

ANALYSIS AND COMPARISON OF WATERFALL MODEL AND AGILE APPROACH
IN SOFTWARE PROJECTS



ELÇİN NUR DEMİRKOL ÖZTÜRK

JANUARY, 2022

ANALYSIS AND COMPARISON OF WATERFALL MODEL AND AGILE APPROACH
IN SOFTWARE PROJECTS

BY

ELÇİN NUR DEMİRKOL ÖZTÜRK

SUPERVISOR

ASSOC. PROF. DR. AŞKIN DEMİRAĞ

DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTERS IN ART IN MANAGEMENT INFORMATION
SYSTEMS

YEDİTEPE UNIVERSITY

JANUARY, 2022

PLAGIARISM

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Date: 06.01.2022

Name/Surname: Elçin Nur Demirkol Öztürk

ABSTRACT

Looking at the history of civilization from past to present, the concepts of project and project management are frequently encountered. Thanks to project management, businesses can work target-oriented, provide high motivation, facilitate internal control and provide a significant increase in quality.

The concept of software, born with the development of technology, penetrates our lives more and more day by day. The globalizing world has brought together the concepts of software and project management and integrated them. Software projects require special management techniques because of their content. For this reason, new methods have emerged over time in the management of software projects.

In this study, the concept of the project, the development of project management from its birth over the years, project management in software projects, software development life cycles, and the Waterfall Model and Agile Approach, which are two methods used in software projects, are discussed. The perception of the Waterfall Model and Agile Approach was evaluated with the survey study created with the participation of 145 employees from the software industry. Thus, it is aimed to support managers to choose an effective and efficient method at the point of deciding which method to proceed in software project management.

Key Words: Project, Project Management, Software Projects, Software Development Life Cycles, the Waterfall Model, Agile Approach

ÖZET

Geçmişten günümüze medeniyet tarihine bakıldığında proje ve proje yönetimi kavramlarına sıklıkla rastlanmaktadır. Proje yönetimi sayesinde işletmeler hedef odaklı çalışabilir, yüksek motivasyon sağlayabilir, iç kontrolü kolaylaştırabilir ve önemli ölçüde kalite artışı sağlayabilir.

Teknolojinin gelişmesiyle doğan yazılım kavramı her geçen gün hayatımıza daha çok giriyor. Küreselleşen dünya, yazılım ve proje yönetimi kavramlarını bir araya getirmiş ve bütünleştirmiştir. Yazılım projeleri, içerikleri nedeniyle özel yönetim teknikleri gerektirir. Bu nedenle yazılım projelerinin yönetiminde zaman içinde yeni yöntemler ortaya çıkmıştır.

Bu çalışmada proje kavramı, proje yönetiminin doğuşundan yıllar içinde gelişimi, yazılım projelerinde proje yönetimi, yazılım yaşam döngüleri ve yazılım projelerinde kullanılan iki yöntem olan Şelale Modeli ve Çevik Yaklaşım ele alınmaktadır. Yazılım sektöründen 145 çalışanın katılımıyla oluşturulan anket çalışması ile Şelale Modeli ve Çevik Yaklaşım algısı değerlendirilmiştir. Böylece yöneticilerin yazılım proje yönetiminde hangi yöntemi izleyeceklerine karar verme noktasında etkin ve verimli bir yöntem seçmelerine destek olunması amaçlanmaktadır.

Anahtar Kelimeler: Proje, Proje Yönetimi, Yazılım Projeleri, Yazıl Geliştirme Yaşam Döngüsü, Şelale Model, Çevik Yaklaşım

ACKNOWLEDGMENTS

I would like to thank my dear advisor Assoc. Prof. Dr. Aşkın Demirağ for his great support and guidance in completing this study. Getting your sincere support every time I lost my motivation kept me going.

Endless thanks to my dear friend, Research Assistant Ceyda Ünal, who supported me at all times throughout my thesis process and encouraged me to continue, to my brothers and sisters who supported me at home, to my family, and to my dear husband Kadir, who faced all the emotional ups and downs of this thesis process with me.

TABLE OF CONTENTS

PLAGIARISM	i
ABSTRACT	ii
ÖZET	iii
ACKNOWLEDGMENTS	iv
TABLE OF CONTENTS	v
LIST OF TABLES.....	viii
LIST OF FIGURES.....	x
1. INTRODUCTION	1
2. PROJECT MANAGEMENT	3
2.1. Project Concept.....	3
2.2. Project Management Concept.....	5
2.3. Brief History of Project Management.....	7
2.4. Benefits of Project Management.....	14
2.5. Disadvantages of Project Management.....	15
2.6. Project Management Processes.....	15
3. SOFTWARE DEVELOPMENT PROJECT MANAGEMENT	22
3.1. Software And Software Development	22
3.2. Software Development Project Management	23
3.2.1. Challenging factors of software project management	24
3.2.2. Software project management processes	25
3.2.2.1. Project initiation and scope definition.....	26

