned to teamwork in the requirements elicitation process as follows:

Table 3.4: Summary of Representatives

Stakeholder Organization	Categories
Teamwork A	Analysis System Development Framework (ASDF) method
Teamwork B	Agile method

3.4.5 Roles and Responsibilities

The "system world" representative play a role to provide automation capabilities as well as to represent the proposed system requirements of the requirements engineer. Providing capabilities concerning the viability and desirability of proposed requirements is the role assigned to the "developer world". The "developer world" is also responsible for ensuring the inclusion of system development industry standards and best practices.
The role of the "user world" representative is to provide expertise with regards to the user interface, intuitive operation, and overall usability of the proposed system. The "subject world" plays a role in providing meeting organization and coordination capabilities in addition to scheduling capability for the proposed system.

3.4.6 Outstanding Issues

Outstanding issues refer to those requirements or aspects of the proposed system which have not been adequately or satisfactorily resolved during the requirements elicitation process. A unique number is assigned to each issue. In addition, each issue contains a title, description, and contact information for the representative sponsoring the issue.
Senior management will consult technical leadership to resolve all outstanding issues. It is conducted through an assessment of impact, analysis of alternatives, or other method deemed appropriate. As part of the requirements specification, the method of analysis, disposition, and contact information of the person responsible for the decision is maintained.

3.4.7 Software System Functional Requirements

The purpose of INOLES is to support the organization of notes lecture. The system shall analyse the quality of each skill in a teamwork required to determine the effectiveness of the intended teamwork capability for system development based on ASDF method.

Table 3.5: Summary of Software System Functional Requirements

Software System Functional Requirements	Description
SFR-1 - PDF	The PDF file for INOLES shall be able to schedule a notes lecture for a list of students, lecturer, and administrator to upload and download the notes through the online system at any location.
SFR-2 - WORD	The WORD file for INOLES shall be able to schedule a notes lecture for a list of students, lecturer, and administrator to upload and download the notes through the online system at any location.
SFR-3 - EXCEL	The EXCEL file for INOLES shall be able to schedule a notes lecture for a list of students, lecturer, and administrator to upload and download the notes through the online system at any location.
SFR-4 - PPT	The PPT file for INOLES shall be able to schedule a notes lecture for a list of students, lecturer, and administrator to upload and download the notes through the online system at any location.
SFR-5 - VIDEO	The VIDEO file for INOLES shall be able to schedule a notes lecture for a list of students, lecturer, and administrator to upload and download the notes through the online system at any location.
SFR-6 - AUDIO	The AUDIO file for INOLES shall be able to schedule a notes lecture for a list of students, lecturer, and administrator to upload and download the notes through the online system at any location.

3.4.8 Software System Non-Functional Requirements

In order to be a good system, INOLES should ensure it has the ability to meet the powerful functional requirements. In addition, it should also pay attention to its non-functional requirements. This section describes the quality attributes that INOLES should have that are reliability, performance, user friendliness, flexibility, extensibility, and security.

3.5 Modules of ASDF Method

There is a need to effectively and efficiently address the various and diverse problems concerning the accessibility of interactive applications used in different contexts. Thus, the concept of user interfaces for all has been proposed (60) to provide the solution for this issue. However, at the moment, there is no development tool that has the capability to support the development of user interfaces for all. Only a core of the user interface can be developed using a unified user interface development platform. Nevertheless, special purpose user interface software tools can play the role to handle the platform and user-specific interface properties. The platform-specific issues can be automatically handled by these tools. In addition, these tools can also adapt the resulting dialogue to the particular user teamwork.
Almost all programming languages that are meant to be used for the real-world applications implementation provide some facilities for structuring programs as a network of smaller modules. In spite of it requires some initial development investment to structure a program in this fashion, it should be noted that, at the same time, it also gains several huge rewards.

3.6 Database of ASDF Method

The remarkably pervasive conversion of data processing to the unified database approach during the past decade has made research and implementation in database management systems the leading growth area in computer software. Keeping up with the rapid development provides a challenge to management, technical and research personnel in industry and in the university community.
System design phase emphasizes on the logical and physical designs. In fact, the model is a logical view of system design on the site. The analysis is performed to determine the information required, the process involved and functionality system information such as input, output, processing and databases. The analysis will be performed using tools such as object-oriented design flow chart, activity diagram, figure sequences, and the use case diagram shows the logical process and the relationships between data. Based on the logical design, the model is generated and later converted into a physical form.

3.7 User Acceptance Test of ASDF Method

User acceptance is a form of confirmation which is done through testing. It is to confirm that the delivered system meets all requirements, functions according to design parameters, and satisfies all business, technical, and management stakeholders. Planning for a user acceptance test starts in the Concept Development Phase. This is the point in which the acceptance criteria are defined which include user acceptance criteria that will be used to test the system during the Test Phase. User acceptance criteria are documented in the Project Scope Statement. The criteria are later updated upon completion of the Requirements Analysis Phase. The user acceptance criteria are utilized as part of the requirements traceability to guide the design, development, testing, and acceptance of a system.
Effective user acceptance criteria are based on functional and non-functional requirements, are specific, unambiguous, measurable, achievable, and realistic, contain specific pass or fail criteria, address all aspects of the system in detail, indicate if a requirement is critical or non-critical that is not required for acceptance, and identify unacceptable errors. In the Planning Phase, the Planning Team defines, schedules, and estimates the cost for a user acceptance test which will take place in the Test Phase. The test plan is progressively elaborated in the Requirements Analysis Phase. It includes the development of the Test Master Plan which addresses planned user acceptance test in detail.
The execution of user acceptance testing is most successful when it is predefined and the acceptance criteria have been approved. In addition, the test is performed by system users, real-world usage conditions are simulated in the test environment, the test is performed on a completed system that has gone through and passed unit testing, integration testing, and system testing, and test is conducted utilizing test cases that cover all scenarios. Test cases describe the functionality (scenario) being tested, input, expected

result, actual result, pass/fail status and rectification strategy for the problems discovered, date and time of test run, name of the person/role who ran the test, all data has been migrated/converted (if applicable) prior to user acceptance test, test case execution is automated with test scripts (when possible). A verification that the delivered system fulfills all the defined user acceptance criteria is an indication of a successful completion of a user acceptance test.

