AGILE SOFTWARE DEVELOPMENT TEAMWORK PRODUCTIVITY- A SYSTEM DYNAMICS APPROACH TO ANALYSE THE PRODUCTIVITY INFLUENCE FACTORS

ISRAT FATEMA
Registration Number: 01 / 2015-2016
Session: 2015-2016

A Thesis
Submitted in Partial Fulfillment of the Requirements for the
Degree of
Master of Philosophy (MPhil)

MASTER OF PHILOSOPHY (MPHIL)

Institute of Information Technology
University of Dhaka
DHAKA, BANGLADESH

AGILE SOFTWARE DEVELOPMENT TEAMWORK PRODUCTIVITY- A SYSTEM DYNAMICS APPROACH TO ANALYSE THE PRODUCTIVITY INFLUENCE FACTORS

ISRAT FATEMA

Approved:

Signature	*Date*
Supervisor: Dr. Kazi Muheymin-Us-Sakib	
Signature	*Date*
Co-supervisor: Dr. Md. Mahbubul Alam Joarder	

Abstract

The agile method emphasizes the people factors and strength of teamwork that simplify the development process. A highly productive team throughout an agile software development process is very instrumental in achieving project success. Consequently, understanding of how individual behavior and productivity are affected by teamwork within an agile team becomes critical. Identifying factors that impact productivity will result in the improvement of teamwork. Hence, a need emerges to recognize the significant ones. Doing so will enable project team management to determine the areas where to concentrate efforts in order to improve productivity.

Improvement in Agile Software Development (ASD) will not be achieved without considering that there is a large number of factors affecting agile teamwork productivity. The objective of this study is to explore what factors influence agile teamwork productivity, and how these factors interacted. This is achieved through a two-phase approach and the use of system dynamics as the modeling tool. The first phase involves reviewing relevant literature, performing a set of in-depth interviews with agile team members and conducting a survey to identify productivity factors. The second phase involves the construction of a System Dynamics (SD) model of agile teamwork productivity with the findings from the first phase to analyze the productivity influence factors.

In the first phase, a survey has been administered to 60 respondents from 18 agile software companies in Bangladesh. The findings from the first phase reveal that from the perspective of agile team members, the most perceived factors impacting their productivity are motivation, team effectiveness, and team management. The culture of social hierarchy in a self- managed agile team obstructs implementation of agile practice. Although, the most followed organizational structure is horizontal, Scrum is leading agile practice among the participating companies. Lack of management support is found to be the most mentioned reason for any failed agile project.

In the second phase, a system dynamics model of agile teamwork productivity is constructed to analyse the productivity influence factors. The complex interrelated structure of different factors affecting agile teamwork productivity is modelled using influence diagram, Causal Loop Diagram (CLD) and stock and flow diagram. The resulting model attempts to capture dynamic characteristics and nonlinearities of ASD teamwork productivity influence factors with an emphasis on the management of agile teamwork. Using the proposed model, the project manager may find the origin of a decrease in productivity, evaluate management strategies along with their effects on teamwork productivity. It also focuses on how well the simulations match the predictions from the theory and survey results from the first phase.

Acknowledgments

All thanks and praise to Allah the Almighty, the Cherisher and Sustainer of the world, the Most Merciful, the Most Gracious.

I would like to thank the many individuals who have assisted in the development of this research. My sincere thanks to my academic supervisor Professor Dr. Kazi Muheymin-Us-Sakib from the Institute of Information Technology, University of Dhaka for his support, feedback, and availability during the entire process of my candidature. I would like to thank him for all his help with the research directions and academic publications. His guidance helped me throughout my research, in writing this thesis and top mostly, pursuing my Ph.D. It was a great privilege to work under his guidance.

I am grateful to my co-supervisor Professor Dr. Md. Mahbubul Alam Joarder, without his support I would not have been able to start my MPhil at IIT. I am grateful to all others staff at IIT, University of Dhaka for always helping me with all the official purposes. I am always thankful to all my junior classmates whose support and cooperation are truly incomparable. There is so much knowledge to gain from them. Many thanks and much appreciation are offered to them also for providing me support with my survey.

Thanks to the eighteen software companies that participated in this research engagement, without them I could not have been achieved the research goals. Thanks to all survey respondents from Bangladeshi software companies.

My gratitude to Ministry of Posts, Telecommunications and Information Technology, Government of the Peoples Republic of Bangladesh for granting me ICT Fellowship 2016-2017.

Profound thanks are due to my family and my beloved children, relatives and friends for their continuous prayers and inspiration.

Lastly, I offer my regards and blessings to all of those who supported me in any respect during the completion of my research.

List of Publications

1. I. Fatema and K. Sakib, "Factors influencing productivity of agile software development teamwork: A qualitative system dynamics approach," in 2017 24th Asia-Pacific Software Engineering Conference (APSEC), 2017, pp. 737-742: IEEE.

2. I. Fatema and K. M. U. Sakib, "Analyse Agile Software Development Teamwork Productivity using Qualitative System Dynamics Approach," ICSEA 2017, p. 71, 2017.

3. I. Fatema and K. M. U. Sakib, "Using Qualitative System Dynamics in the Development of an Agile Teamwork Productivity Model," International Journal On Advances in Software. Vol. 11, no. 12, pp: 170–185, 2018.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

At the present time, all software development companies face the constant challenge of developing faster, cheaper and better. As a consequence, there is a strong need for high productivity in order to stay competitive. Productivity is one of the most common measures of a software organization's competitiveness [1]. Nowadays, software organizations emphasis on methods, tools and best practices that help organizations in productivity improvement. In order to select and implement the proper strategies, it is essential to identify the most relevant productivity factors to achieve productivity improvement. An agile software development process is often claimed to increase productivity [2]. Therefore, a highly productive team is the most important factor in achieving project success at different stages of Agile Software Development (ASD). Software productivity has been changing constantly, yet there is insufficient evidence on the agile teamwork productivity and its influence factors.

This research aims at exploring ASD productivity in contemporary software development that includes an exploration of factors influencing the productivity of agile teams. It is also beneficial to have a dynamic simulation model that tells the project manager in advance the degree of impact that these factors will have on productivity.

Table 1.1: Modern Resolution for All Projects Type [4]

Modern Resolution For All Projects					
	2011	2012	2013	2014	2015
Successful	29%	27%	31%	28%	29%
Challenged	49%	56%	50%	55%	52%
Failed	22%	17%	19%	17%	19%

Table 1.2: Chaos Resolution by Agile Vs. Waterfall [4]

Chaos Resolution By Agile Vs/. Waterfall				
Size	Method	Successful	Challenged	Failed
All Size Projects	Agile	39%	52%	9%
Waterfall	11%	60%	29%	55%

This chapter first briefly presents the motivation of this thesis and introduces the research questions. Later, a brief discussion of the contribution is given. Finally, the scope and the organization of the thesis are provided.

1.1 Motivation

The objective of any software company is to be efficient and productive by being cost-effective and time optimum [2]. Nevertheless, the software industry is troubled by increased cost, delays, lack of customer satisfaction and quality issues that are costing clients and customers worldwide billions of dollars each year [3]. The percentage of successful IT projects worldwide is still too low. Approximately 20% of all projects fail, and about 50% of all projects are challenged: costs and/or time running substantially out of hand, accompanied by unsatisfactory results [4]. Only 30% of the projects are successfully completed [4]. Table 1.1 explains that 70% of the projects are not successful.

Table 1.2 examines how agile relates to waterfall projects. The number of failing projects falls from 29% (waterfall) to only 9% in the case of agile projects. One main reason for failing project is that the initial budgets of projects are often insufficiently defined [4]. Project success rate is also high (39%) in the case of agile compare with waterfall (11%) [4]. Agile methodologies, such as Scrum and Kanban, improve the situation to increase productivity and quality with reduced cost and time [4] [5]. Agile has gained increasing importance in software organizations, as every organization wants to achieve efficiency and effectiveness in reducing the production cost, improving the product services and providing timely deliveries [6]. ASD allows scope changes during the project. Therefore, agile is seen as a solution for companies to remain competitive in a high requirement changing the market and competitive business environment [7]. The 12th annual State of Agile Development survey conducted worldwide (between August – December 2017 with 1492 respondents) shows (Figure 1.1) that the top three reasons for adopting agile are to "accelerate software delivery", "manage changing priorities" and "increase productivity" [8]. Most of those have been in the top three for a few years now. The top five reported benefits of agile adoption include (Figure 1.2): Improved ability to manage changing priorities, better project visibility, better business/IT alignment, faster delivery speed/time to market, and increased team productivity [8].

The above survey results (Table 1.1 1.2 and Figure 1.1 1.2) show that ASD receives high interest in all software industry to enhance the productivity of software development. The agile process places more emphasis on people factors in the project [9]. Therefore, agile teamwork productivity is one of the most important aspects of achieving project success at different stages of a project. An important requirement for accomplishing this is to gain an understanding of those factors that have a significant influence on agile teamwork productivity. Despite the increasing acceptance of the agile methods, insufficient empirical research has been observed on the effect of software development productivity [10]. It is necessary to identify and to understand the most influential ones among the factors and provide a dynamic perspective to manage those factors.

Research has been largely carried out to identify productivity influence fac-

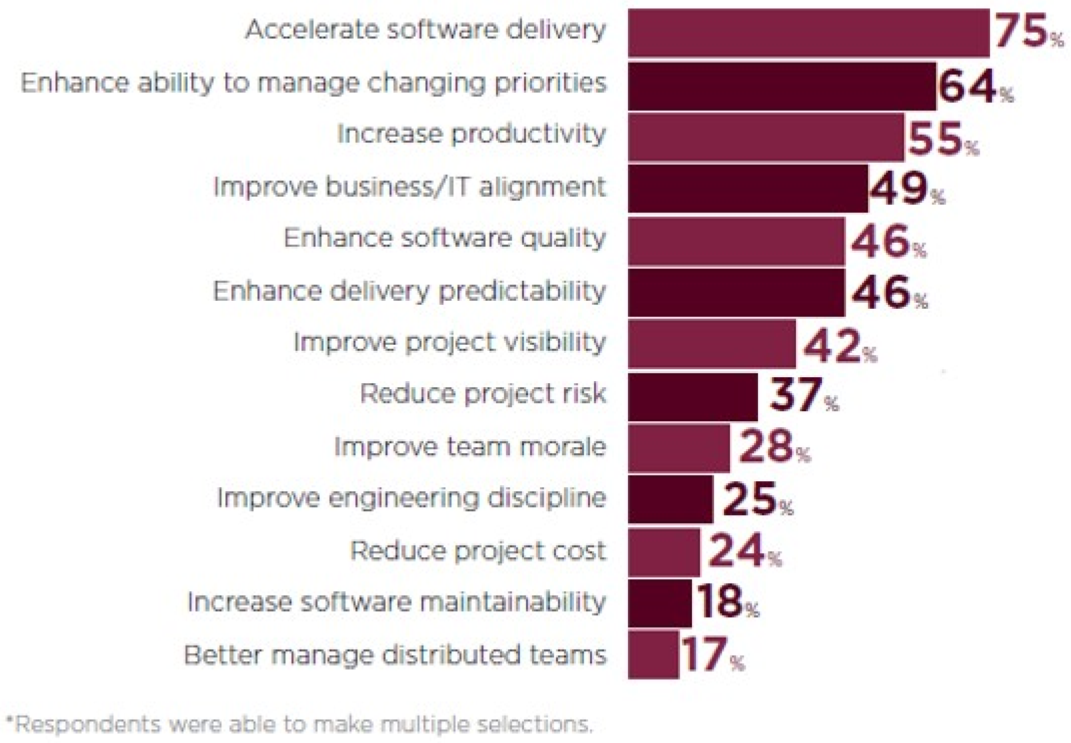


Figure 1.1: Reasons for Adopting Agile [8]

tors in traditional software development. There are four main factors generally discussed [11]: the product being developed (characterization of the specific software), people (team members, capabilities, experience, and motivation), project (task assignment, team size, and schedule) and processes (tools and software methods). However, agile teamwork productivity is a function of various controllable and uncontrollable factors, such as experience and communication skills [12]. The relationships between some of these factors and productivity may change under new software engineering practices and culture [13]. Software productivity across many projects, culture and practice vary extensively even if the same type of software is developed [13]. The factors change over time as expectations change.

The software industry is also different from country to country as are resource availability, the laws which govern it and the developer's cost [14]. In addition, actual productivity measurement becomes more difficult when agile software de-

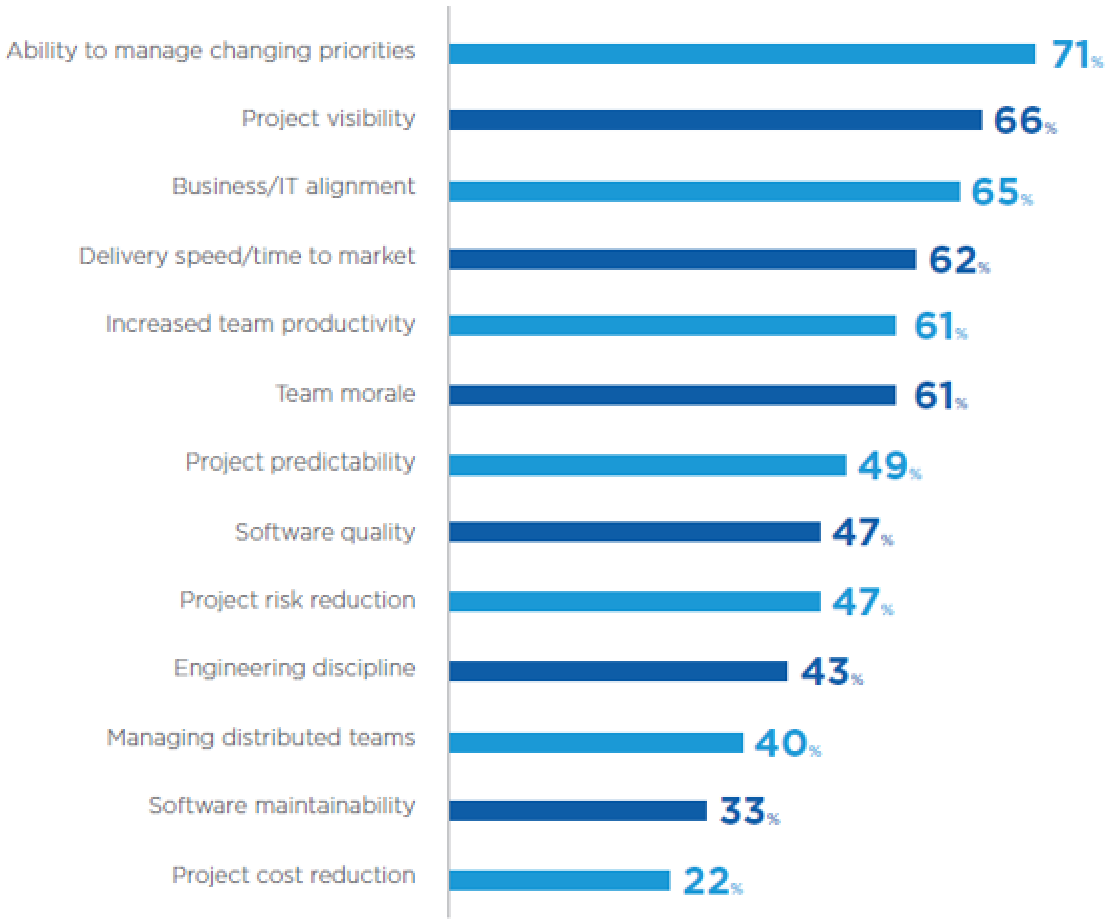


Figure 1.2: Benefits of Adopting Agile [8]

velopers perform knowledge-related tasks (for example, creating, storing, sorting, retrieving, applying and acquiring knowledge) where the product is usually intangible, rarely has a distinct way of doing it, and it is hard to quantify [12]. Since knowledge is complex and hard to evaluate, it is difficult to interpret the productivity of the agile team member's simply by Source Line of Code (SLOC) or function points produced per unit of time/cost [11]. Measuring the same code gives different results with different code counters since there is no universal standard for Line Of Code (LOC) [15]. Story points, used in agile software development, are very subjective and metrics based on story points cannot be used to compare between teams, units or organizations [16].

It would be helpful if the productivity influence factors can be controlled by the Project Manager (PM) when establishing and managing an agile project. "You cannot control what you cannot measure" [17]. Nevertheless, it is difficult to measure agile teams' productivity [16]. In view of the fact that agile software development empowers self-managing teams instead of forming traditional project management. Consequently, the project has less control on the management level. To overcome this limitation and domination project activities, a clear list of influences on productivity in agile software development is needed. Agile team members should also learn to interpret and direct productivity factors regularly as they are self-managed.

Since the agile project team is the most dynamic element in the software development sector, improving teamwork productivity has become a target for software companies everywhere. In addition, the evaluation of individual productivity may not affect the productivity of other developers [5]. These assumptions provide motivation for the study team, not individuals. Throughout the literature review, it has been observed that there is a lack of understanding of the influence factors affecting agile teamwork productivity and well-established dynamic theory of the underlying cause and effect relations. Most of the existing studies have addressed one of the influence factors separately, such as team size, or as a combination of perception about causal relationships, case studies, detailed descriptions of teamwork, effectiveness, and productivity. It does not present a clear, direct statement of dynamic theory and cannot account for the effect of all the influencing factors which have complicated interactions with each other. This study seeks to bridge this gap by developing an integrated model, which accounts for the interactions of these factors and how they affect teamwork productivity and effectiveness.

The motivation behind this study is also driven by the fact that almost all of the agile methods and productivity literature has generally taken their roots in the developed countries, most notably North America and Western Europe [8]. This issue indicates that there is a gap worth filling in the literature resulting from the lack of studies in developing contexts. This study examines the nature of determinants that influence perceived agile teamwork productivity in Bangladeshi software companies. Considering the cultural differences of the Bangladeshi context, this study will contribute towards filling the research gap recognizing that lessons could be learned and knowledge could be transferred to other countries of a similar profile. Finally, the outcome of this study will provide employees, managers, and practitioners with important insights that help them make better decisions concerning agile teamwork productivity aiming at improving organizational processes and fostering strategic goals.

1.2 Research Question

Based on the motivation stated in the previous section, the objective of this study is to analyse and understand the complex interdependences and underlying structures at the team's perception level, which influence agile teamwork perceived productivity over time with the aim of improved productivity.

To reach the higher efficiency level, it is important to look into the factors affecting software development team productivity, and also it is beneficial to have a strategical model that tells the project manager in advance the degree of impact that these factors will have on productivity. The understanding of software productivity is still attached to the earlier perception that is usually not related to the transition and innovation needed from agile development teams [11]. This gap constrains the companies' ability to effectively manage agile software productivity [18]. A better understanding of the agile teamwork productivity influence factors can ensure successful project management which involves ensuring that all company resources perform effectively, provide better support, increase motivation, and enhance team worker's commitment to productivity improvement. This leads to the following questions of this research:

R1. What factors do have an influence on agile teamwork productivity and how is this influence from the team perspective?

ASD consists of many factors that interrelate in the sense that changes made to one of those influence the others either directly or indirectly. To understand the dynamism and complexity inherent in the ASD projects, it is important to identify the relationships between the ASD and its teamwork productivity influence factors. Identification of different factors can be carried out through primary (semi-structured interview and survey of different agile teams' perspective) and secondary information (intensive literature review).

R2. How do the influence factors affect each other, positively, or negatively?

The main purpose of this phase is to determine the causal relationships instead of statistical correlations which possibly do not indicate the cause and effect relationship. With causal structure, the model can be well described, for example, there can be a correlation between two factors, such as team size and programming language, though those may not be causally connected [19].

To analyse the complex inter-related structure of different factors influencing agile teamwork productivity can be modelled using a System Dynamics approach. System Dynamics approach is a simulation methodology enables to model complex system. It can provide more strategic insights and understanding about the effectiveness of different managerial policies, such as increased rewards or training.

However, the identification of causal relationships is the first step in dynamic model conceptualization [20]. In this phase, the answer from the previous question is used as the input. This conceptual model of teamwork productivity can be developed using governing cause and effect feedback loops. These loops can be used to control, link variables and trace the changes.

Then, the relationship that existed between different factors can be determined and the quantitative model of teamwork productivity can be built.

1.3 Contribution of this Research

The aim of this study is to investigate productivity influence factors and define the relationship between the factors at the teams' perception level, which influences agile teamwork productivity over time. These relationships will be used to develop a conceptual model to define and analyse the relationship between cause and effect among the main factors influencing ASD teamwork productivity. Using the proposed model, the ASD project manager may identify the root causes of a decrease in productivity. Thus, agile teamwork productivity may be improved by the implementation of a proper solution. Based on this objective, this study contains two main contributions which are:

Empirical verification of ASD teamwork productivity factors: ASD productivity is influenced by many different factors, such as motivation, leadership, and team size. Identification of productivity factors involves reviewing relevant literature, performing a set of in-depth interviews with agile team members and conducting a survey. The survey has been administered to 70 respondents from 18 agile software companies in Bangladesh. The findings clarify the research questions related to the productivity factors of an agile team. The most perceived factors impacting agile teamwork are motivation, team effectiveness, and team management.

The System dynamic simulation model of ASD teamwork productivity: ASD teamwork productivity model is built to examine the internal dynamics existed within the team and the organizational resources that are used to support those. The model is developed iteratively so that each module can also be used separately depending upon the area of the interest.

1.4 Scope of the Research

The scope of these empirical findings considers the Bangladeshi software companies as a case study, which can, in turn, make the research results beneficial to these companies. However, it is thought that other countries will follow a similar affect to those identified here and its results could be generalized by following the proposed model.

All the data used in this study is collected from the software companies who have voluntarily participated in this research. Therefore, findings from this study should be generalized with caution. While the findings may be specific to the contexts studied, analytic generalization could facilitate the application to other types of culture, background, and environment.

1.5 Thesis Structure

The rest of this thesis answers the above-mentioned research questions in more detail. The organizations of this thesis are described as follows.

Chapter 2: Background Study

To understand the proposed model, the conceptual background of this thesis will be described here. In this chapter, the preliminary knowledge about the agile method, agile software development teamwork productivity factors and measurement, and teamwork and teamwork models are described. It also presents an introduction to Software project management, System Dynamics and its application in software project management.

Chapter 3: Literature Review

The literature review chapter reviews research into agile software development and teamwork productivity, teamwork model and uses of System Dynamics approach in software development.

Chapter 4: Research Methodology

The most appropriate methodology will be selected for the research after identifying /justifying the problem of the research. This chapter will focus on the formulation of the research methodology and the justification of the research approach, survey and System Dynamics approach, adopted for the research.

Chapter 5: Identification of influence factors affecting agile teamwork productivity

An extensive literature review in conjunction with data collected from the survey approach and interviews to identify different factors that influence ASD teamwork productivity is conducted. This chapter will, therefore, present the survey results, determine the major factors that affect agile teamwork productivity in Bangladeshi software companies and what are the factors that should be included in the System Dynamics model.

Chapter 6: Dynamic Modelling of Agile Teamwork Productivity Influence Factors

The objective of this chapter is to provide a full description of the proposed model that explains the relationships suggested between different factors and the ASD teamwork productivity. This chapter presents the system dynamic model beginning with an overview of System Dynamics modelling followed by causal loop diagrams and iterations of the model. The chapter will subsequently explain the methods used to quantify the variables and nature of the relationship between variables. In the last part of the chapter, the behaviour of the proposed model will be discussed to establish how much confidence can be placed in those. This contains the model specification which includes verification and validation as well. Confidence in the usefulness of a model will be established with respect to its purpose. Validation of the model structure and behaviour is an important part of the simulation validation in general and System Dynamics model validation in particular.

Chapter 7: Discussion and Conclusions

This chapter will provide the overall discussion and findings of the research, and concludes it by providing suggestions for future work.

Chapter 2

Background Study

The adoption and implementation of the Agile Software Development (ASD) method enable organizations to adapt to the rapidly changing technologies and worldwide market conditions. A company that can respond to change faster has a better chance of increasing market opportunities and one step ahead of the competition [21]. On this account, in recent years, agile methodology, such as Scrum and eXtreme Programming (XP), have been followed by more and more software companies. Besides, these days, working in an agile team is the criterion for most software developers. Agile teamwork requires added skills than when working as an independent developer. The question is, what these skills are and what are the factors that can influence their skills to work as a team, which eventually affects their productivity. Since the agile project team is the most dynamic element in the software development sector, improving team productivity has become a target for software companies everywhere.

The goal of this chapter is to describe the background needed to understand ASD teamwork productivity. The background of this study is multi-disciplinary, and Figure 2.1 shows a Venn diagram of relevant major research areas. For a better understanding regarding ASD teamwork productivity, firstly agile software development is briefly discussed, followed by ASD teamwork, productivity factors, productivity metrics, and software project management. Lastly, provide a theoretical background for the essential aspects of this research considering modelling and simulation as a methodology.

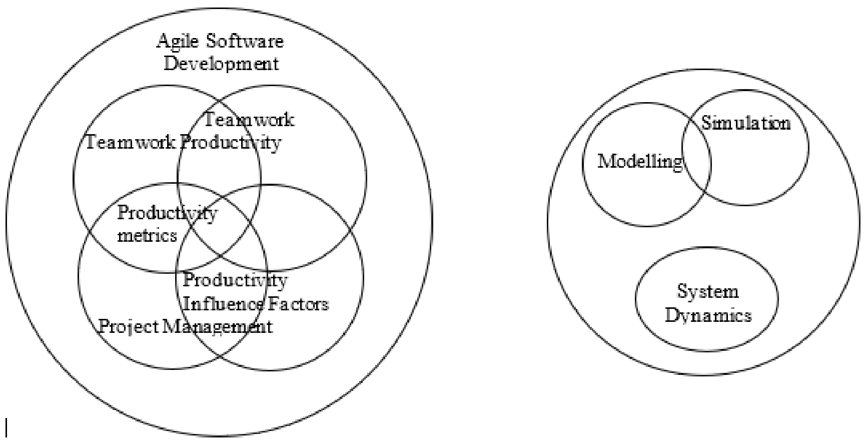


Figure 2.1: Reasons for Adopting Agile [8]

2.1 Agile Software Development

“The most important thing to know about agile methods or processes is that there is no such thing. There are only agile teams. The processes we describe as Agile are environments for a team to learn how to be Agile.” - Kent Beck, creator of eXtreme Programming

Changes in the business environment, such as unstable market pressure, fast-growing system requirements, and advances in technology demand agility in the development of software systems. In response, many firms have replaced their traditional plan-driven (such as waterfall) development with a more adaptive, agile (such as iterative incremental) approach. According to Harvard Business Review 2018, a recent global survey of almost 1,300 IT organizations have reported that most of them are adopting agile within the software development area [22]. Eight out of ten organizations have planned to adopt it. More than half (55 %) are in the process of doing so, while a quarter has already put it into practice [22].

This research aims to produce a System Dynamics (SD) model of ASD teamwork productivity for analysis of its productivity influence factors. In order to model ASD teamwork productivity, first, there is a need to understand the aspect of ASD.

Software companies are attracted by agile methods because it assures to achieve high productivity, deliver high-quality software and high development speed. The term agile comes from agility, which is intended to express the adaptable nature of agile software development method [23].

Some of the principles behind the Agile Manifesto are: [24] [25]

- Customer satisfaction by rapid, continuous delivery of useful software
- Working software is delivered frequently (weeks rather than months)
- Working software is the principal measure of progress
- Even late changes in requirements are welcomed
- Reduce waste (increased productivity and efficiency and reduced development cost)
- Projects are built around motivated individuals (developer satisfaction, well-being)
- Continuous attention to technical excellence and good design
- Simplicity
- Better predictability (visibility)
- Self-organizing teams
- Regular adaptation to changing circumstances

Among the aforementioned agile principles highlight some important features related to productivity itself [26] [27]:

- Agile principle prioritizes customer satisfaction through early and continuous delivery of valuable software. This highlights the importance, not only of the number of developed products but how valuable the product (software) is for its customers. This principle is related to product quality. Quality is the extent to which the software satisfies the essential needs of its various stakeholders by providing value [26].

- Agile principle related to time, recommends the delivery of working software frequently with a preference to the shorter timescale.

- The agile principle establishes that simplicity is essential in ASD. This implies efficiency to perform only the necessary amount of work.

These principles guide all the techniques and rules in the different agile methods, which all aim to make software development more flexible and overall more successful. Agile software development refers to a group of software development methodologies based on iterative development, where requirements and solutions develop through collaboration between self-organizing cross-functional teams. The various agile methodologies share many of the same principles, as well as many of the same characteristics and practices. But from an implementation viewpoint, each has its own set of practices, terminologies, and strategies. Well-known agile software development methodologies include [6]:

Scrum: Scrum is an iterative and incremental framework within which practitioners can employ various processes and techniques to develop a complex product. The main part of Scrum is Sprint, a time-boxed effort usually two weeks to a month. In the sprint, work is divided into subparts and is to be completed in the assigned time limit. Once a sprint has defined no changes can be made to it. A sprint begins with a planning meeting and ends with the demonstrable release. Scrum is a methodology that can be used on small and large projects. It focuses on the management of the development process than coding techniques [26].

Extreme Programming (XP): Extreme programming is a software development methodology based on the set of practices mainly pair programming, customer collocation with the development team and mainly emphasizing customer satisfaction [26]. Extreme programming is most popular among developers because of its simpler rules of iterative planning, daily communication, and team empowerment. Extreme Programming can improve programming quality while shortening delivery schedules.

Feature-Driven Development (FDD): Feature-driven development is an iterative and incremental software development process. FDD combines a number of industry-recognized best practices into a united whole. These practices are all driven from a client-valued functionality (feature) perspective. Its main purpose is to deliver tangible, working software repeatedly in a timely manner.

Crystal Clear Methods: Crystal Clear is an example of a lightweight methodology. Crystal clear states people effects more software development than processes and tools [26]. Crystal clear is a collection of methodologies named toolkit elements that organizations combine to make a suitable methodology for their project. In Crystal clear number of elements used to create a methodology increases and decreases with the size of the project, larger the size of the project more the elements.

Dynamic Systems Development Method (DSDM): Dynamic Systems Development Methods (DSDM) is an agile software development framework. Dynamic systems development method sought to fix cost, quality and time in the beginning and use the prioritization scope into musts, shoulds, could and won't have to adjust the project deliverable to meet the stated time constraint [26]. DSDM is used for developing software and non-IT solutions.

The ASD project teams that apply these aforementioned methodologies also adopt a variety of similar "agile practices" in their organization. The basic set of common attributes that these agile methodologies share are described in Table 2.1 [27] [28].

Table 2.1: Common Practice of Agile Methodologies

Agile Practice	**Short Description**
Story/Feature Driven	Break up of project into manageable pieces of functionality; sometimes named “features”, “stories”, “use cases”, or “threads”.
Iterative-Incremental	Development is performed in repeated cycles (iterative) and in portions at a time (incremental.)
Refactoring	Refinement of the software design and architecture to improve software maintainability and flexibility.
Micro-Optimizing	Teams are empowered to modify aspects of the processor dynamically adapt to changing circumstances. Small improvements and variable changes are made frequently and as needed.
Customer-Involvement	Customer/User involved in demonstrations of functionality to verify/validate features. Higher frequency feedback and clarification of uncertainty. Availability to participate in development meetings.
Team Dynamics	“Soft” factors related to the project team. Daily meetings, agile workspaces, pair programming, schedule/peer pressure, experience gain, etc.
Continuous Integration	Policies and practices related to Configuration Management, and build and test automation.

2.1.1 Agile Software Development Teamwork

These days software development is becoming increasingly complex. The scope of software projects is changing all the time. By adopting an agile method, along with the agile team, software companies stay competitive with market trends. That is why agile teamwork has been widely used and accepted in today's industry of software development. The agile method focuses on people and social interactions. One of its main characteristics is that software development should be organized in small, self-managed teams [29]. Agile teams must be self-organized teams where all members are considered companion at the same level, without a strict hierarchy in practice [30]. The team members are authorized to make decisions as a collective and they should have cross-functional skills, which increases their ability to self-organize. Agile team management aims to be performed in a coordinated approach. Agile teams work better with a smaller team. The main challenge of transition to agile methodologies is the change from the individual work to a collaborative environment that requires a slow transition and not without effort [31]. Traditionally, customer interaction is handled by a single (senior) person in the team. However, in agile projects, the entire team is responsible for communicating with the client/users during sprints/meetings to get the right feedback on the specific part that they are working on. Therefore, it is important that all the team resources are confident (and communicative) enough to contribute to the project for their role. This means that all the team members should be able to communicate with the client since commitments are not made by a single member but by the entire team. Therefore, teamwork and collaboration hold an important influence on a project, an organization, and the overall customer experience. Statistics have also shown that teamwork boosts productivity and increases project success factors rapidly [32].

2.1.2 Software Development Productivity

Productivity is an important feature of any software development project. It has a significant influence on both the cost and the time taken to produce software. Productivity is commonly defined as the ratio of what is produced (output) to the amount of time required to produce it (input). In the field of software development, measuring productivity has proven to be a challenge, mainly due to the individual nature of any given software project and the abstract nature of the software itself. Established productivity metrics reflect this challenge: some metrics are only intended to measure one specific activity of a software development project (such as programming), whereas others are capable of measuring productivity across all probable activities but require the use of time-consuming estimation.

During the last few decades, the Software development field has evolved significantly in terms of development processes and best practices. However, the way in which the productivity of software developers is measured has remained relatively slow [33]. Unfortunately, the software industry is still quite undeveloped in regards to using standards, productivity measurement, benchmarking and continuous improvement [34]. Unexpectedly, the large international system integrators often don't know their productivity, cost-efficiency, quality, and delivery speed, as they don't collect data based on standards [34]. Some organizations even have no idea how they perform against their opponents [35]. Ideally, the productivity of a software team or its members should be measured using a metric that is adaptable enough to take all possible details and activities of a software project. The data needed to assess productivity should be simple to collect in a simple way so that it does not disturb the day-to-day work of a software development team [5]. However, software development is mental work which involves knowledge creation. Knowledge creation is a work of knowledge Worker (KW). The literature has been described KW as a high-level worker. KW puts in their theoretical and analytical knowledge, which is gained by formal education and experience, to construct new products or services [35]. KW tasks (e.g., creating, storing, and sorting, retrieving, applying and acquiring knowledge) where the product is usually intangible, rarely has a single way of doing it, and it is difficult to quantify [36]. KW productivity is related to the extent of autonomy, responsibility and continuous learning process [35]. Table 2.2, adapted from [36], outlines the features which are associated with software development. Since knowledge is complex and hard to evaluate, it is difficult to interpret the productivity of the agile team member's, because [35] [36]:

- It is difficult to measure output (software is not a physical thing, cant be touched and measured with conventional measurement instruments).
- Software projects are much more like an RD (Research and Development) project than manufacturing a product. RD is incredibly hard to measure. It is relatively easy to measure the inputs, but the outputs are hard to measure and unpredictable by nature.

Software development productivity should measure throughout the project instead of leaving it to the end of the project in terms of how the team is doing [5]. To successfully assess the productivity of an agile software development team, it is important to validate the applied metric and its purpose. In addition, the right explanation of the results is also required to understand what can affect productivity and to which level.

2.1.3 Why Measure Software Productivity

"You can't control what you can't measure"- Tom DeMarco, Software Engineer

Productivity measurement is a common activity that drives the success of an organization. It is essential for organizations to understand their capabilities against the industry. According to Scacchi, productivity data is collected and analysed

Table 2.2: Knowledge Worker (KW) Productivity Dimensions (adopted from [36])

KW Productivity Dimensions	**Description**
Quantity	Accounts for outputs and outcomes (quantification of qualitative variables such as customer and worker satisfaction).
Cost	Accounts for profitability, costs, etc.
Timeliness	Accounts for meeting datelines, overtime needed to complete the work, and other time-related issues.
Autonomy	Accounts for independence and how many things a worker can do simultaneously.
Efficiency	Accounts for doing things right. Refers to any task, even if it is not important to the job. The task is completing meeting all the standards of the time, quality, etc.
Quality	Accounts for how good the work is.
Effectiveness	Accounts for doing the right things. This refers just to the tasks that are important to the job, even if they are completed without meeting standards of the time, quality, etc.
Project success	Accounts for the overall result of work, considering decision-making, team interaction, communication, predictability, crisis management, documentation, transferability of work, etc.
Customer satisfaction	Accounts for the fact that the product needs to add value to the customer's business.
Innovation and creativity	Accounts for the ability to create new ideas to improve productivity
Responsibility and importance of work	Accounts for the importance of performing well at a critical time
Learning	Knowledge work requires continuous learning and teaching

for a variety of different reasons and most importantly to identify opportunities for improvement [37]. It is also important to consider that from whose viewpoint productivity is being examined (i.e. stakeholder) [26].

Below are some possible benefits of productivity measurement [37].

- Increase the amount of work successfully achieved by existing team member effort

- Manage to complete the same amount of work with a smaller number of team member

- Develop products of greater difficulty or market value with the same team member workload

- Avoid engaging extra worker to increase workload
- Identify possible product defects earlier in the development
- Minimize the number of defects in delivered products, and minimize the amount of time and effort needed to correct software defects.
- Downsize software production operations
- Proper allocation and time management for underutilized resources
- Identification of weak areas of a less productive team member for training
- Identification of high productive team member for the award

These aforementioned benefits are meaningful to different stakeholders in different ways [38]:

- Development team member - his or her personal productivity data could be used as a benchmark to validate current effort, or seen as an inspiration to improve productivity.
- Team lead - the combined productivity of the team could be used to assess his or her managerial and leadership skills. The individual productivity of team members could be analysed to improve the combined productivity in future projects, for example by introducing a different software development process or transferred specific development work to a different member.
- Software development lead - productivity data for different teams could be compared and used for improvement: intensive projects can be assigned to proven highly productive teams and software projects in a particular domain to teams with a proven track record in that domain.

- Business leaders - the main motivation for measuring software development productivity is that it has a direct impact on cost. A highly productive team can produce results faster, reducing overall costs in terms of man-hours needed to complete a project.

In a software development project, there may exist several stakeholders other than those mentioned above. In order to correctly interpret the results of productivity measurement, it is essential to have knowledge of the factors that impact productivity.