3.2.2.2. Planning software project	27
3.2.2.3. Conducting software project	29
3.2.2.4. Reviewing and evaluating software project	31
3.2.2.5. Closing software project	32
4. SOFTWARE DEVELOPMENT METHODOLOGIES	32
4.1. Software Development Life Cycle	33
4.2. Waterfall Model	34
4.2.1. Waterfall model steps	35
4.2.2. Roles and responsibilities in the waterfall model	37
4.2.3. Waterfall model principles and features	38
4.2.4. Benefits of waterfall model	40
4.2.5. Constraints of waterfall model	41
4.3. Agile Approach	42
4.3.1. Brief history and principles of agile project management	43
4.3.2. Features of agile approach	46
4.3.3. Benefits of agile approach	48
4.3.4. Constraints of agile approach	49
4.3.5. Types of agile approach	51
4.3.5.1. Lean software development	51
4.3.5.2. Kanban	51
4.3.5.3. Extreme programming	53
4.3.5.4. Dynamic software development	55

4.3.5.5. Crystal.....	58
4.3.5.6. Adaptive software development	59
4.3.5.7. Feature driven development	60
4.3.5.8. Scrum	61
4.3.5.8.1. Scrum team and roles	63
4.3.5.8.2. Scrum activities	66
5. RESEARCH.....	70
5.1. Personal Characteristics of the Participants in the Research	70
5.2. Analysis Results Of The Waterfall Model And The Agile Approach In Software Projects	74
6. CONCLUSION.....	101
REFERENCES	105

LIST OF TABLES

Table 1.....	4
Table 2.....	16
Table 3.....	20
Table 4.....	54
Table 5.....	58
Table 6.....	70
Table 7.....	71
Table 8.....	72
Table 9.....	72
Table 10.....	73
Table 11.....	74
Table 12.....	75
Table 13.....	75
Table 14.....	76
Table 15.....	77
Table 16.....	78
Table 17.....	79
Table 18.....	81
Table 19.....	82
Table 20.....	84
Table 21.....	85
Table 22.....	86
Table 23.....	88
Table 24.....	89

Table 25.....	90
Table 26.....	90
Table 27.....	91
Table 28.....	91
Table 29.....	92
Table 30.....	92
Table 31.....	93
Table 32.....	94
Table 33.....	95
Table 34.....	95
Table 35.....	96
Table 36.....	97
Table 37.....	98
Table 38.....	98
Table 39.....	99
Table 40.....	100

LIST OF FIGURES

Figure 1	6
Figure 2	8
Figure 3	9
Figure 4	11
Figure 5	12
Figure 6	13
Figure 7	14
Figure 8	35
Figure 9	47
Figure 10	52
Figure 11	57
Figure 12	66
Figure 13	67
Figure 14	70

1. INTRODUCTION

Today, many businesses or individuals are closely interested in project management. Until the 1980s, project management focused mainly on providing information to senior managers about the resources used. However, especially after the nineties, with the business environment becoming much more complex, project management requires much more in today's conditions. Thus, the role of project management within companies is increasing in the face of increasing needs.

The great progress that information technologies have shown in recent times has led to a remarkable increase in productivity and efficiency, making information technologies an indispensable part of the business world. Therefore, the interest in software projects has increased day by day, and software projects have begun to be accepted and implemented by the wider masses.

The project management system and project management standards developed by the Project Management Institute (PMI), which is the most competent organization in the world in project management, are based on to provide an overview of software project management and guide the way to be followed.

The production of software projects requires a series of engineering and project management processes. The characteristics of the software to be produced, such as the size, complexity, purpose of use, the structure of the organization and project team that will produce the software, and the resources that can be used are important in the methodology selection of these processes. This study aims to examine the waterfall model, which is the pioneer of traditional plan-based methods and all kinds of project management methodologies in software project management, and agile methods that emerged as an alternative to situations where traditional plan-based methods are

insufficient and have gained a very important place today, and are suitable for project-specific situations. The aim is to present a comparison that will guide the choice of methodology.

In the second part of the study, the concept of project management, in the third part, software project management concepts and management processes are discussed together with their knowledge areas. In the fourth chapter, the software life cycle, a brief history of the waterfall model and agile methods, roles and responsibilities, principles and features, benefits, and constraints are explained respectively. In the fifth chapter, the literature research conducted in the other chapters was supported by a questionnaire answered by 145 people who took part in agile projects from the software industry.