3. 8 Final Acceptance Test of ASDF Method

A final acceptance test is the final stage of system development prior to the project handover and project closure. This final stage emphasizes on running the test cases, checking for termination and comparing the test set result with the expected result. In addition, this is the stage in which the Test Log document is produced. The test log focuses on test cases in the sense that what test cases were run, in what order the test cases were run, who ran the test cases and whether each test passed or failed.

The level of formality and necessary documentation during testing varies from project to project. The client or the target audience involved in the project is an important factor that has an impact on the level of formality and documentation. For example, a less formality and documentation may be required for an internal project such as a system developed for use within the developing company compared to an external project in which specific standards are required. Generally, testing documentation is necessary because it could be used to verify and justify the adequacy of a given test suite. It could also be used to repeat test cases for the purpose of regression testing. A good testing documentation should be reviewable, repeatable and achievable. The rest of this chapter continues by describing each of the test levels.

3.8.1 Testing Levels

Each testing level is characterized an environment which includes a type of people, hardware, software, data, and interface. For example, programmers are mainly involved in unit testing while system testing, on the other hand, involves testers. In addition, unit and integration testing are conducted in the programmer's IDE (integrated development environment) while system testing is conducted on a system test machine or a simulated environment.

Each project has different environment variables subject to the type and nature of the project. It is the same for the number of testing levels. Alpha testing, beta testing, compatibility testing, stress testing and security testing are testing levels that can be taken into account other than those five testing levels mentioned above. It is a responsibility of a Test Manager to compile a Master Test Plan prior to system testing. The Master Test Plan should illustrate a detailed structure of how the other test plans would be executed. In addition, the Test Manager has the authority to decide on the number of testing levels of any particular project. Thus, it requires the Test Manager to understand thoroughly the requirements specification of the system. This is to help the Test Manager to avoid overlapping testing levels and also to avoid gaps between testing levels. The budget for a project, the complexity of a project, time scope for a project and staffing are some of the factors that may help the Test Manager to determine the number of testing levels required for a project.

3.8.2 System Testing

The purpose of system testing is to prove that the system implementation does not meet the system requirement specification. It takes place upon completion of integration testing. Test planning for system testing is usually one of the first to be administered. This is normally made available at the early stage of the project development lifecycle. A test team, if any, will be conducting the system test planning and system testing. A high-level design specification in the development process has a high influence on the system test planning phase. Thus, if the requirements specification and design specification are incorrectly translated, it would be very severe as it would propagate downwards to the lower levels of test and development.

According to Mayers (61), system testing is normally run on a system test machine and normally being conducted in a specially configured environment to simulate a realistic end user environment. Several testing categories should be taken into account when designing test cases for system testing. Nevertheless, not all testing categories are applicable to every project as it varies from one project to another.

3.8.2.1 Facility Testing

Facility testing may be conducted without a computer as it is sufficient to use an implemented functionality checklist. This testing is aimed to determine that all functionalities listed in the user requirement specification have been implemented accordingly.

3.8.2.2 Volume Testing

The program to be tested using volume testing is subjected to huge volumes of data in order to see how it copes. For example, an unreasonably large bitmap file to be edited is fed to a graphics application or source files with ridiculous amounts of code is fed to a compiler. This testing is aimed to prove that the system cannot handle the amount of data it has specified in its objectives.

3.8.2.3 Usability Testing

It is crucial to understand the target audience of a program such as the age, educational background and possibly the inclusion of accessibility features for the disabled should be taken into account. Moreover, it is also important to ensure the program produces meaningful, non-offensive messages. This testing is aimed to determine how easily a user would interact with the system or utilize the system.

3.8.2.4 Security Testing

The aim of this test is to attempt to break the program's security. Test cases would normally be designed to attempt to access resources that should not be accessible at all or without the required privileges. For example, an attempt to corrupt the data in a database system. Due to the increased number of e-commerce applications, an internet-based application is a good candidate for security testing.

3.8.2.5 Performance Testing

Performance testing is related to stress testing. The performance of a system is measured based on how fast it works under certain workloads. Performance testing determines if a system works as fast as the stipulated rate under certain conditions while stress testing, on the other hand, determines the performance of a system under extreme workload conditions.

3.8.2.6 Configuration Testing

Due to the advancement of technology, many software systems are now developed to function under multiple operating systems and different hardware configurations. Nevertheless, web applications are a special case in which a given web browser would operate differently on different operating systems due to the presence of different web browsers. Configuration testing, therefore, aims to determine how a system performs under different software and hardware configurations.

3.8.2.7 Compatibility Testing

Many programs are being developed to replace old and obsolete systems. This poses a challenge in which the new programs normally will have compatibility issues with the old ones. A requirement to replace an application that uses a database management

system as it back ends data store would be a good example. It is clear that the new application must be compatible with the old database. Thus, this testing is aimed to show that the system compatibility objectives are not met.

3.8.2.8 Install Ability Testing

This testing is designed mainly for systems with automated installation processes. The whole installation process could go wrong if any of the automated installations is not functioning accordingly. As a result, the system will not be fully functioning as some features are not successfully installed. End users only see a high level end result, thus, may not notice the possibility of system instability or missing features.

3.8.2.9 Reliability Testing

It is crucial for systems that are required to maintain certain uptime to conduct reliability testing. A database system or web hosting system which usually set an uptime of 99.97% over the system lifetime would be a good example for reliability testing. Mean time between failures (MTBF) would be another option to measure the reliability of a system.

3.8.2.10 Recovery Testing

Recovery testing is aimed to show that the recovery functions of a system do not work correctly. It is crucial for systems to have the ability to recover from different forms of failures. For example, database management systems and operating systems need the ability to recover from failures such as hardware failures, human data entry errors, and programming errors. While executing the test, all potential failures could be simulated or emulated. Minimizing the mean time to recovery (MTTR) is also part of the objectives of recovery testing in order to ensure that the system is back up and running as soon as possible after a failure.