2.1.4 Traditional Software Development Productivity Metrics

Measuring productivity is quite simple in traditional software development projects. The quality metric is pretty easy due to the specific set of requirements and the fact that the project's objective is transparent (to achieve the particular set of requirements) [38]. However, all software development process is made up of different activities. Such activities include the elicitation of requirements, analysis, design, coding, testing, and operations [39]. These activities occur regardless of the chosen software development process. Due to the different nature of these activities, the output of a software project does not always consist of only one type of unit, but several. Table 2.3 summarizes the main productivity measurement approaches [39]:

2.1.5 Factors Affecting the Software Development Productivity

Considering a productivity metric, the analysis of its results for the purpose of cross-team or cross-context comparison requires a complete understanding of the influence factors present in that context. When discussing productivity metrics, an

Table 2.3: Main productivity measurement approaches

Technique/Model	**Formula/Description**
Input/output Ratios	Based on the classical productivity model, defined as the ratio of one output divided by one input. The model can be extracted by adding up several inputs as well as outputs.
Weighted productivity factors	Productivity is calculated by weighting factors influencing productivity. A common approach to identify weights of independent variables to determine a dependent variable in regression analysis
Earned value analysis	The output is computed as the percentage of progress towards the final product. The progress can only be measured with clearly defined milestones.
Statistical process control	Uses statistical inferences and control charts to analyse the software process. If data points are outside the control limits, then this indicates that the process is out of control
Balance scorecard	Evaluate an organization based on a long term perspective, e.g., finance, human and management aspects, and customer satisfaction. The measures obtained as compared to the target levels defined based on the goals derived from strategies and visions.

influencing factor is a factor that in some form affects how software development team members are when working on a project. Influence factors contribute either positively or negatively to productivity ratings and often vary between projects of different nature [39]. In COCOMO (COnstructive COst MOdel), the software cost estimation model presents one of the first and most important productivity factors classifications [40]. In general, productivity factors are related to product, project, personnel, and process [41] [42] [43]. Table 2.4 presents productivity influence factors identified from the literature includes:

Product factors: Product is related to a specific characterization of software, such as the domain, requirements, architecture, code, documentation, interface, size, etc. **Personnel factors**: Personnel factors involve team member capabilities, experience, and motivation. **Project factors**: Project factors encompass management aspects, resource constraints, schedule, team communication, staff turnover, etc. **Process factors**: Process factors include software methods, tools, customer participation, software lifecycle, and reuse.

Table 2.4: Key productivity influence factors from literature

Productivity Factors	**Description**
Product Factors	• Resource constraints – Timing – Memory utilization – CPU occupancy – Number of resource constraints • Program complexity • Client interface – Experience – Participation • Size of programming product • Reuse Reuse of software products, processes, and artifacts, including components, frameworks, and software product lines • Software characteristics (System characteristics: architecture, complexity, domain, non-functional requirements, stability requirements, user interface, and software size)

Continued on next page

Table 2.4 – continued from previous page

Productivity Factors	**Description**
Project Factors	• Hardware development concurrent with programming • Development of computer size • Requirements specification • Modern programming practices usage • Personnel experience • Resource constraints (Constraints such as timing, reliability, storage, team size, and project duration) • Schedule concerns schedule compression and expansion • Team composition (Includes team size, team collocation, and staff, Turnover) • Communication (Includes informal and face-to-face communication)

Continued on next page

Table 2.4 – continued from previous page

Productivity Factors	**Description**
Personnel Factors	• Team experience and capabilities (Includes customer experience, domain knowledge and experience, generational experience, i.e., the percentage of the development the team already participating in two or more generations of software projects, programming language experience, staff capabilities, and experience, and experience with tools) • Motivation (Motivation to work on the project and in the company) • Project Management (Includes aspects of quality of management, conflict management, task assignment, and administrative and formal coordination)

Continued on next page

Table 2.4 – continued from previous page

Productivity Factors	Description
Process Factors	• Customer participation (Refers to real customer involvement in the project.) • Daily builds (Frequent integration of system components) • Documentation (Use of artifacts to register project and product knowledge) • Early prototyping (Early Prototyping stage involves prototyping efforts to resolve potential high-risk issues) • Incremental and iterative development (Incremental approaches encompass various ways of producing a sequence of parts of a system, while iterative approaches involve a diversity of ways of producing parts of a system, trying them out, and feedback on the user experience of production of new or revised parts.) • Modern programming practices (Use of top-down requirements analysis and design structured design notation, structured code, etc.) • Programming language abstraction (Level of abstraction of the language (e.g., Java is a high-level language)) • Software methods (Methodologies and practices) • Tools usage (Use of CASE tools, IDEs, etc.)

Table 2.5: Productivity Influence factors from Trendowicz et al. [41]

Productivity Factors	**Description**
Influence factors, team capabilities and experience	• Programming language experience • Application experience and familiarity • Project management experience and skills
Influence factors, software complexity	• Database size and complexity • Architecture complexity • The complexity of the interface to other systems
Influence factors, project constraints	• Schedule pressure • Decentralized/multi-site development
Influence factors, team capabilities and experience	• CASE tools • Testing tools
Context factors	• Programming language • Domain • Development type

Table 2.6: Productivity Influence factors adopted by Sudhakar et al. [44]

Productivity Factors	**Description**
Soft factors (non-technical)	• Team climate: shared perceptions and objectives to achieve organizational goals. • Team diversity: The variation of team member skills, levels of experience, qualifications, gender and race for instance. • Team innovation: new approaches to problem-solving and value-added skills. • Team member competencies and characteristics: Technical and personal competencies of people on the team that impact familiarity and collaboration. • Team leader behavior: Less micro-management approach and more people management and a facilitator role. • Top management support: Commitment from management to the project

Continued on next page

Table 2.6 – continued from previous page

Productivity Factors	Description
Technical factors	Hardware, software tools
Organizational factors	Organization structure, organizational culture
Environmental factors	Socio, political, economic, legal

2.2 Agile Team Productivity Measurement- Why are Agile Projects Different?

It is clear that in order to better control and manage an agile software development project, there is a need to define agile metrics in a measurable form [45]. Concerning the use of Source Line Of Code (SLOC) and Function Points (FP) as renowned metrics for measuring productivity, such metrics do not interpret the true meaning of productivity [46]. Since knowledge is complex and hard to evaluate, it is difficult to interpret the productivity of the agile team member's simply by SLOC or FP produced per unit of time/cost [46]. More specifically, it is not clear how much time or effort in planning, thinking, and information, etc., is earned to develop one SLOC or FP. Moreover, refactoring is an important practice in agile and usually results in reducing SLOC [43]. Therefore, more SLOC does not mean greater productivity [38]. Consequently, considering the ASD method where individuals work in a team for a common goal, individual performance and productivity need to be combined on the team level. Besides, the evaluation of individual productivity may not affect the productivity of other team members [5]. These ideas provide a motivation to study teams' productivity, not individuals.

Usually, agile teams do not document defects. By definition, agile teams are multidisciplinary [38]. The team members sometimes have difficulty recording their efforts on specific tasks. Nonetheless, this is relevant because some activities might be outside the scope of performance measurement. Agile teams sometimes do not change the functional documentation in the same sprint where the functionality is done. This makes it hard to determine the functional size of that sprint. According to the concept, functional code is provided at the end of each sprint that can be applied in a production environment [38]. This is not always the case in reality. Product backlog items are not ready always at the end of the sprint [30]. Since only ready functionality is assessed, this ensures that the sprint provides a low size delivery and perhaps the length of the next sprint is relatively high [6]. As functional size measurement methods only measure the functional specifications, measuring the length of such a sprint would result in a few function points (or even zero) [38]. The standard performance metrics result in wrong values in this situation and the gaols for this sprint are not achieved. Since the product owner decides the most important product backlog, it could be mainly non-functional backlog items need to be done in a certain sprint. Such issues lead to very unequal productivity metric overtime when trying to implement the traditional productivity measurement metric in the agile context [38].

Generally, agile professionals are said to be productive, responsive and more collaborative. Therefore ASD teamwork productivity is an important issue for any software company [39]. Current agile productivity measures are not suitable for all roles (such as scrum master, analyst, and tester) that an agile team consists of and they only concentrate on coding (development) [38]. Moreover, Melo et al. also described ASD teamwork productivity dimensions as KW productivity (Table 2.2) [5].

Correct metric identification is important for understanding and improving the software development process [34]. Because software metrics may be used for assessment or prediction of productivity. The combination of metrics and predictive (simulation) models can add more insights into the project than what is derivable with the help of metrics only [47]. In addition, it can provide information to the project manager to find the root causes of a decrease in productivity [48]. Therefore, software development productivity may be improved by the implementation of proper solutions.

2.3 Software Development Project Management

Software development projects are basically complex socio-technical systems and their practice is motivated by the interactions of people, processes, tools, and policies [28]. Besides, software projects commonly experience uncertainty in terms of quality, cost and time [49]. It becomes even more difficult to understand in dynamic systems with many challenging feedback effects such as planning, budgeting, staffing, and management. Researchers and software practitioners have highlighted on agile software development as an alternative to overcome these problems. Agile methods apply iterative development cycles (typically 3-4 weeks) and periodically get user feedback [6]. The crucial part of agile projects is acting promptly to deliver the product (or parts of the product) at regular intervals [26]. Moreover, agile methods draw the attention of software organizations because they can cope with the changing requirements of the client [23]. Eventually, it increases the business value of products and creates higher client satisfaction [26]. ASD team is a vital part of agile methodology [29]. Project success is highly depended on the performance of the development team [43]. Therefore, the study of the development process is not effective without considering the development team. It is very important to understand the complexities of ASD team dynamics and as well as the effects of management policy on the ASD team.

Sometimes it is necessary to perform experiments with the software development system in order to understand its behaviour, test and compare different conditions, or find the most favourable solutions. However, the experiment is not always possible in reality because it would be too risky in terms of cost as well as time. In such cases, it is suitable sometimes to move from the actual system to the simulation (predictive) model. Simulation allows conducting experiments with the model of the system in a reliable context without actually exploiting the resources and also saves time. After the simulation, draw the solution back to the actual system [50].

2.4 Modelling and Simulation

“A model is a simplified representation of a system at some particular point in time or space intended to promote understanding of the real system.” [51]. To solve an actual system’s problem with simulation modelling, firstly a model is developed, then simulates it, learns from the simulation, revises the model, and continues the iterations until an adequate level of understanding is developed [52].

2.4.1 Modelling

Considering simulation modelling, there are three major methodologies of model development: analytical, continuous and discrete modelling.

Analytical model: The analytical model provides average data on process behaviour and is often used in the software community. An example is the COCOMO model that is used for estimating the schedule and effort for a software product. These analytical models do not consider dynamic interaction and simulation between factors essential in the process [53]. More detailed and realistic predictions of the software development process behaviour require more advanced models, normally based on simulation techniques. Such techniques are either discrete or continuous.

Continuous model: The continuous type modelling technique is based on System Dynamics (SD) and is mostly used to model the project environment [54]. This technique is useful when controlling systems, with dynamic variables, that change over time, such as productivity and defect detection rates. An SD simulation can model the continuous change in, for example, productivity, resource constraints, and schedule pressure, as the project progresses. The continuous model represents the interaction between project factors as a set of differential equations. This initial representation is a qualitative model. Integrating these equations over time using a quantitative model describes the behaviour of project variables such as motivation, resources and rework detection. Both the qualitative and quantitative models lead to numerical data. The qualitative models usually are interpreted as tendencies of an increase or a decrease of influence. On the other hand, the results from the quantitative model can be interpreted numerically. SD models describe the system in terms of 'flows' that accumulate in various 'levels'. The flows can be dynamic functions or can be the result of other 'auxiliary' variables. As the simulation progresses in small evenly spaced time increments, it computes the changes in levels and flow rates. More details about SD simulation will be discussed in chapters 4 and 6.

Discrete model: In the discrete model, the workability of a complex system as a discrete sequence of well-defined events is modelled [53]. Discrete event models describe process steps, but may not have enough events to represent feedback loops correctly [54]. Because the discrete model requires a large amount of detailed information in order to give valid numerical results that are required in a quantitative model. Besides, as discrete models are based on the idea of a sequence of activities, it may be hard to represent simultaneous activities.

Discrete event modelling can be used to find the best possibility of the different operational tasks of the enterprise rather than the enterprise's operations as a whole [55]. Figure 2.2 shows the difference between continuous sequences of points and discrete time intervals.

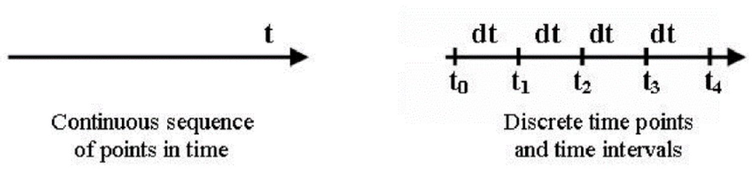

t
Continuous sequence of points in time
dt dt dt dt
t0 t1 t2 t3 t4
Discrete time points and time intervals

Figure 2.2: Discrete vs. Continuous modelling of time (adapted from [55])

2.4.2 Simulation

Simulation is the numerical interpretation of a model [50]. Such a model consists of mathematical relationships describing the studied system. Simulation can be used to explain system behaviour, improve existing systems, or to design new systems [53]. Understanding a system's behaviour and the factors that affect the productivity of the development team is important to a company's management. Many of the inner structure of the system can be revealed during a simulation study [54]. The studies performed before modelling often result in a detailed understanding of the system. Visualization of the model then adds even more understanding of system behaviour. Moreover, a simulation can be shown and explained to others in the organization [53]. Some of the advantages of the simulation are that [50] [53]:

- Simulation helps to understand the complex nature of dynamic systems and can be used for training.
- Mathematical models, by themselves, cannot describe most complex systems, with random elements.
- Experimenting with a system itself is often too expensive, lengthy or impossible.
- Simulation allows studies of a system over a long time period since time is compact.

2.4.3 Why System Dynamics?

As mentioned, three modelling techniques are in practical use: System dynamics (continuous), discrete-event simulation and analytical model. Evaluating the suitability and effectiveness of the three techniques, SD is considered the most appropriate in ASD projects for the reasons discussed in this section.

One major problem in ASD projects is that the description of the causes and effects of incidents in software projects is unclear [56]. Moreover, the input and output of software project components are irregular [1]. Also, the reaction to the project manager's decisions is extremely time-dependent, delayed and nonlinear in software projects [56]. There is a need for support to represent and understand the underlying reasons why ASD project systems behave the way they do.

SD is a powerful methodology for planning, understanding, and discussing complex strategic project management issues and problems [54]. SD has been established as one of the most successful areas for the application is software project management. Because it is important not only to understand the factors of software development technologies, processes, and tools, but also the complexities of project-team dynamics, as well as the effects of management policy (especially the important effects of short-term management actions and decisions) [28]. SD provides a method to model the cause-and-effect relationships among various policy variables. Applying its feedback loops, dynamic and nonlinear behaviours can be effectively represented and explained. Most importantly, SD can be helpful to explore the complex dynamic consequences of different agile-based policies which are not achievable by any other simulation techniques [57]. This is because it simplifies the system design as an integrative feedback system [58]. Through the causal and feedback loops, SD helps to study how the structural changes in one section of the modelled system may influence the overall behaviour of the system. Additionally, SD designs the software project variables in the model as simultaneous activities. This facility is not provided by other simulation techniques, such as the discrete model. This facility is important in ASD projects, specifically in scenarios where the continuous changes in various project variables need to be simultaneously observed as the simulated project progress.

Finally, it will not be suitable to model the ASD process's activity as a discrete-event. Because software projects do not usually focus on the individual activities and flow of the work processes. It rather targets on the quality of the work product in each activity. Observations of the continuous changes in the system are important in determining the most influential variables in the software project system, particularly in determining the ASD teamwork productivity.

2.5 Summary

In this chapter, the basics of agile software development origins, agile teamwork, and its main theoretical underlying concepts are discussed in a nutshell. This chapter has shown software productivity definitions, also adopting knowledge worker productivity as a formal definition of agile team productivity. This chapter also discussed the productivity factors and some popular metrics. Later in this chapter, software development project management is discussed in the context of modelling and simulation. One way of building models is based on system dynamics, which is appropriate for simulating the project environment and its dynamic and qualitative attributes. Details about the SD approach will be discussed in chapters 4 and 6.

The next chapter holds details of the literature review based on the background study of this chapter.

Chapter 3

Literature Review

Over the last decade, the agile software development method has received much attention from researchers and practitioners. Agile as an approach for dealing with change and the unpredictable and hardly controllable elements of software development in a competitive dynamic environment. The intense competition within the software development organization makes agile productivity a topic of high interest. In this chapter, the existing research is described which are attempted to assess the agile productivity and more specifically agile teamwork and its productivity influence factors. Agile team management, workflow, and current modelling efforts related to software development productivity are also presented. Finally, this chapter concludes with the limitations and research gap(s) shown by the review along with the necessary recommendations.

3.1 Existing Research Work

Agile adoption is growing within organizations for accelerating software delivery and productivity, it is essential to discern whether the factors influencing productivity remain the same in all context [12].

Dingsoyr et al. described agile software development as a sociotechnical system comprised of human (socio) and technical entities, the culture of the society in which the system works is crucial [59]. Technological interventions do not increase sociotechnical system effectiveness if they are not supported by social (self-managing team and group) components of the system. Thus, the recent focus on agile software development has increased interest in analysing self-managing agile team and how to effectively make team productive [59].

According to the Agile Manifesto, it focuses on individuals and interactions between people (teamwork) over processes and tools [9]. Therefore, agile software development is influenced by the underlying values and background of the people involved in the development process. These personal characteristics of the people are very much influenced by their local traditions [60]. A survey study by Verner et al. reveals that teamwork productivity factors differ across countries and, culture influences teamwork's decision-making process, problem-solving approach, social interaction, satisfaction and expectation [61].

This research aims at exploring Agile Software Development (ASD) productivity in a current view of software development that includes an exploration of factors influencing the productivity of agile teams. It is also beneficial to have a dynamic model that tells the project manager in advance the degree of impact that these factors will have on productivity. Therefore, focusing on people and teamwork aspects of the agile team; and to better delineate the research aims, the literature review focuses on the following existing research topics:

- Agile teamwork and team effectiveness
- ASD teamwork productivity influence factors
- Team management
- Dynamic modelling to analyse productivity influence factors

3.1.1 Agile teamwork and team effectiveness

Agile software development methodology facilitates to produce high-quality software in a shorter period of time. The term “agile” refers to the software development team as active, swift, and flexible and agile methods put more emphasis on the people factors [5]. Thus, agile development focuses on individuals' talents and skills that simplify the software development process, potentially leading to increased team productivity. Boehm reported in his productivity estimation model, Constructive Cost Model (COCOMO), that productivity of a software development project is mostly affected by the development team, teamwork and team management [40]. Scacchi also identified that poorly managed or organized project's productivity was mostly lower than those projects, which were well managed [37].

Moreover, evaluation of individual productivity may not affect the productivity of other team members [5]. These ideas provide a motivation to study teams' productivity, not individuals. A number of studies exist on teamwork in agile software development on a range of topics relevant to the composition of the team [62], task-effective norms in teams [63], team member's motivation [64], and the importance of a team vision. Yet others have focused on how the team uses daily stand-up meetings to communicate [62], how the team makes decisions [65] and how to achieve self-management [62].

Systems that include technical systems but also operational processes and people who use and interact with the technical system is called socio-technical system [59]. The socio-technical system intends to support some organizational activity [59]. It includes managers, operators, hardware, software, etc. Software development can be seen as a socio-technical system consisted of human and technical entities. This system functions as an integrated, coordinated unit, can address a wide range of problems that are too complex to be addressed by individuals or technologies working alone. Mostly the design and implementation of complex software systems place primary emphasis on technological innovation. It hardly considers the social component – the teams and groups of software developers – that use that technology to create the software systems.

Three key issues characterize complex socio-technical systems [59] [66]:

- First, such systems adapt to environmental uncertainty through self-managed processes without a central executive or outline guiding that adaptation.
- Second, technological interventions do not increase sociotechnical system effectiveness if they are not supported by the social (team and group) components of the system.
- Third, in complex sociotechnical settings is that adaptation comes from coordinated and collaborative interactions at a team or group level. Such team interactions are essential in software development. Therefore, recently the main focus on agile software development has increased interest in how to effectively organize small self- managing teams.

Members of self-managed teams are usually responsible for managing and monitoring their own processes and executing tasks [59]. Many researchers have shown that team effectiveness is the result of the team's processes. However, it is less clear what those processes are and how they result in improved outcomes. Research has shown that different types of team expresses teamwork processes differently.

Considering these characteristics of software development, and focusing on human and cooperative aspects in software development teams, Dingsøyr and Dybå aimed to provide a better understanding of team effectiveness models [59]. Dingsøyr and Dybå, tried to find in their research that whether established models from other disciplines relevant for agile teams or there is a need to develop a new one [59]? Their aim was to improve theoretical understanding and priorities for future studies of teamwork. In order to perform their research, the authors discussed three teamwork models concerned with team effectiveness from an internal perspective. Team effectiveness (or performance) models aim to describe causal relationships between a variable that result in productivity outcomes or at least provide actionable advice for managing productivity [59].

Among the three teamwork models (Figure 3.1, 3.2 3.3), two models are taken from psychology and one is taken from management science.

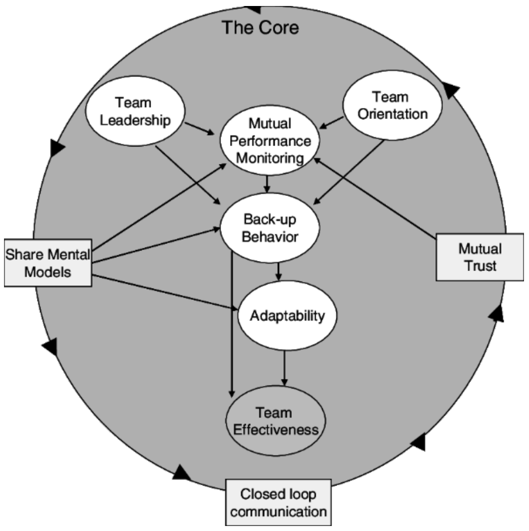


Figure 3.1: Big Five Model (From Salas. E. 2005 [59]

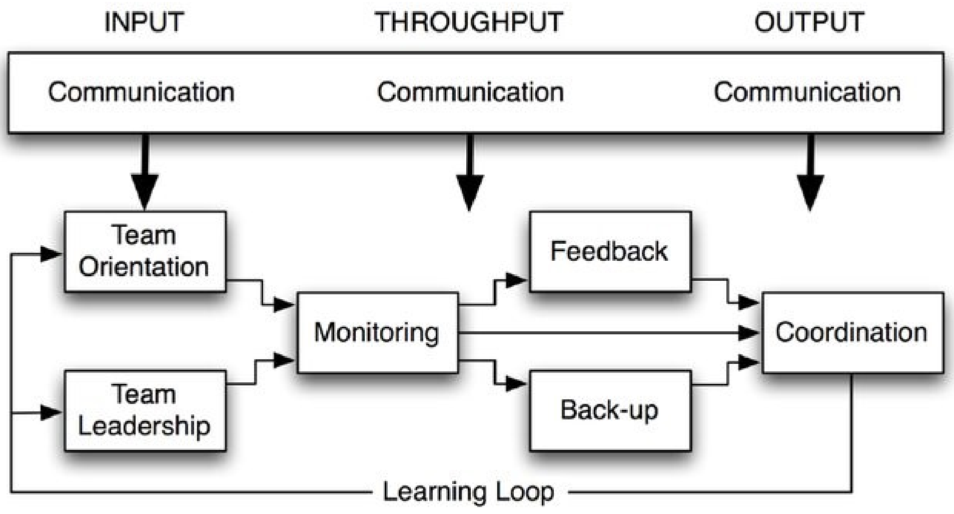


Figure 3.2: Teamwork Model (From Dickinson McIntyre 1992 [59])

Salas at el. undertook an extensive review of the literature to define common teamwork components and as a result identified five core components of teamwork: team leadership, mutual performance monitoring, backup behavior, adapt-

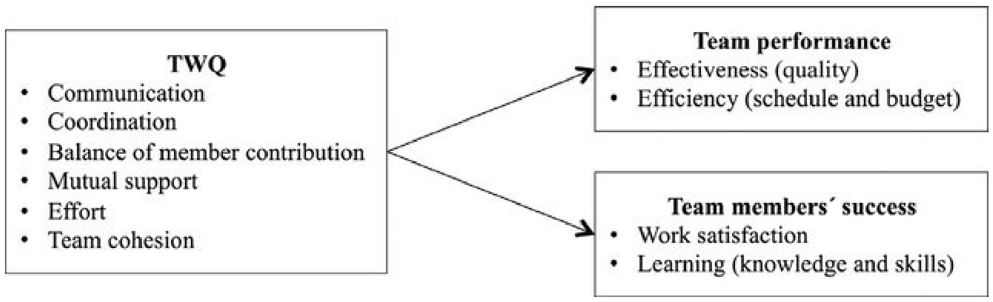


Figure 3.3: Team effectiveness Model (FROM Hoegl 2001 [59])

ability, and team orientation [62][67]. This model (Figure 3.1) also defined a well-established coordinating mechanism common to all effective teamwork: shared mental model, mutual trust and closed-loop communication. While an advantage of their model is the solid grounding in literature, a critique of their work is that their model is not focused on self-managing teams which is an important goal of the agile development team.

One teamwork model that focuses on practical use and self-organizing teams is the Dickinson and McIntryre model [59]. This model (Figure 3.2) uses many of the same mechanisms as Salas et al (Figure 3.1): Team leadership, team orientation, back-up behaviour, and communication. Monitoring is similar to what Salas et al. described as mutual performance monitoring, while feedback and coordination are other characteristics than what is used in the Salas model [62] [67].

The Hoegl model (Figure 3.3) is based on a literature review. This model is currently using for a survey on teamwork in agile development. Teamwork quality is described as consisting of communication, coordination, the balance of member contributions, mutual support, effort, and cohesion [59]. The three models were developed for different purposes. The Salas model (Figure 3.1) was developed for the US Army to summarize team research in a practical model inspired by the "big five" personality factors in psychology [67]. The Dickinson and McIntyre model (Figure 3.2) similarly aimed at practical use but focused on self-managing teams [59] [62]. The Hoegl model (Figure 3.3) was developed for a survey study on the effect of teamwork quality on team performance in innovative projects [59].

These three models use different terminology but share some similar mechanisms. Communication is central in all models; team orientation in the first two models is related to the effort (defined as "Do team members apply all efforts to the team's tasks?"). Back-up (behaviour) can be found again in mutual support. Hoegl puts more emphasis on the importance of motivation in work effectiveness through the focus on team cohesion [59]. In the other models' motivation is placed in team leadership. This involves "motivating team members" in Salas, and "establishing a positive atmosphere". The Salas model is the only one to clearly focus on trust and shared mental models as requirements for effective teamwork. Learning and feedback is a characteristic of the Dickinson and McIntyre model [59] [62].

Dingsøyr and Dybå's study reports that although several team performances (or effectiveness) models exist in other disciplines, there are many open questions regarding their use in software engineering [59]. The relationship between team performance and project success also remains an open question. The authors identified five issues as fundamental for better understanding and future studies of software team effectiveness [59]:

- Practical implementation of a better measurement approach needed for teamwork and team performance in software development.
- More Rigorous Industrial Case Studies can provide higher quality context-specific guidance to socio-technical (software team) systems.
- Work is needed to better understand the dynamics of distributed or virtual software teams.
- Improved theories and models of software teamwork
- Team effectiveness needed to better represent and capture complex tasks performed by the software team.

Dingsøyr and Dybå research raised the question of whether the existing models are representative and useful for software teams with different cultural compositions. As software development globalization progresses, there is an increasing need to better understand the role of culture in software team performance. To answer the research question, their study established that there exists a large number of relevant theories and team effectiveness models in other disciplines. The software development team's research should connect to this pool of research. With respect to empirical studies of software teams, there is a need to test, extend and update theories from other fields [59].

Software development depends significantly on team productivity and performance. Setting up a work team is usually motivated by benefits such as increased productivity, innovation, and employee satisfaction. Research on software development teams has found that team performance and productivity are linked with the effectiveness of teamwork coordination [68]. Most current development methods argue that teams should be self-managed. The use of self-managing teams has become a popular cause of their use to promote more satisfied employees, lower turnover and lower absenteeism [68]. Many organizations, and especially large organizations, still base their software development around plan-driven or component teams (opposite of cross-functional multi-disciplinary). The transition of such teams to agile teams and how to overcome difficulties that occur during that process has been the subject of many case studies. All of them agree that this kind of transition is a hard process.

Another stream of research has focused on team performance in agile software development to analyse the teamwork. Team performance refers to the evaluation of the results of the teamwork. Moe et al. used the Dickinson and McIntyre's input-throughput-output teamwork model (Figure 3.2) to explore the nature of self-organizing agile project team adopting Scrum [65] [68]. The objective of their study was to provide a better understanding of the nature of self-managing agile teams and focused on the interrelations between essential teamwork components. Their study can, in turn, benefit the effective application of the agile method in software development and the teamwork challenges that arise when introducing self-managing agile teams. Moe et al. tried to find the teamwork challenges that arise when introducing a self-managing agile team [68]. Moe et al. conducted fieldwork for an extended period of time (9 months) in a software development company that introduced Scrum [68].

The author's research method focused on the human sense-making, on how mechanisms of teamwork were understood by the people involved. It followed an interpretative field study, used seven principals for interpretive field research [68]. The authors used seven principles- hermeneutic circle, contextualization, the interaction between researchers and subjects, abstraction and generalization, dialogical reasoning, multiple interpretations, and suspicion [68]. Besides the observation, interviews were conducted with different roles at a different level and all team members. In scrum, self-organizing teams work in iterations called sprints with a high degree of self-management and decision making power placed to the operational level, unlike traditional control and power oriented methods [69]. The project is divided into several sprints or iterations depending upon its complexity. Highly specialized skills and a corresponding division of work effects teamwork [29]. There can be a negative effect on team performance when the team members have high individual autonomy [31].

Performance monitoring drives both the content of feedback and timely backup behaviour [67]. In the studied project of Moe et al, the team members did not monitor each other much [68]. There was less feedback and almost no backup. If a group member plays a specialized role, then it reduced team redundancy and backup. It reduced timing for team members to spend working with our communication with each other. Three ways of providing effective backup if required by teammates are verbal feedback or coaching, physically assisting in carrying out a task, or completing a task when it is observed that the workload is too much for a team member [68]. These serve as an effective way for team coordination [67] [68].

Teams having communication problems likely to experience difficulties with coordinating their work [31]. The Dickinson model (Figure 3.2) presented communication as a "glue" that links together all other teamwork processes [59] [68]. In the studied project of Moe et al, the daily stand-up was mostly used by a Scrum Master to obtain an overview of the progress and ongoing project activities and the most important mechanism for such communication [68]. Problems with team orientation, team leadership, and coordination in addition to highly specialized skills and corresponding division of work were important barriers for achieving team effectiveness in the studied project of Moe et al [68]. These are also essential for successful software development. In addition to Dickinson and McIntyre's teamwork components (Figure 3.2), Moe et al. research has found trust and shared mental models are important components for communication, monitoring and team orientation [68].

Transitioning from individual work to self-managing teams requires a reorientation not only by developers but also by management [32]. This transition takes time and resources, but it is essential for the success of any kind of agile method based on self-management [32]. Agile methods depend on teamwork to face conflicts and to develop adaptability (team effectiveness, Figure 3.1) [59]. According to Melo et al., knowledge worker productivity is the best way to describe ASD teamwork productivity (chapter 2, table 2.2) [5]. Since teamwork productivity (and performance) in dynamic environments influenced by different factors, there is a need for more studies on agile teamwork productivity influence factors.

3.1.2 Agile teamwork productivity influence factors

There are several studies that attempted to assess the impact of some of the influencing factors on agile teamwork productivity. Besides, agile surveys have been conducted mostly on the development process and overall view of agile practices [5] [8] [66]. However, these surveys do not elaborate much on productivity factors and do not consider how they are related. Only Melo et al. analysed the major factors influencing agile teamwork productivity using the team's perception as one potential dimension to understanding their overall productivity [5] [12]. Through perceptions, they found that team management is the most influencing factor in agile team productivity. Melo et al. conducted multiple case studies for six months based on three companies in Brazil. Selection criteria for Companies chosen were that they have used agile methods for at least 2 years, projects in place for at least 6 months with at least 4 workers co-located, along with each business in different segments and geographical locations [5].

Productivity is controversial and differs due to context [21]. There is no general agreement on the measurement of software products for this reason. Melo et al. define productivity using team perception to quantify team productivity as a common measurement across the three companies [12].

The authors chose a research model which is started with Input-Process-Output (IPO) teamwork effective frameworks (Figure 3.4) [12]. IPO is a well-known generic model in software development which has more recently been adopted for teamwork effectiveness and conceptual framework modelling for quantitative and qualitative studies [5]. This study first time proposed a conceptual framework based on this IPO model where inputs, processes, and outputs are based on the agile principles [12].

In Figure 3.4, Inputs used are Individual group characteristics, stage of team development, and nature of the task, organizational context, and supervisory behaviours. Outcomes are Agile productivity, attitude and behavioural. Group processes are internal and external. Note that inputs directly, or indirectly via group processes, handle the outcomes as shown in this conceptual framework.

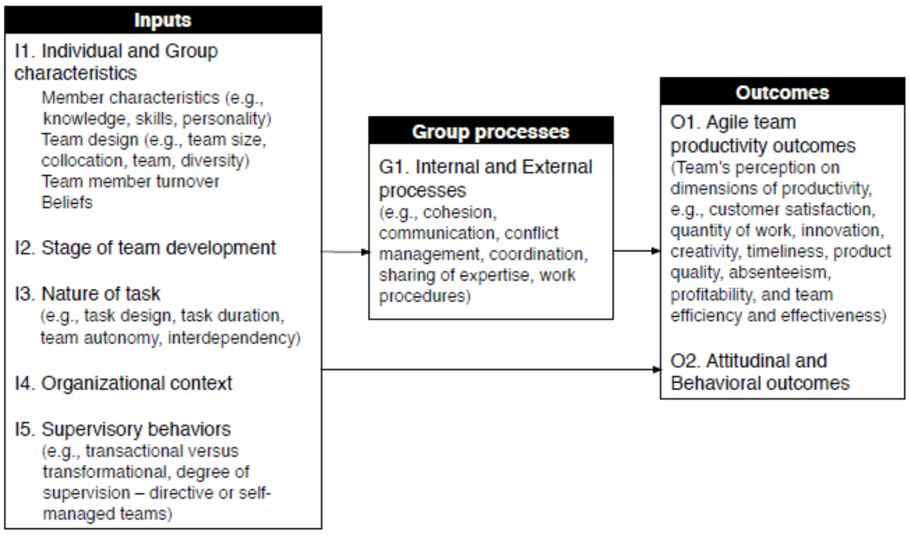
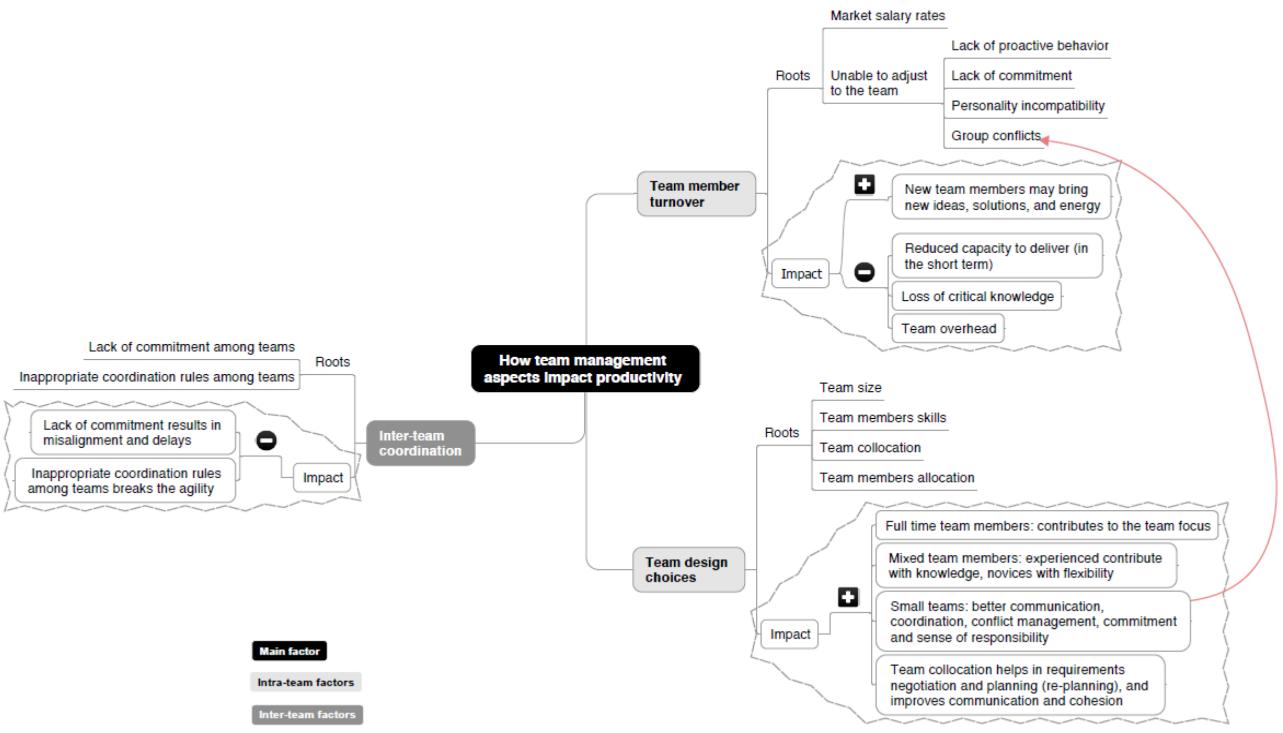

Inputs
I1. Individual and Group characteristics
Member characteristics (e.g., knowledge, skills, personality)
Team design (e.g., team size, collocation, team, diversity)
Team member turnover
Beliefs
I2. Stage of team development
I3. Nature of task (e.g., task design, task duration, team autonomy, interdependency)
I4. Organizational context
I5. Supervisory behaviors (e.g., transactional versus transformational, degree of supervision – directive or self-managed teams)
Group processes
G1. Internal and External processes (e.g., cohesion, communication, conflict management, coordination, sharing of expertise, work procedures)
Outcomes
O1. Agile team productivity outcomes (Team's perception on dimensions of productivity, e.g., customer satisfaction, quantity of work, innovation, creativity, timeliness, product quality, absenteeism, profitability, and team efficiency and effectiveness)
O2. Attitudinal and Behavioral outcomes

Figure 3.4: Input-Process-Outcome Agile team productivity conceptual framework (Adopted From [12])

The length of the study was chosen to be six months to remove or identify productivity factors over time. Company 1 provides financial systems, uses XP/Scrum and has one-week cycles, employs 400 people and has 33% staff turnover. Company 2 provides E-Commerce, uses XP/Scrum/Lean principles and has 3 or 4-week cycles, employs 120 people and has 40% staff turnover. Company 3 is an Internet content provider, uses Scrum/XP/Lean principles and 3-week cycles, employs 200 people, and has 35.3% staff turnover. The data collected was retrospective document analysis, semi-structured interviews, and nonparticipant direct observation field studies.