2. PROJECT MANAGEMENT

2.1. Project Concept

According to Project Management Institute (PMBOK, 2017), "a project is a temporary endeavor undertaken to create a unique product, service, or result." The term "temporary" in this definition means that the project has a certain beginning and an inevitable end. In contrast, the term "unique" means that the product or service must be different from other products or services. (PMBOK, 2000, p. 4)

Each project creates a unique product, service, or result, the outcome of which can be tangible or abstract. No task repeats itself in the same form and conditions. Although repetitive elements may be found in some project outputs and activities, this does not change the fundamental, unique features of the project work again. According to Horine(2009), projects exemplifies ;

- Constructing a new house
- Developing a new software application
- Evaluation of existing production processes
- Creating a new radio advertising

Unlike projects, operations and operational work belonging to the organization are the ongoing, repetitive activities that keep the organization alive. As an example of ongoing transactions;

- Realization of accounts receivable and accounts payable activities
- To carry out the daily production processes.
- Executing customer order processes. (Horine, 2009, p. 8)

The comparison of project and operation terms is shown in Table 1 below. (Horine, 2009, p. 9)

Table 1

Comparison of Projects and Operations

FEATURE	PROJECTS	OPERATIONS
Key Similarities	Planned, executed, and controlled. Performed by people. Resource-constrained.	Planned, executed, and controlled. Performed by people. Resource-constrained.
Purpose	Attain objectives and terminate.	Sustain the organization.
Time	Temporary. Definite beginning and endpoints.	Ongoing.
Outcome	Unique product, service, or result.	Non-unique product, service, or result.
People	Dynamic, temporary teams formed to meet Project needs. Generally not aligned with organizational structure.	Functional teams are generally aligned with organizational structure.
Authority of Manager	Varies by organizational structure. Generally minimal, if any, direct line authority.	Generally formal, direct line authority.

Projects are not ongoing but a collection of activities completed within a certain period. However, the product or service obtained at the end of the project is not temporarily applied; the products or services produced in the projects are continuous. Temporality in projects indicates that a project has a specific beginning and end. The intention of the temporary does not mean that a project is short-term.

Every project has a specific, well-defined goal to be achieved. In projects, the aim is a focal point, the reason for the existence of the project. Without a plan, there is

no project. Every work is done under certain constraints and using specific resources. Every job is done for one purpose.

All projects need to deal with uncertainties. Because it is planned before implementation and therefore carries a risk element. Defining projects with high tension and setting realistic goals will probably be difficult. The planning process is challenging if all phases of a project are subject to change sequentially. Resources can be given, times can be agreed upon, but all plans made before the bike will need to be redrawn if some activities are delayed. When it is high, all project planning should be flexible enough to deal with change results. (Slack et al., 2007, p. 497)

The conclusion to be drawn from all these definitions is that the project responds to a need, is for a purpose, is made only once, has a start and end date is realized within an organization, and has a flexible structure.

2.2. Project Management Concept

Project management is defined by Kerzner (2017) as a whole of planning, management, organizing, and oversight activities for the usage of resources, reducing costs and risks, maximizing the value while a project is executed for a business goal.

According to Butler, project management has been strengthened with complex initiatives. It is designed to provide concentrated and integrated management and includes the following;

- Focusing on a significant portion of total organizational resources on a particular subject
- Specialized activities highly interconnected

- Relative power constraints on cost and end-of-product performance (Butler, 1973,p.85)

The project constraints figure is below (Kerzner, 2017, p. 5). The purpose of the figure is to show that project management is designed to manage or control company resources in a particular activity over time, within cost, and within performance/technology. Time, cost, and performance are considered constraints in the project. It must be recognized that it is possible to manage a project within time, cost and performance. These are considered primary constraints and are generally considered as criteria for a project by which success is a measure. (Kerzner, 2017, p. 6)

Figure 1

Project Constraints



It is tried to determine how long it will take to complete the project within time constraints. The time required for the completion of the product or service is planned. Failure to comply with the initially agreed timetable may result in compensation to the client or sponsor. This will cause an increase in costs.

In the cost constraint, it is tried to calculate the cost of completing the project. Thus, the cost of a working system to be obtained with the available resources can be determined. Cost intent includes the wages of the people working on the project, equipment purchases, software, hardware, rental, computer networks, management, stationery expenses necessary for the execution of the project. When it comes to cost

management of the project, it should not be understood that only viewing and recording the expenses incurred. Project cost management is the whole of the well-thought-out works, well designed, and aiming to finish the project on the agreed date, considering the time-cost relationship.

When the performance/technology constraint is evaluated as scope, the answer to "Which goal is aimed to be achieved" is sought. It is decided which product or service the sponsor or customer wants to achieve when the project is completed. The constraints of project management are interrelated, and changes made in one criterion affect the others.

Project management is software from the information, technology, and tools used to meet the project requirements for the project. Project management brings together the appropriate version and application and the project management processes necessary for the project. Thanks to project management, it is ensured that organizations carry out projects effectively and efficiently. (Kerzner, 2017, p. 9)

2.3. Brief History of Project Management

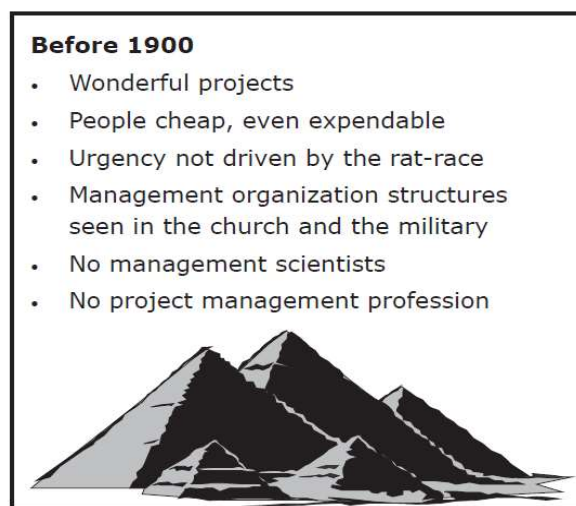
The history of modern project management started with the emergence of the concept of "project management." Considering the developments experienced in project management will be examined under five main titles.