3.8.2.11 Documentation Testing

Documentation testing is also part and parcel of system testing as the accuracy of the user documentation is also important in the system testing process. Test cases are usually designed by using the documentation. For example, the documentation is used as a guide to writing an actual stress test case once a stress case is identified. In order to have a clearer view of the system behaviour which would be the major focus in design system test cases, testers normally model the system under test as a finite state machine.

3.9 Case Study through Questionnaire Approach

Rather than treating it as a method, a case study research, according to Eriksson and Kovalainen (62), should be treated more as a research strategy. Moreover, case studies can also produce quantitative data in spite of its qualitative nature. However, case studies are not meant to produce statistical generalizations, such as experimental, quantitative and deductive research aims (62). Case studies aim to investigate the case concerning its organizational and social environment. Methodologically classic case studies are connected to the interpretative, ethnographic and field research studies (63). Case study research approach is opted as a research strategy when the purpose is addressing complex organizational, managerial or other business issues, and when the emphasis is trying to achieve a holistic and detailed understanding on the issue in question (62).

According to Eriksson and Kovalainen (62), made a distinction between intensive (classic) and extensive case study research. The intensive research focuses on finding out as much as possible on a small number of cases whereas the extensive design, on the other hand, aims at finding out common patterns and properties across cases. Moreover, intensive case study draws on the qualitative and ethnographic research traditions. It aims at developing an understanding of the case “from the inside” by listening to people that are involved in the case. The researcher plays the role of an interpreter of the multifaceted and rich phenomena and creates a narrative, a story out of the results (62). An extensive case study research, on the other hand, aims to test and develop theories by relying more on quantitative and positivist research. Thus, the main focus is to be seen as instruments in exploring certain business-related phenomena and developing theoretical propositions that could be tested and generalized to other contexts or to as a part of a theory rather than on detailed prescriptions of the “real life” cases (62).

This research falls somewhere in between intensive and extensive case study through questionnaire research. Although the aim is to understand the cases quite thoroughly from the perspective of individuals experiences and interpreting them (intensive case study approach), this research also aims to compare the results and map out similar or contradicting patterns based on existing research on agile software development as well as on the claims that the creator of ASDF method have presented (extensive case study approach).

In order to address the research question to understand the quality of skills in a teamwork based on ASDF method and analyse the possibility of having an impact on certain themes (URS, SRS, Module, Database, UAT, and FAT), this research opts to conduct thematic interviews through the use of a questionnaire.

3.10 Thematic Interview

Normally, based on their predetermined levels of organization and construction, qualitative interviews are categorized under three structural types that are structured, semi-structured, and unstructured. Under the structured category, the researcher determines the questions to be asked and the order in which the interviewer will ask them (64), and they are usually conducted in the form of a survey or a questionnaire. The more structured the interview, the closer the method gets to a quantitative study, providing a narrow platform for respondents to respond in their own words (64). The flexibility and spontaneity of collecting systematic data on variables from each respondent are compromised once the interviews are highly structured (64). Unstructured (open-ended) interviews are at the other end of the continuum, in which the interviewer does not use a pre-determined interview schedule and the interviewee usually leads the conversation.

Thematic interviews are semi-structured in their nature, which means they have the flexibility to adjust the wordings and sequence of questions during each interview although the interviews kick off with certain predetermined themes, topics, and issues. It gives room to the respondents to express themselves freely but at the same time the data obtained are still systematic (62, 64) pointed out that people are able to express complex feelings and thoughts through language, which would not be possible to understand through observation or questionnaires. This is an interesting notion concerning this research as well since it is not only aiming at getting an insight into people’s general perceptions and experiences but also understanding the underlying factors that lead to a certain kind of thinking. The format of the interviews is semi-structured, in a way, that this research used mostly pre-planned questions during the interview, but altered the wording according to the situation and prompted additional and inquisitive questions in order to engage the respondents and to gain more comprehensive data.

At the most basic level, interviews are conversations (65). His defines qualitative research interviews as "attempts to understand the world from the subjects' point of view, to unfold the meaning of peoples' experiences, to uncover their lived world prior to scientific explanations". Interviews for research or evaluation purposes differ in some important ways from other familiar kinds of interviews or conversations. Unlike conversations in daily life, which are usually reciprocal exchanges, professional interviews involve an interviewer who is in charge of structuring and direct-

ing the questioning. In some professional interview situations, such as job interviews or legal interrogations, the power of the questioner is much greater than the power of the one being questioned. Therapeutic or clinical interviews are another special kind of professional interview, in which the purpose is to increase understanding and produce a change in the person being interviewed. While interviews for research or evaluation purposes may also promote understanding and change, the emphasis is on intellectual understanding rather than on producing personal change (65).

After presenting a brief overview of the complexity of the qualitative interviewing process used some of the major ideas that psychotherapy researchers using such interviews must consider both before and during the interview process. They then offer thoughts regarding approaches to strengthen qualitative interviews themselves. Therefore, much of qualitative psychotherapy research relies on spoken interviews with participants to gather detailed information regarding the phenomenon under examination (66). In an activity that calls for not only strong interviewing techniques but also the very skills used when working with clients, interviewers confront challenges inherent in both domains, "How do they conduct an incisive interview and simultaneously helping participants to feel safe enough to explore in depth often difficult experiences with a relative stranger?". Perhaps complicating this process, qualitative psychotherapy researchers also must attend to the ethics of interviewing. The ethics of interviewing are beyond the scope of the article, but interested readers are encouraged to see (67).