Interviews were conducted by an author experienced in requirements elicitation and interviewing techniques. Interviews lasted one hour and were recorded. Interviewees were informed of the importance of the study and included developers, project managers, product owners with differing experiences. No details were given to bias their information. The analysis first consists of constructing a thematic map, often used in qualitative data to identify, analyse and report themed information. The thematic map shown in Figure 3.5, identifies three themes. Inter-team coordination, Team member turnover, and Team design choices along with reasoning [12]. At a later stage, further information is added regarding impacts affecting these themes and whether they have a positive or negative impact. The thematic map findings were discussed and reported using the conceptual framework.

Team member turnover is a productivity factor producing cost in separation, advertising and recruiting, The relevance to this research is the team productivity is affected until the team member is integrated and up to speed, and loss of effective contributors when someone leaves.



How team management aspects impact productivity
Team member turnover
Roots
Market salary rates
Unable to adjust to the team
Lack of proactive behavior
Lack of commitment
Personality incompatibility
Group conflicts
Impact
New team members may bring new ideas, solutions, and energy
Reduced capacity to deliver (in the short term)
Loss of critical knowledge
Team overhead
Inter-team coordination
Roots
Lack of commitment among teams
Inappropriate coordination rules among teams
Impact
Lack of commitment results in misalignment and delays
Inappropriate coordination rules among teams breaks the agility
Team design choices
Roots
Team size
Team members skills
Team collocation
Team members allocation
Impact
Full time team members: contributes to the team focus
Mixed team members: experienced contribute with knowledge, novices with flexibility
Small teams: better communication, coordination, conflict management, commitment and sense of responsibility
Team collocation helps in requirements negotiation and planning (re-planning), and improves communication and cohesion
Main factor
Intra-team factors
Inter-team factors

Figure 3.5: Thematic map on agile productivity factors (Adopted From [12])

There was a positive side found when new members contribute new ideas and experience to the team. These positive and negative effects were clearly identified in interviews and retrospectives from the companies.

Team design choices affect productivity. This research identified desirable team attributes as full-time allocation, diversity, skills, team size, and collocation. Interviews identified that experienced and less experienced members provide more cohesion and flexibility, fewer conflicts. Small team size was found to improve communication and alignment, responsibility and commitment. Bigger teams have more communication and conflict resolution, more people to understand and align with the big picture. Social interaction of collocation improved communication and removed boundaries within hierarchically organized companies. Workspace layout considering desk positions and proximity also affected how well collocation worked. Although considerable influence is applied by team design choices, most of the decisions are made by people outside the team or their control.

Inter-team coordination affected productivity due to shared resources, prerequisite constraints, and simultaneity constraints, relationships of tasks and subtasks dependencies between teams. Inter-team coordination processes are then needed to manage these dependencies. Examples of inter-team dependencies found were external customers, quality assurance, releasing, testing, integration, production, reuse of components under development or maintained in other teams. All three companies in interviews and retrospectives identified limitations waiting for a decision by external dependencies like this. Some teams owning components to be reused were not committed to the same goal, only to the task to be completed. Other dependencies working on different schedules, like quality assurance and integration.

In further discussion, the influencing factors for team productivity are further explored. Productivity was particularly sensitive to team management. Agile teams take responsibility for managing their own work and behaviour, yet others usually make a decision about goals, team structure, and organizational support.

Melo et al. developed a conceptual framework in their study, using thematic analysis to understand the possible mechanisms behind such productivity factors [5]. Agile team management was found to be the most influential factor in achieving agile team productivity. At the intra-team level, the main productivity factors were team design (structure and work allocation) and member turnover. At the inter-team level, the main productivity factors were how well teams could be effectively coordinated by proper interfaces and other dependencies and avoiding delays in providing promised software to dependent teams.

Dingsøyr and Lindsjørn conducted a focus group study with 92 participants in 18 groups [62]. Their study was performed to identify factors that have an influence on agile software development team performance. The focus group's study was chosen to quickly obtain information on rising phenomena through structured, moderated discussions with groups of practitioners. The advantages of focus groups include the ability to collect large and rich amounts of research data [62]. The researcher can interact directly with respondents for clarification of responses or follow-up questions and that focus group participants can react to and build upon responses from other focus group members.

Salas et al. model of team performance were used to structure the output from focus groups [67]. In Salas' model, the "Big Five" core components of teamwork (team leadership, mutual performance monitoring, backup behaviour, adaptability, and team orientation) interact to produce performance. Because teams require a complex mixture of factors that include organizational support, individual skills, and also teamwork skills. Three coordinating mechanisms (shared mental models, closed-loop communication, and mutual trust) are proposed as a means to raise the level of performance.

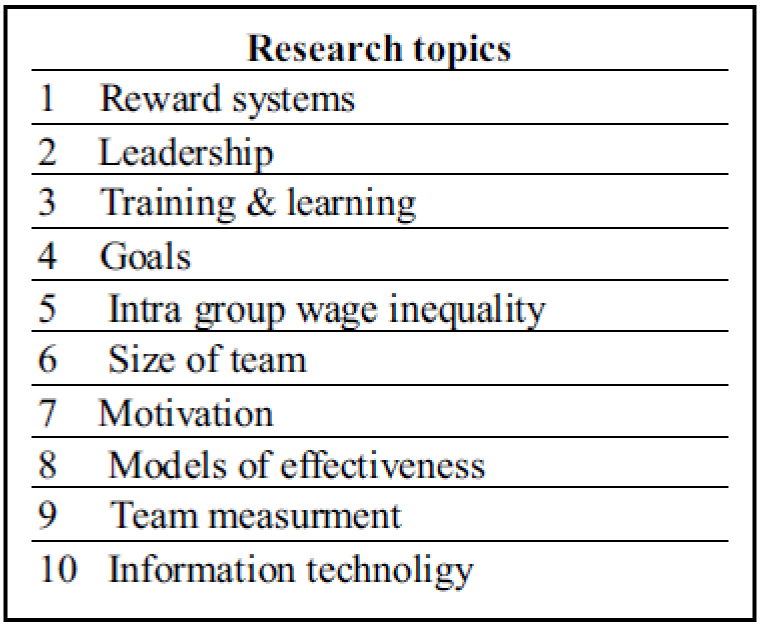
In [67], the main steps of conducting the focus groups were planning, recruitment, moderation, documentation of results, processing and analysis. The Planning stage developed a plan for each focus group including the agenda (introduction, 2 group exercise, and presentation team performance model and summery). In the Recruitment stage, among the 18 focus groups, 11 groups were taken from three conferences on agile software development [67]. The participants signed up for a workshop on "Effective Agile Teamwork" and were divided into groups of

4-6 on arrival. The rest of the focus groups conducted within four companies that participated in a research project on effective teamwork in Norway [67]. According to Dingsøyr and Lindsjørn, to get the correct result, creating the focus group to conduct the study is the crucial part [67].

People involved in any organization interact and communicate with each other during their work. The quality of work often depends on the ability of team members from all levels of the organization to create a shared understanding of the task, process and the respective roles of its members [17]. The organization must know how to get the best out of the team. In turn, teamwork challenges also increasing due to globalization and competitiveness of the global economy [70]. Teamwork must now look for new ways to adapt quickly, operate more efficiently and better prepare themselves for the future.

Many organization believes they can have remarkable benefits by redesigning the structure of their organization. Benefits include enhanced profitability, overall performance, productivity and better alignment of teamwork with business needs [70]. Hamid Tohidi in his study aimed to support those organizations considering undertaking this kind of design effort [70]. In order to provide this assistance, Hamid Tohidi conducted a survey of the research studies on teamwork productivity and effectiveness [70]. The purpose of the survey was: 1) review literature, 2) classify the literature based on teamwork productivity. Ten major factors were considered for classification schemes that impact on teamwork productivity and effectiveness. Factors included reward systems, leadership, and training and learning, goals, intragroup wage inequality, size of the team, motivation, models of effectiveness, team measurement and information technology.

Extensive work has been done on applying teamwork productivity and effectiveness in an organization using the classification scheme (Figure 3.6) [70]. The author found that having an effective reward system can improve the level of motivation. Because motivation has a huge effect on increasing productivity during



	Research topics
1	Reward systems
2	Leadership
3	Training & learning
4	Goals
5	Intra group wage inequality
6	Size of team
7	Motivation
8	Models of effectiveness
9	Team measurment
10	Information technoligy

Figure 3.6: Classification schemes for teamwork productivity and effectiveness. (Adopted From [70])

a project's implementation. An important element to consider when implement team rewards systems are the size of the reward. Reward size has been shown to correlate with pay satisfaction and motivation. The role of leadership is fundamental to facilitating individual and organizational performance. Investment in training and development will positively impact organizational performance. Training directly impacts organizational productivity and business continuity, and possibly could increase organizational competencies [70].

The findings from Hamid Touhidi's study are: a team will perform less effectively without a goal. Goal setting is critical for the success of a team. Team success is more important than the success of the company as a whole. The goal must be measurable. Most studies focused on the performance of the goal that included goal process and setting. In this context also, most studies on the organization goal serve the discussion on the organization's performance, setting and behaviour [70].

Wage inequality damage team bonding and decrease an organization's performance. Team workers withdraw their effort as their actual wage fails to meet their fair wage [70]. The size of an organization mainly refers to the number of staff in an organization. The size of the organization may have an influence on teamwork productivity and effectiveness. Organizations with different sizes have different ways to construct their information systems. Poor performance by one employee can invalidate good performance by the rest of the group. Indirect motivation generated by perfect monitoring and repeated interaction within the group solve the free-rider problem [70]. Reward, effort recognition and incorporate group beliefs and norms are collective forms of work motivation.

Effectiveness model has been defined by most of the studies in terms of productivity, employee and customer satisfaction, and manager judgments. Job design, interdependence, composition, context, and process are the main theme that contributes to the effectiveness model.

Team measurement is the basis for management to know what is going on in their organization. It helps to identify performance gaps that should be analysed and eliminated. Research has found that there are some basic criteria needed for defining and measuring performance [70]. These are the establishment of goals and objectives, planning of work, identification of skills and tools and resources needed by performers. Performance measurement can be divided into four basic categories: quality, quantity, timeliness, and cost. High performing teams who have mastered technology improvements and developed the corresponding human/social systems have made significant improvements in their productivity. There are five major challenges that global software development teams face: 1) adjusting to geographical distribution of IT personnel and users 2) loss of communication richness with less face-to-face interaction, 3) coordination breakdown in project management, 4) loss of "teams", 5) dealing with cultural differences among globally distributed staff and business organizations [70].

Hamid Touhidi has discussed a large number of literature review and survey of research studies on teamwork productivity and effectiveness [70]. It has identified (1) the factors that influence the effectiveness of teams at work in organizations (2) It provides some of the strongest support for the value of teams to organizational effectiveness. It also reported that there is a close relationship between motivation and effectiveness in organizations [70].

The author expected that the study will assist researchers currently engaged in teamwork productivity and effectiveness [70]. Eventually, that may lead to the identification and stimulation of areas requiring additional research [70].

3.2 Software Project Management and System Dynamics

Though agile approaches have received wide attention, empirical research that evaluates their effectiveness and appropriateness is insufficient. Much of the research to date has focused on studying the impacts of select agile practices such as pair programming and methods such as SCRUM or Extreme Programming [47] [69] [71]. However, there is a need to study the dynamic nature of agile practices as an integrated system rather than as individual (isolated) practices. For example, agile practices magnify the impacts of dynamic project features such as feedback, (e.g., iterative feedback from customers), time delays (e.g., delays in implementing change requests), and nonlinear cause-effect relationships among project components (e.g., the relationship between schedule pressure and adding new personnel). Successful management of agile projects requires an understanding and exploitation of such dynamic features [26] [28] [47]. Existing literature does not help integrate the understanding of individual practices and their implications for the entire development process. The simulation model provides an integrated environment within which the impact of an agile practice on the entire development process can be evaluated, simulated in the real-life environment [53]

[72]. Agile practices are closely related to each other. For example, refactoring is supported by unit testing, continuous integration, and design, while unit testing is supported by pair programming [28] [47]. SD provides an environment in which such interconnection between agile practices can be fully represented and studied. Motivated by this concern, this research uses System Dynamics (SD) simulation to investigate the influences of agile teamwork productivity factors by modelling them as a dynamic system.

SD refers to the simulation technique developed by Forrester [20]. SD model contains interrelations and dependencies of software development at a more extensive level than traditional analytic summary models [19] [20]. SD models are developed using continuous quantities that are expressed as levels, rates and information links that represent the feedback loops and are comprised of coupled, nonlinear first-order differential equations [20]. Details of the SD approach is defined in chapters 4 and 6.

SD technique has been applied in software engineering fields for modelling purposes, which is important for the organization and the project. SD is well suited for studying complex systems where unknown attributes of system properties are less visible [19] [20]. For example, the SD simulation technique can create integrative models for software projects to analyse interactions between project-related activities such as testing, hiring, training, project management, etc. [20]. SD modelling is also recognized as a useful tool for conducting research in dynamic decision making such as how software project managers handle staffing delays and how their decisions affect the outcome of a project or how the presence of unreliable initial estimates lead to real prediction [73] [74].

Typical problems in software development include poor planning, lack of risk identification and mitigation, constantly changing requirements, and various managerial problems such as poor hiring and training practices [74]. These problems are related and an understanding of their dependence can provide information for making improvements [75]. An integrative view of management-type functions such as planning, controlling, and staffing and production-type functions such as designing, coding, reviewing, and testing is useful and can identify different factors that may be interacting to cause problems [72] [73] [74]. There are hundreds of variables that can affect software development productivity and these variables are often related to one another [11].

There are a few types of research that attempted to evaluate the impact of some of the influencing factors on productivity separately using SD [10][72][73][76]. However, the complex inter-related structure of all the major factors influencing the teamwork productivity was not considered by the previous works. Abdel-Hamid and Madnick attempted to integrate system dynamics modelling and project dynamics insights with traditional software development processes [74]. Their widely known system dynamics model for software development is given in their book, Software Project Dynamics: An Integrated Approach [73]. In [73], Abdel-Ahmed investigated the effect of various management policies on development cycle time, quality and effort. This high-level model simulates the typical waterfall process after requirements have been obtained. It is intended for medium-sized projects. This model integrates multiple functions of the software development process and includes both management-type functions (e.g., planning, control, and staffing) and software production-type activities (e.g., design, coding, review, and testing). This model has been used to investigate a wide range of areas in software development including software cost and schedule estimation [73], the economics of the quality assurance function [77], project staffing [73] [78], software reuse [73] and project control with faulty information [73] [74] [78]. More recently, SD modelling has been used extensively in research on the software development process [75].

In [73], Abdel-Hamid created a model that combined his software development SD simulator with the software effort estimator, COCOMO. In [79], the author demonstrates how the model can be used before a project begins to adapt COCOMO estimates to reflect the true nature of the staffing limitations, during development to adapt product-sizing assumptions.

Abdel-Hamid's models do not concentrate on different process models and architectures and this is needed since the waterfall process does not always lead to successful performance. Details of the actual development process are missing from the model and it uses only software development rate and allocated staff to determine changes to the completed task level. It is important to examine the relationship between development phases. This work was an important first step and base model for this research, but more detail is needed for managerial decision-making and planning purposes. In addition, Abdel-Himid works adopt the waterfall method, which limits their applicability in a recent software project and more importantly, does not focus on the agile principles.

In another research, Rodrigues proposed methods by which system dynamics modelling can be integrated with principles of project management [80]. In [68], the authors discussed whether the agile project will fit within the common system dynamics project management structures or it has a unique structure. An analysis of factors that impact on productivity during agile web development and maintenance phases was conducted by Kong et al. [10]. However, it does not explicitly show the interrelations of different variables that influence the effectiveness of teamwork. Cao et al. created an integrative system dynamics model of agile software development for investigating refactoring and its impact [81]. The authors investigated the dynamics of agile software development and the impact of agile practices on cycle time and customer satisfaction using SD [81]. Modellers have also investigated schedule pressure effects on the dynamics of iterative development cycles [82]. In [47], Glaiel et al. presented an Agile Project Dynamics model that captured the agile natures as a separate component of the model and allows experimentation with combinations of practices and management policies. The agile natures were identified as Story/feature-driven, iterative/incremental, refactoring, and micro-optimizing, customer involvement, team dynamics, and continuous integration. The goal of this model was to gain insights and recommendations to integrate agile practices into a large-scale software engineering organization. Glaiel et al. concluded that team Dynamics, feature-driven, and iterative-incremental natures are relatively easy to implement or adapt, as most of these practices rule the behaviour of the software development team alone [28] [47].

In a different stream, a wide range of SD models can be found that is concerned with modelling and simulating mental aspects of human beings in an organizational context. Sterman offers a model of how individuals manage their workloads [20]. Another model by Sterman depicts how workforce quality and loyalty are influenced by perceived career opportunities and wages [54]. Henk Akkerman and Kim van Oorschot model among others how employee's motivation, satisfaction, and training influence productivity [83]. Jeffrey Vancouver et al. apply SD to model how a new team member to an organization seeks to build up job-relevant knowledge [84]. Andreas Gregoriades presents a model to study how factors like fatigue, motivation, and stress result in human error [85]. Block and Pickl have done research using SD on human behaviour and its connections with human resource policies [86]. They have modelled based on ability, motivation and opportunity theory. They established causal links and stock and flow diagram showing the influence of ability, motivation, and opportunity on the performance of an individual.

3.3 Summary

All the above discussions show that SD can be successfully applied to model working team member's behaviour and mental processes in a job context of an individual. However, throughout the literature review, it has been observed that there is a lack of well-established dynamic theory about agile teamwork. This study seeks to fill this gap by developing an integrated model, which represents the interrelated structure of productivity influence factors and how they impact (positively or negatively) agile teamwork's productivity. In order to do so, this study applies a system dynamics approach, which can study complex systems by exploring underlying associations and connections between the components of a system that normally are not discovered by the input-output-process type of models used in organizational studies. Focusing on people and teamwork aspects of the agile team, this paper makes use of two team effectiveness models for better analysis of agile software development teamwork productivity. Two models, the Salas (Figure 3.1) and the Dickenson McIntyre models (Figure 3.2) are used, which focus on team effectiveness, and mainly on internal aspects of the team.

Chapter 4

Research Methodology

The aim of this chapter is to give a description of the methodology and research strategies followed for the creation of this research work. This means that this chapter only contains the planning of how the research should be performed and not the execution itself, which is described in chapters 5 and 6. The objective of this research is to identify the most suitable approach for identifying the influence factors and relations that determine the productivity of the ASD teamwork with positive or negative consequences. For this purpose, this research follows an exploratory sequential (mixed-methods) two-phased strategy. This chapter briefly discusses the strategies adopted in this research: interview, survey and System Dynamics (SD) modelling.

This study aims to identify Agile Software Development (ASD) teamwork productivity influence factors (such as motivation, team management efficiency, and team effectiveness) and develop a productivity model to analyse the interactions among the identified main factors. Consideration of the research methodology will lead the research methods for collecting data and the modelling approach to fulfill the research aims. The shortcoming of previous research studies lies in their not considering the complex inter-related structure. In addition, causal relationships of different factors (hard and soft) affecting the agile teamwork productivity.

Modelling the relationship has been proposed as a potential solution. The development of a model should allow a better understanding of the influence factors and exploring the key relationship. The following subsections discuss the approach to investigating the feasibility of this proposal. An appropriate methodology will be explored to achieve the goals of this study. Therefore, the following subsections give a brief overview of the research methods used in this study. Firstly, it describes the survey method which is used to investigate ASD teamwork productivity influence factors and their influence from a team perspective. Secondly, it describes the System.

4.1 Scope of research

Before selecting a research methodology, the focus and scope of the study are carefully considered. This is to ensure that research findings will be appropriate to the applied context. The research scope has been limited to Bangladeshi software companies as the collection of data is conducted by Bangladeshi software companies. This can, in turn, makes the research results beneficial to these companies.

All the data used in this study is collected from the software companies who have voluntarily participated in this research. Therefore, findings from this study should be generalized with caution. While the findings may be specific to the contexts studied, analytic generalization could facilitate the application to other types of culture, background, and environment.

4.2 Research Strategy

This research followed an exploratory sequential (mixed-methods) two-phased strategy. A mixed-methods research strategy is suitable to achieve the objectives of this study since ASD is a complex event, usually shows unusual results and few reliable observations [5]. A mixed-methods research strategy combines both qualitative and quantitative approaches. This research strategy is selected as a result of the following criteria:

- Exploratory sequential design is useful when a researcher first begins by exploring with qualitative data and analysis and;
- Then uses the findings in a second phase. The second phase in the sequential design builds upon the results achieved in the initial phase [87]

. Figure 4.1 summarizes the high-level view of the methodology that has been used to carry out the research:

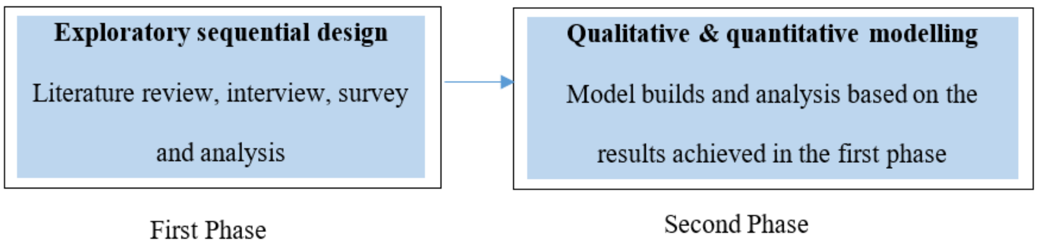

Exploratory sequential design
Literature review, interview, survey and analysis
Qualitative & quantitative modelling
Model builds and analysis based on the results achieved in the first phase
First Phase
Second Phase

Figure 4.1: Research methodology diagram

Exploratory research runs for a problem that has not been considered more clearly and there are few earlier studies [5]. The outcomes do not completely contribute to the final solutions. It rather provides insight into a given situation and a better understanding of the existing problem [5]. The main sources of information collection in exploratory research are trials, interviews, focus group discussions, observation, or questionnaires/surveys [5]. This study chose to develop a questionnaire in order to answer the research questions with responses from agile practitioners in Bangladeshi software companies. This study adopted survey research for the reason that survey questionnaires are relatively inexpensive and do not require excessive amounts of time from respondents. The standardized answers from the questionnaire can provide data that can be statistically analysed.

In the first phase, considering the research question R1 (what factors do have an influence on agile teamwork productivity and how is this influence from the team perspective?), the survey method is primarily used for collecting and exploring the data. Compare to other research methods, such as case studies or experiments, survey research is suitable when [5]:

- Research questions take the form of "what is happening?" and "how and why it is happening?" Survey research is appropriate for answering questions about what, how and why [5];
- Independent and dependent variable is not possible to control all the time;
- Point of interest occurs in current time or the recent past;
- The sample size is small and non-representative;
- Primary data analysis is qualitative.

In the second phase, considering the research question R2 (How do the influence factors affect each other, positively, or negatively?), the System Dynamics (SD) method [20] is used:

- As it helps to obtain a basic understanding of the feedback concepts in ASD.
- SD is a simulation methodology enables to model complex system considering the influencing factors, including the soft (subjective) factors which can have a critical influence on the whole project. Factors such as motivation, team management, learning or training and defined specifically within causal feedback loops analysis [76]. Causal feedback loop analyses how one or more factors affect changes in another factor.
- SD model uses symbols and feedback loops to express the influence factors in a specific though qualitative manner. It also provides the opportunity to incorporate simple, quantitative estimation of their effects.

- SD approach is based on the assumption that these important influences are the essential factors to project management and need much greater attention [76].

There are many modelling techniques developed and used so far such as analytical, continuous or discrete modelling, according to the modelling objective and perspective. However, SD modelling is chosen for this research because it provides a systematic method for description, exploration, and inspection of the dynamic behaviour of complex systems [48]. Figure 4.2 summarizes and provides an overview of the methodology that has been used to carry out the research:

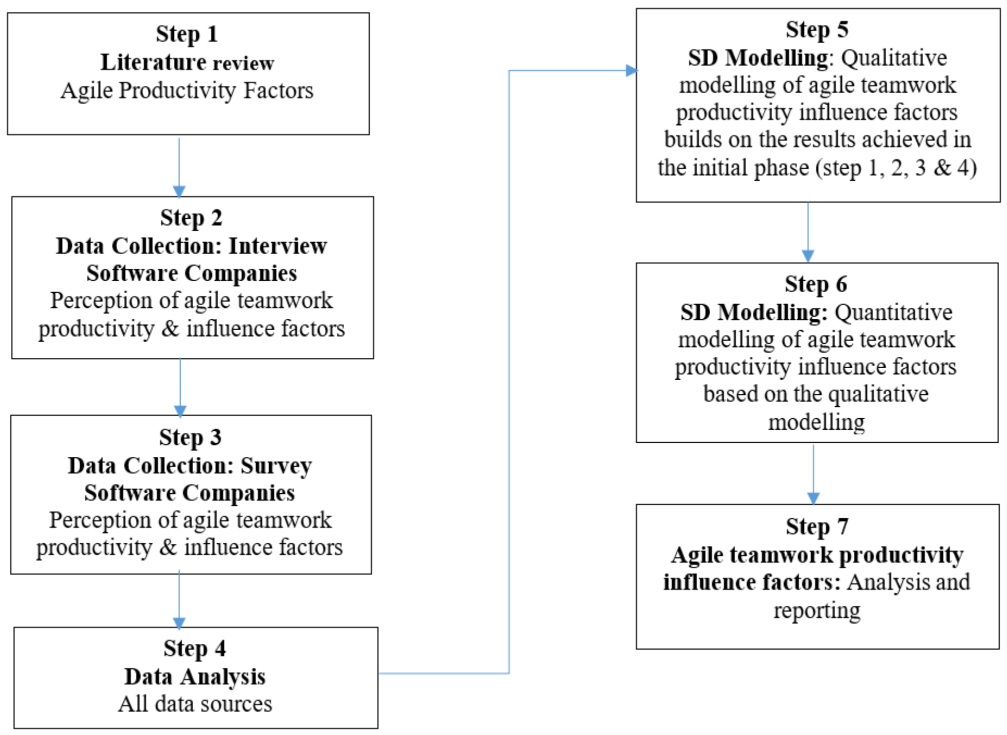

Step 1
Literature review
Agile Productivity Factors
Step 2
Data Collection: Interview Software Companies
Perception of agile teamwork productivity & influence factors
Step 3
Data Collection: Survey Software Companies
Perception of agile teamwork productivity & influence factors
Step 4
Data Analysis
All data sources
Step 5
SD Modelling: Qualitative modelling of agile teamwork productivity influence factors builds on the results achieved in the initial phase (step 1, 2, 3 & 4)
Step 6
SD Modelling: Quantitative modelling of agile teamwork productivity influence factors based on the qualitative modelling
Step 7
Agile teamwork productivity influence factors: Analysis and reporting

Figure 4.2: Overview of Research methodology

The following subsections give a brief overview of the research methods used in this study to answer the research questions. Firstly, it describes the primary (literature review) and secondary (interview and survey) information collection approach used to investigate the influence factors.

Secondly, it describes the SD methodology which involves the construction of CLD (qualitative) and a simulation model (quantitative) for ASD teamwork productivity influence factors analysis. The detailed procedures are given in the respective chapters (5 and 6). In these subsections, mostly focus on the characteristics of each research method.

4.2.1 Survey Approach

A survey is not just the process for collecting data through a questionnaire or an interview [88]. Rather the objective of a survey is to produce quantitative or numerical data, which describe aspects of the target population [89]. "The target population is the group or the individuals to whom the survey applies" [89]. According to Kitchenham, the main steps for conducting a proper survey are [89]:

- Setting the objectives;
- Survey design;
- Developing the survey instrument (the questionnaire);
- Evaluating the survey instrument;
- Collection of valid data;
- Analysing the data.

The first step in any survey research (or any research) is to determine objectives. Each objective is the expected outcome of the survey [88]. Survey objectives can be derived from literature reviews and other surveys [88] [89] [90]. According to Kitchenham, the types of survey objectives [89]:

- evaluate the rate or frequency of some features that develop in a target population

- assess the severity of some features or status that develops in a target population
- identify factors that influence a feature or status

This research focuses on the last type of survey which looks at the relationship existing among factors and conditions within a target population. According to Kitchenham, there are two common types of survey design [89]:

- **Cross-sectional**: This type of study analyses data of variables collected at one fixed point in time across a target population. These surveys offer researchers a sort of snapshot in time and give an idea about how things are for the respondents at a particular point in time that the survey is administered. For example, a researcher may poll all the members of a software development organization at a particular time to find out what activities they are working at that exact time. This type of information gives a snapshot of what is going on in the organization.
- **Longitudinal**: This type of study is an observational study, providing information about changes in a specific population over an extended time. For example, a researcher wants to find out which disease affects young boys (in the age group of 10-15) then the researcher will observe the individuals over that period of time to collect meaningful data.

This research performed a cross-sectional survey in Bangladeshi software companies to gather quantitative data regarding the common agile practice used in the companies and the perception of productivity influence factors. Survey administration is another point to decide when designing a survey. Options include to administer survey:

- Telephone surveys;
- One-to-one interviews;

- Self-administered questionnaires (email and web-based surveys).

This study is more interested in interviews and self-administered questionnaires (online-based surveys). The online questionnaire is an effective way to collect information quickly and relatively inexpensively from the target group. The next step involves the development of a survey instrument. Survey instruments, which are usually questionnaires, are developed using the following steps [88]:

- Search the relevant literature;
- Construct (create or re-use) a questionnaire;
- Evaluate the questionnaire;
- Document the questionnaire.

Once the instrument is created, it is essential to evaluate and pre-test the questionnaire. The pre-testing is done to check that the questions are understandable. It helps to assess the likely response rate and the effectiveness of the follow-up procedures. This study has done the pre-testing with the selected students of the Institute of Information Technology (IIT), Dhaka University, who already have working experience in software companies with agile practice. The collection of valid data requires a suitable sample group that is:

- Really represent a larger population;
- Suitable to involve in the survey;
- It is not highly expensive to query.

The last step, analysing the survey data and there are many types of survey analysis. In this study, the analysis relies only on simple descriptive statistics which mainly allow for the presentation of frequency distributions and reliability statistics. Each question group have been analysed for reliability.

4.2.2 System Dynamics Approach

It is found in the software production literature that the production environment is complicated. For this reason, the management of software development has turned out to be challenging [56] [76]. Despite the increasing acceptance of the agile methods, insufficient research has been empirical on the effect of software development productivity [10]. In order to better understand the system and come up with better policies, there is a need to establish a method that allows us to model and understand the nature of the system. The method must allow to represent complex systems, and simulate the relationship between variables over time [28].

The System Dynamic (SD) approach is a method that focuses on just the above-mentioned necessity. SD introduced by Jay Forrester of the Massachusetts Institute of Technology (MIT) in the 1960s as a modelling and simulation methodology for studying complex systems [20]. SD methodology has been applied by many researchers for studying and managing complex feedback systems, where feedback is understood as a closed sequence of causal relationships [48] [56] [57] [91]. The concept of a feedback loop reveals that any actor in a system will eventually be affected by its own action over time. According to Sterman [20], the concept of multiple feedbacks with time delays can create incorrect perceptions in a system. This incorrect perception can be analyzed and corrected if the SD model is properly defined as key factors and influences inside the system itself. Therefore, the SD approach allows testing hypotheses and policies in order to better understand the system behaviour or to change the perceived behaviour [20].

SD modelling consists of qualitative (or conceptual) and quantitative (or numerical) analyses [55]. The qualitative analysis includes the formulation of studied problem structures and identifies system variables and cause-and-effect relationships (causal feedback loops). The causal feedback loops are represented by a Causal Loop Diagram (CLD) capturing the underlying feedback loop structure of the studied situation/problem. CLDs can be transformed into a simulation model, also referred to as stock and flow and calibrated for quantitative analysis using computer simulation [19] [20]. The building blocks in SD that are essential for modelling behaviour and providing policies: feedback and causal loops, stock, and flows, are described in the following sections.

Feedback and Causal loops

Each factor that affects the variable is affected by other factors [5]. Some factors may be the reaction of the same action [48]. For example, variable X affects variable Y and, in turn, how variable Y affects variable Z through a chain of causes and effects. In system dynamics, this reaction is called feedback. A feedback loop is a closed sequence of causes and effect's actions and reactions [20]. All dynamics begin from the interaction of feedback. There are two types of feedback, reinforcing (positive) and balancing (negative) feedback. The positive feedback loops reinforce or increase the processes in the system. Negative feedback loops are self-correcting systems that prevent change [20]. Causal loop diagrams (CLD) are often used in SD to capture and track feedback in the given system. CLDs can express a hypothesis about the causes of dynamics [19]. Causally related variables indicate how the dependent variable performs when the independent variable changes [20]. In CLD, this behaviour is represented with the links' polarity which can be positive or negative. Technically, a CLD consists of words or phrases, which are linked by curved arrows, each of which has attached the sign (positive or negative) and occasionally a time delay symbol [20]. The arrow represents a causal relationship between two factors. The sign is symbolized by '+'indicating the two related variables change in the same direction, or '-' showing the two linked variables change in two different directions. In Figure 4.3 and Figure 4.4, two kinds of arrows used in a causal loop diagram to portray the relationship between two variables, birth, and population. The polarity is given by the $+$ or $-$ sign at the arrowhead. When there is a positive relationship between two variables, an increase in one variable provides an increase in the other. If there is a negative relationship between the variables, an increase in one provides a decrease in the other.

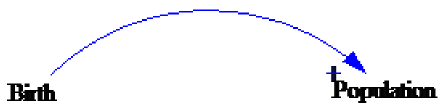


Figure 4.3: Example of a positive cause-and-effect relationship

Figure 4.3, an example is the relationship between births and population. An increase (or decrease) in births increases (or decreases) the population above (or below). Here, the variable birth has a positive effect on the variable population.

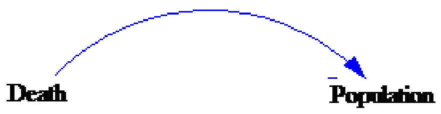


Figure 4.4: Example of a negative cause-and-effect relationship

Similarly, in Figure 4.4, the variable of death has a negative effect on the variable population. An increase (or decrease) in deaths decreases (or increases) population below (or above).

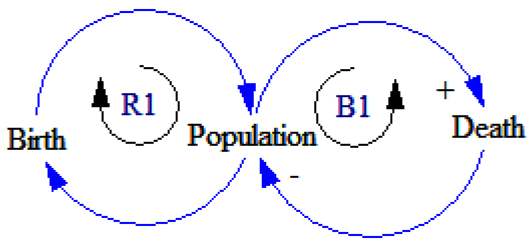


Figure 4.5: Positive (reinforcing) and negative (balancing) feedback loops

Figure 4.5 shows the feedback loops between the total population and births and deaths. When the population increases, the number of human increase and more births will occur. With more births, the population increases. When the population increases, more humans reach old age and will die. Increased population leads to an increased death-rate, and the overall population will decrease.

This simple system provides the reinforcing and balancing aspects of a CLD, and shows how two loops are causing the behaviour in the population-stock.

Stock and Flows

Another main idea of SD modelling is stock and flow along with feedback. CLD is useful in many situations but it has some limitations. CLD is unable to capture stock and flow variables which is an essential step for simulating the dynamic behaviour of the given system [20]. The commonly used constructs in SD models are a level, a rate, a source and a sink [20].

Stocks (also called levels or state variables) are an accumulation, or integration, of flows over time and characterize the state of the system. Stocks generate information needed for further decisions and actions [20], e.g. work to be done, developed software. Stocks can change their content only through inflow or outflow. In SD these flows, which increase and decrease the stocks, are called rates, e.g. software development rate, assimilation rate, etc.

These rates will always start somewhere and end somewhere. However, there are situations where the origin or the destination of the flow is outside the scope of the developed system, e.g. software delivered to customers. In such cases, the flow's start point is called a source and the flow's endpoint is called a sink.

There are three kinds of variables commonly used in the SD model when a system is constructed and simulated. The definitions of each of them are described below and graphical notations are presented in Figure 4.6.

- A constant is a variable that is initialized at the first time-step of a model and kept constant during the simulation run.
- The stocks of a system are also initialized at the beginning of a simulation. Unlike the constants, stock accumulates its value over the model run. The operating inflows and outflows will change this value.

- An auxiliary variable calculates information and forwards it through the system. The auxiliary variables are calculated by each time-step through the model run, e.g. communication overhead %, or simply constants, e.g. average meeting time.

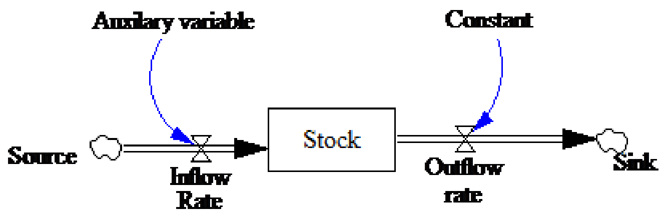


Figure 4.6: Representation of auxiliary and constant variables with rate and stock

Nonlinear relationship

In a general mental model, the relationship between the two processes is linear. Such as, if X amount is doubled, then the effect on Y is doubled as well. In the real world, however, it is unlikely that effects are proportional to the cause over time [19]. This is one of the difficult aspects of understanding the cause and effect relationships around any real system. The feature for a relationship at once may be different from its feature at another time [54]. The software development system also applies similar kinds of relationships as it involves sequences of iteration of the development activities. These iterations make it difficult for the management to locate the exact point for planning and reporting [69]. As a result, it becomes critical for management to understand the non-linear relationship between variables.

SD simulation model allows identifying the non-linear relationships over time with the help of stock and flow and CLD diagram. The ability to simulate policies and their influences on software development productivity will provide valuable insight for project managers [76]. Using the SD model, the project manager may understand the relationship between factors over time and when to adjust their approach due to non-linearity. This is the strength for the SD method, and why this research chooses to utilize it for modelling agile teamwork productivity influence factors.

4.2.3 System Dynamics Modeling Procedure

This research has adopted the five steps of SD modelling described in [20] for the development of ASD teamwork productivity influence factors, that covers:

- Problem statement;
- Data-gathering and analysis;
- Conceptual (qualitative) model building;
- SD (quantitative) model building;
- Scenario planning and model validity.

A flowchart representing different stages of the ASD teamwork productivity simulation by the proposed SD approach is shown in Figure 4.7

The first step, problem formulation delimitation involves the formulation of the problem and the description of the objectives and goals. Determining the model boundary is an important issue and an iterative process. This study formulated and defined the original problem at the beginning of this thesis.

The second step, data collection, and analysis start with collecting and analysing data for SD modelling. SD literature proposes several qualitative and quantitative methods for data collection such as survey, observation, discussion, interviews, existing data and observation [20]. Data collection is done in the first phase of the research, described in Chapter 5.