- 1900 and Before

It is challenging to obtain the architectural and industrial cultural heritage left by the ancient projects quickly and cheaply without using today's technology. Except for a few employers, there were no measures necessary for the welfare and safety of

workers. Workers seemed to be a cheap resource. As an outcome of the industrial revolution, the working class was just emerging, so the formal government organizational structures consisted of military and civilian administration, excluding workers. This has resulted in many workers being injured, sick, and dead from physical exhaustion. (Lock, 2007, p. 2). In Figure 2, a brief summary of project management in the period before 1900 is seen. (Lock, 2007, p. 2)

Figure 2



Before 1900 Project Management

It is challenging for us to obtain the architectural and industrial cultural heritage left by the ancient projects quickly and cheaply without using today's technology. Except for a few employers, there were no measures necessary for the welfare and safety of workers. Workers seemed to be a cheap resource. As an outcome of the industrial revolution, the working class was just emerging, so the formal government organizational structures consisted of military and civilian administration, excluding workers. This has resulted in many workers being injured, sick, and dead from physical exhaustion. (Lock, 2007, p. 2)

- 1900 - 1949 Period

Methodologists and industrial engineers emerged to study people and their productivity in factories to meet the need for ammunition during rapid industrialization and World War I. Henry Gantt. They worked for Taylor and were essential to the Model T car and project managers. Henry Ford made production line manufacture famous by developing the charts that are still popular and used universally today. Without computers, they were not flexible about changing, and they were boring to translate into study programs and therefore difficult to implement and use. For this reason, Gantt bar charts are often selected and are often based on custom charts that allow reprogramming using removable magnetic or plug-in inserts or cards. From job assignments to staff and machines, every detail to vacation planning is controlled by programs displayed in the office. (Lock, 2007, p. 3). In Figure 3, a brief summary of the project management for the period 1900 - 1949 can be seen. (Lock, 2007, p. 3)

Figure 3

1900 - 1949 Project Management

1900 - 1949

- Emergence of management science
- People begin to study work and people at work
- Henry Gantt introduces his famous planning charts
- Early development of critical path networks

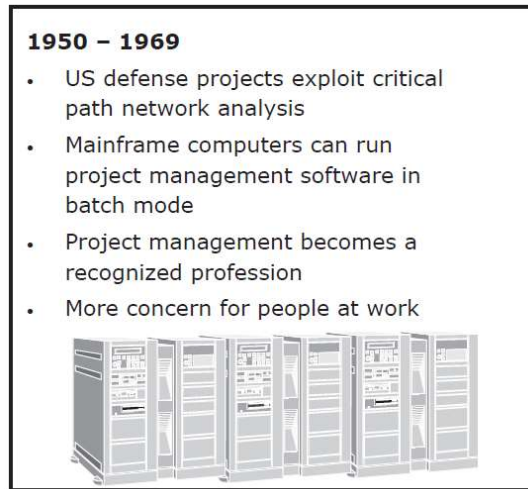


- 1950 - 1969 Period

With the development of technology, introducing digital computers into our lives has enabled us to fit the processing and updating of serious road networks more shortly and practically. One of the organizations that benefited from this robust scheduling and planning tool that emerged with the developing technology in the 1950s was the American defense industry and Du Pont. In addition, the manufacturing and construction industries were quickly getting to know these new techniques.

Computers at that time were both costly and extensive, requiring large, air-conditioned, clean rooms. Computers were costly, and the budget required to operate was huge. It was far beyond what the most significant organizations could afford. For this reason, many planners in small organizations have benefited from the offices where these computers are operated collectively. These offices were set up by both computer manufacturers and large companies with available computers.

Although project management is not seen as a respected profession, it is included in the profession definition. Although discrimination continued due to race, gender, and age, organizations cared about the well-being of people in the workplace. (Lock, 2007, p.3). In Figure 4, a brief summary of the project management for the period 1950 - 1969 can be seen. (Lock, 2007, p.3)

Figure 4*1950 - 1969 Project Management**- 1970 - 1989 Period*

During this period, there were significant developments in information technology. Industrial project management continued as before, but the spread of more project management software and IT created a different type of project manager. They were IT project managers with the technical and mental skills needed to lead teams developing IT projects, rather than people with no project planning or programming knowledge and not interested or willing to learn these methods. These IT project managers were usually senior systems analysts. The high demand for their services allows them to change jobs frequently and rapidly raise higher salary scales.