The first half of the interview questions, starting from Part I of the questionnaire, are more open and less guided than at the end. The objective was to let the respondents tell in their own words what they think is important in their opinion. The second half of the questions, starting with Part II until Part X of the questionnaire, are more structured and standardized, as this research aims at getting information on certain themes, such as to validate the quality of skills in a teamwork of user requirement specification (URS) based on ASDF method, to validate the quality of skills in a teamwork of system requirement specification (SRS) based on ASDF method, to validate the quality of skills in a teamwork of module integration based on ASDF method, to validate the quality of skills in a teamwork of database implementation based on ASDF method, to validate the quality of skills in a teamwork of user acceptance testing (UAT) based on ASDF method concerning the interface, programming syntax, module and database before handing over to end users, to validate the quality of skills in a teamwork of final acceptance testing (FAT) based on ASDF method concerning the final interface, programming syntax, module and database after handing over to end users, to validate the quality of skills in a teamwork concerning the effectiveness of ASDF method, to validate the quality of skills in a teamwork concerning the efficiency of ASDF method, and overall quality of skills in a teamwork based on ASDF method. At the end of the interview, interviewees are asked to sum up their feelings towards ASDF method and to add any essential points concerning their perceptions towards ASDF method.

According to Silverman (68), interview questions should be formulated based on three different categories that are positivist, emotionalist and constructionist. The emotionalist approach, which is being used in this research, focuses on understanding respondents' authentic experiences on a given issue. In other words, rather than focusing on the hard facts, the interview questions focus more on people's perceptions, conceptions, understandings, viewpoints and emotions (62). Open-ended interview questions, compared to the closed ones, give more rooms for respondents to express their thoughts and give them more control on the issues being discussed (62). In general, many open-ended questions tend to start with "why" or "how" and they need to be structured in such a way that respondents need to provide answers more than just "yes" or "no". In this research, after the first four background questions, most of the questions are formulated in this way. It is an important part of interviewing to seek to understand through active listening skills. The interviewer should be able to prompt respondents for clarity on what is being said throughout the interview, and reflect on what they are saying (69). This technique is being adopted during the whole interview by asking additional questions, for example, "What do you feel of this newly developed system?" in order to seek clarity and understand the holistically people's experiences. When conducting a qualitative interview, it is highly recommended that the responses of respondents are questionnaire survey, audio-recorded and transcribed. All the interviews in this research were recorded and the data obtained were transcribed word by word, which resulted in more than a hundred pages of data to be analysed.

3.11 Sample

In thematic interviews for qualitative studies, since the objective is not to simplify the data obtained beyond the respondents of the research study, non-probability sampling is often opted as a recruitment strategy for the interviewees. In probability sampling techniques, a probability for the sample to represent a larger population can be calculated. Non-probability sampling, on the other hand, does not meet this criterion, and the sample may or may not represent a larger population. Nevertheless, determining the important characteristics of respondents is important and the sample should be formed according to these (64). In this study, the researcher wanted to obtain diverse opinions and standpoints, and especially people who have contrasting viewpoints on ASDF method. It was done by asking the interviewees if they know anyone who is against ASDF method within the teams. The respondents did not know any of these kinds of people, apart from Teamwork B. The respondent in Teamwork B pointed out two people, who supposedly were a bit resistant towards ASDF method, of which the researcher interviewed one of them (Developer, Teamwork B). In addition, the researcher wanted to see if respondents' opinions varied depending on their experience on ASDF method compared to the agile method. For example, the amount of experience they had on applying the agile method as well as ASDF method. Therefore, the researcher opted to interview respondents from teamwork that have adopted ASDF method (Teamwork A and B). The samples of this study comprised of 222 and 116 respondents for the ASDF and agile methods respectively based on the feedback received during the interview through the use of a questionnaire across two different groups of respondents.

In Teamwork A, the new proposed ASDF method has been adopted quite recently and people have been working in ASDF mode for less than a year, apart from the test interviewees who were interviewed only a few weeks after the new proposed ASDF method transformation process was completed. The Teamwork A comprised of people with 1 – 8 years, and the average being 3 – 7 years' experience of working in the agile mode in the public and private sector. Thus, after reviewing the new environment of development, which is based on ASDF method, this method is much better and completed the development requirements for less than three months and above for working development rules.

In Teamwork B, in general, the people have been working a bit less in the public sector, with the average being 3 – 7 years and the experience varies from 1 to 8 years inside the public sector. Therefore, when comparing the agile method to the new proposed ASDF method concerning the quality of skills in a teamwork and the development work progresses smoothly in between three months and above. The sample comprised of both sexes with females being the minority. In order to protect people's anonymity, every teamwork will be referred as "me", and people's opinions will be implied as their assigned number and role.

Here is the sample according to people's roles and the assigned number:

Teamwork A – Analysis System Development Framework (ASDF) method

Teamwork B – Agile method

In addition to acquiring data from the interviews with software development teams, the researcher also conducted informal interviews and conversations with other stakeholders, who were somehow connected to the projects of pursuing agile development in public and private sectors or to the teams that were the focus of this research. Hence, the introduction of the new proposed ASDF method development. The communication took place via face-to-face, telephones, and emails. The people whom the researcher communicated with are those who being as a source to explain the situational background information on the agile and new proposed ASDF methods. Therefore, Software Quality Manager, System Analyst, Developer and Programmer who has an insight into the situation in the organization that Teamwork A and B operate in, and External Consultant who was part of executing the change process in Teamwork A and B will remain anonymous in order to protect the anonymity of the people within Teamwork A and B.

The researcher also has the access to the intranet of public and private sectors, in which textual and visual materials (PowerPoint presentation) on the new proposed ASDF method development are obtained, particularly those concerning the new proposed ASDF method development process of Teamwork A and B. The materials were used in order to get some background data on the agile development in public and private sectors, as well as to understand the motivational factors for initiating the change process development environment.

Table 3.6: Demographic Information of Sampling

Teamwork A	Experience in Development System	Sector		Designation	No.	Gender	
		Public	Private			Female	Male
New proposed ASDF Method	1 – 8 years	22	3	Software Quality Manager	32	94	128
				System Analyst	57		
				Developer	49		
				Programmer	43		
				External Consultant	41		
Teamwork B	**Experience in Development System**	**Sector**		**Designation**	**No.**	**Gender**	
		Public	**Private**			**Female**	**Male**
Agile Method	1 – 8 years	22	3	Software Quality Manager	18	29	87
				System Analyst	34		
				Developer	37		
				Programmer	17		
				External Consultant	10		

3.12 Validity and Credibility

The objective of thematic interviewing for qualitative research is to bring light to the quality of skills in a teamwork. Thus, the notion of validity as applied in quantitative research has a very little impact on qualitative research practices (64). The aim is not to transfer the knowledge of similar contexts rather than creating generalizable study findings or external validity(64). This is applicable to this study as it tries to complement previous research findings on the agile software development method at public sector to the new software development environment using ASDF method.