The third step, conceptual model building is about the qualitative or conceptual model building. This includes combining the relationships between the

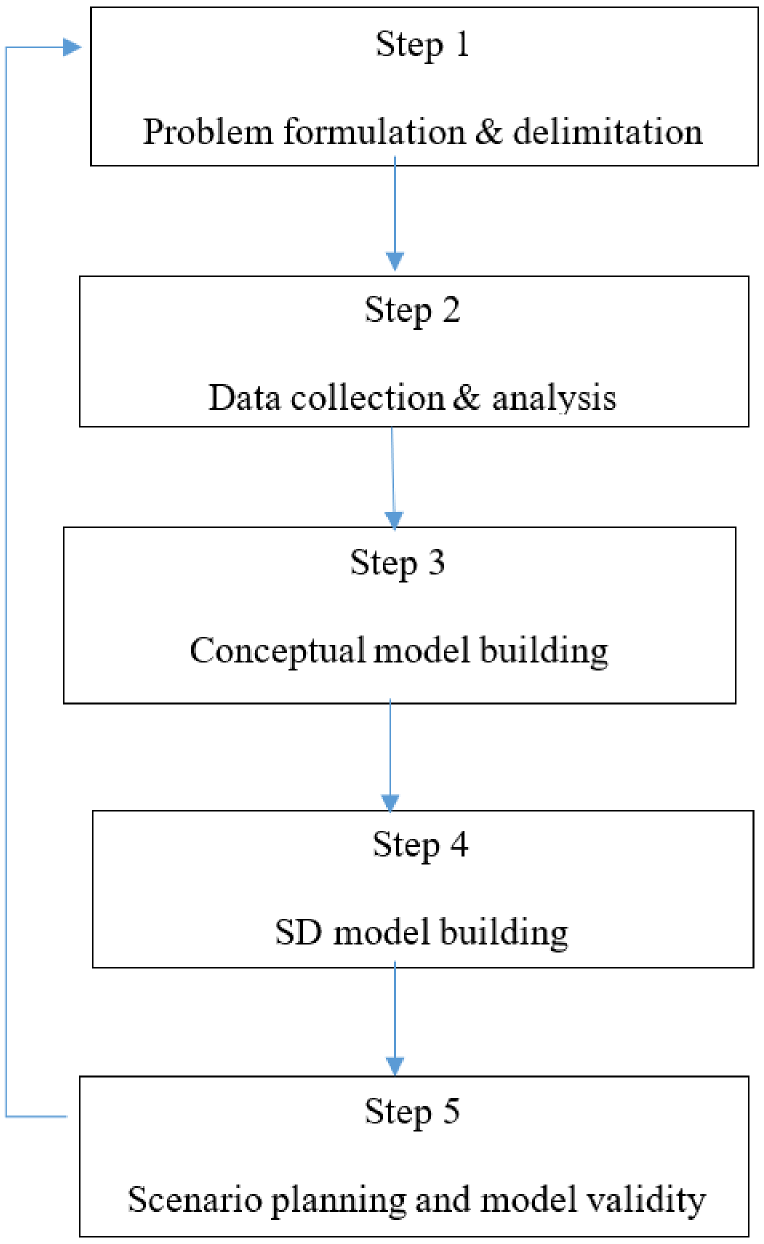

Step 1
Problem formulation & delimitation
Step 2
Data collection & analysis
Step 3
Conceptual model building
Step 4
SD model building
Step 5
Scenario planning and model validity

Figure 4.7: Flowchart of different stages of ASD teamwork productivity simulation by SD approach

relevant variables. In order to get a basic understanding of the feedback concepts in the given system, these relationships are depicted with the help of CLD.

The fourth step, SD model building includes the conversion from qualitative to the quantitative simulation model. This includes the conversion of data into simulation elements, such as stocks and flows and the quantification of these data.

Before running the simulation model, logical checks and variables tests are done.

The last step of scenario planning and model validity includes various simulation tests. This is accomplished to answer different kinds of questions by running the final simulation model in various scenarios. This is needed to validate the model and to check suitability for the intended purpose. The model validity step aims at building the right model.

4.3 Summary

This chapter only contains the planning of how the research has been performed and not the execution itself, which is described in chapters 5 and 6. Therefore, this chapter has discussed the most appropriate approach to conduct the research and method of collecting data. This research chose two different research approaches to answer two main research questions in an appropriate way.

This chapter has discussed the survey approach as a data collection method and system dynamics as a modelling approach. Firstly, the method and steps to accomplish the survey are described. In the later sections, it provides an introduction to system dynamics so that different concepts used in the modelling can be easily understood in the later chapters.

Chapter 5

Identification of influence factors affecting agile teamwork productivity

The productivity of the development team is important for a successful software project. The agile team, which is the most dynamic element and the human input in the software development industry, gain more interest to study their productivity. The aim of this research is to identify and understand the complex interdependences and underlying structures at the team's perception level, which influence agile teamwork productivity over time. In order to achieve this, the main factors that affect teamwork productivity are identified. Through a systematic literature review, interview and surveys of different agile teams were conducted to collect and select impacting factors, and they were evaluated and ranked to identify the most influential ones.

5.1 Survey Approach

In order to answer research question R1 (what factors do have an influence on agile teamwork productivity and how is this influence from the team perspective?), this research chose the survey method primarily for collecting and exploring the data. This research performed a cross-sectional survey (see section 2.2.1) in Bangladeshi software companies to gather quantitative data regarding the common agile practice used in those companies and the perception of productivity influence factors.

When designing the survey questionnaire, three other goals were added with the research question:

- Which agile practice (s) are adopted in software companies to impact on a given team's productivity?
- What is the criterion for measuring or perceiving productivity in their software company?
- What were the main reasons for the failed agile project (if any)?

Considering the current popularity of agile practices, it is relevant to investigate the adoption, self-organization, application domain, effect of agile and agile project team's interaction with regard to agile productivity. This research work investigated the above-mentioned domains in the Bangladeshi IT context with the help of a survey.

5.1.1 Identification of different factors affecting agile teamwork productivity

Data collection

There are three important objectives of collecting information; to determine what factors affected the productivity of agile team members, to determine how these factors impacting project productivity in the team's perception and to determine the significance of the factors.

1. **Literature review**: Keywords such as "productivity", "agile productivity influence factor", "system dynamics" and "agile teamwork" were used to search for related work in digital libraries. Significant findings from related work were not only helped in identifying some factors but also helped in the determination of the impact the factors have (positive or negative) on other variables in the project. The estimation of this impact would be vital in the calibration of the SD model.

2. **Interview**: Primarily, to collect quantitative data, a set of semi-structured interviews and face-to-face discussions were conducted with twelve key project members from four software companies in Bangladesh. All of the respondents had experiences in agile software development methods, such as XP and Scrum. The roles of the respondents included project managers, scrum masters, developers and project owners. The semi-structured interviews mainly focused on their working team, their team productivity influence factors and experience of introducing agile practices in Bangladesh.

3. **Questionnaire/survey**: In an attempt to identify the perceived influencing factors and their impact on agile team members, data was collected with the help of an online survey.

   (a) **Questionnaire design**:

   Using the factors identified in the literature review and interview, a questionnaire consisting of 17 questions was developed [92]. The questionnaire details are described in Appendix A. Most of the questions were based on a previous global survey on agile methods conducted by [8] and country-specific survey on agile productivity factors [5]. The questionnaire was structured in 4 parts. The first section was on demographic data, information about the project and organization. The respondents' details included their experience with agile methods, current position, current working project and status, working team size and organization name. The organizational profile included details about its' main activity, structure, size, mostly followed the agile method, an agile practice adopted and most used programming language. The second section was on the perception of success/failed project and the criterion for measuring/perceiving productivity. The third section was a set of 35 productivity influence factors. The last section was taken for any additional comments in order to allow the respondents to express their opinion more freely. To measure the significance of agile teamwork productivity influence factors, the respondents were asked to indicate the strength (high, medium or low) for each factor that they perceived influenced their productivity.

(b) **Questionnaire administration and selection of respondents**:

The questionnaire was emailed to a total of 25 software companies in Bangladesh, requested for distribution within the organization through Human Resources departments. The company selection criteria for this preliminary study were: companies using agile methods for at least 1 year, developing software for both offshore and local markets, and top listed software companies in Bangladesh [93]. Survey notifications were also sent to members of the Agile-related group (Agile Bangladesh) with announcements on the Facebook group. 60 responses from 18 companies responded to the questionnaire. In the online survey, respondents were requested to fill up the questionnaire based on an ongoing project or they had completed recently (regardless of whether the project outcome was positive or negative). Data were collected throughout a period of eight months in 2017 (January-August). In order to ensure the quality of data, team members were all self-selected by their organization based on their work roles as members of existing agile teams. Therefore, respondents who responded to survey questionnaires were already aware of the agile team environment and most experienced. The filled in questionnaires were then analysed to identify factors, which have major influences on agile teamwork productivity. Currently, more software companies are being requested to participate in this survey, as the plan is to collect more than 150 responses from different agile teams.

4. **Author's assumption**:

   Where necessary, the author's assumptions are used in the development of the model. Such assumptions will be permitted and perhaps, moderated by experienced agile practitioners via interviews and questionnaires.

5.1.2 Selection of factors for inclusion in the model

Data analysis

Factors affecting agile teamwork productivity are rarely independent of the others, but a set of factors interacting with each other to build the final result [74]. The important factors identified in literature and interviews were taken as a starting point for the system approach in this research. In total, 35 factors were chosen for preliminary analysis. In order to create a system model to analyse the teamwork productivity, it is required to determine the importance of the individual factor, their correlation with one another and their effects on productivity itself. The agile team members were asked to fill in the questionnaire to indicate the strength (high, medium or low) of the factors that they perceived influenced their productivity [91].

The procedure followed to extract the agile team member's perception of the influence factors affecting their productivity can be summarized in the following steps:

1. Convert the qualitative scale to a quantitative one. The qualitative scale of high, medium or low was converted to a number scale of 3, 2, and 1, respectively.

2. Find the total score of each factor for frequency analysis. Then, the arithmetic mean of the total counts were calculated to eliminate the factors below the average / (Table 5.1) mean 2.26.

3. Cronbach's Alpha () coefficient for internal consistency reliability was calculated for the identified factors [94]. Cronbach alpha () is widely used as an estimator for reliability tests [94]. In a good solution for indicating high internal construct validity, Cronbach alpha ranges between zero and one - the larger the value, the more stable the factors. Generally, the value of 0.70 is accepted as the minimum desired value of reliability [94]. In this study, the 35 factors were tested for internal consistency, using the 60 respondent's data. The results, shown in Table I had values ranging from 0.877 to o.887, all of which were considered acceptable (Cronbach's alpha higher than 0.70) and Cronbach's alpha for 35 factors was .885

4. From step 2, twenty factors (Table I, highlighted) were selected as the most influential ones (above the average/mean).

5.2 Survey Results

This study used reliable survey instruments that can be helpful for comparing new results with the previously studied results [5] [8]. However, there is no data

Table 5.1: Agile Teamwork Productivity Influence Factors- Questionnaire Results From Frequency Analysis: Arithmetic Mean, Std.Dev and Internal Consistency Test

Sl	Factors	Description	Mean	Std. Dev	Cronbach's Alpha
1.	Staffing	The right persons should be selected	2.73	.482	.882
2.	Size of team	Small and mixed team	1.93	.362	.885
3.	Project Complexity	Database size, architecture, complexity of interface to another system, code, interface complexity to hardware and software, logical problem	1.97	.551	.883
4.	Team Leadership	Shared leadership can be shown by several team members	2.57	.621	.880
5.	Mutual performance monitoring	Being aware of other team members' performance	2.37	.637	.881
6.	Backup Behaviour	Being available to assist other team members when needed	2.32	.651	.879
7.	Team orientation	Assigning high priority to team goals and participating willingly in all relevant aspects of the team	2.48	.651	.881
8.	Adaptability	Response to changing conditions, internal or external	2.45	.622	.883
9.	Feedback	Giving, seeking, and receiving of information among team members	2.48	.624	.880
10.	Mutual trust	Shared belief that team members will perform their roles and protect the interests of their team-mates	2.62	.524	.881
11.	Coordination	Team members executing their activities in a timely and integrated manner	2.75	.474	.880
12.	Communication	Exchange of information between two or more team members in the prescribed manner and using appropriate terminology	2.65	.606	.882
13.	Team members are appreciated for working long hours	Team incentive for working overtime to finish a job	1.72	.761	.884
14.	Team reward	Overtime reward for working extra time (then or later)	1.93	.733	.882
15.	Adequate technical training for team	Offering appropriate training for new technologies	2.57	.563	.880
16.	Adequate team skills training for team	Communication, organization, interpersonal, etc.	1.78	.415	.885
17.	Turnover	Staff leave or entry in the project team	1.93	.733	.884
18.	Key personnel Stayed throughout the project	Impact of personnel turnover on team	2.37	.610	.882
19.	Reuse	Software products, processes, artifacts, including components, frameworks, and software product line	2.38	.585	.879
20.	Goals	Establishment is critical for the success of the team	2.37	.637	.879
21.	Intra group wage inequality	Fair wage	1.90	.775	.883
22.	Dealing Cultural differences	Cultural differences among the offshore organization	2.15	.659	.882
23.	Self-management	Most work-related decisions are made by the members of the team rather than a manager	2.13	.430	.887
24.	Task variety and Innovation	The team get a chance to learn the different tasks the team perform to meet the workload needs of the team	2.40	.694	.877
25.	External Dependencies	Waiting for customer acceptance/for a component; interacting with external customers; publishing version of system/of data model across different environments	1.90	.511	.884
26.	Tool Usage	Use of CASE tools	2.13	.623	.880
27.	Programming Language	Programmer's experience and skills	2.13	.747	.883
28.	Schedule Pressure	The impact of intangible project pressure	1.95	.429	.884
29.	Pair Programming	Two programmers working collaboratively to develop software	1.80	.514	.883
30.	Resource constraints	e.g. timing, reliability, storage, team size, and project duration	2.37	.637	.878
31.	Team Management	Quality of management, conflict management, task assignment, administrative and formal coordination	2.55	.565	.880
32.	Motivation	To work on the project and in the company	2.57	.593	.880
33.	External project factors	Customer involvement, Customer expectation, Customer satisfaction	2.30	.696	.879
34.	Culture	Agile requires a true cultural change from plan-based approach, not only a simple change in the processes used	2.10	.796	.879
35.	Working environment	Suitability of the workplace to do creative work, collocation, e.g., windows, natural light, size of room and desk, meals provided	2.33	.629	.878

available on the state of agile development in Bangladeshi software companies to interpret this study sample representativeness. Interestingly, this study also found some similarities between Bangladesh and worldwide surveys [5] [8].

This section presents a summary of the results found in this research. Characteristics of the sample software companies and respondents can be found in Figure 5.1 to Figure 5.11. As can be seen from Figure 5.1 – Figure 5.3 summarizes the respondent's profile. The results show that 35% of the respondents cover the range of 2-5 years of experience using agile methods.

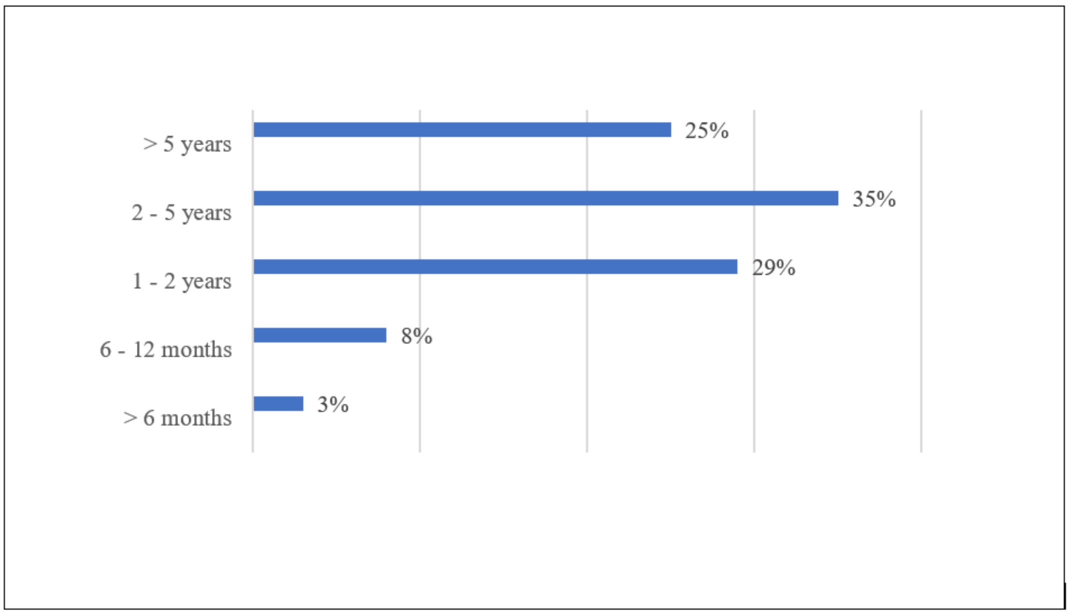


Figure 5.1: Agile practices experience

The respondents were working in various positions in their organizations, ensuring diversity in the survey. Figure 5.2 and Figure 5.3 present the team role of the respondents in their organizations and the respondent's main team assignment.

60% indicated themselves as a developer, 17 % as team leader/ manager, 10% as QA engineer, while the remaining 13% of the respondents are active in other roles, such as Scrum master, product owner and software architects. The majority of respondents (85%) are working on a development project and 12% on maintenance projects (Figure 5.3).

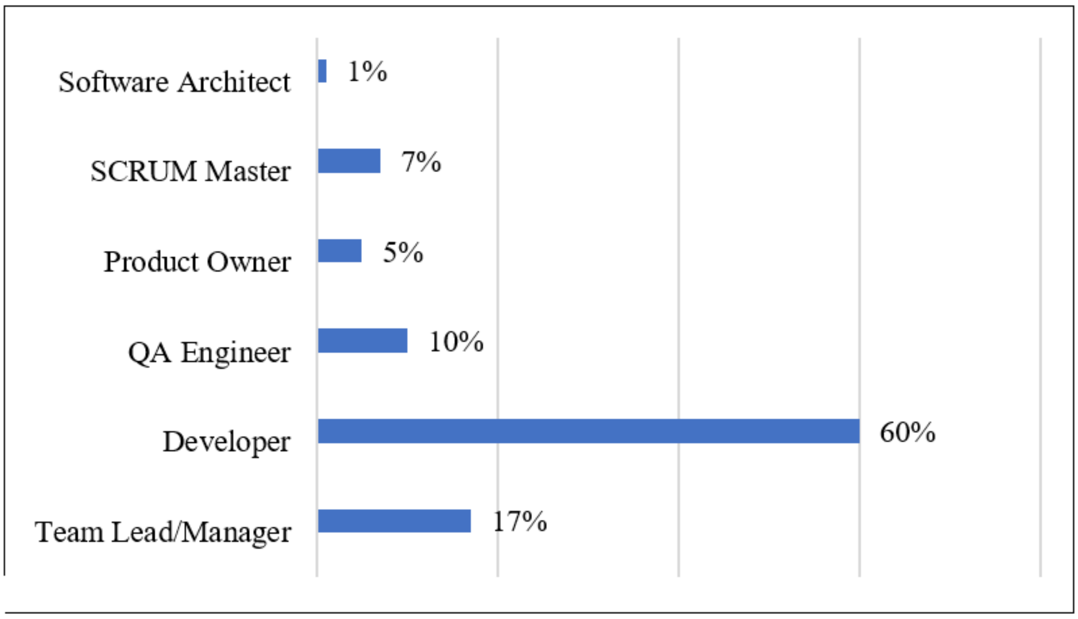


Figure 5.2: Team role in the project

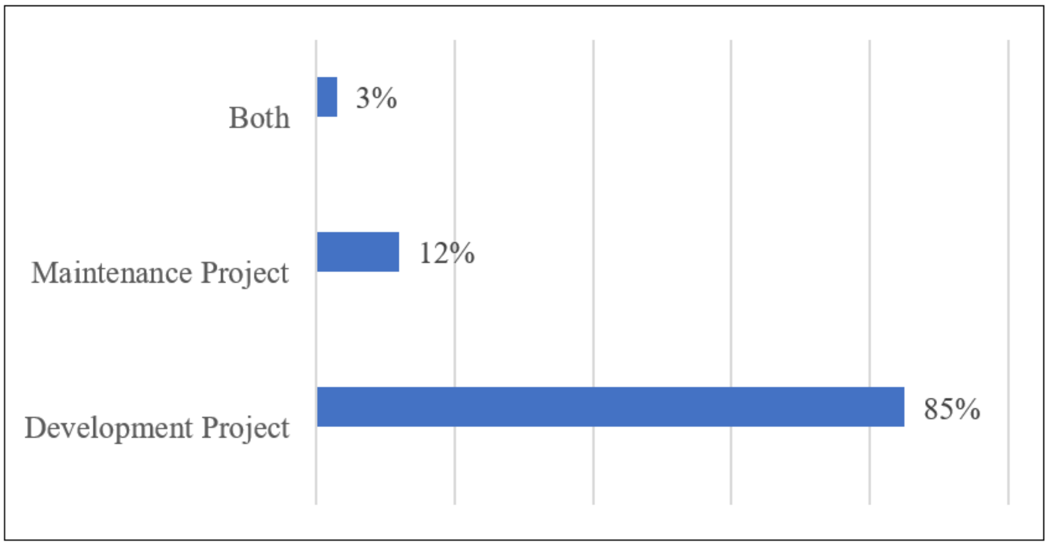


Figure 5.3: Main team assignment

The majority of the respondents' (42%) software organization's size is small, between 30-50 people (see Figure 5.4). However, 30% of organizations employ 100-150, and 12% employ more than 150 people.

As can be seen from Figure 5.5, Scrum is extensively used by software companies. 97% indicated Scrum and 3% chose Kanban and XP. Scrum is the most

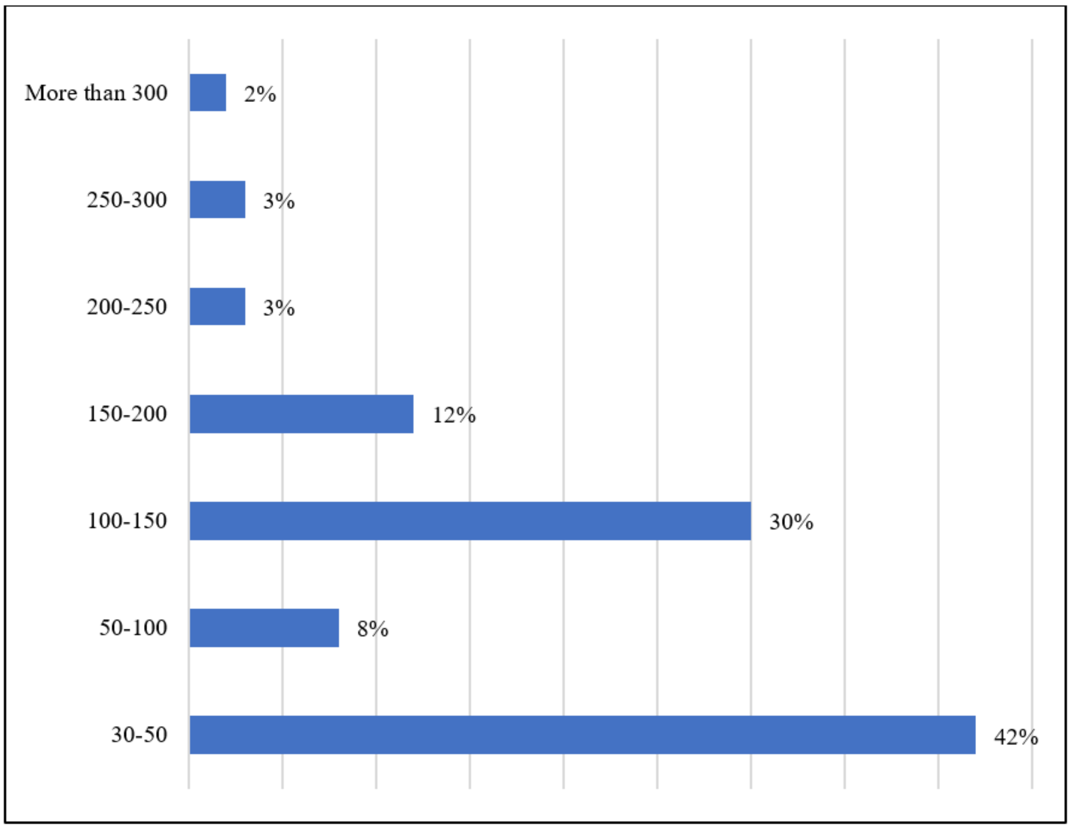


Figure 5.4: Size (people) of the software companies

popular Agile methodology also in [5] [8].

According to the respondents (50%), the frequently used programming language in their organization is C and then JavaScript followed by Java (see Figure 5.6).

Regarding the agile practices in use by the participating software companies, the results are well aligned with the results of a similar survey [8]. Figure 5.7 presents the most adapted practices are daily stand-up meeting, release planning, stories, and retrospective.

Figure 5.8 shows that lack of management support (e.g., resource constraint, team design choice, and motivation) is the main reason for failure in agile projects. In addition to this, respondents have mentioned another three more reasons in this extension of the previous study [2] [95] [96]. Integration failure, frequent change requests, and substantial funding crises are mentioned by the survey respondents.

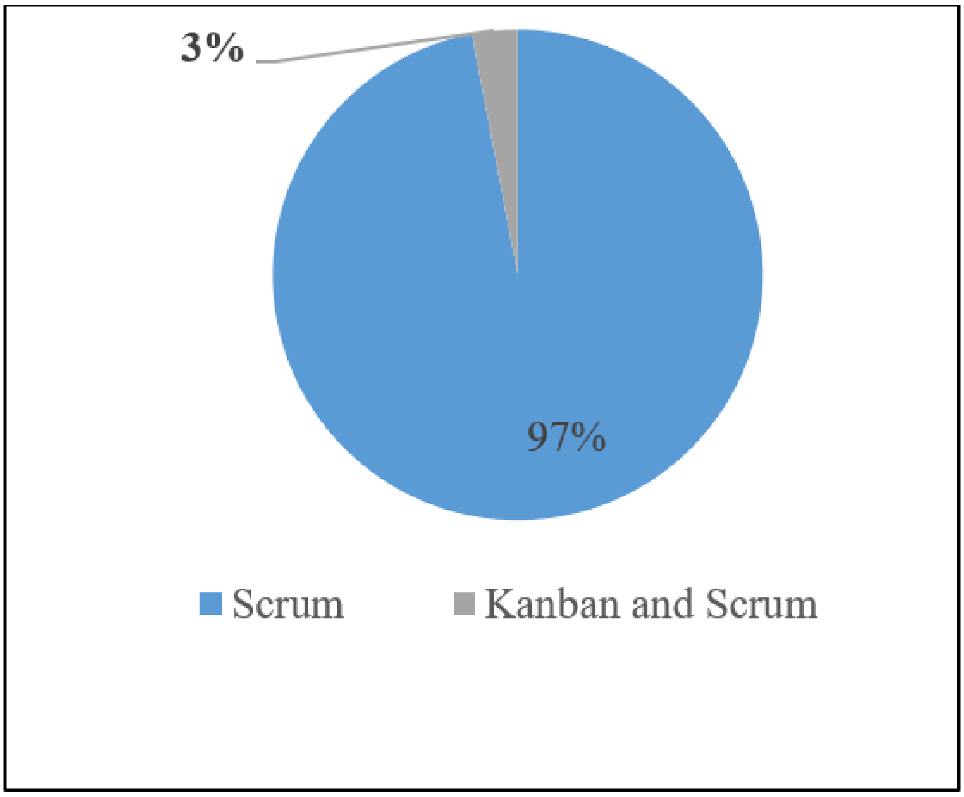


Figure 5.5: Most followed agile development method

Lack of experience with agile methods and the company culture are indicated as project failure reasons in similar surveys [8]. The most recent survey on agile acceptance and success or failure project results indicate a lack of experience with agile methods and the company culture is the main project failure reasons [8].

In most of the interviews, the team members could not define productivity as their performance measurement. A large number of them mentioned that team management has their own ways of measuring productivity. Although at the end of the project, the management assessed their productivity on the basis of timeliness and quality. At the same time, ten interviewees and survey respondents (Figure 5.9) also mentioned customer satisfaction as a criterion for measuring or perceiving productivity. Customer satisfaction is very important to software development companies in Bangladesh as a rising market for outsourced software destination. This study result also confirms the latest worldwide survey studies

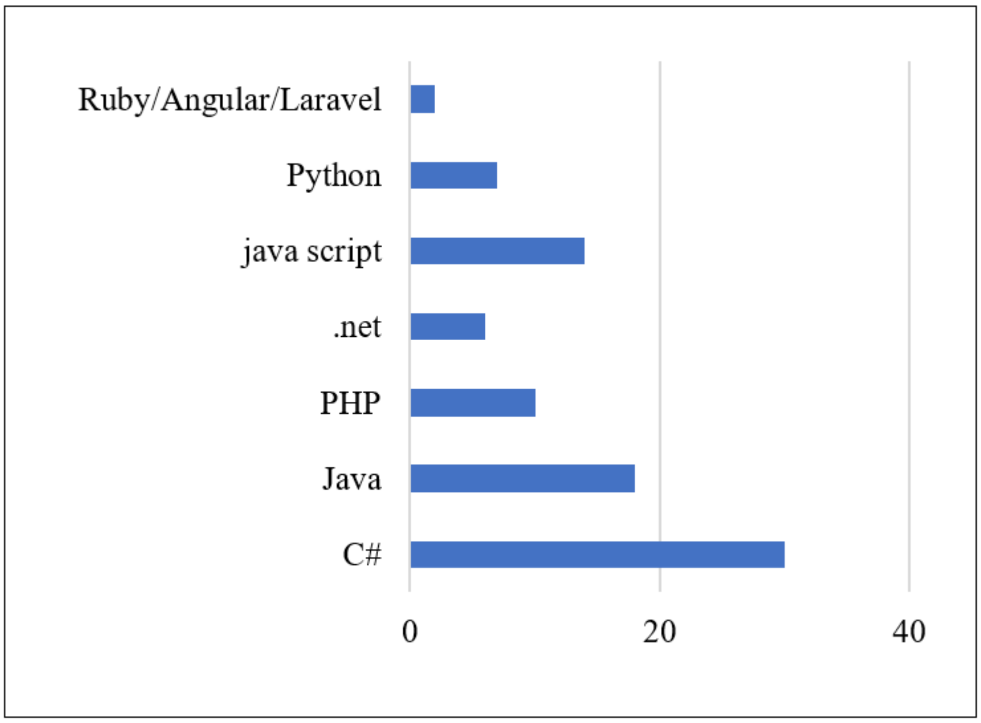


Figure 5.6: Programming languages used in software companies

that have shown customer/user satisfaction is the number one measure of an agile project's success [8].

According to the product owner interview, dealing with cultural differences among offshore organization influences teamwork productivity. Two main reasons behind this are time and culture differences. Sometimes, it becomes difficult to keep contact with the offshore client on urgent issues due to time difference between places. Moreover, offshore client's expectations are different, both in terms of their general culture and their views on life and work. The project developed within western cultures is different from eastern cultures. For example, daily traffic condition consumes most of the working time in Bangladesh, which makes the developers less motivated. Since staff is not rewarded enough for working long hours. However, schedule pressure can be easily dealt with overtime working because it is inexpensive in Bangladesh.

Five interviewees (project leads and managers) mentioned that culture is a big

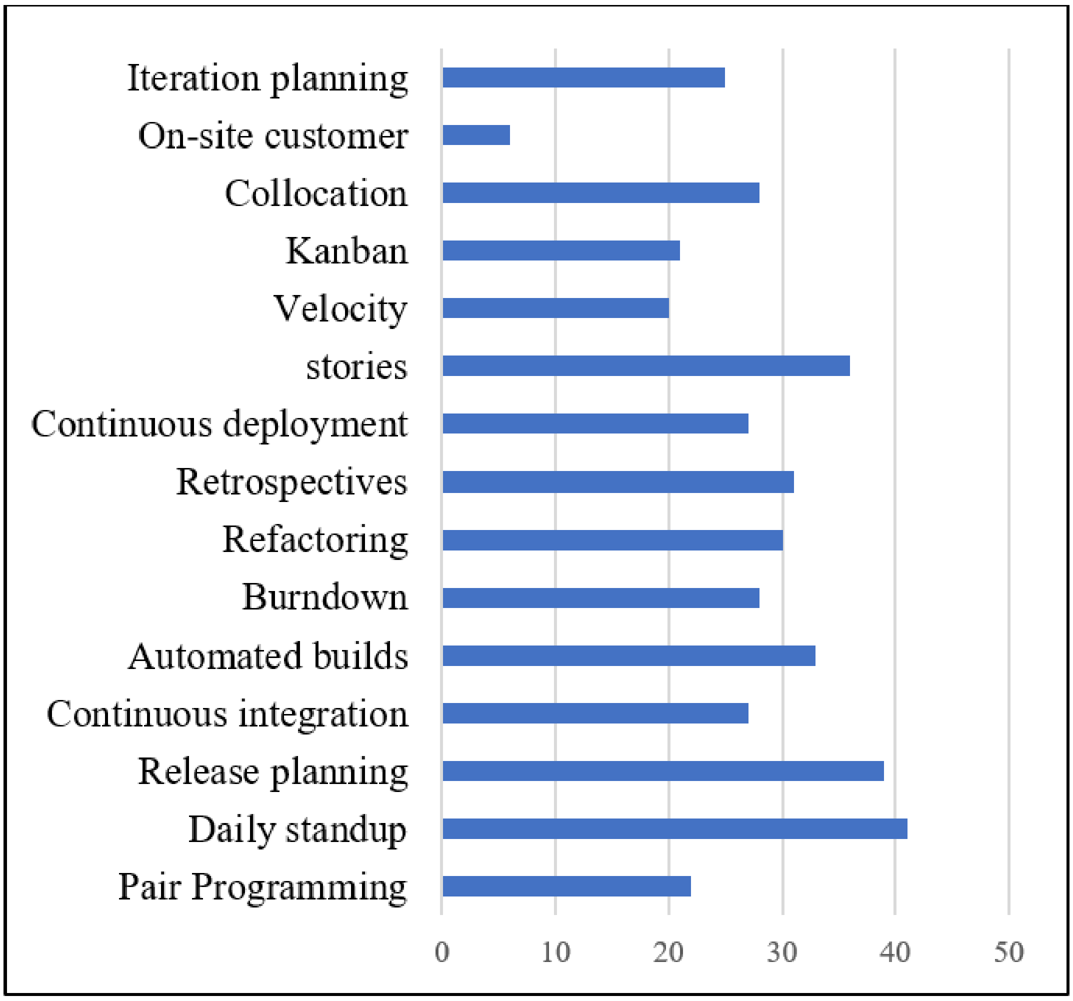


Figure 5.7: Agile practice adopted in software companies

barrier for working in an agile team. Even though it is not one of the most influential factors mentioned by survey respondents. The survey result shows (Figure 5.10) that the participating software companies' organizational structure and coordination are primarily horizontal (68%), where coordination processes are usually provided by an individual team member who communicates directly with other members or users on a one-to-one basis [5][97]. On the other hand, vertical coordination (32%) is usually implemented through project managers. The horizontal structure of agile involves self-organizing teams that work in an iterative fashion and deliver continuous additional value directly to customers [97]. Although the practice of self-organized teams conflicted with the cultural responses of social

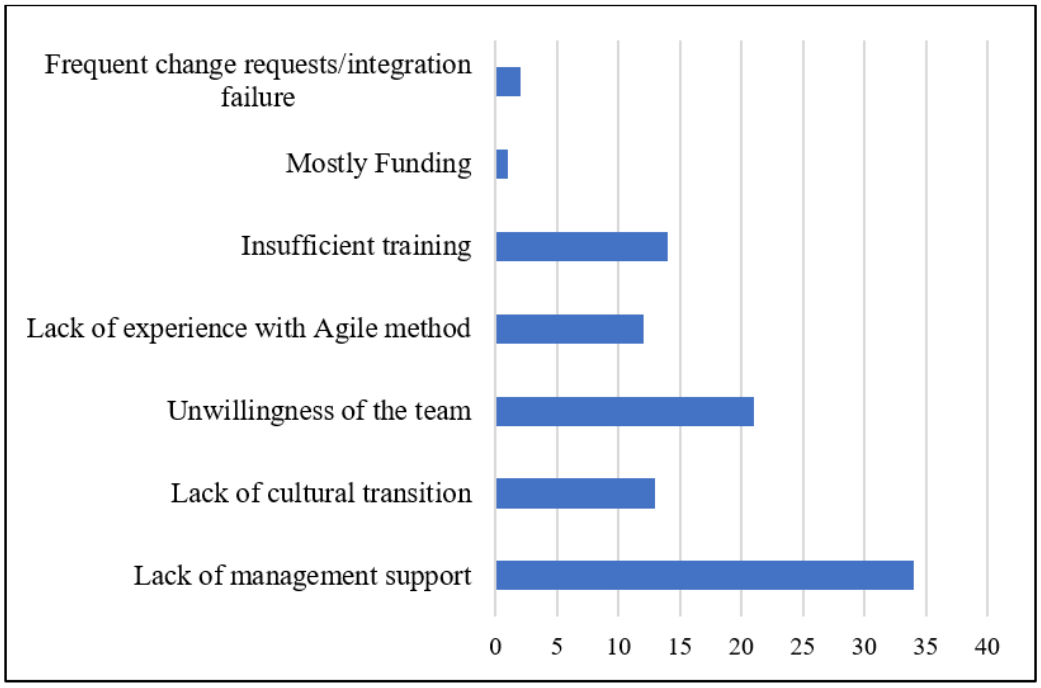


Figure 5.8: Main reason for failure in an agile project

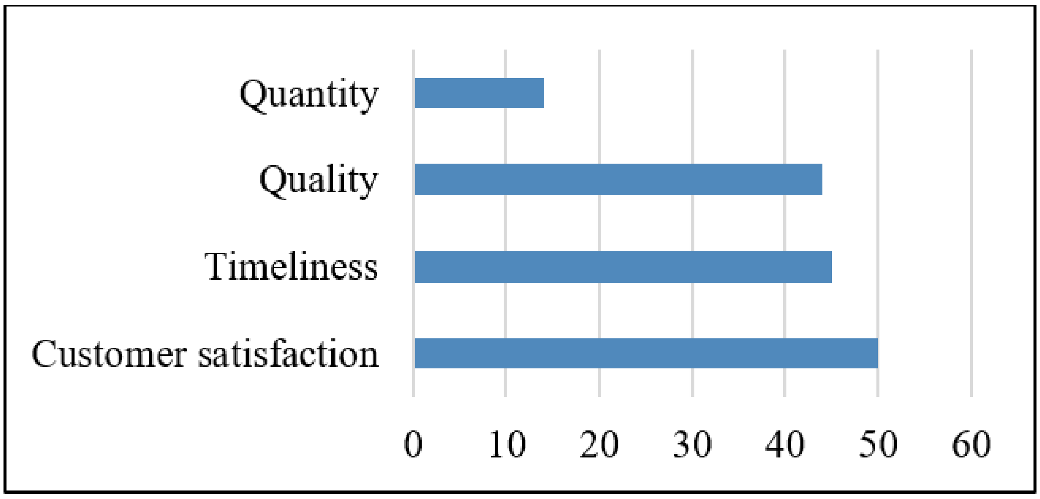


Figure 5.9: Criterion for measuring or perceiving productivity

hierarchy. According to Balasubramaniam et al., Social hierarchy recommends a top-down approach to decision making, which is different from a participatory approach and hinders teamwork [97]. In Eastern culture, workplace hierarchies are a common practice of being superior to others in authority, power, or status that are commonly accepted by subordinates. Team members look for clear instructions and accept their supremacy, also their own dependency on the superiors.