Project management software became widespread, but operations continued to be carried out collectively on the large, expensive, most basic computers. Graphics were primitive and scanty compared to modern equipment. Sometimes data entry was done by requiring two cards for each network activity because the data on cards in the network diagrams were punctured. These punch cards had to be brought to trained

computer operators working in clean and air-conditioned rooms where project managers were generally prohibited. The initial process results seemed to form an enormous stack of output listing several errors, such as several new card punctures and fixes. Still, this process was exciting and fun. (Lock, 2007, p. 4). In Figure 5, a brief summary of the project management for the period 1970 - 1979 can be seen. (Lock, 2007, p. 4).

Figure 5

1970 - 1979 Project Management



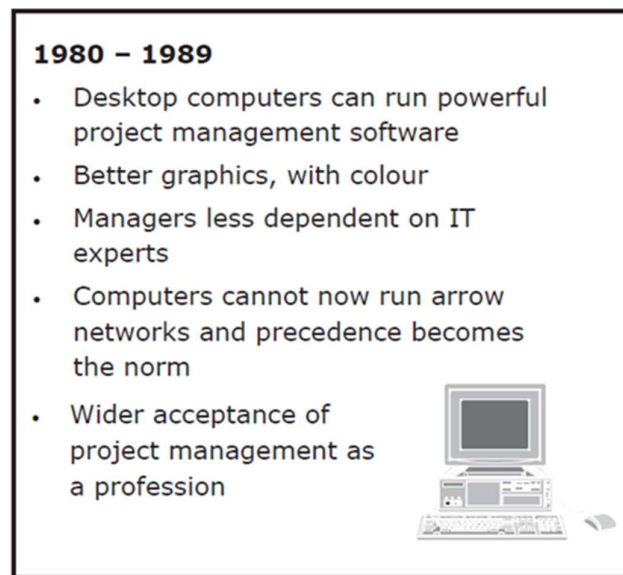
- 1980 - 1989 Period

Over this decade, the dependence of project managers on IT professionals has declined. Desktop computers and small printers for graphics were now available that could run most project management software. However, productivity did not go hand in hand with this growth in technology because managers could not move from engaging in technology to what they aimed to manage. People often gathered around each other's desks, asked each other about their experiences, and learned to depend

much less on IT professionals. Processing times were so shortened that the IT expert could quickly prepare and run programs almost instantly from their keyboard. (Lock, 2007, p. 4). In Figure 6, a brief summary of the project management for the period 1980 - 1989 can be seen. (Lock, 2007, p. 4).

Figure 6

1980 - 1989 Project Management



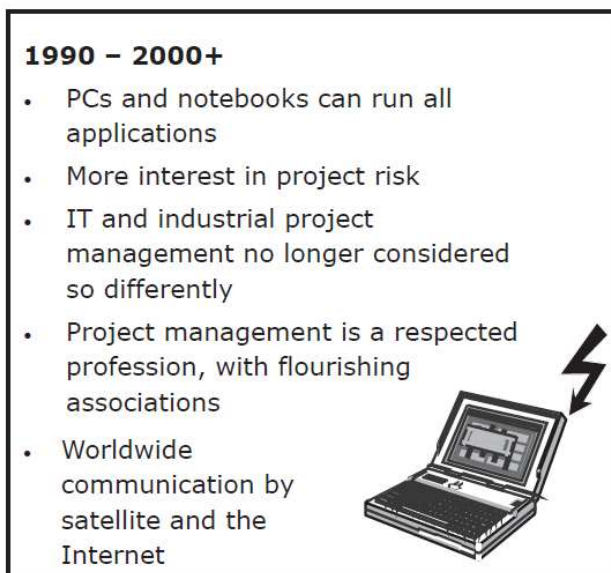
- 1990 - 2000+

In this period, technology continued to be a driving force for change and greatly influenced the actions of project managers. In 1996, PRINCE was upgraded to PRINCE2; shortly after 1997, an alternative method called Critical Chain Project Management (CCPM) was introduced. CCPM is the “planning and managing projects” method developed by Eliyahu M. Goldratt. Derived from TOC, unlike CPM and PERT, the method emphasized essential resources rather than the essential features to complete the project.

In this period, project management is no longer seen as two separate branches: industry and IT. For projects to bring faster and better results, companies prefer a broader and more satisfying approach. In 2001, the Agile Manifesto was written. Agile Manifesto is built on core values aimed at software development teams to perform well as a team. (Lock, 2007, p. 4). In Figure 7, a brief summary of the project management for the period 1990 - now can be seen. (Lock, 2007, p. 4).