According to Goodman (64), a qualitative study aims for rigorous trustworthiness and credibility. The methodology used to obtain data is found to be the most common threat to achieving the trustworthiness and credibility. The outcome of thematic interviews may be misrepresented due to the reluctance of interviewee to share the information or the inability of the interviewer to be neutral in which the interview may be interrupted with the interviewer personal viewpoint (64). One way to mitigate the latter is by using self-monitoring activities during the interview which includes an open-ended, neutral and clear questionnaire. The researcher has applied this in this study when designing the questionnaire to ensure there are no "either-or" options. For example, instead of "has your UAT level increased or decreased", the questionnaire uses "ASDF method is very easy to comprehend in the identification process applied in system development". Another aspect that needs to be taken into account is to have a good rapport and trust with the respondents. This is to ensure that they feel comfortable in revealing personal information about themselves. The researcher built the trust by assuring the interviewees that they can remain anonymous so that they feel free to speak their mind.

In addition, the interviewing situation itself may also have an impact on respondents. They may not be able to express themselves naturally if the situation itself is unnatural such as being intimidating or weird. Moreover, the interviewers should also realize that their voice, body language and facial expressions also have an impact on the relationship, and they are responsible, as much as possible, to ensure the situation is comfortable.

To summarize, the researcher who conducts the interviews him/herself is more involved in the research process than when using other business research methods. Therefore, the data obtained from the interview can be seen (more or less) as jointly constructed by the interviewee and the interviewer. In addition, the researcher can choose which aspects of the interview to be extracted and presented in the report. In this study, the researcher wanted to present all the perspectives that found to be essential concerning the development of ASDF method. Even though the interviewees seemed to have a very positive attitude towards ASDF method, the researcher would like to include debatable and contrasting viewpoints and issues in the report as well. The different viewpoints may help to provide some kind of understanding on why certain people might not enjoy working in ASDF mode or what are the shortcomings that people perceive in the development of ASDF method.

3.13 Summary of the Study

The structure of a system development teamwork will become broader and more complex due to increasing enrolment and growth in line with current technology advancements. The new proposed ASDF method is one way to facilitate the existing system analysis solution process to be able to detect any problem in improving the quality of skills in a teamwork.

This chapter concludes the achievement of the research objectives, the results of the study conducted, and the recommendations for future use of the new proposed ASDF method.

This chapter will summarize the conclusions of this study concerning the research questions presented in Chapter 1. In addition, it will also analyze how these conclusions were derived from previous research on agility, particularly related to the study conducted. The conclusions will also include the implications and recommendations based on the results obtained that could be taken into account while adopting the new proposed ASDF method to be practiced in a new system development environment.

Previous research have indicated that the quality of system development is, to a certain extent, depending on a good teamwork. It is, therefore, important to understand the factors of software development teams that have a significant impact on the quality of skills in a teamwork since the success rate of software development projects is low (70). Based on general system development models and enquiries on the quality of skills in a teamwork, a new proposed system development model known as Analysis System Development Framework (ASDF) is developed and evaluated which comprises of six steps that are user requirement system (URS), system requirement specification (SRS), module, database, user acceptance test (UAT) and final acceptance test (FAT).

This study contributes mainly to the findings, more than previous research have found, that significantly support the perception that

a high quality of skills in a teamwork results in better performance of software development. This, in turn, converts a higher success rate for software development projects. The results of the study show that the p-value of all samples is <0.05 which indicates that all sample distributions are non-normal. Hence, the study uses non-parametric test (Mann-Whitney) for data analysis.

3.14 Implications

The study has the implications towards the practitioner's perceptions and experiences concerning agile software development method in a large company applying ASDF method as their main methodology. Although it has been debated whether ASDF method can be successfully implemented in large companies, this study has successfully proved the opposite. This study, however, has a limitation in which the number of respondents (222 and 116 in Teamwork A and Teamwork B respectively) is rather small representing only two teamwork known as Teamwork A and B. Nevertheless, all the 222 and 116 respondents who experienced the new proposed ASDF method were of the opinion that the adoption of agile methods has changed for the better.

3.15 Research Limitations

Based on the respondents' responses, there are some manageable limitations that could be considered for future system development especially when it comes to using the new proposed ASDF method in a new environment of the industry. After going through the training and when the team members feel that they are actively involved in the change process, it is noticed that their attitudes towards agile practices are improved as they understand better the benefits. The study by stated that the cohesion of agile teams will not exist if team members are not involved in close communication in which it supports the findings of this study. Thus, it is very important to enable open communication from the very beginning. In addition, this is also applicable to training sessions and available support from agile coaches throughout the change process. Teamwork A felt that this, in some ways, contributes to the successful transformation of the new proposed ASDF.

It should be emphasized that this study has a few limitations. First, the data for this study are obtained from 22 Malaysian industries and three multinational software development teams. Thus, the scope of the study is quite limited. Due to the limitation, it is not that easy to claim the outcome of this study can be replicated in a bigger population. However, the outcome at the team member level contributes to the outcome at the team level. This somehow opens a room for the outcome of this study to be applied to a bigger population. Nevertheless, this is only applicable to the new proposed ASDF method part. Software development teams need to work on complex tasks, in a constantly changing environment (71). In order to work effectively and efficiently, a good teamwork collaboration is highly required. Other factors might be important for performance, however, quality in a teamwork may not be crucial if the tasks for the team are less complex. The new proposed ASDF method may also be applicable to other domains, for example, engineering development. However, minor changes may be required in order to make it more applicable to other domains.

Second, this study could not provide a clear association with the relationship between the new proposed ASDF method and agile method. A more comprehensive study may be able to provide a definite information on the association with the relationship as well as a clear description of perceptions of teamwork and performance.