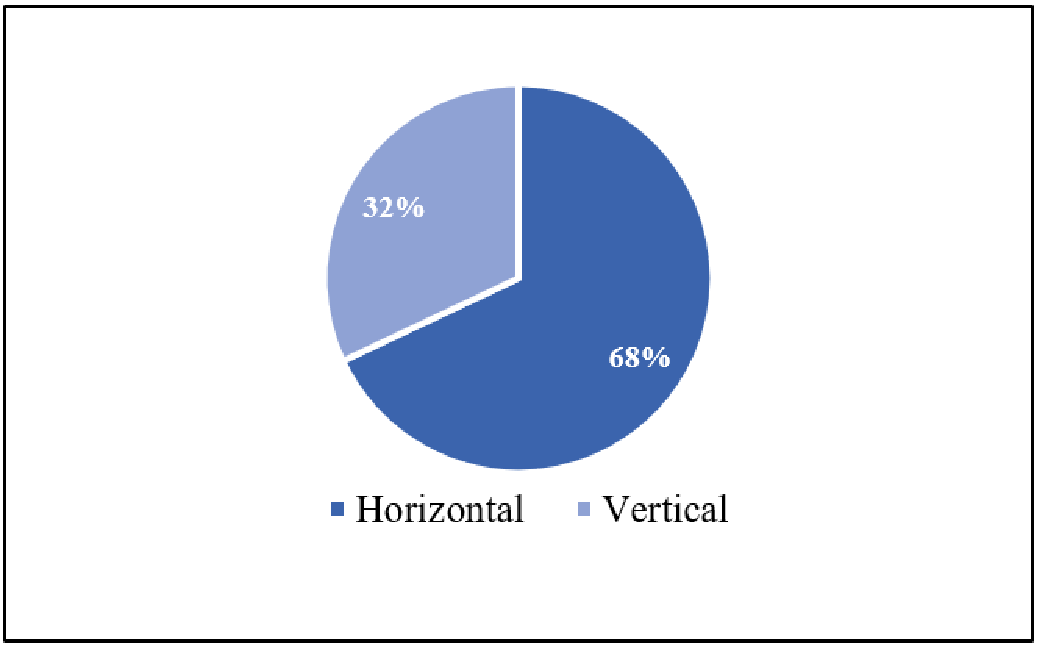


Figure 5.10: Organizational structure

Based on results found in the interview and survey of this study, it is perceived that social hierarchy is embedded in Bangladeshi software organizations and affects the implementation of the agile principle. In agile development, communication links together all other teamwork processes. Therefore, regular and informal meetings should take place among team members. The survey result shows that project/team management has more influence on productivity than self-management. Besides that most popular agile practice among the participating companies is Scrum and there is no such role as the project manager. In the agile approach, the team should be self-managed and work is coordinated by the team members [68].

From this scenario, it is evident that, even though the most followed organizational structure is horizontal, the social hierarchy culture significantly influences agile teamwork productivity. It is because the way team members communicate with team and customers, and more often to respect official hierarchy/top management (the cultural norm in Bangladesh), communication occurred between members at the same levels of the organizational hierarchy [97].

In addition, sometimes the language barrier hinders communication. Cultural transitioning from individual work to self-management team requires a reorientation not only by developers but also by management. This changeover of organizational culture and institutional process take time and resources. These begin from changes of individual perception and for this reason, project managers prefer fresher as a team member. Their software companies like to groom up with several activities such as training, community, and conference than changing the traditional mind set up of the team members.

Stable teams are associated with higher productivity, so avoiding changing team members to keep key personal throughout the project has a great influence on productivity (Table 5.1 and Figure 5.11). Sustainable pace is an essential part of agile development, and only by working regular hours at a reasonable level a team can produce a good flow of work [12]. Productivity grows over time through the development of the teamwork practices by the team learning process and not by doing overwork or compromising the quality to increase the team's productivity [5]. Moreover, teams are not rewarded enough for working long hours (Table I). This study's findings also indicate that schedule pressure has less impact on productivity and; timeliness and work quality are the most mentioned criterion for measuring or perceiving productivity (Figure 5.11).

Figure 5.11 provides highlights of the most influencing productivity factors that are perceived by the agile team members. These study results show that agile teamwork is highly dependent on team effectiveness. Offshore clients' satisfaction (external factors dependency) is very important for the organization. Team leadership and team orientation are very important for teamwork motivation. The factors impacting on agile teamwork productivity mentioned by the team mem-

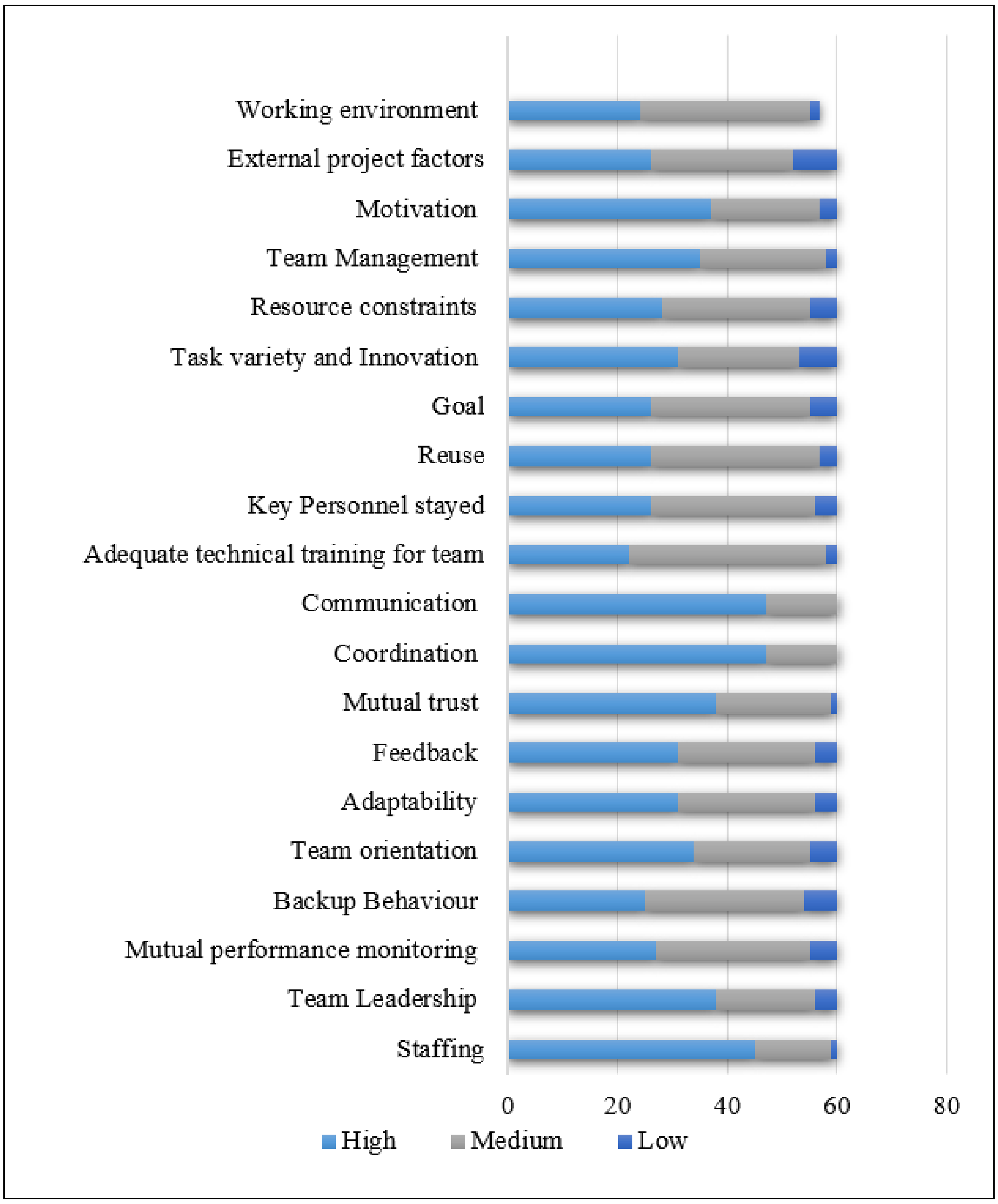


Figure 5.11: Agile team perception of productivity influence factors

bers suggested that feedback, team orientation, communication, coordination, and mutual trust improve team effectiveness. Eventually, this will enable the team to learn how to effectively manage relationships within the team in order to become more productive.

In sum, the study results show that some traditional productivity factors (from

Table 5.1) are still influential factors to agile software development teamwork productivity, even with the adoption of the agile practices. A transition to the self-managing agile team is one of the biggest challenges when introducing agile development in Bangladeshi culture. Agile implementation needs the mindset change of all the team members; investment in training and learning-oriented activities will make the agile team members more productive.

5.3 Summary

This chapter presents the preliminary results that address productivity factors in agile software development teams. The findings of this stage are the main influencing factors, which are motivation (external factors, customer satisfaction), team effectiveness (communication, coordination, mutual trust, leadership) and team management (staffing, Key personnel Stayed throughout the project, team design choice). The most cited and influential factors are Coordination, Communication, mutual trust and staffing (right person selected for the team). These factors are the most important for effective teamwork and team management in agile teamwork productivity.

According to the study results, lack of management support is found to be the most mentioned reason for any failed agile project. The most followed organizational structure is horizontal and the most followed agile method is Scrum. In addition, this study finds that due to social hierarchy culture influences, the self-managed agile team may not fully fit in their organization. This factor also hinders agile transformation from plan driven to a self-managed agile team.

The next chapter involved the development of a system dynamics model of agile teamwork, highlighting and consolidating the different productivity influencing factors identified in this chapter.

Chapter 6

Dynamic Modelling of Agile Teamwork Productivity Influence Factors

The objectives of this research are to identify Agile Software Development (ASD) teamwork productivity influence factors, to identify the relationships between cause and effect of these factors and to analyse the identified influence factors using the SD approach. On the basis of these objectives, the research wants to answer questions such as 'What are the main factors influencing ASD teamwork productivity and How do the influence factors affect each other, positively, or negatively?

This part of the thesis adopts System Dynamics (SD) approach to draw a model for analysing the dynamic behaviour of the ASD teamwork productivity influence factors. The results obtained from the survey approach described in chapter 5 (the second step of the SD modelling procedure) was used to construct these models.

Therefore, this chapter presents the second phase of the thesis which involves the construction of the conceptual model (causal loop diagrams) and simulation model (SD model) building for Agile Software Development (ASD) teamwork productivity influence factors. Finally, Scenario planning and model validity are done as part of the SD modelling procedure, discussed in chapter 4.

6.1 Introduction

This chapter presents the second phase of the research which involves the construction of causal loop diagrams and a simulation model for ASD teamwork productivity influence factors. This part of the research has adopted the System Dynamics (SD) approach to visualize a model for analysing the dynamic behaviour of ASD teamwork productivity influence factors, using the insights gained from the previous chapters. The results obtained from the survey results analysis described in chapter 5 are used to construct these models. This chapter first provides an approach to developing SD models so that different steps used in modelling can be easily understood. The objective of this chapter will then be to construct the qualitative model of ASD teamwork productivity using cause and effect feedback loops. The chapter will subsequently explain the methods used to quantify the variables and nature of the relationship among variables that determine the influence on the productivity of ASD teamwork. The inter-relationships that existed between different factors are defined by mathematical equations and the quantitative model of ASD teamwork productivity is built. Dynamic simulation is performed using a developed quantitative model to determine ASD teamwork productivity. Sensitivity analysis is conducted to assess the impact of different factors on ASD teamwork productivity. Using the developed SD model, the impact of alternative solutions to improve ASD teamwork productivity could be assessed.

6.1.1 Approach to developing the SD model

The system dynamics modelling approach adopted for this research was discussed in Chapter 4. As part of the modelling formulation process, it is necessary to identify factors that will be used for the model. Therefore, SD principles will be used to map and analyse the major factors that influence ASD teamwork productivity.

Formulation and construction of the SD model involve iterative elaboration, construction, and simulation events [20]. Figure 6.1 shows the structured process of creating an SD model based on previous chapter 4. In this phase of the study, firstly the qualitative model of agile teamwork productivity is constructed using the Causal Loop Diagram (CLD). The formation of CLD will be based on the data obtained from the survey approach (chapter 5). This CLD will assist to explain how the factors are influencing agile teamwork productivity directly or through other intervening factors. Based on the CLD, the quantitative model of agile teamwork productivity is constructed using stock and flow diagrams. Dynamic simulation of agile teamwork productivity is performed using a developed stock and flow model.

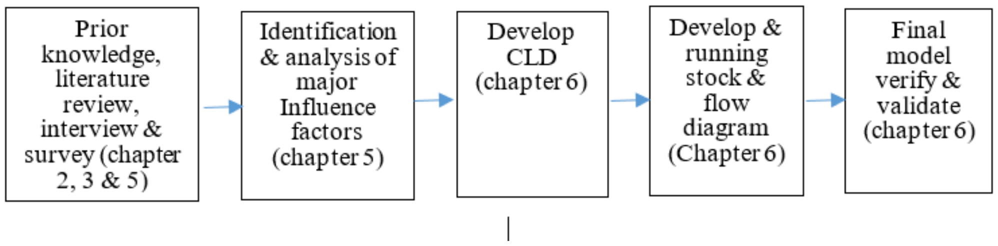

Prior knowledge, literature review, interview & survey (chapter 2, 3 & 5)
Identification & analysis of major Influence factors (chapter 5)
Develop CLD (chapter 6)
Develop & running stock & flow diagram (Chapter 6)
Final model verify & validate (chapter 6)

Figure 6.1: Process of creating the system dynamics model

6.1.2 System Dynamics Modelling Software Packages

The SD Society mentioned Dynamo, iThink/STELLA, Powersim Studio, and Vensim under the main tools sector on their website as SD modeling software packages [98]. Vensim [99], a free SD modelling software package is used for this research.

This software program can facilitate the development, exploration, analysis, and optimization of SD models. Vensim models graphically display the connections and feedback loops of the system. It is possible to instantly see simulation results for all variables on the screen and view more detailed results of any selected variable of interest with different analysis tools. It is very easy to perform simulation tasks and the powerful SyntheSim mode provides attached sliders for each model constant that can be used to adjust values and to instantly observe the effects of the adjustments [99].

6.2 Qualitative Modelling of Agile Software Development Teamwork Productivity

The objective of this research is to explore what factors influence ASD teamwork productivity, and how these factors interacted. Factors affecting agile teamwork productivity are rarely independent of the others, but a set of factors interacting with each other to build the final result [74]. Identification of different factors affecting agile teamwork productivity was carried out in the first phase of this research (chapter 5) i.e. through literature review, interview, and survey approach.

Software development productivity is a function of a complex set of "hard" and "soft" factors [20]. Most of the data required to understand the development and dynamics needed to determine the factors that influence agile teamwork productivity mainly are concerned with soft factors [9][44]. The SD approach is capable of incorporating the soft factors, which can have an important influence on the agile teamwork. Soft factors such as productivity, motivation, team management efficiency, customer satisfaction, and skilfulness and team effectiveness may be included and represented visually as a CLD. In the following section, the complex inter-related structure of different influence factors, such as motivation, team effectiveness, and team management are modelled using a qualitative SD approach.

6.2.1 Causal Loop Diagram (CLD)

Each factor that affects agile teamwork productivity is affected by other factors [5]. Some factors may be the reaction of the same action [48]. In system dynamics, this reaction is called feedback. There are two types of feedback – reinforcing feedback and balancing feedback. Sometimes feedback (or a reaction) does not occur immediately – the process contains delays. The dynamic system can be drawn as a model with circles of causality – including actions, feedbacks and delays [20] [54]. Technically, a CLD consists of words or phrases, which are linked by curved arrows, each of which has attached the sign (positive or negative) and occasionally a time delay symbol [20]. The arrow represents a causal relationship between two factors. The sign is symbolized by '+'indicating the two related variables change in the same direction, or '-' showing the two linked variables change in two different directions; and the time delay is shown by '//'crossing the arrow.

6.2.2 Initial conceptual model

The overall conceptual model (influence diagram) of agile teamwork productivity is presented in Figure 6.2, which is developed by identified influence factors in the first phase (chapter 5). All the factors in the influence diagram are directly taken from table 5.1. The factors are linked and connected to show their influences directly and indirectly. In Figure 6.2, the arrows indicate causal relationships and the positive or negative signs at the arrowheads indicate that there exists a positive or negative relationship between two factors/variables, respectively. For each of these links, the relationship is indicated as positive in the case of the same variation for both connected factors and negative for the opposite case. The SD is based on the ground that these underlying influences are crucial to project management and need special attention [80]. This resulting model is used to understand and explain factors and feedback relationships between the influencing factors over time.

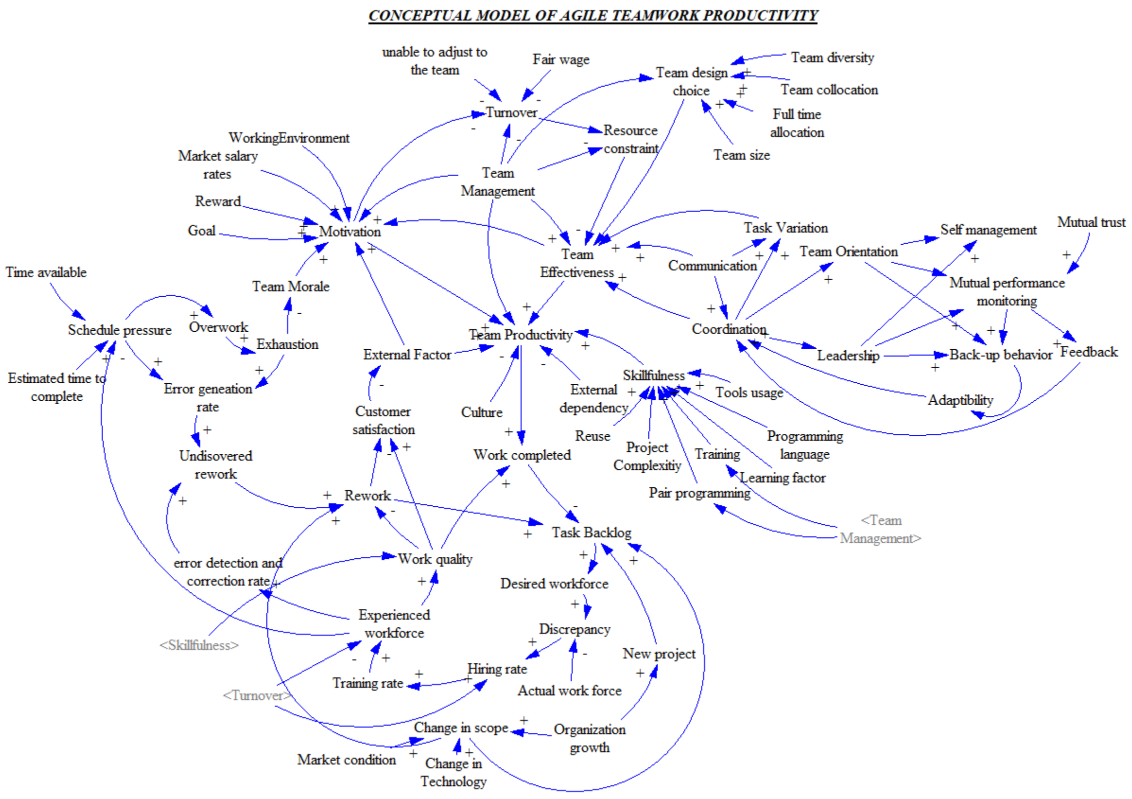


Figure 6.2: Overall Conceptual Model of Agile Teamwork Productivity Influence Factors

The previous study did not explain ASD teamwork productivity influence factors within causal relationships. This study describes the relationship in the initial conceptual model detailed in the causal loop diagram.

Below, this section presents the causal loop diagrams seeking to capture the dynamics of ASD teamwork productivity.

6.2.3 Analysis of causal links between agile teamwork productivity influence factors

A CLD of an identified research problem is developed by already established ideas and research in addition to the researcher's mental model [20][54]. Focusing on people and teamwork aspects of the agile team, this paper makes use of two team effectiveness models for better analysis of ASD teamwork productivity influence factors. This study considers an adapted form of Salas Big Five teamwork theory [62] and the Dickinson and McIntyre model of team effectiveness [68] for the betterment of agile teamwork (previously discussed in chapter 3). These two models focus on team effectiveness, and mainly on internal aspects of teamwork. The Big Five theory is highly cited, well-grounded in the research literature, and has previously been used in the agile software development context. At the same time, both of the models consider teamwork activity as a learning loop in which teams are identified as self-managed, adaptable and dynamically changing over time [68]. These self-managed agile teams are usually responsible for managing, monitoring and executing their own tasks. It also requires double-loop learning, which is a characteristic of self-managing agile teams to change underlying values and assumptions [62]. The findings from the survey include the productivity influence factors that also comply well with Salas [62] and Dickinson and McIntyre's [68] teamwork components.

Based on these two teamwork models' theory (detailed in the literature review chapter 3, Figure 3.1 and 3.2), this study aims to develop an SD model to analyse ASD teamwork productivity influence factors. This model should allow studying how different factors influence ASD teamwork and hence their productivity.

This section presents the CLD exploring to capture the dynamics of the agile teamwork productivity. To keep the readability of CLD, it has been divided into three sub-models motivation, team effectiveness, and team management.

6.2.4 The influence of motivation on productivity sub-model

Agile development emphasizes the importance of the human aspects of developers [9]. Managing developer motivation has become critical to achieving successful agile projects [26]. Agile teamwork motivation to work on the project and in the company is several dimensional concepts [70]. It includes aspects like working environment, customer satisfaction, and management support, and team effectiveness.

As seen in Figure 6.3, team morale positively influences work quality, as a highly motivated team generates fewer errors and less rework. Expecting higher quality and high team morale, in turn, increase customer satisfaction [100]. The result of increased customer satisfaction is a decrease in external factor's influence on teamwork productivity and thus has an indirect (positive) effect on motivation.

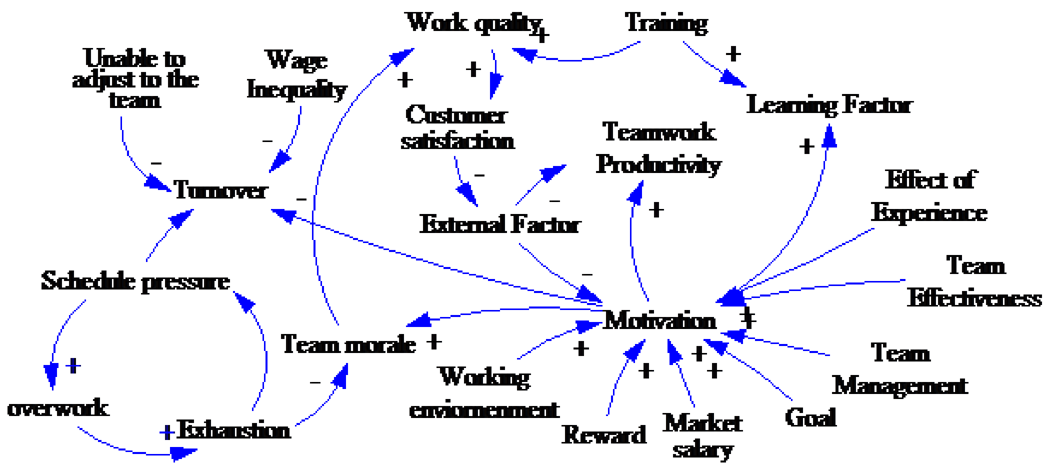


Figure 6.3: Causal Loop Diagram "The Influence of Motivation on Productivity" Sub-Model

Customer satisfaction is one of the indicators of productivity [12] and less external factors along with team morale and motivation positively influence the overall teamwork productivity. Working environment, reward, and salary directly influence motivation. Goals set by team management is a future condition to motivate them to work towards its accomplishment and moral development in the team [70]. A high level of team morale in the project will increase development motivation. The impact factors to the level of motivation include the relationships of the team, team management, individual salary, working environment, reward, etc.

Team motivation is also affected by the behaviour of the team management. According to Melo, agile team management is the most important factor in achieving agile teamwork productivity [10].

Schedule pressure is another influencing factor in this model. Team morale is negatively affected by this factor. According to Abdel-Hamid, schedule pressure can play a significant motivational role in productivity [73].

The influence of team effectiveness on productivity sub-model

Salas Big Five [62], and the Dickinson and McIntyre model of team effectiveness [68] acknowledge that team requires a complex mixture of factors that include organizational support and individual skills, and also teamwork skills. Therefore, these two models have combined the knowledge on teamwork into five components; team leadership, mutual performance monitoring, backup behaviour, adaptability, and team orientation. Their opinion is that each of the models' components is required for good team effectiveness and good teamwork can be achieved if the right sets of components are there. These two models focus on team effectiveness, and mainly on internal components of teamwork. At the same time, both of the models consider teamwork activity as a learning loop in which teams are identified as self-managed, adaptable and dynamically changing over time [68]. These self-managed agile teams are usually responsible for managing, monitoring and executing their own tasks. It requires double-loop learning, which is a characteristic of self-managing agile teams to change underlying values and assumptions [62].

Figure 6.4 compiles the cause-effect relationships connecting team effectiveness, team management, motivation, learning factors with teamwork productivity. Agile software development emphasizes teamwork in self-organizing teams more than traditional development methods do. It is useful to learn how the team works effectively to better understand the influence factors of agile teamwork productivity.

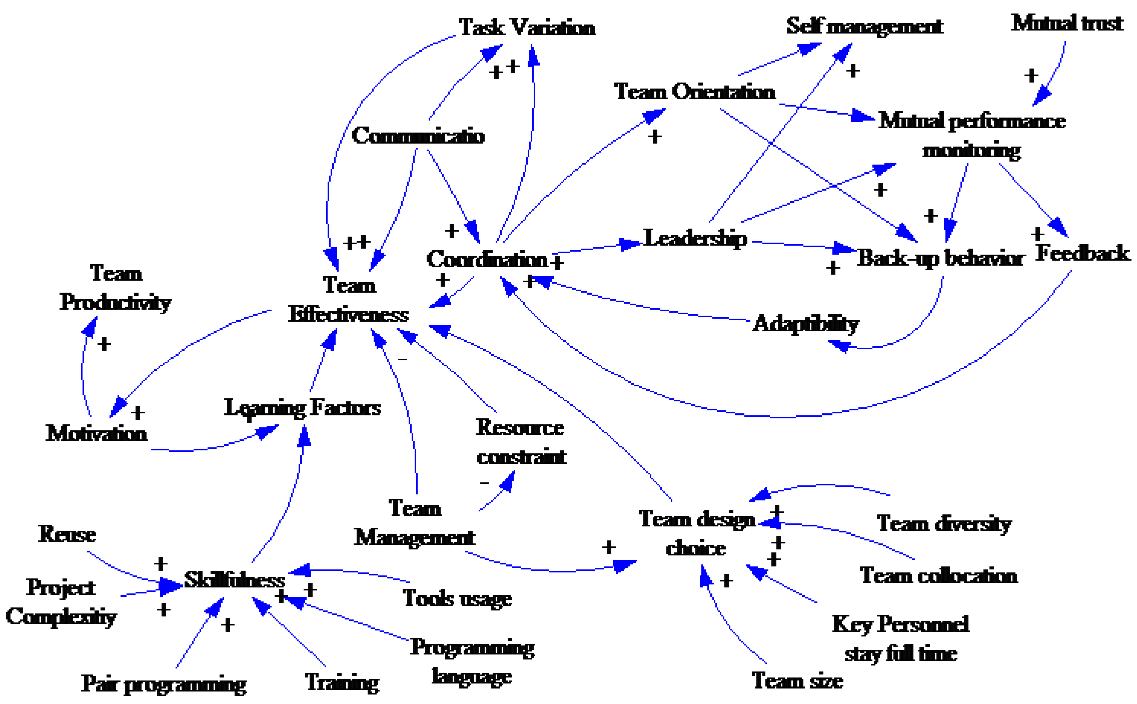


Figure 6.4: Causal Loop Diagram "The Influence of Team Effectiveness and Team Management, on Productivity" Sub-Model

Figure 6.4 shows how the team effectiveness is built within a team and how it affects the engagement of the team in learning-oriented activities (learning factors). The mutual trust concept is based on a shared belief that the team members feel accepted and respected for their feedback. Without sufficient trust, team members will spend time and energy protecting, checking, and inspecting each other as opposed to mutual performance monitoring [31]. It is evident that trust is a prerequisite for shared leadership, feedback, and communication. Team members may not be willing to participate or share information if they fear being perceived for incompetent performance. The degree of mutual trust, adaptability, team orientation, coordination, and communication can be impacted by the experience of working together. More the team members understand each other, the higher the ability of the team to identify a problem in a short time frame and hence increase teams' productivity [10].

The team inspiration to engage with the learning factors is positively related to team effectiveness in regard to team orientation and mutual performance monitoring and feedback present in an agile team. This perception is represented in the CLD (Figure 6.4) by the factor motivation, which offers support to team members to overcome the fear that arises when they face difficult situations. Therefore, the higher the level of motivation, the more secure team members feel, and the more willing they become to involve in learning-oriented activities. As the project proceeds, the team members increasingly engage in learning activities, they interact and coordinate more, hence the team productivity increases.

Dickinson and McIntyre's model suggests that team leadership and team orientation promote team members the capability to monitor their team members' performance [59] [68]. Consequently, it enhances team effectiveness, which leads to improved team productivity.

The team effectiveness, including team management efficiency, are both influenced by skilfulness and might be enhanced by training. Training strengthens the teams' process knowledge, which in turn improves team members' skills and capabilities. Teams' expertise is further influenced by individual learning, which is characterized by individual work experiences [101]. Individual learning positively influences organizational learning, which can be further created through shared experiences [101].

The influence of team management on productivity sub-model

According to Melo, agile team management is the most important factor in achieving agile team productivity [102]. Supportive team management tends to provide constructive feedback and encourage the team to involve task variation. Teamwork productivity depends highly on the design of the working environment and the social system the team is part of [11] [12]. Team design, resource, task variation, technology, leadership, training, cooperation and motivation impact a team's productivity to complete the job, assuming the team has favourable working conditions and supportive team management. Team design refers to the formation (diversity, skills, backgrounds, and experiences) and structure (relative size, the formal role of each member, clear goals, and group norms) of the team. These factors of team design have been found to be relevant to a team's ability to work together and to involve in learning behaviours [11] [102].

Within the factors in Figure 6.4, that influences team motivation represented through a direct link from the factor team management. A clear management perception enables team workers to better understand the project goals or project/scope changes, which in turn has a positive effect on team morale and commitment [70]. Change in scope occurs due to changes in business such as a change in technology or in market conditions. Customers often request requirement changes to be incorporated into the project. Software companies are preferring the agile method due to its capability to incorporate requirements change easily [26]. If a software company has to deliver the product to its full effectiveness then it is mandatory for them to effectively manage change during the changes in scope. Hammer and Champy observed that management support also positively influences team-workers' change understanding and consequently positively influences team morale [103]. Training represents another important factor influencing team morale, especially when developers gain new skills for the changed processes and tasks. [100]. Furthermore, successful developers increase skilfulness and capabilities by training organized by management. Skilled team worker is capable to increase teamwork productivity and transform the software companies' vision.

The skilfulness has a positive influence on the work quality of the products. The results of higher quality are satisfied customers i.e. the improvement of the quality also positively influences team motivation, in the model represented by an indirect link from customer satisfaction through external factors. Increased customer satisfaction eventually reduce external factors' influence (negative influence) and hence increase in team motivation and productivity (positive influence).

The factor resource constraint presents one of the important influence factors that impact ASD teamwork productivity (chapter 5 Table 5.1.). Along with resources and training, the team's experience positively influences the work quality and has an indirect positive effect on customer satisfaction and team motivation. In turn, positive team motivation causes productivity measurement is performed more efficiently.

However, team management, which follows a social hierarchy, promotes a top-down approach to decision making, as opposed to a participatory approach, significantly influenced the way team members to communicate with each other. Under this kind of team management, agile team members will avoid any unwanted situation where they can face a problem and restrain from learning-oriented activities. As a result, team productivity decrease, the team management efficiency decrease, indicating a perceived need for team/technical training, as represented in Figure 6.4.

Another factor that influences skilfulness is pair programming, which is one of the key practices influencing team productivity [5] [8]. However, this factor is not encouraged in Bangladeshi software companies. Management does not want to engage two resources for single work due to an increase in expenses. It is mostly practiced by the developers when they need assistance to complete a difficult work.

Getting the right person with suitable skills and knowledge for an agile team is a difficult job for the software companies in Bangladesh. Staffing (right person selected) happened to be one of the most important factors impacting teamwork productivity, as Table 5.1 and Figure 5.11 show. Consequently, team design choice becomes a significant influencing factor for agile teamwork productivity (Figure 6.2). It affects the amount of knowledge that team members must apply to improve the team task (Figure 6.4).

Causal loop diagrams are a compelling method to summarize and communicate the structure of the model. These conceptual models are the basis for a quantitative specification of a system. The next step of the SD methodology involves the mapping of the causal diagram into a dynamic simulation model for a more detailed analysis of the studied system behaviour.

6.3 Quantitative Modelling of Agile Software Development Teamwork Productivity

The purpose of this section is to construct System Dynamics (SD) models to simulate the affects of ASD teamwork productivity influence factors, using the insights gained from the previous chapters. The proposed SD model is based on the conceptual model; i.e. on the CLD, developed in the previous section 6.2 (step 3 of SD modelling approach) and data obtained from chapter 5 (step 2 of SD modelling approach; literature review, interview, and survey). The SD simulation is carried out with the Vensim software, which is capable of modelling SD simulation [99]. The factors from the conceptual/influence diagrams are added as auxiliaries and constants to control the flows. All of the factors in the influence diagrams are not included in the simulation model to avoid getting a too complex model.

The proposed SD model is created by adapting and extending the pioneering model of Abdel-Hamid and Madnick, in Software Project Dynamics, which includes a validated Software Project Dynamics [73]. Their model provides the core systems and behaviours of software project management as a conceptual reference baseline. The model of Abdel-Hamid and Madnick is based on the classic Waterfall process, i.e. sequential software development model [73]. Therefore, the model has been adjusted for this research purposes to follow the agile software development process.

6.3.1 Dynamic Model Formulation

This section presents a step-by-step discussion about the dynamic representation of the ASD teamwork productivity model. The model captures the dynamics of teamwork productivity influence factors and its affect on teamwork productivity.

Based on the CLD represented in the previous sections 6.2, the model basically implements the stock and flow diagram. Most of the factors present in the CLD (Figure 6.2) are also represented in the dynamic model. The factors from the conceptual/influence diagrams are added as auxiliaries and constants variable to control the flows. To avoid getting a too complex model, all of the factors in the influence diagrams are not included in the simulation model. Some factors and loops are disaggregated in order to better capture the dynamics of the system. Some factors are aggregated because they present relatively similar dynamics and/or collapse into a group of factors (variable) to avoid getting a too complex model.

To keep the readability of the overall SD stock and flow diagram, it has been divided into six interrelated sectors before integrating them into a complete model, namely project workflow (software production), teamwork productivity, team effectiveness, performance, team management and organizational context, and motivation. These interrelated sectors are explained briefly below.

Project Workflow (software production)

Agile development processes usually divide a project into multiple parts of similar iteration lengths to be executed sequentially [6]. When all parts are developed, the complete project is also finished. Figure 6.5 shows the stocks and flows of the agile subsystem. This model has been evolved from the model proposed by Abdel-Hamid [73] and represented by [82]. Their modified model [82], however, has been further developed to simulate the ASD processes.

The following stocks of work exist during a common agile software development project [82]:

- **Project Backlog**: A project backlog is a set of requirements that needs to be done within the project.
- **Project iteration**: The list of tasks/work that the development team identified during the upcoming iteration/sprint. It basically consists of a list of tasks required to complete a feature.
- **Project completed**: A project begins with an initial amount of work, initial load as the input for the stock Project Backlog. A subset of those tasks is moved from Project Backlog to the stock Project iteration. This subset of tasks from Project backlog is selected depending upon the velocity of the team. The requirements from Project iteration are developed depending upon the productivity of the available workforce and the accomplished work is accumulated into the stock Project completed.
- **Rework**: Some fraction of the work is either incorrect or incomplete and requires rework, and represented by the variable rework.

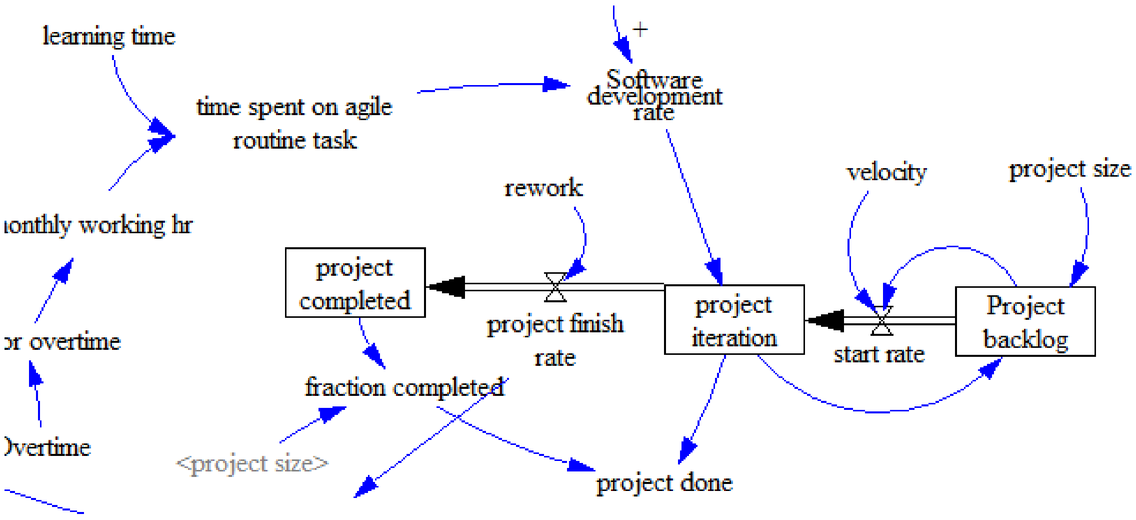


Figure 6.5: ASD project workflow sector

Under the ideal situation, the project starts with an initial scope of work, Project backlog. The hiring/forming of the project team takes place according to the variation in team members (workforce). All along with the project, rework occurs due to requirements (scope) change, quality standards, and errors. Customer satisfaction increases when the project is completed on time, on budget, and with the expected quality. Change in scope occurs due to changes in business such as a change in technology or in market competition. The agile method accepts requirements change throughout a project at regular intervals [6] [100]. These change requirements add more work to the existing rework and project backlog. This rework will have a causal affect on the completion of the project, but most of the time the customer wants on-time completion of the project. All these aspects have been considered in the development of the stock and flow diagram of the project workflow section.