Figure 7

1990 - Now Project Management



2.4. Benefits of Project Management

According to PMBOK (2017), project management enables organizations to carry out projects effectively and efficiently. In addition, effective project management helps individuals, groups, and public and private organizations to:

- Achieve business goals;

- Meeting stakeholder expectancy;
- Be more presumable;
- Increase chances of success;
- Deliver the right products at the right time;
- Solve problems and issues;
- Immediately respond to risks;
- Optimize the use of corporate resources;
- Identify, recover, or terminate failing projects;
- Manage constraints scope, quality, schedule, costs, resources;
- Balance the impact of restrictions on the project;
- Manage change better.

2.5. Disadvantages of Project Management

Besides many advantages of project management, there are also some disadvantages:

- Complex organizations
- Establishment tendency to deteriorate
- Management strength
- Difficulty using staff (Butler, 1973, p. 88)

2.6. Project Management Processes

According to PMBOK (2017), “A Project Management Process Group is a logical grouping of project management processes to achieve specific project objectives. Process Groups are independent of project phases. ”

Projects usually go through the same stages, from inception to completion. In the first phase of the life cycle that begins with implementing the project, a manager is appointed to the project and the key players to participate in the project, in other words, the team members. Then, the main objectives of the project to be outlined in the project mission are defined. At this stage, the duties of the team leader, managers, and other staff in the team are defined. Project management processes are divided into five project management process groups: initiating, planning, executing, controlling, and closing. (Cleland et al., 2006, p. 1-17).

The description of project management process groups is shown in Table 2 below. (Horine, 2009, p. 10)

Table 2

Description of the Project Management Process Groups

Process Group	Description	Common Terms
Initiating	Authorizing the project or phase.	“preliminary planning” “kicking off”
Planning	Defining and refining objectives of the “defining” project and selecting the best course of “developing the plan” action to attain those objectives.	“defining” “developing the plan.” “setting the stage.”
Executing	Coordinating the people and resources to “make it happen” implement the plan.	“making it happen.” “getting it done.” “coordinating”
Controlling	Ensuring project objectives are met by “tracking progress,” monitoring and measuring progress regularly “keeping on course” to identify variances from	“tracking progress” “keeping on course.”

	the plan so that corrective actions can be taken.	
Closing	Formalizing acceptance of project or phase “client acceptance” and bringing to an orderly end.	“client acceptance,” “transition” “closeout

Initiating Process Group: By obtaining the project or phase start authorization, the processes performed to define a new project, or a new phase of an existing project is initiated. In the preparation phase, the project is determined, evaluated for the purposes, and the initial approval is given. It includes initial formal studies that determine the working and environmental conditions that will apply to the project life cycle, determine the basic rules and checklists to create the official starting document of the project, called the project specification. The first registrations of project stakeholders that may affect the project or be affected by the project results are completed in this process.(PMBOK, 2017, p. 53)

Planning Process Group: Planning processes are required to establish the scope of the project, refine the objectives, and define the course of action required to attain the objectives that the project was undertaken to achieve. Planning processes are required to establish the scope of the project, refine the objectives, and define the course of action required to attain the objectives that the project was undertaken to achieve. (PMBOK, 2017, p. 53)

Planning involves the establishment of clear and precise objectives (and the work activities that will have to take place to accomplish them) to reach a final, stated goal. For example, the goal may involve the solution of a problem or the achievement of some state or condition different from the present one). After the project sponsors or the client approves the planning, the implementation process starts. (Weiss et al., 1992, p. 4-5)

Executing Process Group: Organizing processes performed to complete the work defined in the project management plan to satisfy the project requirements. During the implementation phase, everyone involved in the project plays an active role. All planned stages of the project are carried out in this process. Besides organizing people, project management involves assembling the necessary resources, such as human resources, materials, and money, to perform the work described in the plan. It also ensures that the structure needed to implement the plan is established. (Weiss et al., 1992, p. 4-5)

Monitoring and Controlling Process Group: These processes are necessary to monitor, review and regulate the progress and performance of the project. Identifying areas that need to be modified in the plan and initiating relevant changes are matters in this process group.

When resources are combined into a cohesive structure, this structure will need to be monitored and maintained as the project progresses. The definition and creation of a reporting structure at specific points throughout the project lifecycle are achieved through control and monitoring. The resulting reports create historical records and contain early warnings of situations and events that fall outside the performance criteria. (Weiss et al., 1992, p. 4-5)

Closing Process Group: Those processes performed to formally complete or close the project, phase, or contract. Process flow diagrams are used throughout this guide. The project management processes are linked by specific inputs and outputs where the result or outcome of one process may become the input to another process that is not necessarily in the same Process Group. (PMBOK, 2017, p. 53). The end of the project phase is started when the product or service produced is ready for testing

in a real environment. The project is delivered with the project sponsors and client approval. (Horine, 2009, p. 11)

2.7. Project Management Knowledge Areas

Project Integration Management is the coordination of all elements of a project. This includes managing conflicts between different aspects of a project, making trade-offs between competing demands and evaluating resources, and coordinating tasks, resources, stakeholders, and other project elements. (PMBOK, 2017, p. 54).