Third, this study is lacked objective assessment of team performance. Thus, this study tries to make the assessment more objective by using performance assessment from different perspectives. Future research may take this into account in order to ensure team performance is objectively measured. Moreover, due to the absence of objective assessment in this study, it may be one of the factors that contribute to the high differences in the findings of this study. This study is focusing on the quality of skills in a teamwork, thus, does not include other aspects such as development method and complexity to be part of its discussion. That is the reason why the main bulk of the new proposed ASDF method only focus on describing the differences concerning the quality of skills in a teamwork. Future research may revisit the outcome of this study in a broader perspective by including those factors that are not discussed in this study due to the scope constraint.

It is observed that, in multiple teams, team members tend to have different perspectives concerning the development methodology based on the new proposed ASDF method. Thus, the measures concerning the quality of skills in a teamwork based on the new proposed ASDF method in this study open a room for further enquiries. The findings of this study may be due to the outcome of the respondents might not have a good understanding of the principles of the development methodology. It might be a case too that the respondents may not understand the overall application in terms of function points. There is also a possibility that the function points may not be a common measure for the quality of skills in a teamwork.

Finally, future research should further investigate the performance efficacy and the effectiveness of the quality of skills in a teamwork as a continuation of this study since this study has provided the experimental evidence of the new proposed ASDF method influence on software development. Future research should provide the answer on what system analysts can do to enhance the quality of skills in a teamwork in which at the same time will improve the work quality.

3.16 Future Work

Although the findings of this study revolve around the new proposed ASDF method as a whole, some recommendations for future research have been provided to enhance the findings. The researcher plans to conduct a comprehensive survey to identify issues and challenges experienced in software development projects based on the new proposed ASDF method.

Another interesting research goal to be considered is to identify human and social factors and to investigate their impacts on the success or failure of software development projects based on the new proposed ASDF method. This study covers only the new proposed ASDF method, whereas the tools of this method are not included in the scope of this study. Thus, to further extend the current context, another possible research topic could be to study the tools used to support the new proposed ASDF method.

3.17 Conclusion

The present research of new proposed ASDF method is accomplished. Regarding objectives for system development are concluded to verify the main problem of extended project timeline teamwork and the decrease of a high team collaboration and understanding in system development are proved and accomplished. Therefore, concluded for determine the project development is on tie in oder to control and increase the quality of skills in a teamwork collaboration and understanding in system development.

The present survey found TWQ and team performance to be highly related when team members rated these two concepts. Furthermore, the correlation between TWQ and team members' success - their work satisfaction and learning - unity. One interpretation is that the team members consider TWQ and team members' success as indistinguishable concepts.

The team leaders' perception of team performance had a medium correlation with TWQ. In contrast, no effect of TWQ on team performance . The effect of TWQ on team performance was higher for product quality (in particular regarding team members and team leaders) than for project quality. Despite the emphasis on TWQ in the agile community, in the traditional and the agile surveys alike, both the evaluation of TWQ itself and its effect on team performance and team members' success were similar.

However, the agile survey showed lower agreement among the raters regarding evaluation of team performance than was the case in the traditional survey. An implication of this survey is that the quality of teamwork is a major factor in improving team performance, especially for improving the quality of the team's product. Note that when trying to optimize team performance, one needs consensus of whose view of team performance should be considered. For the future, we recommend that more research efforts be made to validate the TWQ construct and to advance the measurement of team performance.

References

[1] H. Ingram, " Lin King Teamwork with Performance ", Journal of Team Performance Management, Vol.2, No.4, 2000, pp.5-10.

[2] C. W. Langfred, "The Paradox of Self-Management: Individual and Group Autonomy in Work Groups. Journal of Organizational Behavior, 21, 2000, pp. 563 – 585.

[3] Hoegl, and Gemuenden, Liang et al., Henderson and Lees, "The Influence of Teamwork Quality on Software development Team Performance", 2001, 2012, 1992.

[4] M. A. Marks, J. E. Mathieu, and S. J. Zaccaro, E. S. Salas, and C.S. Burke, "A Temporally Based Framework and Taxonomy of Team Processes, Academy of Management Review, Vol. 26 No.3, 2001, 2005, pp. 356 - 376.

[5] E. Salas, D. E. Sims, and C. S. Burke, "Is There A Big Five in Teamwork?, Small Group Research,Vol. 36, 2005, pp. 555–599.

[6] T. L. Dickinson, and R. M. McIntyre,"A Conceptual Framework of Teamwork Measurement," in Team Performance Assessment and Measurement: Theory, Methods, and Applications", M.T. Brannick, E. Salas and C. Prince (eds.). NJ: Psychology Press, 1997, pp. 19–43.

[7] R. E. Kraut, and L. A. Streeter, "Coordination in Software Development", Communications of The ACM, Vol. 38,1995, pp. 69-81.

[8] J. R. Katzenbach, and D.K. Smith, "The Discipline of Teams Harvard Business Review", March-April, Vol. 7, No. 2,1993, pp. 111-120.

[9] J. Rodd, "Leadership in Early Childhood (4th Edition) Maidenhead", Open University Press, 2012.

[10] T. Dingsoyr, S. Nerur, V. Balijepally, and N. B. Moe. Agile Software Development: An Introduction and Overview. Ed: Springer Berlin Heidelberg",2010, pp. 1-13.

[11] T. Dyba, and T. Dingsoyr, "Empirical Studies of Agile Software Development: A Systemati Review. Information and Software Technology, Vol. 50, No. 9,2008, pp. 833-859.

[12] V. Gulliksen Stray, N. B. Moe, and T. Dingsoyr, "Challenges to Teamwork: A Multiple Case Study of Two Agile Teams", A. Sillitti, O. Hazzan, E. Bache, X. Albaladejo, (eds.) XP, LNBIP, Springer, Heidelberg vol. 77, 2011, pp.146–161.

[13] J. R. Hackman, "The Design of Work Teams", In J. Lorsch, eds., Handbook of Organizational Behavior," New York: Prentice Hall. 1987, pp. 315-342.