Performance

Organizations aim to increase the developer's performance by influencing team behaviour with different soft factors [86]. Such as motivation, team effectiveness, team management, team leader efficiency, working environment, and training, have an effect on software development teams' performance. Team performance influences the team to perform in a way that contributes to organizational goals [62]. Performances are measured in terms of time, quality and cost [32]. However, it is hard to measure team performance (quantifying the efficiency and effectiveness of developing tasks) instantly [101]. Taking into account that, it depends not only on the team output but also on the aforementioned internal soft factors. That is why this model shows performance as perceived and not actual performance and can be defined as a function of achieving performance rate and work completion rate.

Performance= work rate/pt

The parameter performance target (pt) is a constant that the organization wants to achieve and does not change during the project execution. Performance positively influences the quality of products and services and has an indirect positive effect on customer satisfaction and team motivation. Due to increased customer satisfaction, there is a decreased external factors' effect, which in turn positively influence team motivation.

Figure 6.10 shows that an increase in the performance may increase the skilfulness and reduce the perceived need to engage in learning activities. The difference between team performance and the performance target (pt) generates a gap that allows the team to assess its skilfulness in order to reach its target. Therefore, as team performance increases, the performance gap likely to decline, leading to a decrease in the engagement of the team in learning activities. On the other hand, if the team performance decreases, the performance gap will increase, making the system work in the opposite direction in an attempt to improve team performance.

However, the effective team tends to maintain the performance at a higher level because such team are more likely to engage in learning activities, such as pair programming and reusing of software, that aid to detect errors for reworking and take the necessary actions to complete team tasks, completing team performance rate. Conversely, ineffective teams' performance is likely to decrease over time due to the infrequent involvement of team members in learning activities. Individual perceptions that motivate self-centered and unwilling to take the interpersonal risk, compromise team effectiveness and goal achievement. The more the team members understand each other, the higher the ability of the team to detect an error in a short time frame. For example, the performance of a pair who have long term experience of working together is greater than the performance of a new pair [10].

Team management and organizational context

Team management (TM) is not modelled as a stock. It has been assumed that work conditions do not accumulate but change instantly by managerial intervention and provide adequate training.

Organizational context (OC) refers to factors such as organizational culture, rewards, information systems resources, adequate tools, equipment, material, and supplies. It represents a supportive environment for the team to complete the work with the right person selected and team design choice that mixes skills and experiences, and right-sized teams with a proper resource provided that is important to the completion of the tasks.

This study includes all these different aspects of the Organizational Context (OC) and is considered as external variables. Because groups generally have no direct control over such factors.

Motivation

In this SD model, Motivation M is a stock representing the accumulation over time of the concept that the team is confident for interpersonal risk-taking. Motivation is crucial for an ASD team since they are self-managed. As team members become more confident and comfortable to work together, they are more motivated with the project they are working on [10]. Then the team is more willing to begin improvement activities that increase motivation and contribute to learning factors. Learning factors depends on team member's enthusiasm and effort to achieve new knowledge and skills. For this reason, a link between motivation and learning factor initiate.

The level of a stock is changed by the rates. The rate of change in the level of motivation is defined by the influence of team effectiveness, OG context, external factors, and the affect of team management. Motivation value varies from zero to one.

Team Effectiveness

As agile team began to work together and make progress, team members become more engaged to do the work and in turn learned more about their tasks, roles, constraints, objectives, and need in continuing participation willingly in all relevant aspect of the project and its implementation (through teams' feedback, behaviour, monitor, trust and leadership). As teams' knowledge of their own work increased by feedback and backup-behaviour, the error rate decreased. As their knowledge of the other's work increased by mutual performance monitoring and team orientation, the productivity rate increased. By working together (team orientation)—i.e., collaboratively—they also learned about the other's role in the project and increase mutual trust. As mutual trust increased, each member shares more information, thus mutual performance monitoring increases for each other and so allowing the other team members to learn more about its objectives and constraints, and thereby providing more feedback and backup.

When team members do not perceive progress from their work together, they become less engaged and do not learn more about their own work and others'; mutual trust does not increase, and individuals do not provide feedback, and backup behaviour about one another's work slows, further slowing the project work and hence team productivity. Therefore, team effectiveness is a crucial influence factor that contributes to motivating the team to engage in leaning oriented activities.

Team Effectiveness, in Figure 6.6, is not a single variable in the model; rather, it represents at an abstract level the effects of team effectiveness in the model. All the influence factors affecting the team effectively are collapsed into the variable team effectiveness. The team process reflects the level of team effectiveness required of a team to work effectively.

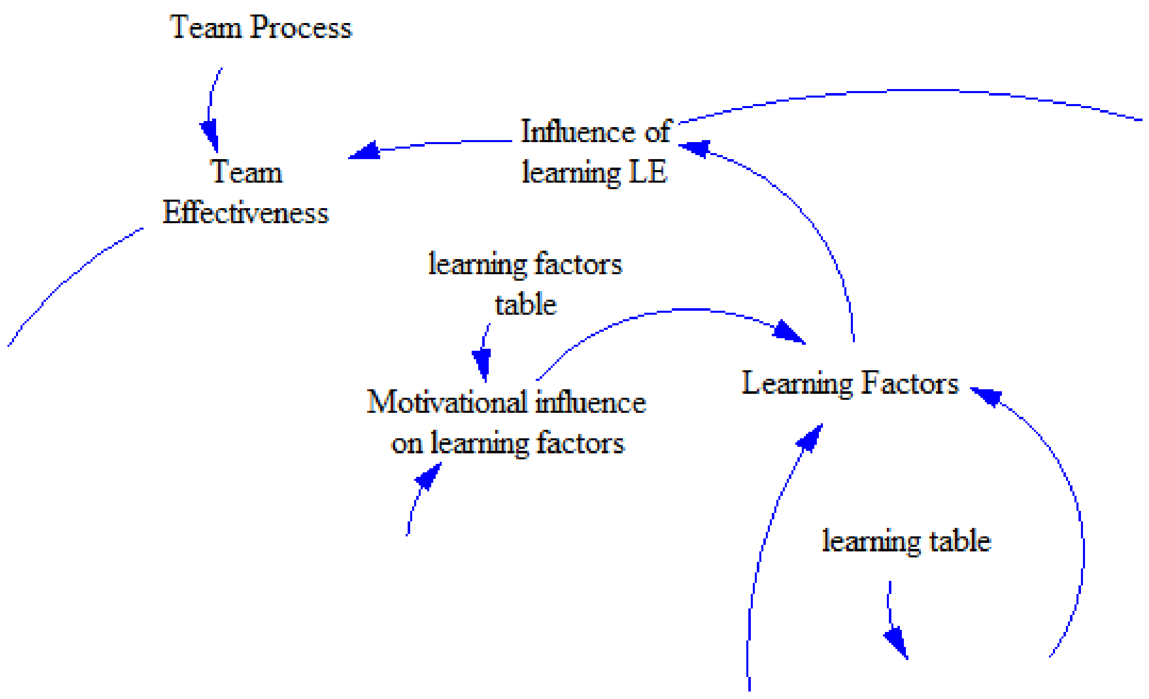


Figure 6.6: Team Effectiveness Subsector of ASD Teamwork Productivity Model

Teamwork Productivity

During the software development phase, it is important for management to understand and anticipate the software development rate (productivity). The development rate affected by different factors such as motivation, training, reward, and technology. Therefore, software development productivity is an outcome of a complex set of different factors than just being a function of how much workforce is allocated to the project [73]. The teamwork productivity sector is based on the following state from Abdel-Hamid [73]:

Actual productivity = potential productivity − losses due to faulty processes

Actual productivity is indicated as the potential productivity subtracted losses due to the faulty process [73]. The Actual productivity is influenced by the potential productivity and a number of complex factors as identified in this study. The losses (losses due to faulty process) refer to mainly communication overhead and motivational losses. Potential productivity represents the best possible use of resources and the maximum level of productivity the team can produce. It is the level of productivity that can only be achieved when teamwork at maximum efficiency in an organization for a particular project. When there is no loss due to faulty processes, team productivity is maximum.

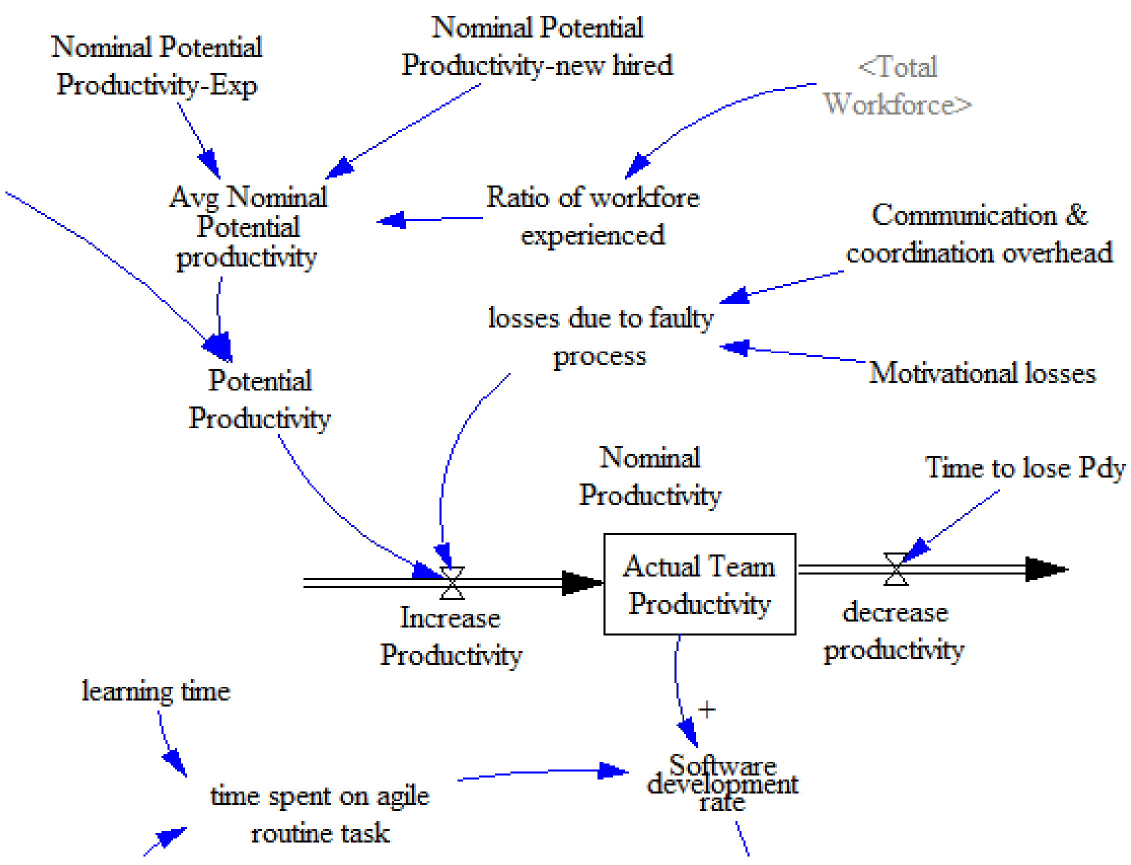

Nominal Potential Productivity-Exp
Nominal Potential Productivity-new hired
<Total Workforce>
Avg Nominal Potential productivity
Ratio of workforce experienced
Communication & coordination overhead
losses due to faulty process
Motivational losses
Potential Productivity
Nominal Productivity
Time to lose Pdy
Actual Team Productivity
Increase Productivity
decrease productivity
learning time
+
time spent on agile routine task
Software development rate

Figure 6.7: ASD Teamwork Productivity Subsector

Experience and learning factors have an impact on potential productivity. In any software development project, there are mainly two different kinds of the workforce, newly hired and experienced workers. Experienced workers are usually more productive than newly hired workers [73]. Therefore, the overall potential productivity changes with the mixing of the experienced and newly hired workforce. Hence, the ratio between new workers and experienced workers (in the model ratio of workforce experienced) in the assigned workforce to software production will influence the potential productivity rate [73] [74].

The nominal potential productivity is influenced by the experience level of the workforce. Here an experienced person is set to 1 task/man-day productivity and new people to 0.5 task/man-day. The nominal productivity rate is then the maximum level of productivity measured in the number of tasks performed per day [73].

There is another factor affecting potential productivity is a learning factor effect that occurs throughout the project [74]. As the project proceeds, the team knowledge of how to do the project increases, hence the potential productivity increases. It is an S-shaped multiplier curve set to 1 at the beginning and ends up at 1.25 at the end of the project (Figure 6.8).

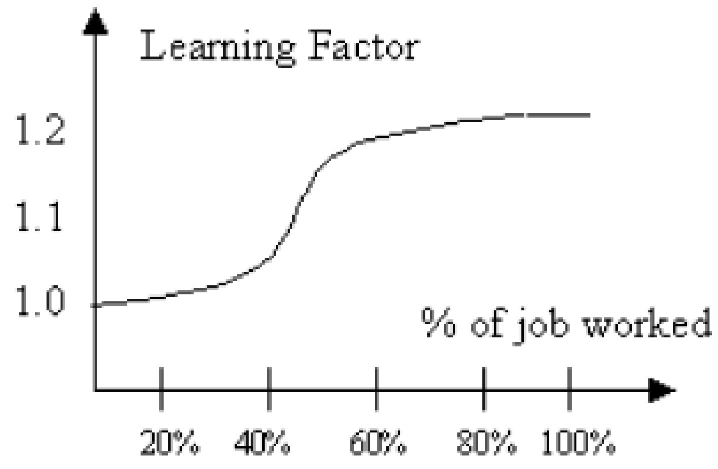


Figure 6.8: Learning Factor [73]

During the software development process within a single project, some motivational factors (views and goals) remain constant, while others change during production [73]. Motivational factors such as promotion, team responsibility, salary, etc. are all constant factors that influence the overall condition of the organization.

Slack time is a motivational influence factor that changes over time due to schedule pressures. It is defined as the time lost to non-project activities such as coffee-breaks, email reading, and personal activities. During the software development process, some hours will be lost to these activities and will influence actual productivity. These are calculated as motivational losses.

The second factor of loss to actual productivity derives from communication overhead. The agile team requires more frequent communication and meeting, such as daily stand-ups. As team size increases, the required number of communication links between team members also increase. Moreover communication overhead also increases if the team members involved have a diverse interest. On the other hand, Team coordination reduces communication overhead.

The communication overhead (a function of the square of the number of people) is expressed as a nonlinear function of the total number of people that need to communicate (Figure 6.9). Therefore, it becomes more problematic to communicate effectively with a larger team. The time spent communicating between team members such as meeting, documentation and any routine work, decreases the productivity rate. This is defined as the average drop in a team member's average nominal productivity as a consequence of team communication [74].

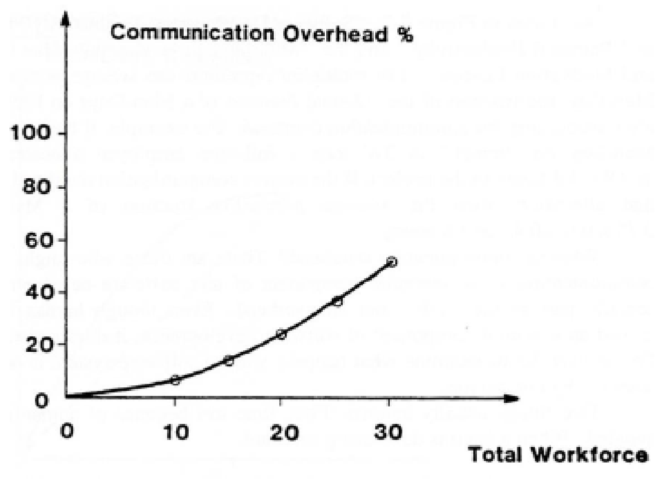


Figure 6.9: Communication Overhead [73]

This dynamic is defined by the variable software development rate:

*Software development rate = Actual team productivity * (1- time spent on agile routine task)*

Unlike Abdel Hamid [73], in this SD model Team Productivity is a stock defining the productivity level of a team. Similar to the other stocks, team productivity stock is the total of all previous changes in it and its initial value.

The major activities of a software development project consist of four sectors: manpower allocation, software development (design and coding), quality assurance and rework, and system testing [73]. Each sector is alone too complex to explain as one piece. Therefore, only the software development sector is covered in the SD model.

6.3.2 Simulation analysis - Model Validity and Scenario Planning

Using the proposed SD model, the ASD teamwork productivity can now be simulated considering the affects of the influencing factors. Model experiments help to identify influences in several different behaviour modes of motivation, team effectiveness, team management, performance, workflow, and teamwork productivity. Therefore, this study designs the model to simulate different agile project management practices for analysing the simulation results with respect to ASD teamwork productivity. Moreover, the proposed model has the capability to predict the value of ASD teamwork productivity considering the influencing factors such as motivation, team effectiveness, and team management efficiency. Although, the purpose of this modelling is not to quantify or to predict ASD productivity. The productivity prediction is only a tool here to explore and analyse the influence of productivity factors. It also focuses on how well the simulations match the predictions from the theory and survey results (chapter 5).

Model verification and validation process have been performed for the developed model to build confidence in the simulated results. To validate the proposed model and resulting consequences, this research has conducted a variety of standard verification and validation tests that have been proposed in the literature [20] such as boundary adequacy, structural assessment, dimensional consistency, parameter assessment, extreme conditions, and integration error.

After the initial estimation, the model is validated using the tests described in the next section. Following that, the model is simulated and the results obtained are analysed.

Model Verification and Validation

It is an important aspect to establish confidence in the usefulness of any model-based methodology with respect to its purpose [19]. The validity of the results of a specific study is dependent on the validity of the model. Therefore, validation of the model structure and behaviour are important parts of the simulation validation, and especially in the System Dynamics methodology [19] [20]. All models are represented as a simplified version of the real system and therefore, no model is fully correct [20]. Hence, the verification and validation of this study are focused on building confidence in the model as an acceptable representation of the system.

Model verification is concerned with building the model right, i.e. confirming that the right parameters have been used. For this purpose, the model components are reviewed and analysed during its incremental development stage. The model validation is concerned with building the right model, i.e. determining that the model is a correct representation of the real system. Validation involves checking that the model meets the expectations of the modeller. The proposed SD model is created by adapting and extending the model of Abdel-Hamid, which is an extensively validated and widely accepted representation of Software Project Dynamics [73]. The model has also been used as a research tool in several studies [79] [81]] [90], as well as in actual project settings in organizations [82].

There are several techniques and tests (such as structural and behavioural) that enable us to build confidence in the proposed model in terms of both logic and numerology [19] [20]. This section will now proceed with the verification and validation effort for the proposed Agile Software Development (ASD) teamwork productivity model. The rest of this section will discuss how sensitive the SD model is to changes in parameter values (productivity influence factors). Additionally, it is important to know what factors have the greatest influence on the model and to validate these findings by using survey results. The techniques applied here follow the recommended tests for both the structural aspects and for the internal dynamics of the developed model. The tests are provided by Sterman, Coyle, and Forrester [19] [20] [104].

Model Verification

The verification of a model consists of activities that focus on its internal workings. The proposed productivity model verification activities fall into the following categories [19] [20]:

1. Verifying the structure of the model. The test focuses on exploring the dimensional consistency of equations in the model. This test shows a typographical error, an inverted ratio, or a missing time constant [19]. It is important to specify units of measure for each variable while building the model. Because units errors discover important limitations in understanding of the structure or decision process that is trying to model [19]. The dimensional consistency check is used to verify the consistency of all related variable units. Moreover, this test is automatically carried out by the Vensim software [99]. If there is any dimensional inconsistency the model will not run in the Vensim environment. The only errors that remain are caused by using dimensioned variables as input for non-linear functions. Nevertheless, the rest of the auxiliaries, levels, rates, and constants show dimensional consistency.

2. Verifying the behaviour of the model. This test focuses on numerical integration by using a finite time step and average the resulting approximation in the interval.

3. Verifying the behaviour anomaly of the model. This test verifies and establishes the significance of important relationships by determining whether

anomalous behaviour arises when the relationship is deleted or modified [20]. A common method for performing the test is by deleting loops. It is also helpful for determining the importance of a loop [20]. Only significant loops were tested which have an influence on other sectors of the model.

Model Validation

Validation is used to refer to the model usability. To increase confidence in SD models several tests can be done for model validation. According to Forrester and Senge [104], there is no single test that can be used to validate an SD model. The model's parameter estimation is needed to test the inside behaviour of the model and to find the similarity between observed and simulated structure. Parameter's estimation is easy when there is access to data-sets from real projects. At the same time, the data needed to build and test the model are hardly available without significant cost and effort. It is important to constantly make judgments about whether the time and cost of additional data gathering are justified. Soft variables such as motivation, perception, productivity, and performance cannot be measured directly and objectively. However, when no such data sets are available then the problem can be resolved through judgmental estimation and using data-sets that have been previously made available online. In this study, some of the model elements are estimated based on data collected from a survey (Chapter 5), some are from available developed SD model [2] [95] [96] and the remaining elements are estimated using judgmental estimation.

There are two types of validation tests; direct structure test and structure-oriented behaviour test [20]. Both tests have been applied in this model:

- The first step, direct structural test, involves a structural assessment of close by elements, model subsystems, and overall model structure.

- Once enough confidence in the structural validity of the model has been established, in the second step, the structural behavioural validity of the system is tested. The focus of this step is to establish and relate the model's inside behaviour with the information gathered from the survey in Chapter 5.

Direct Structural test In Direct Structural test, the model is tested without running any simulation rather each relation in the model is studied. This study has evaluated three direct structural tests; theoretical structure and parameter confirmation, extreme condition and boundary adequacy of structure. The tests are discussed below.

Theoretical structure and parameter confirmation

The structure assessment test is conducted to ensure that the system structure is consistent with the knowledge of the system (both physically and mentally). Both of the model structure and the parameters must correspond to the relevant understanding of the system. The parameters must also be checked on the basis of any numerical knowledge [19]. There is no quantitative literature available with the focus of this thesis other than what is used to build the model. There are no studies to compare with the simulation outputs of the whole system model. However, a comparison is possible for parts of the model. Part of the model (workflow and productivity) is modified from the model developed by Abdel-Hamid [73]. This validates the relevant structural descriptions used in the model with reference to the previous model [73] [74].

Direct extreme condition

In this test, the model equation is tested individually under extreme conditions. Every equation can be checked this way by entering extreme values for the input variables and comparing the output variable to the expectations in real situations.

Boundary adequacy of the structure

The boundary adequacy test involves drawing the model boundary. That is, deciding which variables will be included in the model. This test is used to check whether the model has contained the most important concepts required to analyse the problem [19].

Structural behavioral oriented test

In this test, the model is evaluated by simulation running. This study has evaluated two structure-oriented tests; extreme condition test and sensitivity analysis.

Extreme condition test

The extreme condition test involves assigning extreme values to selected parameters and comparing the model-generated behaviour with the expected or observed behaviour of the real system under the same extreme conditions [20]. The purposes of this test are to make sure that the model's behaviour in extreme conditions still makes sense and can provide information on any possible structural defects. This test is much more suitable than a direct structure test for formalizing and quantifying [20]. The extreme condition test for this study is conducted by considering four extreme conditions; no team effectiveness and very high team effectiveness. No project management efficiency and high PM efficiency. The model simulation results under these conditions were compared with the expected behaviour of some of the variables like motivation, learning factors, and teamwork productivity. Table 6.1 summarizes the expected and actual results of this test.

Sensitivity analysis

To help the research to assess the model behaviour, the sensitivity analysis is performed on the influencing factors of the model. Among the factors identified in chapter 5 influencing the ASD teamwork productivity, the model presented in chapter 6 indicated that the following factors (as they are the only constant factors in the model) are directly affecting the ASD teamwork productivity:

1. Team process TP
2. Team Management efficiency TME
3. OG context
4. Motivation M

5. Performance Target PT

The sensitivity analysis available within the Vensim [99] modelling software(s) can automatically change variable values and evaluate new results. This process analyses how sensitive the results are to changes in these uncertainties, and thus discover how sensitive the model is. The validation process using the sensitivity analysis is performed by selecting first each variable influencing the ASD teamwork productivity model and the selected affecting variables. It also produces distributions of values for the affected variables. The changes of results in different ranges can be studied, and below or above certain percentiles [20]. In this way, the chances of unwanted results can be seen, besides favourable results [20].

Table 6.2 shows the degree of the variation in the output of the model (i.e. productivity changes) when the value of the variables is changed within the specified range. The influence of the variation of the most influencing factors is studied within its range on the model and the results are displayed with the graphs showing the variation in section 6.3.2.2.2 Scenario Planning. Table 6.1 summarizes the verification and validation tests used in this study.

Table 6.1: Summary of Verification and Validation Tests

Name of the test	**Purpose of the test**	**Procedure conducted in this research**	**Test results**
1. Dimensional Consistency	Determines whether each equation is dimensionally consistent	· The test was performed during the model creation and after the finalization of it using Vensim's inbuilt tool · Inspected model equations for suspect parameters, meaningless names, and strange combinations of units.	Passed
2. Integration Error	Determine whether the results are sensitive to the time step	· Used different time steps to ensure proper results	Passed
3. Behavior Anomaly	Establishes the significance of important relationships by determining whether anomalous behavior arises when the relationship is deleted or modified	If Effect of learning activities or productivity rate are removed, the model exhibits anomalous behavior	Performed well

Continued on next page

Table 6.1 – continued from previous page

Name of the test	**Purpose of the test**	**Procedure conducted in this research**	**Test results**
4. Boundary Adequacy of structure	Determines whether the model has contained the most important concepts required to analyze the problem. The major concepts identified in the theoretical build-ups like team effectiveness, project management and efficiency, learning factors, and cultural impacts are sufficiently captured by the model.	causal diagrams and stock-flow diagrams were reviewed using survey results	Model was improved
5. Structure and parameter confirmation	Both of the model structure and the parameters must correspond to the relevant understanding of the system The parameters must also be checked on the basis of any numerical knowledge.	· The major relationships, input variables, and output variables are reviewed using available similar kind of models · Other parameter values are set based on data collected from the survey, some standard laws and judgmental estimation	Passed

Continued on next page

Table 6.1 – continued from previous page

Name of the test	**Purpose of the test**	**Procedure conducted in this research**	**Test results**
6. Extreme Conditions (Direct structure and simulation)	Each equation makes sense on extreme input values	· Inspecting each equation · Testing response to extreme values of each input. The extreme condition test is conducted by considering 4 extreme conditions; no team effectiveness and very high team effectiveness. No project management efficiency and high PM efficiency). The model simulation results under these conditions are compared with the expected behavior of some of the variables like motivation level, expected work completion time and productivity.	The simulated behavior coincides with what is expected
Continued on next page			

Table 6.1 – continued from previous page

Name of the test	**Purpose of the test**	**Procedure conducted in this research**	**Test results**
7. Sensitivity Analysis	The impact of changing assumptions	The analysis will be carried out by varying values of: · Team process · Team Management efficiency · Organizational context · Performance rate	Performed well
Soft factors (non-technical)	test		
Technical factors	Hardware, software tools		
Organizational factors	Organization structure, organizational culture		
Environmental factors	Socio, political, economic, legal		

Scenario Planning for Model Simulation

A systematic approach has been used to explore the simulation model. Firstly, a base case is introduced and analysed. Secondly, different sets of scenarios are conducted to explore the dynamics of the system. Finally, the applicability and performance of the developed SD model are evaluated by its implementation in a real software project.

The initial parameter values for simulation are mostly based on the qualitative data available from chapter 5 (literature review, interview, and survey). However, the project workflow sector (section 6.3.1), that is built on several validated models (such as Abdel Hamid [73]) and contains elements that are comparatively easier to quantify. On the other hand, it is not practical to validate parameters used for modelling soft factors such as teams' motivation, effectiveness, and team management. Since soft factors deal with mental beliefs that are difficult to measure [20]. To keep the value within limits and to represent the real-world system, these soft factors are simulated with values between zero (no influence) and one (high influence) [20]. Few assumptions:

- The model does not make changes in team members.
- It is assumed that the simulations run with the same team members.
- Even if there is any turnover, the team are replaced by others with the same level of experience.

Base Case

The first set of simulation runs is a base case. The base case simulates a scenario in which the software organization provides the required resources, information and supports for the ASD team to complete their tasks. The time horizon of simulation is one year (12 months) which is a reasonable time frame for a medium to small size agile project activities.

The base model data are based on several assumptions:

1. the newly hired employees are only half (.5) as productive as experienced employees (1)
2. The ratio of workforce experience 1.5
3. Losses due to faulty process .5
4. The team and tasks are assigned to support Team effectiveness. The team is designed to effectively work as an agile team and has previous experience to work as a team (represented by team process = 0.7)

5. Organizational (OG) context provides supportive job design and work environment at the minimum conditions for learning activities to take place (OG context = 0.7)

6. Management is fairly supportive (capture by Team management efficiency TME= 0.7)

7. Since management is fairly supportive (Motivation M =.5), indicating that the team starts out working together with some level of motivation.

8. Since management is fairly supportive, it sets the performance target at pt = 1 task/month, as it is minimum productivity.

9. The fraction of rework is expected to be 10% of the project completed.

The results of the base run are shown in Figure 6.10

The results of the base case run show that team members involve learning factors once their motivation and productivity increase. Initially, motivation, productivity, and performance increase quickly. As the team perceives their performance rate has increased, team members put less effort into learning activities, resulting in a decrease in motivation and productivity (shown after month 3 in Figure 6.11), and a slight decrease in performance (shown after month 5, in Figure 6.11). In this base case run, this team has favourable team management and OG context. Therefore, management support influences the agile team to increase their engagement in learning activities (training and skilfulness) when team productivity, motivation, and performance decreases.

The base case run shows a possible functional condition generated by the team management's target for the ASD team productivity, which may function as a motivation for team members to engage in team process improvement by learning factors. Effect of team learning activities LE will help the team to increase its productivity and reach the target performance. This study has performed simu-

Table 6.2: Simulated values of ASD teamwork productivity for different values of influencing factors

Case No.	Team Process	OG Context	Team Management Efficiency TME	Motivation M	Performance Target PT	Productivity
Base Case run	.7	.7	.7	.5	1	1.20652
Base case run 2	.3	.3	.3	.4	1	1.04422
Base case run 3	1	1	1	.5	1	1.22165
Moderate Management & low Team Process	.1	.8	.8	.5	1.5	1.21344
Poor Management Support & High Performance target	.4	.1	.1	.4	1.5	1.01196
Test Scenario 1	.9	.9	.9	.6	1	1.22034
Test Scenario 2	.5	.5	.5	.5	.8	1.02928

lation on the other possible behaviour modes considering the base case run as a standard. In Table 6.2, the simulated values of the ASD teamwork productivity are presented for different values of influencing factors in different cases.

Figures 12 - 16 show the results of the different runs for motivation, team process, productivity, performance, team management support, work quality, OG context, and learning factors. Both runs (Base case run 2 and Base case run 3) start with some level of motivation, organizational and management support. But the agile team who has higher management support and team effectiveness builds up its motivation faster than another team (compare with other runs).

A higher motivated team is more comfortable than another team to engage in learning activities. This team is more likely to be comfortable with work environment and culture, and can respond to changes and correct their course of action more promptly through team effectiveness, and build an adequate level of team productivity. Table 6.2 shows the variation in team productivity.

Scenario Planning

Moderate Management Low Team Effectiveness and Poor Management Support High-Performance target Two different scenarios have been planned for this simulation purpose:

1. The first scenario, Poor Management Support High-Performance target,

the organization ignore enhancing organizational context and management support (both are very low and set as 0.1) and set up higher performance rate (set as 1.5 task/month)

2. In the second scenario, Moderate Management low Team Effectiveness, for the same performance rate, the organization pays attention to enhance organizational context and management support (both are very low and set as 0.8. On the other hand, overlooked the team process improvement (set as 0.1 task/month).

Team management support has been recognized to be of high importance for teams (chapter 5). The team management can influence team effectiveness by helping team members learn to work interdependently (self-managing), building responsibility to the team (team orientation) and its task, and coordinating the implementation of the work plan [62]. This kind of working environment of a team helps determine the team's effectiveness and influence the team works to get through any difficult and unwanted situation [31] [59]. In the same way that a supportive management and OG context can motivate a team, a conservative working environment can slow down team productivity. In addition, supportive team leaders do communicate horizontally in conversation, not through top-down commands.

The effectiveness of an agile team is influenced by the shared perceptions of its members and by the team dynamics and functioning. Agile development teams have confidence in their own skills, build trust, self-managing, and maintain a quality of working environment [26] [29]. Giving agile team members the chance to act on their own might result in several positive implications in the process outcome, such as increased productivity in the team because of increased ownership in their work, increased performance, reduced costs, job satisfaction and increased team motivation [12] [29] [38]. Thus, the goal is to provide a positive working environment where team members easily establish motivation and agile practices. Without doing so, the higher performance target, in turn, harms team learning, team productivity, and time effectiveness.

The aforementioned scenario is simulated for the constant team process, management efficiency OG context and performance target. The simulation results support the survey findings that team management, motivation, and team effectiveness play a significant role in ASD teamwork productivity.

The survey result from chapter 5 shows that two of the main reasons for failure in agile projects are organizational culture and lack of management support. In addition, this study finds that due to social hierarchy culture influences, the self-managed agile team may not fully fit in their organization. The horizontal structure of agile involves self-organizing teams that work in an iterative fashion and deliver continuous additional value directly to customers [97]. Although the practice of self-organized teams conflicted with the cultural responses of social hierarchy.

Team members look for clear instructions and accept their supremacy, also their own dependency on the superiors. Based on results found in the interview and survey of this study, it is perceived that social hierarchy is embedded in Bangladeshi software organizations and affects the implementation of the agile principle. In agile development, communication links together all other teamwork processes. The survey result shows that ASD team management has more influence on productivity than self-management.

From this scenario, it is evident that, even though the most followed organizational structure is horizontal, the social hierarchy culture significantly influences agile teamwork productivity. It is because the way team members communicate with team and customers, and more often to respect official hierarchy/top management (the cultural norm in Bangladesh), communication occurred between members at the same levels of the organizational hierarchy [97].

Figures 17 - 21 show that the agile team management cannot just set up a higher performance rate, and achieve operational success without enhancing management support and team process. ASD teams present poorer team learning and motivation, lower work quality and productivity when they merely set up higher performance goals without considering the management support, organizational context, and team process.

Test Scenario

The project used for the test scenario experiments is a simulation of a real-life ASD project, which is a running large-scale "Insurtech" project. However, comparison with project data is problematic because no reported data sets contain all the parameters incorporated in the system dynamics model. No project data was found to be complete enough for a comparison of total affect. For instance, published data does not include project size, productivity constants or rework/error rates and usually does not include phase effort and schedule. Due to these limitations, the model output can only be validated in certain regards as shown in the following sections.

Two different scenarios have been planned for this simulation purpose. It is assumed that both teams share the same structural characteristics. Both the test scenarios are based on:

1. Experienced staff > 10
2. Inexperience staff < 5
3. Team size > 15
4. Sprint velocity 280 points (avg)
5. Productivity is high
6. Quality is high

7. Performance is high

Test scenario 1:

Since the team is highly productive, presents high performance and quality, it is assumed that the project has a highly effective team. Organizational (OG) context provides a supportive job design and works environment, and management is highly supportive.

In test scenario 2:

All the constant variables' values are taken as average value (Table 6.2) for the same project. The results of the test scenarios are shown in Figures 22 - 24.

Figures 22 - 24 show the results of this run for team motivation, team productivity and team involvement in learning factors. In Test scenario 1, the team builds up its motivation faster given the higher level of OG context and Team management efficiency. The higher the motivation increased of the team, the more comfortably team members are to engage in learning activities, and the earlier the results will appear because team members can respond to changes and correct their course of action more promptly, increasing their potential productivity and hence actual productivity [10].

Figures 22 - 24 show that the engagement of team members in Test scenario 1, learning activities fluctuates more than the engagement of the other teams' scenario (Test scenario 2 and base case). The team involvements in learning-oriented activities are considerably higher at the beginning of the project and adjust downwards as team members learn and gain experience with their work. However, in Test scenario 1, team members are able to respond to decreases in the productivity level more quickly than Test scenario 2 and base case.

Compare to the Test scenario 2, the simulations show that though the team has been given lower performance target (PT = 0.8, OG context= 0.5, Management Skill = 0.5) than the Test Scenario 1, the team couldn't improve motivation and productivity. The reason behind is poor management support. Project management is one of the most influencing factors for ASD teamwork productivity. Supportive and responsive management can help build motivation and team effectiveness with team members by helping them learn to be self –managing an agile team. The simulation results establish the theory that the OG context and TM effect play a relevant role within the ASD team. A defensive and non-motivational TM can prevent team motivation and productivity.

6.4 Summary

This chapter presented a System dynamics model of the ASD teamwork productivity influence factors and performed the relevant tests to ensure its validity. The model is designed on the basis of empirical information gathered (in chapter 5) through literature, interview and survey; and obtains influence factors and their relationships among each other. It also addresses the research question R2: How do the influence factors affect each other, positively, or negatively?

The resulting model attempts to capture dynamic characteristics and nonlinearities of ASD teamwork productivity influence factors with an emphasis on the management of agile teamwork. The scope of the model is limited to analysing the influence of productivity factors and observing productivity rates.

Firstly, the complex inter-related structure of different factors affecting ASD teamwork productivity is modelled using cause and effect feedback loops and the qualitative model of ASD teamwork productivity is constructed. Secondly, the relationships that existed between different factors (such as motivation and team management) are then determined and the quantitative model of the project is built. The relevance and performance of the proposed method in the modelling of ASD productivity are evaluated by its implementation in a real software development project. Using the developed SD model, the value of ASD teamwork productivity is predicted considering the main influencing factors. A sensitivity analysis is then conducted to assess the impact of different factors on ASD teamwork productivity. The proposed SD approach offers a flexible and robust method for the simulation of ASD teamwork productivity with the possibility of finding the root causes of a decrease in productivity. Therefore, teamwork productivity may be improved by the implementation of proper solutions. Although more sample project data are needed to validate the outputs of the model. Moreover, considering the complex structure and dynamic behaviour of different influencing factors may provide the management decision-maker with valuable information.