Project Scope Management consists of project scope studies to plan, define, improve, and manage the scope. Project scope tasks; planning scope management is collecting needs, defining the scope, establishing the work breakdown structure, verifying the scope, and controlling the scope. (PMBOK, 2017, p. 54).

Project Schedule Management area is used to generate and retrieve information about the project time and schedule, determine the project timetable's status, and update the plan and re-prepare the foundation if any changes in the project cause time changes. (PMBOK, 2017, p. 54).

Project Cost Management is the field of knowledge that uses planning, managing, controlling, and recalculating costs and the budget, such as planning cost management, estimating costs, determining the budget, and controlling costs. (PMBOK, 2017, p. 54).

Project Quality Management is used to plan, manage and control the quality in the project, which is to plan the quality management, manage the quality and control the quality. (PMBOK, 2017, p. 54).

Project Resource Management is the knowledge that shows how the project, project team, and resources will deal with key production parameters. This knowledge area consists of resource management planning, estimating operational resources, obtaining resources, developing a team, managing the team, and controlling resources. (PMBOK, 2017, p .54).

Project Communication Management is the area of knowledge about planning communication management, managing communications and monitoring communication tasks, and how to organize communications from the beginning to the end of the project. (PMBOK, 2017, p. 54).

Project Risk Management is the knowledge area about how to handle risks. Project risk management identifies risks, plans risk management, performs qualitative and quantitative risk analysis, plans risk responses, implements risk responses, and monitors risks. (PMBOK, 2017, p. 54).

Project Procurement Management is the field of information used to manage project procurement, plan procurement management, execute procurements, and control supplies. (PMBOK, 2017, p. 54).

Project Stakeholder Management is the information space for managing the project's stakeholders by identifying stakeholders, planning stakeholder engagement, managing stakeholder engagement, and monitoring stakeholder engagement. (PMBOK, 2017, p. 54).

The knowledge areas matrix is depicted below in Table 3. (PMBOK, 2017, p. 55)

Table 3

Project Management Knowledge Areas

Knowledge Areas	Process Groups				
	Initializing	Planning	Executing	Monitor and Controlling	Closing
Project Integration Management	Develop Project Charter	Develop Project Management Plan	Direct and Manage Project Work Manage Project Knowledge	Monitor and Control Project Work Perform Integrated Change Control	Close Project or Phase
Project Scope Management		Plan Scope Management Collect Requirements Define Scope Create WBS		Validate Scope Control Scope	
Project Schedule Management		Plan Schedule Management Define Activities Sequence Activities Estimate Activity Durations Develop Schedule		Control Schedule	
Project Cost Management		Plan Cost Management Estimate Costs Determine Budget		Control Costs	
Project Quality Management		Plan Quality Management	Manage Quality	Control Quality	
Project Resource Management		Plan Resource Management Estimate Activity Resources	Acquire Resources Develop Team Manage Team	Control Resources	
Project Communication Management		Plan Communications Management	Manage Communication	Monitor Communication	
Project Risk Management		Plan Risk Management Identify Risks Perform Qualitative Risk Analysis Perform Quantitative Risk Analysis Plan Risk Responses	Implement Risk Responses	Monitor Risks	
Project Procurement Management		Plan Procurement Management	Conduct Procurements	Control Procurements	
Project Stakeholder Management	Identify Stakeholders	Plan Stakeholder Engagement	Manage Stakeholder Engagements	Monitor Stakeholder Engagements	

3. SOFTWARE DEVELOPMENT PROJECT MANAGEMENT

3.1. Software And Software Development

According to ISO definitions, the software is programs, procedures, and documents that depend on the operation of a data processing system. Therefore, the software is an abstract product. The software is based on the plurality of logical paths in the program modules and the combination of all combinations of interface details.

Creating software requires innovative problem solving to create unique solutions. Software is a cognitive and human-based development process that requires the sharing of documents. (Ruhe et al., 2014, p. 11)

On the other hand, software development is the process of discovering whether and how the chosen technology can fulfill the role required of it. Software development is not just a software creation process; it is also the process of learning how to create the most suitable software for its purpose. Software development is considered among the most complex activities carried out by a human. Software development is often described as “a learning process in which knowledge is acquired and knowledge generated during a project.” Definitions made to understand software and software development, in general, can be expressed as follows;

- Software is unique and intangible in that its most significant issue is its complexity.
- It is not easy to define the requirements for the software before starting development.
- Software development technologies change fast.
- Software development has much technology, and these technologies are more complex than a single individual can hope to gain expertise.