[14] R. Katz, and T. J. Allen, "Investigating The Not Invented Here (NIH) Syndrome: A look at The Performance, tenure, and communication Patterns of 50 R&D Project Groups. In M. L. Tushman, W.L. Moore, eds., Readings in The Management of Innovations, Cambridge, MA: Ballinger Publishing Company,1988, pp. 293-309.

[15] A. Griffin, and J. R. Hauser, "Patterns of Communication Among Marketing, Engineering and Manufacturing: A Comparison Between Two New Product Development Teams. Management Science, Vol. 38, No. 3, 1992, pp. 360-373.

[16] M. B. Pinto, and J. K. Pinto, 1990, "Project Team Communication and Cross-Functional Cooperation in New Program Development. Journal of Product Innovation Management, Vol. 7, No. 3, pp. 200-212.

[17] Lu, Y. Xiang, C. Wang, B. & W. Xiaopeng,"What Affects Information Systems Development Team Performance? An Exploratory Study from The Perspective of Combined Socio-Technical Theory and Coordination Theory", Computers in Human Behavior, Vol. 27, No. 2, 2010, pp. 811- 822.

[18] D. L. Gladstein, "Groups in Context: A Model of Task Group Effectiveness", Administration Science Quarterly, Vol. 29, 1984, pp. 499–517.

[19] E. Salas, and J. A. Cannon-Bowers,"Methods, Tools, and Strategies For Team Training", In M. A. Quinones and A. Ehrenstein, eds., Training for a Rapidly Changing Workplace: Applications of Psychological Research Washington, DC, American Psychological Association, 1997, pp. 249 - 279.

[20] S. L. Jarvenpaa, and D. E. Leidner, "Communication and Trust in Global Virtual Teams", Journal of Computer-Mediated Communication, Vol. 3 No. 4, 2006.

[21] S. Faraj, and L. Sproull, "Coordinating Expertise In Software Development Teams. Management Science, Vol. 46, No.12, 2000, pp. 1554 - 1568.

[22] J. He, B. S. Butler, and W. R. King, "Team Cognition: Development and Evolution In Software Project Teams. Journal of Management Information Systems, Vol. 24, No. 2, 2007, pp. 261- 292.

[23] V. B. Hinsz, R. S. Tindale, and D. A. Vollrath, The Emergent Conceptualization of Groups as Information Processors, Psychology Bulletin, Vol. 121,1997, pp. 43-64.

[24] J. Murnighan, and D. Conlon, "The Dynamics of Intense Work Groups: A Study of British String Quartets", Vol. 36, June 06, 1991. Graduate School of Management, University of California, Irvine,1992.

I. Nonaka, and H. Takeuchi, "The Knowledge Creating Company: How Japanese Companies Create The Dynamics Of Innovation". Vol. 29, January 01, 1991.

[25] K. H. Roberts, and C. A. O'Reilly, III. 1974. Failures in Upward Communication in Organizations: Three Possible Culprits, Academy of Management Journal, Vol. 17, No. 2, pp. 205-215.

[26] K. Crowston, and E. E. Kammerer, "Coordination and Collective Mind In Software Requirements Development", IBM Systems Journal, Vol. 37, No. 2,1998, pp. 227 - 245.

[27] K. A. Bollen, and R. H. Hoyle, "Perceived Cohesion : A Conceptual and Empirical Exandnation", Social Forces, Vol. 69, No. 2,1990, pp. 479 - 504.

[28] B. Mullen, and C. Copper, "The Relation Between Group Cohesiveness and Performance: An Integration", Psychological Bulletin,Vol. 115, 1994, 210 - 217.

A. V. Carron, W. N. Widmeyer, and L. R. Brawley, "The Development of An Instrument to Assess Cohesion In Sport Teams: The Group Environment Questionnaire, Journal of Sport Psychology, Vol. 7, No. 3,1985, pp. 244-266.

[29] S. M. Gully, D. J. Devine, and D. J. Whitney, "A Meta-Analysis of Cohesion and Performance: Effects of Level of Analysis and Task Interdependence, Small Group Research, Vol. 26, No. 4, 1995, pp. 497-520.

[30] A. Chang, and P. Bordia, "A Multidimensional Approach to The Group Cohesion Group Performance Relationship. Small Group Research, Vol. 32, No. 4, 2001, pp. 379-405.

[31] F. Friedlander, "The Primacy of Trust As A Facilitator of Further Group Accomplishment", Journal of Applied Behavioral Science, Vol. 6, No. 4, 1970, pp. 387- 400.

[32] J. L. Pearce, S. M. Sommer, A. Morris, and M. Frideger, "A Configurational Approach To Interpersonal Relations: Profiles of Workplace Social Relations and Task Interdependence.

[33] R. C. Mayer, J. H. Davis, and F. D. Schoorman, "An Integrative Model of Organizational Trust. The Academy of Management Review", Vol. 20, No. 3, 1995, pp. 709 - 734.

[34] K. T. Dirks, "The Effects of Interpersonal Trust on Work Group Performance Published in : Journal of Applied Psychology", Vol. 84,1999, pp. 445 - 455.

A. K. Mishra, "Organizational Responses To Crisis: The Role of Mutual Trust and Top Management Teams, Unpublished Dissertation. Ann Arbor, MI: The University of Michigan School of Business Administration, 1992.

[35] R. W. Boss, "Trust and Managerial Problem Solving Revisited. Group Organization Management, Vol. 3, No. 3, 1978, pp. 331-342.

[36] D. Zand, "Trust and Managerial Problem Solving", Administrative Science Quarterly, Vol. 17,1972, pp. 229-239.

[37] G. Jones, and J. George, "The Experience and Evolution of Trust: Implications for Cooperation and Teamwork, Academy of Management Review, Vol. 23, No. 3,1988, pp. 531-546.

[38] D. Bandow, "Time To Create Sound Teamwork Journal for Quality and Participation", Vol. 24,2001, pp. 41-47.

[39] R. Cooper, and A. Sawaf, "Executive EQ: Emotional Intelligence in Leadership and Organizations", New York, Grosset, Putnam, 1996.