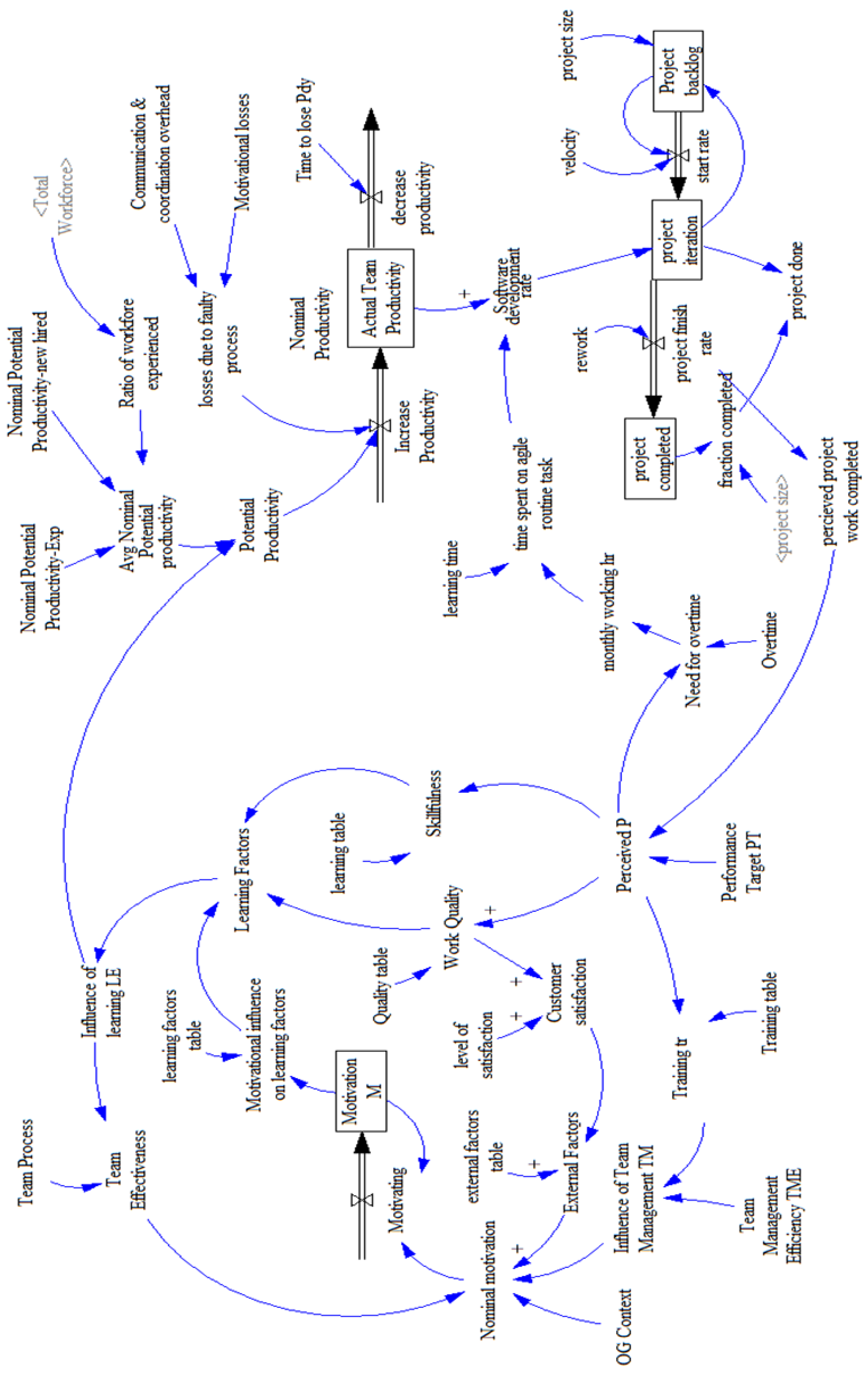


Figure 6.10: Overall System Dynamics Model of Agile Software Development Teamwork Productivity Influence Factors

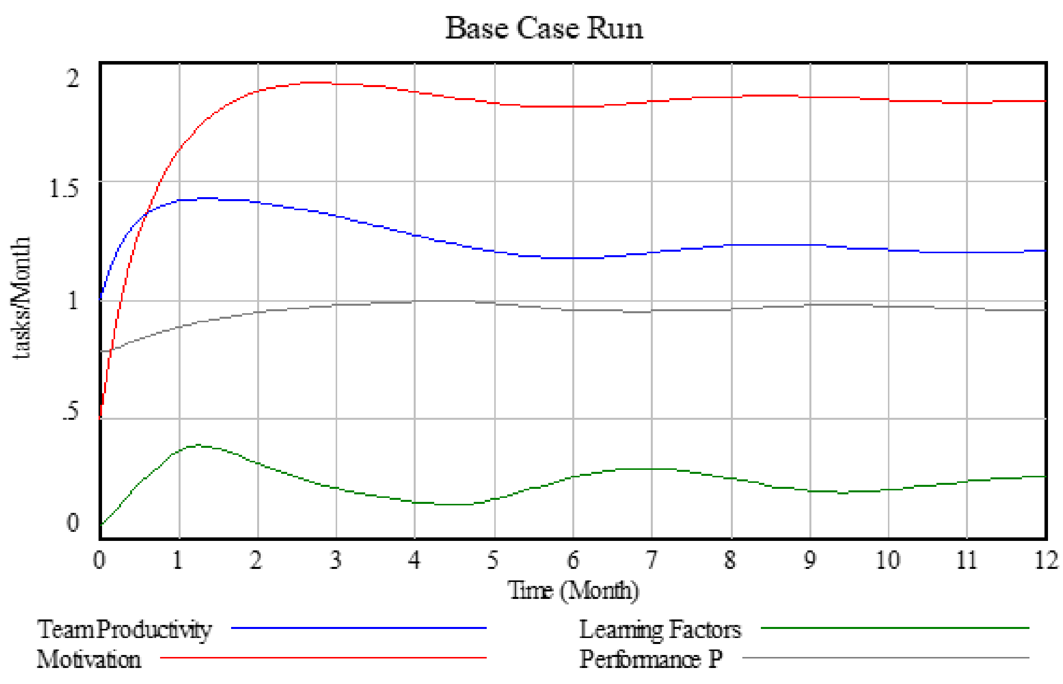


Figure 6.11: Base Case Run

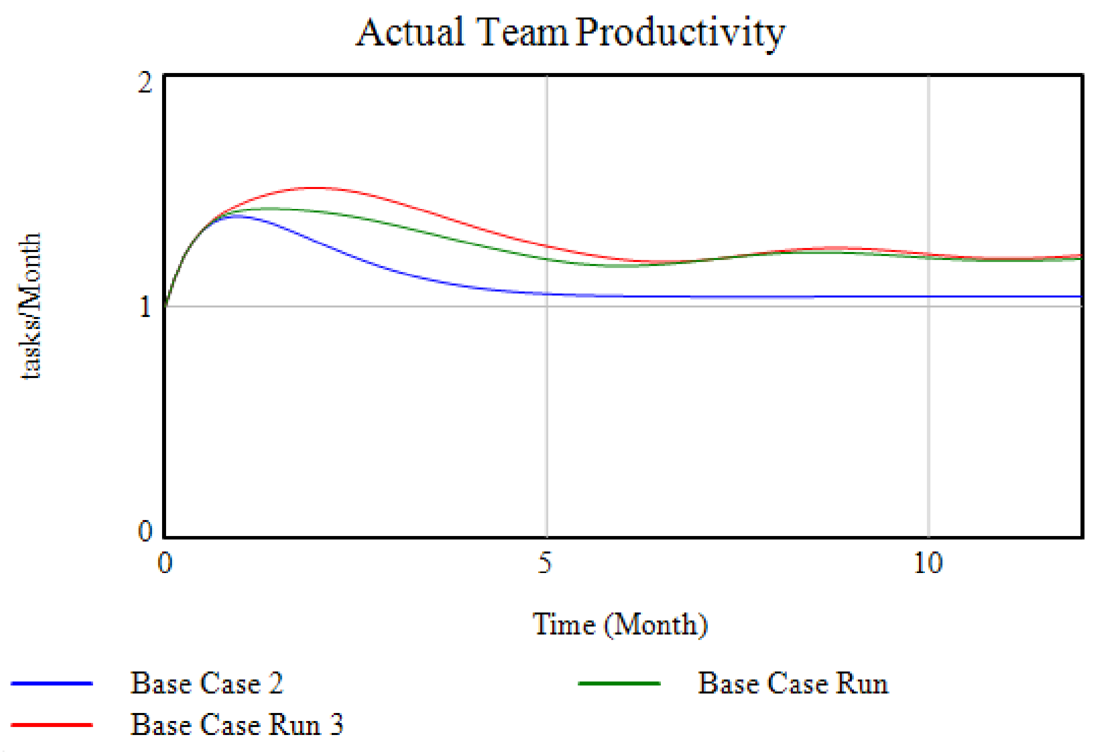


Figure 6.12: Actual team Productivity

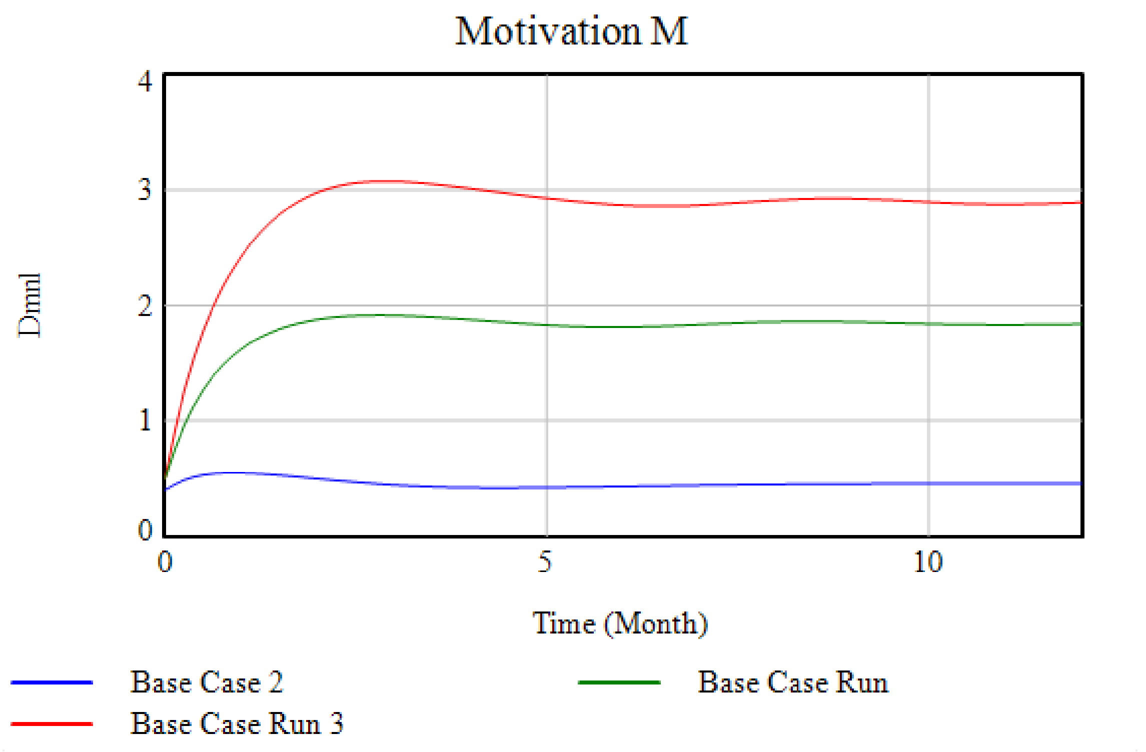


Figure 6.13: Motivation M

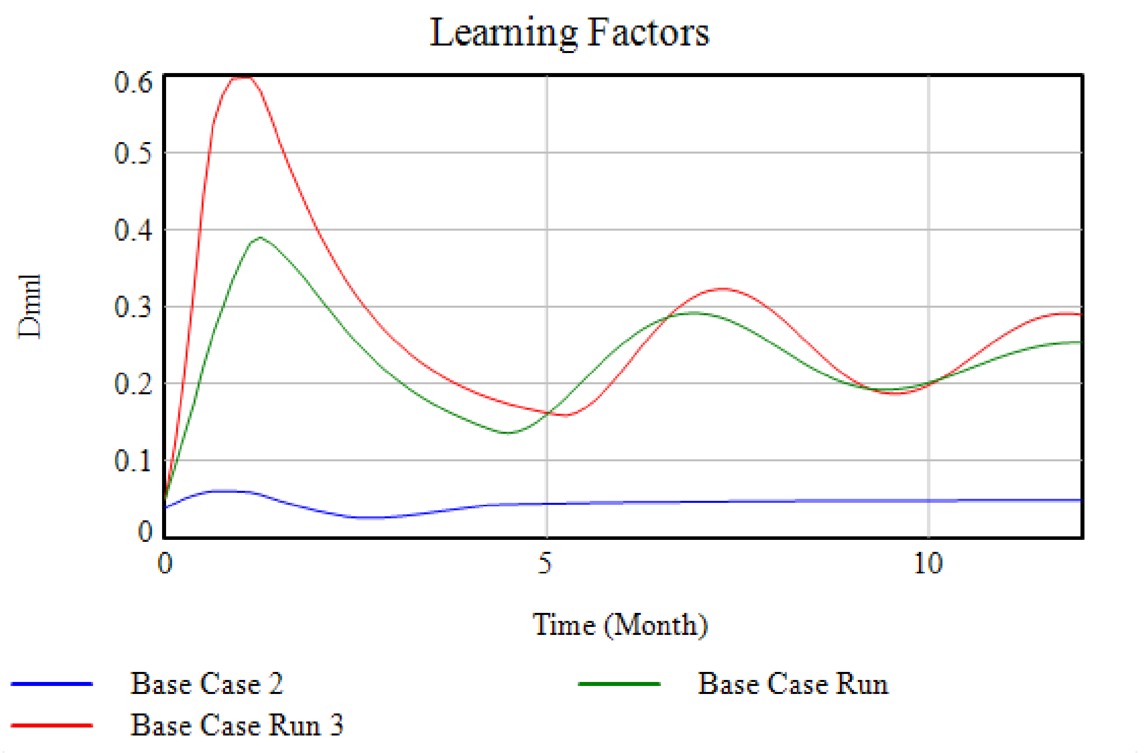


Figure 6.14: Learning Factor

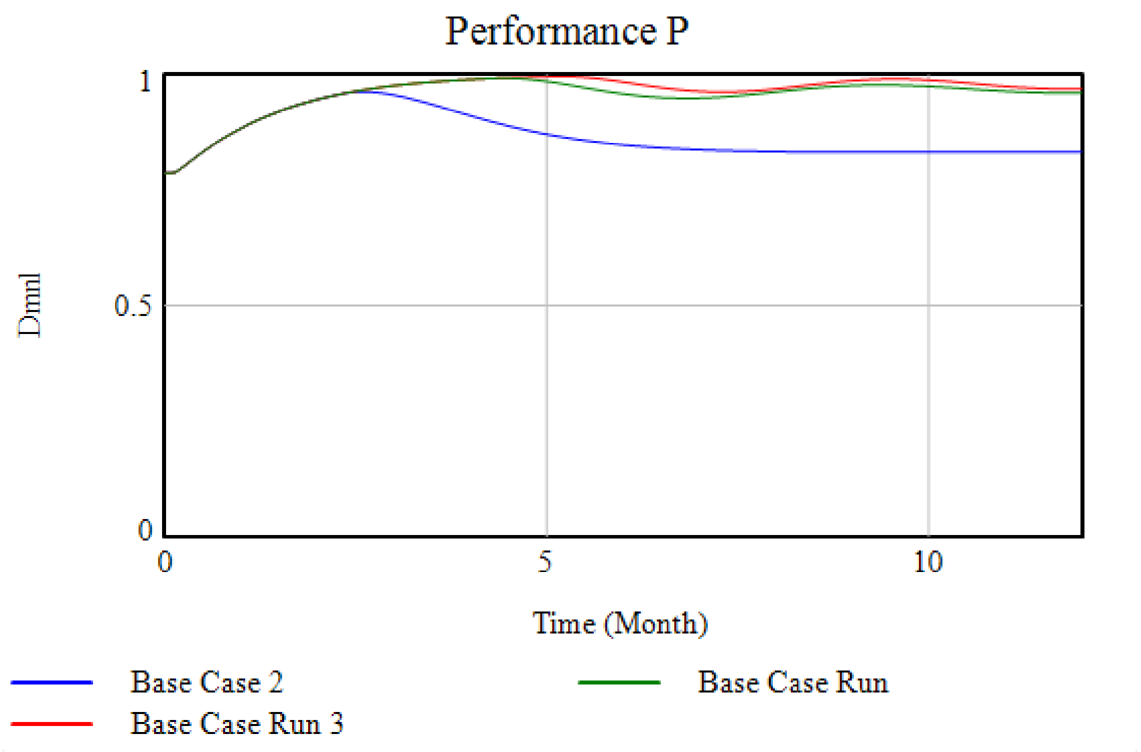


Figure 6.15: Performance P

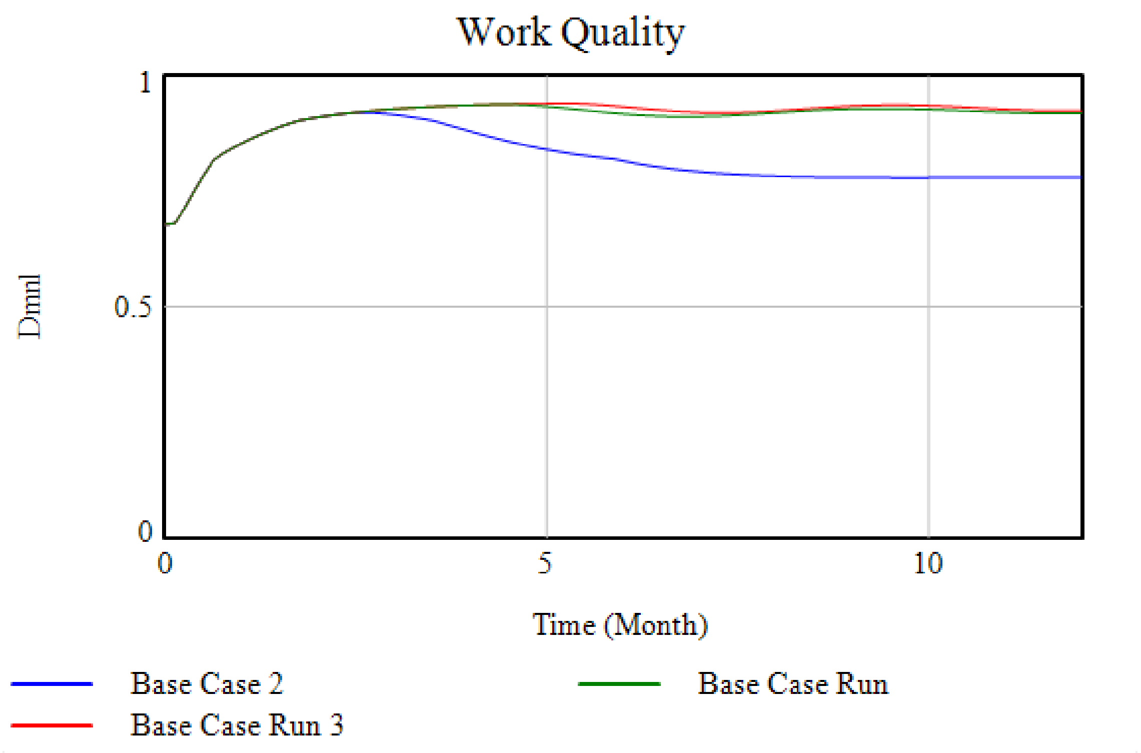


Figure 6.16: Work Quality

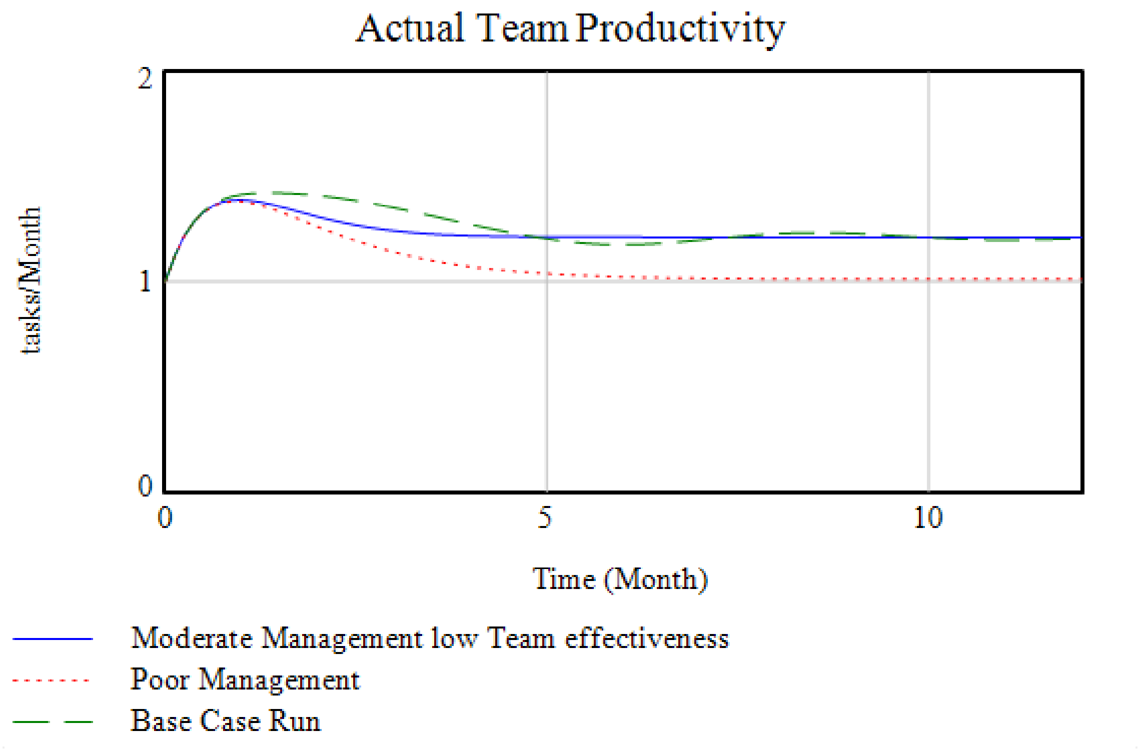


Figure 6.17: Actual team Productivity

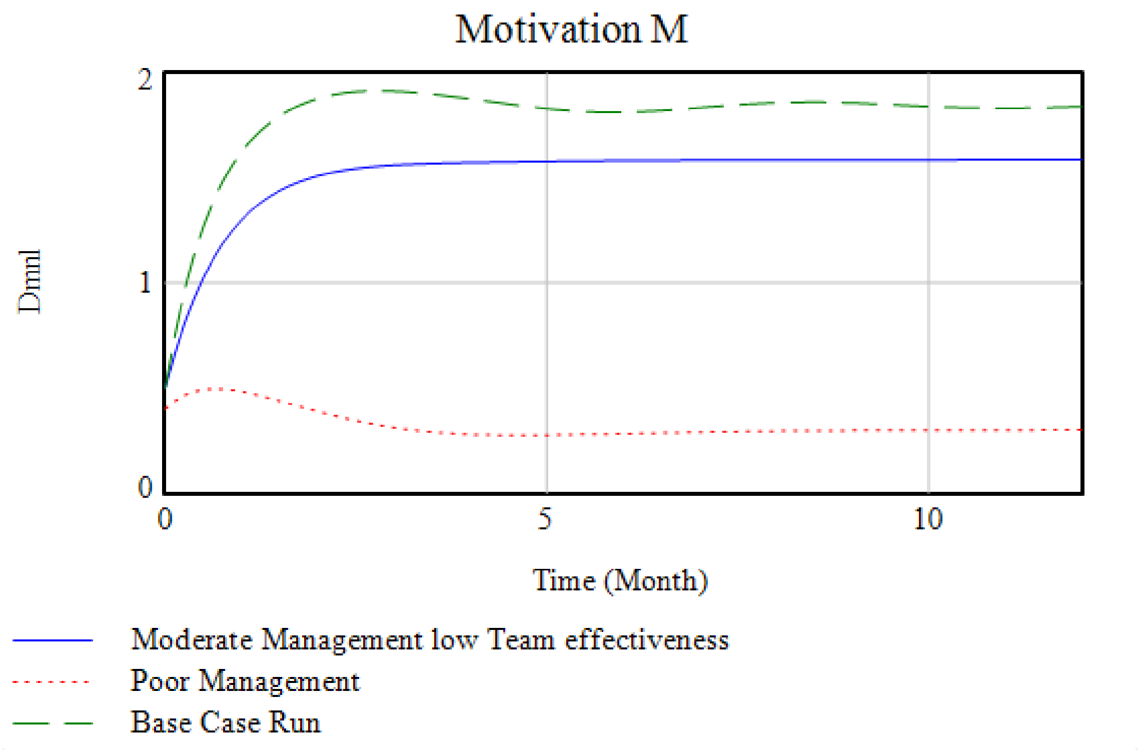


Figure 6.18: Motivation M

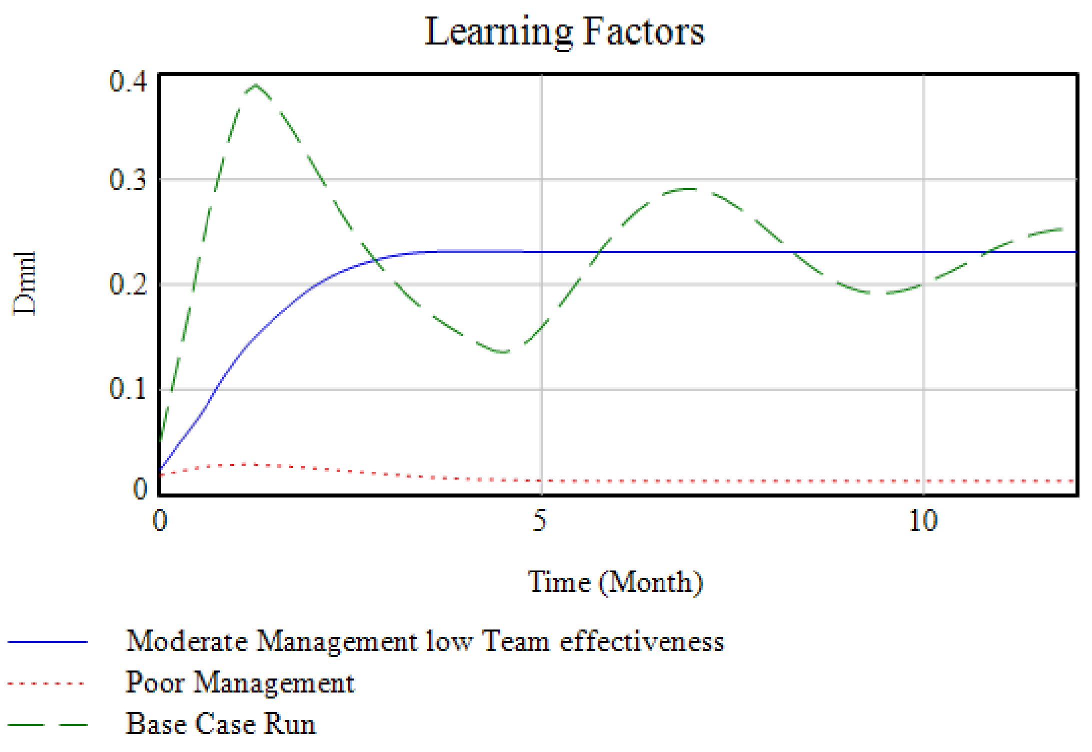


Figure 6.19: Learning Factor

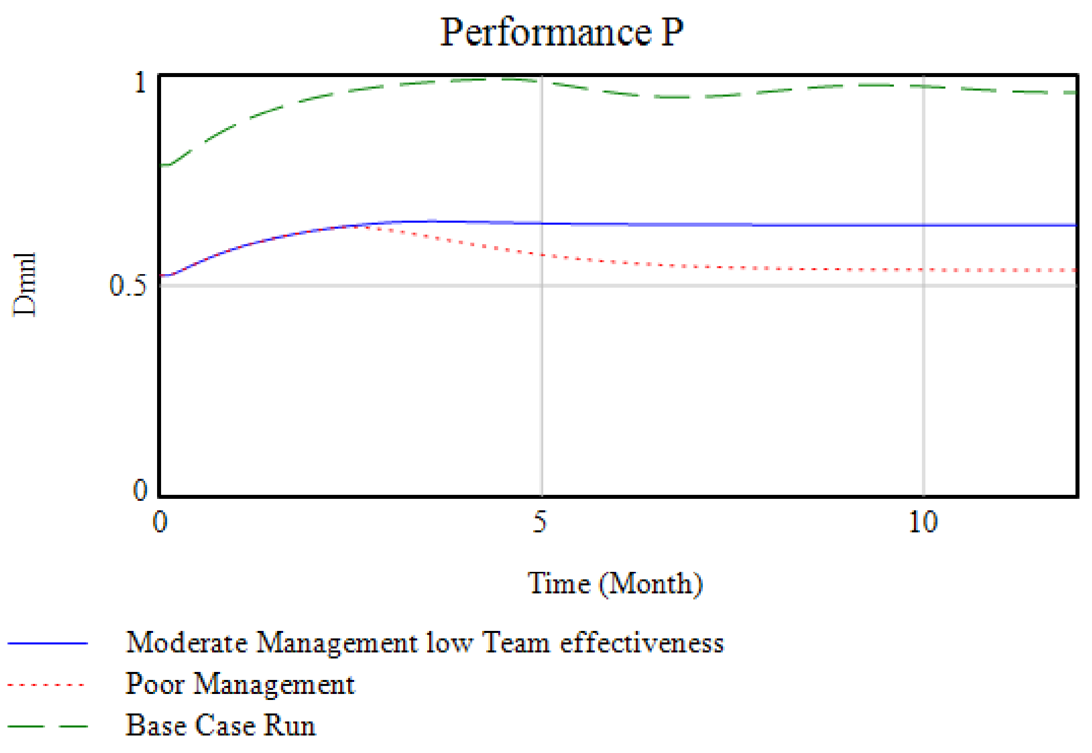


Figure 6.20: Performance P

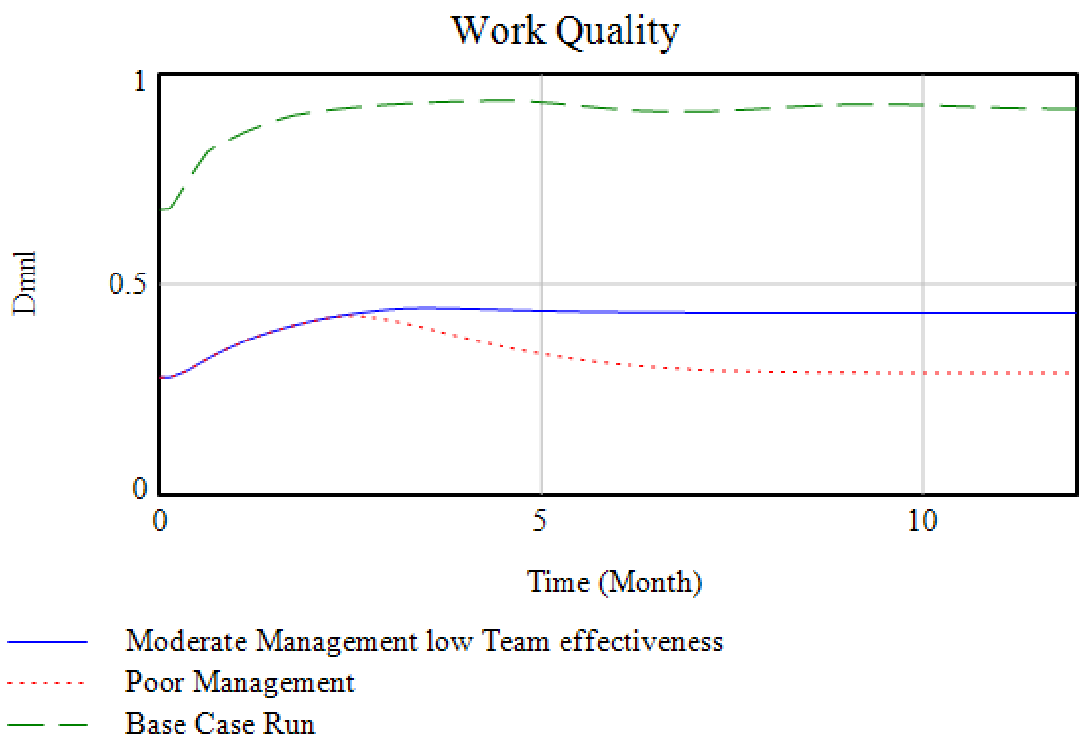


Figure 6.21: Work Quality Test Scenario

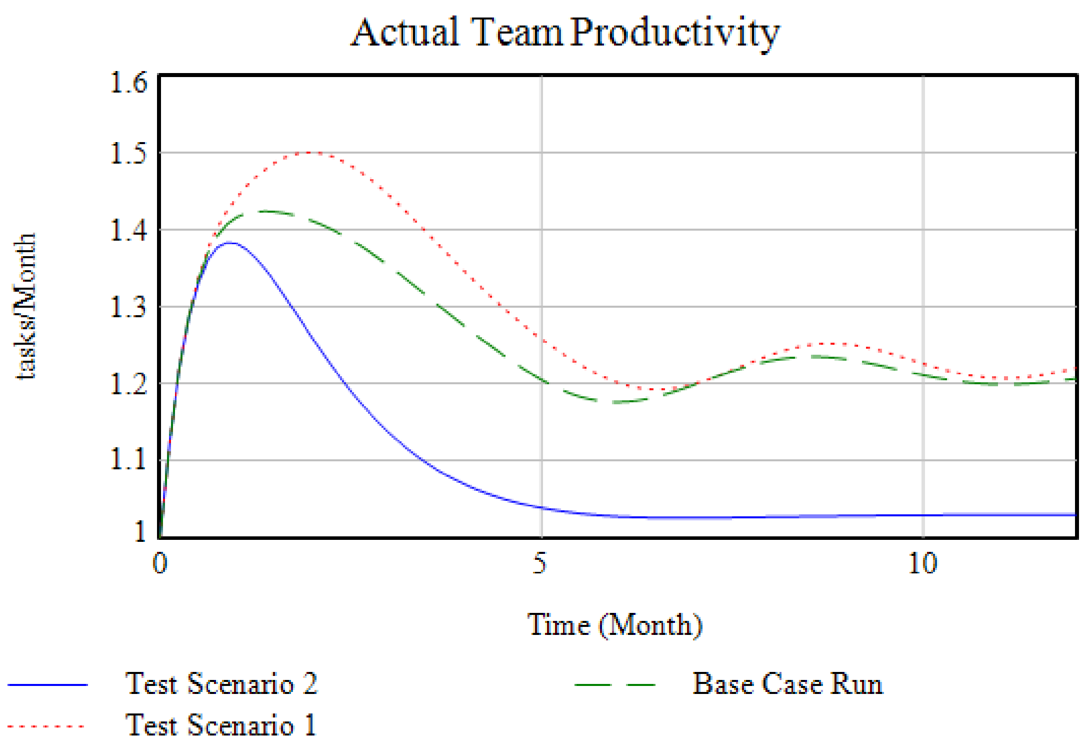


Figure 6.22: Actual team Productivity

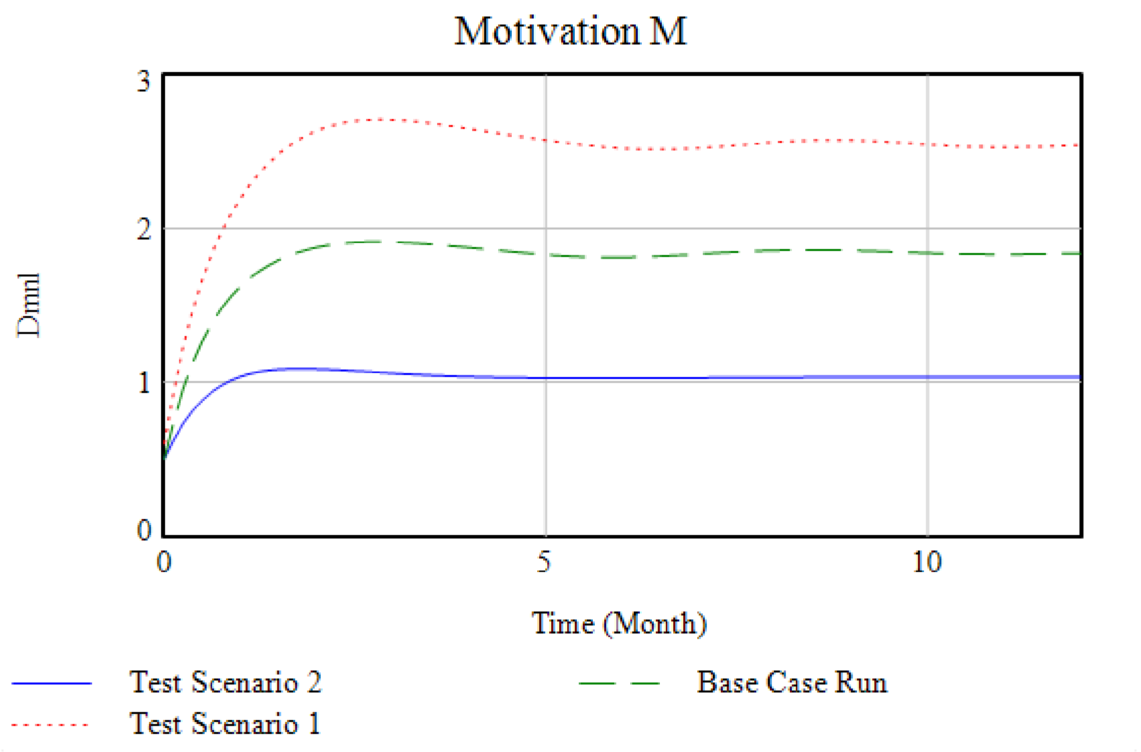


Figure 6.23: Motivation M

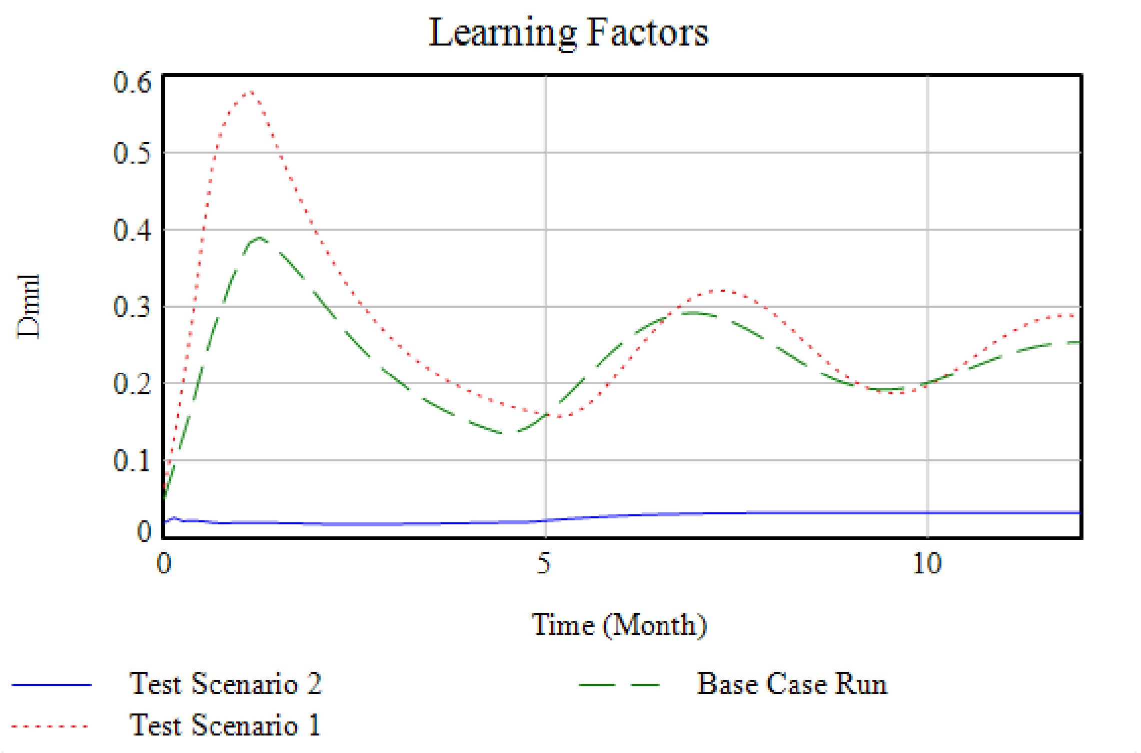


Figure 6.24: Learning Factor

Chapter 7

Discussion and Conclusion

The productivity of the development team is important for a successful software project. The agile team, which is the most dynamic element and the human input in the software development industry, gains more interest to study their productivity. A better knowledge of the factors and the mediators that influence agile teamwork productivity could help determine where to focus management efforts to improve productivity. This research is motivated by an insufficient understanding of dynamic and complex interactions arising from the Agile Software Development (ASD) teamwork productivity influence factors and their relationships. Therefore, this research aimed to identify and understand the complex interdependences and underlying structures at the team's perception level, which influence agile teamwork productivity over time. This is achieved through a two-phase approach. The first phase involves the identification of ASD teamwork productivity influence factors. The second phase involves the construction of the System Dynamics (SD) model of ASD teamwork productivity with the findings from the first phase to analyse the productivity influence factors. The contributions of this thesis build on the results and findings of the research studies presented in chapters 5 and 6. This chapter presents the major results achieved in each of the phases of this study and how they helped to answer the specific research question. The chapter also summarizes the main study limitations and includes ideas for the future development of the SD model to increase its reliability.