[40] W. E. D. Creed, and R. E. Miles, "Trust In Organizations: A Conceptual Framework Linking Organizational Forms, Managerial Philosophies, and The Opportunity Costs of Controls", In R. M. (1996).

[41] C. Handy, "Trust and The Virtual Organization", Harvard Business Review, Vol. 73, No. 3,1995, pp. 40–50.

[42] D. Tjosvold, and M. M. Tjosvold, "Cross Functional Teamwork: The Challenge of Involving Professionals, In M. M. Beyerlein, D.

A. Johnson, and S. T. Beyerlein, eds., Advances in Interdisciplinary Studies of Work Teams. Vol. 2. Greenwich, CT: JAI Press, 1995, pp. 1-34.
[43] R. A. Cooke, and J. L. Szumal, "The Impact of Group Interaction Styles on Problem-Solving Effectiveness", Journal of Applied Behavioral Science, Vol. 30, No. 4,1994, pp. 415 - 437.
[44] K. A. Jehn, G. B. Northcraft, and M. A. Neale, "Why Differences Make A Difference: A Field Study of Diversity, Conflict, and Performance In Workgroups, Administrative Science Quarterly, Vol. 44, No. 4,1999, pp. 741-763.
[45] J. E. McGrath, J. L. Berdahl, and H. Arrow, "No One Has It But All Groups Do: Diversity As A Collective, Complex, And Dynamic Property of Groups. In Susan E. Jackson, N. Marian, and Ruderman , eds., Diversity in Work Teams: Research Paradigms for a Changing World", Washington, DC: APA Publications. Model For The Agile Entreprise. Whitepaper, Leffingwell, LLC, 1996, pp. 42 - 66.
[46] J. R. Hackman, "Groups That Work And Those That Don't", San Francisco, Jossey Bass,1990.
[47] N. Agarwal, and U. Rathod, "Defining Success For Software Projects: An Exploratory Revelation, International Journal of Project Management, Vol. 24, No. 4, 2006, pp. 358 - 370.
[48] J. Wateridge, "IT projects: A Basis For Success, International Journal of Project Management, Vol. 13, No. 3, 1995, pp. 169-172.
[49] L. Buglione, "A Quality Factor For Software", European Software Institute Parque Technologico De Zamudio #024 E-48170 Zamudio, Bizkaia, Spain, 1999.
[50] O. K. Molokken, and J. Magne, "A Comparison of Software Project Overruns Flexible Versus Sequential Development Models", Published in:IEEE Press Piscataway, NJ, USA, Journal IEEE Transactions on Software Engineering, Vol. 31, No.9, pp. 754-766.
[51] B. W. Boehm, "A Spiral Model of Software Development and Enhancement", TRW Defense Systems Group, Published in : IEEE Computer, Vol. 21, No. 5,1988, pp. 61-72.
[52] P. Abrahamsson, O. Solo, J. Ronkainen,, and J.Warsta, "Agile Software Development Methods, VTT Technical Research Centre of Finland, 2002.
[53] J. Deepshikha, "Analysis of Software Quality Models for Organizations University of Jammu Department Of Computer Science and IT, 2010.
[54] D. K. C. Chan, and K. R. P. H. Leung, " Software Development As A Workflow Process, Data Source: Software Engineering Conference, Asia Pacific and International Computer Science Conference, APSEC and ICSC, Proceedings, 1997.
I. Sommerville, "Software Engineering, 7th", ed., Addison:Wesley, 2004.
[55] Sommerville, "Software Engineering, 9th", ed., Addison:Wesley, 2011.
[56] C. Stephanidis, A. Savidis, and D. Akoumianakis, "Towards User Interfaces For All", Conference Proceddings of 2nd TIDE, Congress,1995, pp. 167-170.
[57] G. J. Mayers, "The Art of Software Testing", Second Edition, Published by John Wiley & Sons, Inc., Hoboken, New Jersey, 2004.
[58] P. Eriksson, and A. Kovalainen, "Qualitative Methods in Business Research", Sage, London, 2008.
[59] W. Dyer, and A. Wilkins, "Better Stories, Not Better Constructs To Generate Better Theory: A Rejoinder To Eisenhart", Academy of Management Review, Vol. 16, No. 3,1999, pp. 613 - 619.
[60] H. Goodman, "In-depth interviews", In B. A. Thyer, ed., Sage Publications: The Handbook of Social Work Research Methods Thousand Oaks, 2001, pp. 308 - 319.
[61] S. Kvale, "Interview: An Introduction to Qualitative Research Interviewing", London, Sage, Chapter 7: The interview situation,1996, pp. 124 - 135, Chapter 8: The Quality of The Interview, pp. 144 - 159.
[62] D. E. Polkinghorne, " Language and Meaning: Data Collection in Qualitative Research", University of Southern California , Journal of Counselling Psychology, Vol. 52, No. 2, 2005, pp. 137-145.
[63] B. E. Haverkamp,. Ethical Perspectives On Qualitative Research In Applied Psychology, Journal of Counseling Psychology, 52, (2005)146-155.
[64] D. Silverman, " Interpreting Qualitative Data: Methods for Analysing Talk, Text and Interaction ", Vol 2, No,3, 2001.
[65] L. Guion, D. Diehl, and D. McDonald, "Conducting An In-Depth Interview, University of Florida, IFAS Extension, 2009, Retrieved: http://edis.ifas.ufl.edu/pdffiles/FY/.FY39300.pdf. Accessed March 28th, 2012.
[66] The Standish Group, "Chaos Summary Report: The 10 laws of CHAOS", Technical Report. Boston, MA: The Standish Group International, 2009.
[67] E. J. Barry, T. Mukhopadhyay, S. A. Slaughter, "Software Project Duration and Effort: An Empirical Study", Vol. 3, No. 1-2, 2002, pp. 133 - 136.
[68] B. Conti, and B. Kleiner,"How to Increase Teamwork in Organizations, Journal of Quality, Vol. 5, No. 1,2003, pp. 26 - 29.