7.1 Summary of Results

Teamwork productivity determines the overall project performance in an agile software development process. Therefore, it has gained more interest to study team member's productivity. Agile team members should be taught to interpret and manage productivity factors regularly as they are self-managed. Productivity improvement programs would become effective only if all the variables are simultaneously controlled and monitored. Researchers have tried to quantify and measure how soft factors and social aspects affect teamwork productivity. The agile Software development process must be analysed as a behavioural process [23]. Therefore, coordinating and managing an agile team is a vital activity for software companies and team dynamics have a direct influence on teamwork productivity. One effective solution to improve productivity is to look into the factors influencing productivity and also have a dynamic strategical model that tells the project manager in advance the degree of impact that these factors will have on team productivity.

In order to achieve that, the main factors that affect teamwork productivity were determined via a two-phased approach. In the first phase a systematic literature review, interview and survey of different agile teams were conducted to collect and select impacting factors. The collected factors were evaluated by simple descriptive statistical analysis and ranked to identify the most influential ones. The second phase involved the development of the SD simulation model of ASD teamwork productivity.

The research questions posed in Chapter 1 were answered and a brief discussion follows:

7.1.1 Answers to Research Questions

To identify influence factors, this study chose to develop a questionnaire and conduct a semi-structured interview with responses based on the perception and observation of agile practitioners in 18 Bangladeshi software companies. This study now discusses the survey results in light of the research questions.

R1. What factors do have an influence on agile teamwork productivity and how is this influence from the team perspective?

The findings of this stage are the main influencing factors, which are motivation (external factors, customer satisfaction), team effectiveness (communication, coordination, mutual trust, leadership) and team management (staffing, key personnel stayed throughout the project, team design choice). Moreover, this study used two team effectiveness models- Salas [68] and the Dickenson McIntyre [59] model for better understanding and analysing inter aspects of an agile team. The most cited and influential factors are Coordination, Communication, mutual trust and staffing (right person selected for the team). These factors are the most important for effective teamwork and team management in agile teamwork productivity. Among the identified influential factors, staffing (aggregates team design choice) remains an important factor affecting agile teamwork productivity. It is even more obvious on agile teams that depend on teamwork and people factors. This study results indicated that staffing that includes: team size, diversity, personality, skills, collocation, and full-time allocation are key factors to be considered when designing agile teams.

According to study results, lack of management support is found to be the most mentioned reason for any failed agile project. The most followed organizational structure is horizontal and the most followed agile method is Scrum. In addition, this study finds that due to social hierarchy culture influences, the self-managed agile team may not fully fit in their organization. This factor also hinders agile transformation from plan driven to a self-managed agile team. Among agile practices, the daily stand-up meeting is the most cited practices impacting teamwork productivity. Customer satisfaction is found as the main criterion for measuring or perceiving productivity by the interviewees and survey respondents.

R2. How do the influence factors affect each other, positively, or negatively?

The findings from the first phase were compiled into the System Dynamics (SD) model for quantitative analysis of the teamwork productivity influencing factors. The complex inter-related structure of different factors affecting ASD teamwork productivity was modelled using cause and effect feedback loops and the qualitative model of ASD teamwork productivity was constructed. The relationships that existed between different factors were then determined and the quantitative model of the project was built.

Before interpreting the results of the model, the model has to be verified and validated. Although there was no actual data for validation, some of the simulated results from literature for a similar kind of problem were used to validate the model. The result of the verification and validation test shows the viability of the model for the intended purpose.

To better understand how the model works, important parameters of the base case have been presented and their behaviour has been explained. Several scenarios and sensitivity analyses were simulated to study the affect of the influencing factors on ASD teamwork productivity. The applicability and performance of the proposed method in the modelling of ASD teamwork productivity were evaluated by its implementation in a real-world agile software development project. Using the developed SD model, the value of ASD teamwork productivity was predicted considering the influencing factors. Although more sample projects are needed to validate the outputs of the model, yet, accounting for the complex structure and dynamic behaviour of different influencing factors may provide the decision-maker with valuable information. The proposed SD model will provide more strategic insights and understanding about the effectiveness of different managerial policies based on non-straight forward cause-effect relationships hidden in the system.

Results of the model simulations show that motivation, OG context, and adequate resources have an important role in providing a supportive environment for an agile team. It positively influences the agile team to perform their tasks more effectively. A motivated team is more likely to involve in learning activities. Even though, the minimum level of motivation can foster team learning and hence positively influence potential productivity. Team management plays a key role in driving the dynamics of the system. The simulations show that a high-performance target without adequate team management support is not favourable. Because they lead to high expectations and productivity and consequently to less quality and customer satisfaction. High customer satisfaction positively affects team motivation.

The simulation results found that team effectiveness can be achieved if the team has enough management support and OG context. In this kind of scenario, the team presents mutual trust, a high level of self-confidence that lessens the affect of a lack of training.

To summarize the above dynamics, emphasize that the perception of agile team management should be revised and more emphasis should be given to preparing agile team members to work effectively and independently to achieve desirable teamwork productivity. Because inherent social hierarchy culture in Bangladeshi software companies usually influence self-managed agile teams excessively that lessen agility, restrict creativity and resist change.

This study tries to show that how SD simulation models might be used for analysing and observing the dynamic complexities in ASD projects and its' teamwork. The interrelationships between the factors are more apparent and so their affects might be more easily observed by such a model. Decision-makers and project managers can experiment with the model in order to improve their understanding of the important affects, the interrelationships, and the complex feedback loops.

7.2 Overall Observation for software companies in Bangladesh

This study observed the experiences and perceptions of software development companies that have used or are using agile methods or practices in Bangladesh. The observation indicates that:

- The motivations behind the adoption of agile are similar to most companies worldwide. The main motivation is the flaws in the traditional development methods, such as the issues of late delivery, over budget, and being unadaptable to changes. In contrast, the agile development methods have the capacity for addressing such issues, and hence, are chosen as the optimal solution by most companies.

- There are some obstructions to the adoption of agile software development in software companies in Bangladesh. Specifically, there are some internal obstructions surround the organizational issues, namely team member knowledge, management, and team cooperation and communication. The most common internal obstacle to the adoption process is the lack of knowledge. Since agile software development is a relatively new concept to the Bangladeshi developer community, the issue of lacking knowledge is more or less understandable. At the same time, the lack of information sessions and professional training courses has deteriorated the situation.

- This research found that project management plays a vital role in the success of the adoption process, in which the managers introduce the optimal plan to motivate their agile team and help them overcome the difficulties. Besides encouraging the team, the manager is also responsible for the execution of a project as well as dealing with the customers. Hence, the achievement of a good management strategy requires great effort and skills from the managers.

- Together with the inappropriate management strategy, the inactive characteristic of the Bangladeshi people also brings negative effects to the software companies. This is due to the fact that software developers in Bangladeshi companies are accustomed to working individually rather than in a team. As a result, the communication and collaboration factors cannot be sufficiently utilized. Moreover, the software developers feel safer when working on a hierarchy basis, in which they do the tasks assigned by the managers. In other words, because of the social hierarchy culture influences, they are afraid of taking more responsibilities because it may risk their work performance.

- From the research, the issues of project compatibility and the customer's perspectives are discovered and classified as the external dependency obstruction. Because agile offshoring has recently become an issue of global import [2], more studies are needed to investigate the factors affecting agile offshoring success.

7.3 Limitations of the Study

In interpreting the findings of this study, some limitations need to be acknowledged:

7.3.1 Survey Methodology Limitations

There are a number of limitations to this study. Firstly, this study was limited to 60 respondents and 12 interviewees from 18 software companies. It was challenging to get access to more software companies due to time constraint and its access to appropriate resources was limited. Respondents were carefully chosen from different roles within the agile team in order to get different perspectives on productivity in the context of the Bangladesh software Industry. Another limitation of this study is the agile team members' perceptions used as a response. However, with the survey, this study relies on what the respondents provided to the researcher. It is possible that the respondents' perceptions may change and be different after the end of the project. To minimize the impact of this effect, the survey and interviewees' responses were compared for factors selection to include in the model. The questionnaire used for this study had been used successfully in other research and was developed after a detailed literature review [5] [61]. Some of the questions were included in the survey after getting knowledge about the working conditions of software companies in Bangladesh from the interview sessions. The scope of these empirical findings considers the Bangladeshi software companies as a case study, which can, in turn, make the research results beneficial to these companies. All the data used in this study is collected from the software companies who have voluntarily participated in this research. Therefore, findings from this study should be generalized with caution. While the findings may be specific to the contexts studied, analytic generalization could facilitate the application to other types of culture, background, and environment.

7.3.2 SD Limitations

Firstly, most of the time the SD model reflects the subjective perceptions of the researcher rather than a reproduction of the real-world system. This study nearly overcame this limitation by including assumptions based on empirical data (collected by survey and interview) rather than only on subjective assumptions. The second limitation is the use of hypothetical data when developing the SD model. According to Barlas, an exact matching between real data and model data is not necessary for a system dynamics model [105]. In SD, the main concern is to show whether the parameters have an increasing or decreasing affect on another parameter. The third limitation is about model validation. SD has often been criticized for relying too much on informal, subjective and qualitative validation procedures [104]. In this research, limited access to information for both the creation of the production sector and lack of direct access to primary data put-on problems for replicating the behaviour and estimating values of some variables (such as team process), several assumptions had to be made.

7.4 Further Research

Finally, this chapter concludes by providing some ideas to extend and improve this research in the future. The model developed in this research represents a formal case and would require calibration to specific project environments. The proposed model needs to be further elaborated and validated.

The model can be validated against data obtained from any real-time software development project developed by some companies. This can give deeper insights and might also improve the model. The model can be extended by including all the variables described in causal loop diagrams. SD modelling and simulation is an inexpensive way to gain deep insights when real-time data is unavailable.

This research outlined the work on team effectiveness model of teamwork and on supporting theory. Thus, different teamwork theories can explain this research work.

The simulation model can be extended to include other influence factors and feedback loops. Keeping the simulation model up-to-date with regard to new markets, competitors, and organization changes.

Future research can use more statistical data to estimate parameters and assess the ability of the model to replicate historical data when numerical data are available.

Appendix A

Survey Questionnaire

Agile Teamwork Productivity Influence Factors

The questionnaire consists of statements about your team, how your team functions as a group and your perception of agile teamwork productivity influence factors.
*No personally identifiable information will be associated with your responses to any reports of these data. All information provided will be treated strictly as confidential and purely for academic purpose.

* Required

1. Name of your Organization *

____

2. What is the main activity of your organization?
Mark only one oval.

- ◯ Multimedia
- ◯ Games/Entertainment
- ◯ Embedded systems
- ◯ Mobile application
- ◯ Financial
- ◯ Education
- ◯ Consulting/services
- ◯ Communications
- ◯ Other: ______________________

3. What are you working on now?
Mark only one oval.

- ◯ Development project
- ◯ Maintenance project
- ◯ Other: ______________________

4. What is the status of the project?

____

5. What is your role in the project and how long have you been working on it?

____

6. How long have you personally been practically using Agile development method?
Mark only one oval.

- ◯ Less than 6 months
- ◯ 6 - 12 months
- ◯ 1 - 2 years
- ◯ 2 - 5 years
- ◯ > 5 years

7. How large is your total software organization ?

____

8. What is your team size?

____

9. What is the most followed agile method in your company?
Mark only one oval.

- ◯ Scrum
- ◯ XP
- ◯ Other: ______________________

10. What are the Agile practices adopted in your company?
Check all that apply.

- ☐ Daily standup
- ☐ Release planning
- ☐ Continuous integration
- ☐ Automated builds
- ☐ Burndown
- ☐ Retrospectives
- ☐ Refactoring
- ☐ Continuous deployment
- ☐ Velocity
- ☐ Kanban
- ☐ Pair Programming
- ☐ stories
- ☐ Iteration Planning
- ☐ On-site customer
- ☐ Collocation (placing team members near each other)
- ☐ Other: ______________________

11. What are the mostly used programming languages in your organization?

____

12. What were the main reasons for failed agile project (if any)?
Check all that apply.

- ☐ Lack of cultural transition
- ☐ Lack of management support
- ☐ Unwillingness of the team
- ☐ Insufficient training
- ☐ Lack of experience with Agile method
- ☐ Other: ______

13. Criterion for measuring or perceiving productivity in your organization
Check all that apply.

- ☐ Timeliness
- ☐ Quantity
- ☐ Quality
- ☐ Customer satisfaction
- ☐ Other: ______

14. Teamwork productivity influence factors:Please indicate the level to which each statement describes your perception by circling the number to the right of each factor.

Mark only one oval per row.

	Low	Medium	High
Culture (Agile requires a true cultural change from plan based approach, not only a simple change in the processes used)	◯	◯	◯
Staffing (the right persons should be selected)	◯	◯	◯
Size of team (small and mixed team)	◯	◯	◯
Project complexity	◯	◯	◯
Team Leadership (Team leadership can be shown by several team members)	◯	◯	◯
Mutual performance monitoring (being aware of other team members' performance)	◯	◯	◯
Backup Behaviour (being available to assist other team members when needed)	◯	◯	◯
Team orientation (assigning high priority to team goals and participating willingly in all relevant aspects of the team)	◯	◯	◯
Adaptibility (response to changing conditions,internal or external)	◯	◯	◯
Feedback (giving, seeking, and receiving of information among team members)	◯	◯	◯
Mutual trust (shared belief that team members will perform their roles and protect the interests of their team-mates)	◯	◯	◯
Coordination (team members executing their activities in a timely and integrated manner)	◯	◯	◯
Communication (exchange of information between two or more team members in the prescribed manner and using appropriate terminology)	◯	◯	◯
Staff are appreciated for working long hours	◯	◯	◯
Staff are rewarded (then or later) for working long hours	◯	◯	◯
Adequate technical training for team	◯	◯	◯
Adequate team skills training for team (communication, organization, interpersonal, etc.)	◯	◯	◯
Team member turnover	◯	◯	◯
Key personnel stayed throughout the project	◯	◯	◯
What is the staff turnover rate in the project?	◯	◯	◯
Reuse (of software products, processes, artifacts, including components,frameworks, and software product line)	◯	◯	◯
What is the software reuse level in the project?	◯	◯	◯
Goals (establishment is critical for the success of the team	◯	◯	◯

	Low	Medium	High
Intra group wage inequality (fair wage)	◯	◯	◯
Team measurement	◯	◯	◯
Self-management (most work-related decisions are made by the members of team rather than manager)	◯	◯	◯
Task variety and Innovation (team get chance to learn the different tasks the team perform to meet the workload needs of the team)	◯	◯	◯
External Dependencies (waiting for customer acceptance/for a component; interacting with external customers; publishing version of system/of data model across different environments)	◯	◯	◯
Tools usage	◯	◯	◯
Programming language	◯	◯	◯
Schedule pressure	◯	◯	◯
Impact of Pair programming on productivity	◯	◯	◯
Resource constraints (e.g. timing, reliability,storage, team size, and project duration)	◯	◯	◯
Project Management (quality of management, conflict management, task assignment, administrative and formal coordination)	◯	◯	◯
Motivation (to work on the project and in the company)	◯	◯	◯
External project factors (Customer involvement, Customer expectation, Customer satisfaction)	◯	◯	◯
Dealing with cultural differences among offshore organizations	◯	◯	◯
Working environment (suitability of the workplace to do creative work, e.g., windows, natural light, size of room and desk, meals provided)	◯	◯	◯

15. What do you think that most influences your team's productivity?

16. Is there anything else you would like to add that you think is interesting in this context, but not covered by the questions asked?

Powered by Google Forms

Bibliography

1. A. Hernandez-Lopez, R. Colomo-Palacios, and A. Garcia-Crespo, "Software engineering job productivity—a systematic review," *International Journal of Software Engineering and Knowledge Engineering*, vol. 23, no. 03, pp. 387-406, 2013.

2. I. Fatema and K. M. U. Sakib, "Using Qualitative System Dynamics in the Development of an Agile Teamwork Productivity Model," *International Journal On Advances in Software*. Vol. 11, no. 12, pp: 170–185, 2018.

3. V. Nguyen, L. Huang, and B. Boehm, "An analysis of trends in productivity and cost drivers over years," in *Proceedings of the 7th International Conference on Predictive Models in Software Engineering*, 2011, p. 3: ACM.

4. Percent-Successful-IT-Projects-MetricsQuest. Oct 18, 2017. [Online]. Available from **Percentage successful IT projects worldwide is still too low**. [retrieved: June, 2019].

5. C.O. Melo, "Productivity of agile teams: an empirical evaluation of factors and monitoring processes," Ph.D. dissertation, Universidade de São Paulo, 2015.

6. C. Schmidt, "Agile Software Development," in *Agile Software Development Teams*: Springer, 2016, pp. 7-35.

7. F. Maurer and S. Martel, "On the productivity of agile software practices: An industrial case study," in *Proceedings of the International Workshop on Global Software Development*, 2002, vol. 20.

8. VersionOne: 12th annual state of Agile survey (2017). [Online]. Available from: https://www.versionone.com/about/press-releases/12th-annual-state-of-agile-survey-open/ [retrieved: May, 2018]

9. V. Lalsing, S. Kishnah, and S. Pudaruth, "People factors in agile software development and project management," *International Journal of Software Engineering & Applications*, vol. 3, no. 1, p. 117, 2012.
10. X. Kong, L. Liu, and D. Lowe, "Modeling an agile web maintenance process using system dynamics," in *11th ANZSYS/Managing the Complex V conference, ISCE Publishing, Christchurch, NZ*, 2005: Citeseer.
11. A. Trendowicz and J. Münch, "Factors influencing software development productivity—state-of-the-art and industrial experiences," *Advances in computers*, vol. 77, pp. 185-241, 2009.
12. C. D. O. Melo, D. S. Cruzes, F. Kon, and R. Conradi, "Interpretative case studies on agile team productivity and management," *Information and Software Technology*, vol.

55, pp.412-427, Feb.2013.

13. K. Petersen, "Measuring and predicting software productivity: A systematic map and review," *Information and Software Technology,* vol. 53, no. 4, pp. 317-343, 2011.

14. Y. W. Ramírez and D. A. Nembhard, "Measuring knowledge worker productivity: A taxonomy," *Journal of intellectual capital,* vol. 5, no. 4, pp. 602-628, 2004.

15. How to estimate the required team size of agile teams? ISBSG. [Online]. Available from http://isbsg.org/wpcontent/ uploads/2017/08/Estimate-Agile-team-size-v1.0.pdf [retrieved: May, 2018].

16. Productivity measurement in Agile teams – why is it so difficult? [Online]. Available from http://isbsg.org/isbsgagile/ [retrieved: May, 2018].

17. T. DeMarco and T. Lister. Peopleware. Productive Projects and Teams. Dorset House Publishing, 1987.

18. J. E. Hannay and H. C. Benestad, "Perceived productivity threats in large agile development projects," in *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, 2010, p. 15: ACM.

19. E. F. Wolstenholme and R. G. Coyle, "The development of system dynamics as a methodology for system description and qualitative analysis," *Journal of the Operational Research Society*, vol. 34, no. 7, pp. 569-581, 1983.

20. J. Sterman, "Busines Dynamics: System Thinkng and Modeling for A Complex World. Boston: The McGrau Hill Companies," ed: Inc, 2000.

21. Practical Guide: Productivity Measurement of Software Projects. [Online]. Available from
https://www.isbsg.org/wp-content/uploads/2019/09/Productivity-Measurements-of-Software-Projects.pdf [retrieved: September, 2019].

22. **Survey Data Shows That Many Companies Are Still Not Truly Agile.** [Online]. Available from https://hbr.org/sponsored/2018/03/survey-data-shows-that-many-companies-are-still-not-truly-agile [retrieved: September, 2019].

23. P. Abrahamsson, O. Salo, J. Ronkainen, and J. Warsta, "Agile software development methods: Review and analysis," *arXiv preprint arXiv:1709.08439,* 2017.

24. K. Beck *et al.*, "Manifesto for agile software development," 2001.

25. M. Fowler and J. Highsmith, "The agile manifesto," *Software Development,* vol. 9, no. 8, pp. 28-35, 2001.

26. T. Dybå, T. Dingsøyr, and N. B. Moe, "Agile project management," in *Software project management in a changing world*: Springer, 2014, pp. 277-300.

27. S. Misra, V. Kumar, U. Kumar, K. Fantazy, and M. Akhter, "Agile software development practices: evolution, principles, and criticisms," *International Journal of Quality & Reliability Management,* vol. 29, no. 9, pp. 972-980, 2012.

28. F. Glaiel, "Agile project dynamics: a strategic project management approach to the study of large-scale software development using system dynamics," Massachusetts Institute of Technology, 2012.

29. V. G. Stray, N. B. Moe, and T. Dingsøyr, "Challenges to teamwork: a multiple case study of two agile teams," in *International conference on agile software development,* 2011, pp. 146-161: Springer.

30. Y. Kozak, "Barriers against better team performance in agile software projects," *Chalmers University of Technology, Sweden,* 2013.

31. N. B. Moe and T. Dingsøyr, "Scrum and team effectiveness: Theory and practice," in *International Conference on Agile Processes and Extreme Programming in Software Engineering*, 2008, pp. 11-20: Springer.

32. T. J. Gandomani and M. Z. Nafchi, "Agile transition and adoption human-related challenges and issues: A Grounded Theory approach," *Computers in Human Behavior,* vol. 62, pp. 257-266, 2016.

33. G. Robiolo and D. Grane, "Do agile methods increase productivity and quality?," *American Journal of Software Engineering and Applications*, vol. 3, no. 3, pp. 1-11, 2014.

34. D. S. Nguyen, "Success factors for building and managing high performance agile software development teams," *International Journal of Computer,* vol. 20, no. 1, pp. 51-

82, 2016.

35. R. Antikainen and A. Lönnqvist, "Knowledge work productivity assessment," *Institute of Industrial Management. Tampere University of Technology. PO Box,* vol. 541, pp. 79-102, 2006.

36. Y. W. Ramírez and D. A. Nembhard, "Measuring knowledge worker productivity: A taxonomy," *Journal of intellectual capital,* vol. 5, no. 4, pp. 602-628, 2004.

37. W. Scacchi, "Understanding and improving Software Productivity," *Advances in Software engineering and Knowledge engineering*, 2005.

38. H. van Heeringen, T. Prins, and E. van Gorp, "Productivity Measurement of Agile Teams–Overcoming the Issues with Non-Functional Requirements," *Collegium of Economic Analysis Annals,* no. 43, pp. 117-134, 2017.

39. M. Pagels, "Assessing the viability of implicitly estimated velocity for measuring the productivity of software teams," 2013.

40. B. Barry, "Centre for Systems and Software Engineering," Oct. 2012. [Online] Available From http://sunset.usc.edu/csse/research/COCOMOII/cocomo_main.html. [retrieved: August, 2018].

41. A. Trendowicz and J. Münch, "Factors influencing software development productivity—state-of-the-art and industrial experiences," *Advances in computers,* vol. 77, pp. 185-241, 2009.

42. S. C. de Barros Sampaio, E. A. Barros, G. S. de Aquino Junior, M. J. C. e Silva, and S. R. de Lemos Meira, "A review of productivity factors and strategies on software development," in *2010 fifth international conference on software engineering advances,* 2010, pp. 196-204: IEEE.

43. D. Nguyen, "Success factors that influence agile software development project success," *American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS),* vol. 17, no. 1, pp. 171-222, 2016.

44. G. Purna Sudhakar, A. Farooq, and S. Patnaik, "Soft factors affecting the performance of software development teams," *Team Performance Management: An International Journal,* vol. 17, no. 3/4, pp. 187-205, 2011.

45. C. Lokan, T. Wright, P. Hill, and M. Stringer, "Organizational benchmarking using the ISBSG data repository," *IEEE Software,* vol. 18, no. 5, pp. 26-32, 2001.

46. K. Petersen, "Measuring and predicting software productivity: A systematic map and review," *Information and Software Technology,* vol. 53, no. 4, pp. 317-343, 2011.

47. F. S. Glaiel, A. Moulton, and S. E. Madnick, "Agile project dynamics: A system dynamics investigation of agile software development methods," 2014.

48. F. Nasirzadeh and P. Nojedehi, "Dynamic modeling of labor productivity in construction projects," *International journal of project management,* vol. 31, no. 6, pp. 903-911, 2013.

49. S. M. A. Shah, E. Papatheocharous, and J. Nyfjord, "Measuring productivity in agile software development process: a scoping study," in *Proceedings of the 2015 International Conference on Software and System Process*, 2015, pp. 102-106: ACM.

50. M. I. Kellner, R. J. Madachy, and D. M. Raffo, "Software process simulation modeling: why? what? how?," *Journal of Systems and Software,* vol. 46, no. 2-3, pp. 91-105, 1999.

51. R. G. Coyle, "System dynamics modelling: a practical approach," *Journal of the Operational Research Society,* vol. 48, no. 5, pp. 544-544, 1997.

52. A. Ortiz, J. M. Sarriegi, and J. Santos, "Applying modelling paradigms to analyse organisational problems," in *Proceedings of the 24th International Conference of the System Dynamics Society*, 2006.

53. C. Andersson and L. Karlsson, "A system dynamics simulation study of a software development process," *CODEN: LUTEDX (TETS-5419)/1-83,* 2001.

54. J. D. Sterman, "System dynamics modeling: tools for learning in a complex world," *California management review,* vol. 43, no. 4, pp. 8-25, 2001.

55. G. Ossimitz and M. Mrotzek, "The basics of system dynamics: discrete vs. continuous modelling of time," in *The 26th Conference of SD*, 2008: Citeseer.

56. A. Rodrigues and J. Bowers, "System dynamics in project management: a comparative analysis with traditional methods," *System Dynamics Review: The Journal of the System Dynamics Society,* vol. 12, no. 2, pp. 121-139, 1996.

57. S. Kuppuswami, K. Vivekanandan, and P. Rodrigues, "A system dynamics simulation model to find the effects of XP on cost of change curve," in *International Conference on Extreme Programming and Agile Processes in Software Engineering*, 2003, pp. 54-62: Springer.

58. M. Ruiz, I. Ramos, and M. Toro, "A simplified model of software project dynamics," *Journal of Systems and Software,* vol. 59, no. 3, pp. 299-309, 2001.

59. T. Dingsøyr and T. Dybå, "Team effectiveness in software development: Human and cooperative aspects in team effectiveness models and priorities for future studies," in *2012 5th International Workshop on Co-operative and Human Aspects of Software Engineering (CHASE)*, 2012, pp. 27-29: IEEE.

60. S. C. Misra, V. Kumar, and U. Kumar, "Identifying some important success factors in adopting agile software development practices," *Journal of Systems and Software,* vol. 82, no. 11, pp. 1869-1890, 2009.

61. J. M. Verner, M. A. Babar, N. Cerpa, T. Hall, and S. Beecham, "Factors that motivate software engineering teams: A four country empirical study," *Journal of Systems and Software,* vol. 92, pp. 115-127, 2014.

62. T. Dingsøyr and Y. Lindsjørn, "Team performance in agile development teams: Findings from 18 focus groups," in *International Conference on Agile Software Development*, 2013, pp. 46-60: Springer.

63. A. Teh, E. Baniassad, D. Van Rooy, and C. Boughton, "Social psychology and software teams: Establishing task-effective group norms," *IEEE software*, vol. 29, no. 4, pp. 53-58, 2011.

64. B. Tessem and F. Maurer, "Job satisfaction and motivation in a large agile team," in *International Conference on Extreme Programming and Agile Processes in Software Engineering*, 2007, pp. 54-61: Springer.

65. N. B. Moe, A. Aurum, and T. Dybå, "Challenges of shared decision-making: A multiple case study of agile software development," *Information and Software Technology*, vol. 54, no. 8, pp. 853-865, 2012.

66. T. Dybå and T. Dingsøyr, "Empirical studies of agile software development: A systematic review," *Information and software technology*, vol. 50, no. 9-10, pp. 833-859, 2008.

67. E. Salas, D. E. Sims, and C. S. Burke, "Is there a “big five” in teamwork?," *Small group research*, vol. 36, no. 5, pp. 555-599, 2005.

68. N. B. Moe, T. Dingsøyr, and T. Dybå, "A teamwork model for understanding an agile team: A case study of a Scrum project," *Information and Software Technology*, vol. 52, no. 5, pp. 480-491, 2010.

69. E. S. Cardozo, J. B. F. A. Neto, A. Barza, A. C. C. França, and F. Q. da Silva, "SCRUM and Productivity in Software Projects: A Systematic Literature Review," in *EASE*, 2010.

70. H. Tohidi, "Teamwork productivity & effectiveness in an organization base on rewards, leadership, training, goals, wage, size, motivation, measurement and information technology," *Procedia Computer Science*, vol. 3, pp. 1137-1146, 2011.

71. P. Wernick and T. Hall, "The impact of using pair programming on system evolution a simulation-based study," in *20th IEEE International Conference on Software Maintenance, 2004. Proceedings.*, 2004, pp. 422-426: IEEE.

72. M. Wu and H. Yan, "Simulation in Software Engineering with System Dynamics: A Case Study," *JSW*, vol. 4, no. 10, pp. 1127-1135, 2009.

73. T. Abdel-Hamid and S. E. Madnick, *Software project dynamics: an integrated approach*. Prentice-Hall, Inc., 1991.

74. T. K. Abdel-Hamid and S. Madnick, "Software productivity: potential, actual, and perceived," *System Dynamics Review,* vol. 5, no. 2, pp. 93-113, 1989.

75. R. J. Madachy and B. Khoshnevis, "A software project dynamics model for process cost, schedule and risk assessment," University of Southern California, 1994.

76. J. M. Lyneis and D. N. Ford, "System dynamics applied to project management: a survey, assessment, and directions for future research," *System Dynamics Review: The Journal of the System Dynamics Society,* vol. 23, no. 2-3, pp. 157-189, 2007.

77. T. K. Abdel-Hamid and F. H. Leidy, "An expert simulator for allocating the quality assurance effort in software development," *Simulation,* vol. 56, no. 4, pp. 233-240, 1991.

78. T. K. Abdel-Hamid, "Adapting, correcting, and perfecting software estimates: a maintenance metaphor," *Computer,* vol. 26, no. 3, pp. 20-29, 1993.

79. K. Sengupta and T. K. Abdel-Hamid, "Alternative conceptions of feedback in dynamic decision environments: an experimental investigation," *Management Science,* vol. 39, no. 4, pp. 411-428, 1993.

80. A. Rodrigues and J. Bowers, "The role of system dynamics in project management," *International Journal of Project Management,* vol. 14, no. 4, pp. 213-220, 1996.

81. L. Cao, B. Ramesh, and T. Abdel-Hamid, "Modeling dynamics in agile software development," *ACM Transactions on Management Information Systems (TMIS),* vol. 1, no. 1, p. 5, 2010.

82. K. E. Van Oorschot, K. Sengupta, and L. van Wassenhove, "Dynamics of agile software development," in *Proc. Int'l Conf. of the System Dynamics Society 2009*, 2009.

83. H. A. Akkermans and K. E. Van Oorschot, "Relevance assumed: a case study of balanced scorecard development using system dynamics," in *System Dynamics*: Springer, 2018, pp. 107-132.

84. J. B. Vancouver, K. B. Tamanini, and R. J. Yoder, "Using dynamic computational models to reconnect theory and research: Socialization by the proactive newcomer as example," *Journal of Management,* vol. 36, no. 3, pp. 764-793, 2010.

85. A. Gregoriades, "Human error assessment in complex socio-technical systems-system dynamic versus Bayesian belief network," in *System Dynamics Conference, Manchester*, 2008.

86. J. Block and S. Pickl, "The mystery of job performance: a system dynamics model of human behavior," in *Proceedings of the 32nd international conference of the System Dynamics Society, Delft, Netherlands, July*, 2014, pp. 20-24.

87. S. R. Terrell, "Mixed-methods research methodologies," *The qualitative report,* vol. 17, no. 1, pp. 254-280, 2012.

88. A. Pinsonneault and K. Kraemer, "Survey research methodology in management information systems: an assessment," *Journal of management information systems,* vol. 10, no. 2, pp. 75-105, 1993.

89. B. Kitchenham and S. L. Pfleeger, "Principles of survey research: part 5: populations and samples," *ACM SIGSOFT Software Engineering Notes*, vol. 27, no. 5, pp. 17-20, 2002.

90. T. Chow and D.-B. Cao, "A survey study of critical success factors in agile software projects," *Journal of systems and software,* vol. 81, no. 6, pp. 961-971, 2008.

91. M. J. Mawdesley and S. Al-Jibouri, "Modelling construction project productivity using systems dynamics approach," *International Journal of Productivity and Performance Management,* 2010.

92. I. Fatema, “Agile teamwork productivity influence factors,”Jan. 2017. [Online] Available from https://goo.gl/forms/I5xGdQGqFMk9he5f2 [retrieved: May, 2019].

93. A. S. Ami, A. Imran, A. U. Gias, and K. Sakib, "Effects of Internship on Fresh Graduates: A case study on IIT, DU students" doi: 10.13140/RG.2.2.13745.68967. [Online]. Available from https://www.researchgate.net/publication/323393923_Effects_of_Internship_on_Fresh_Graduates_A_case_study_on_IIT DU_students. [retrieved: May, 2018].

94. V. Sridhar, D. Nath, R. Paul, and K. Kapur, "Analyzing factors that affect performance of global virtual teams," in *Second International Conference on Management of Globally Distributed Work*, 2007, pp. 159-169.

95. I. Fatema and K. Sakib, "Factors influencing productivity of agile software development teamwork: A qualitative system dynamics approach," in *2017 24th Asia-Pacific Software Engineering Conference (APSEC)*, 2017, pp. 737-742: IEEE.

96. I. Fatema and K. Sakib, "Analyse Agile Software Development Teamwork Productivity using Qualitative
System Dynamics Approach," The Twelfth International Conference on Software Engineering Advances (ICSEA 2017), Oct. 2017, pp. 60-66, ISBN: 978-1-61208-590-6

97. B. Ramesh, L. Cao, J. Kim, K. Mohan, and T. L. James, "Conflicts and complements between eastern cultures and agile methods: an empirical investigation," *European Journal of Information Systems*, vol. 26, no. 2, pp. 206-235, 2017.

98. System Dynamics Tools. [Online]. Avilable from https://www.systemdynamics.org/tools [retrieved: May, 2018].

99. Ventana Systems Inc, [Online]. Avilable from http://vensim.com/ [retrieved: May, 2018].L. L. Rodrigues, N. Dharmaraj, and B. Shrinivasa Rao, "System dynamics approach for change management in new product development," *Management Research News*, vol. 29, no. 8, pp. 512-523, 2006.

100. E. Lizeo, "A dynamic model of group learning and effectiveness," in *System Dynamics Society Conference*, 2005.

101. C. Melo, D. S. Cruzes, F. Kon, and R. Conradi, "Agile team perceptions of productivity factors," in *2011 Agile Conference*, 2011, pp. 57-66: IEEE.

102. M. Hammer and J. Champy, "A manifesto for business revolution," *Reengineering the Corporation*, 1993.

103. P. M. Senge and J. W. Forrester, "Tests for building confidence in system dynamics models," *System dynamics, TIMS studies in management sciences*, vol. 14, pp. 209-228, 1980.

104. Y. Barlas, "Formal aspects of model validity and validation in system dynamics," *System Dynamics Review: The Journal of the System Dynamics Society*, vol. 12, no. 3, pp. 183-210, 1996.