- Unlike other products, the software is not constructed but instead designed into existence.
- The software can be changed quickly, and it is necessary to implement the changes correctly.
- The software is not as perfect as it was initially designed; It will always require changes to be optimized for its purpose. (Stepanek, 2005, p. 22)

3.2. Software Development Project Management

Software projects are specialized projects, and while they have the basic features of project management, they require special management techniques due to their content. There are significant differences that distinguish software projects from other projects. While the final product's physical existence is created in other project management areas, the product that occurs in software projects is abstract due to the intellectual processes. Therefore, it is more difficult to calculate the workforce and cost to be consumed in software projects, and the possibility of failure is higher since uncertainties are higher than the general project management. Projects carried out in other project management areas do not differ much according to time and place. The experience is gained in project management for these projects, and these experiences are used in new projects. In software projects, the rapid development of information technologies and changes in customer requirements may cause changes even in the current project and almost invalidate the experience gained in previous projects. For these reasons, success rates in software projects are low. The criteria determining whether projects are successful are time-resource-scope items, also known as the project management triangle. (Sommerville, 2011, p. 94)

3.2.1. Challenging factors of software project management

- The software is an abstract and malleable product.
- The key features that complicate software projects are the project and its complex features, unreal scaling of resources, project and product value, initial uncertainty in the project and product scope, and information obtained as the project evolves.
- The emergence of the software project can change as the project gains knowledge of the software requirements, the scope of the project, and as it emerges.
- The intellectual capital of software personnel may differ.
- Communication and coordination with software teams and project stakeholders often lack clarity.
- Since software projects require innovation, they are abstracted, and stakeholders may not express or agree on them communicating with the software product because it involves risk and uncertainty.
- Primary planning and forecasting for software projects are complex because of imprecision or link to incomplete or impractical needs of these activities. Preparing accurate estimates is also tricky because the efficiency and effectiveness of software developers vary greatly.
- Product complexity makes it challenging to develop and modify software for combining multiple logical paths within program modules with combinations of data and interfaces between program modules that use the paths.
- Extensive software testing is impractical because of the time required to test all logical paths and interfaces under all combinations of input data and other input stimuli.

- It may cause integration and performance problems with other software products.
- Information security techniques are necessary, as most software is interconnected. As a result, software security is a vast and growing problem.
- Objective quantification and measurement of software quality are not accessible due to the abstract nature of the software.
- Software developers can use constantly evolving and frequently updated processes, methods, and tools.
- A software product may need to run on a variety of hardware platforms and infrastructure software.
- Running software is not an independent product. Instead, it is executed in computing hardware and is usually part of various hardware, other software, and manual procedures.
- Platform technologies, infrastructure software, and vendor-supplied software are frequently changed or updated, which may require changes to the software under development. (PMI, 2013, p. 6)

3.2.2. Software project management processes

There are many moving parts in different scale projects. There is a lot to coordinate and monitor to get from point to point and run a successful project. For this reason, projects are broken down into smaller, more digestible pieces, also known as project stages. Project phases allow for taking and organizing large-scale projects so that they can make progress more efficiently. Software project management processes are processes affected by high-level processes and manage lower-level processes related to software development. Project management processes are processes that help the management to see what is happening in the project and at the same time allow them to

control the project. So these are management processes. These processes also include processes such as project risk management, effort, cost estimation. (Ahmed, 2012, p. 10)

3.2.2.1. Project initiation and scope definition

It covers the processes related to the initiation of the project and the definition of the scope, and the decision to start the software project. The focus of these processes is to obtain the requirements effectively using various methods and the analysis of the project feasibility study. The activities carried out are negotiating and determining the requirements, feasibility analysis, and reviewing and, if necessary, correcting the requirements.

✓ Discussing and determining the requirements:

It ensures that the requirements are discussed and decided, the tasks to be fulfilled, and the boundaries are drawn. Requirements are obtained from the customer with various techniques, analyzed, definitions are made, and finally verified. In this way, constraints and targets are determined, and the scope of the project is revealed.

✓ Feasibility analysis

The purpose of the feasibility analysis is to clearly define the project objectives and evaluate alternative approaches to decide whether the solution applied is the best solution within the framework of existing technology, resources, and socio-cultural factors.

✓ Review and renewal of requirements

Within the framework of developments that occur as the project progresses, the requirements should be reworked and renewed if necessary. If the changes are to be

accepted, traceability and risk analysis should be performed to determine the effects of the change. (SWEBOK, p. 7-5)

3.2.2.2. Planning software project

The first activity to be done in the project planning process should be the selection of an appropriate software development methodology and, if necessary, adaptation of the methodology within the framework of the project scope, software requirements, risk assessment, usage area of the software, functional and technical complexity level and software quality requirements. A risk assessment with the participation of all stakeholders should be made, and a risk profile should be drawn up to which all stakeholders agree. Software quality management processes should be determined as part of the planning process. The activities performed in the planning process are planning the process, determining the outputs to be delivered, effort-calendar-cost estimates, resource allocation, risk management, quality management, and plan management.

✓ Planning the process

Process planning differs according to the software development methodology applied. Since predictive methods such as the waterfall model are plan-based methods, they require intense planning at the beginning of the project. In these methods, software project activities are applied in a linear order. In adaptive methods such as agile methods, much time is not spent on planning, and activities are implemented in an iterative sequence.

✓ Deciding the outputs

The products to be obtained at the end of all project activities, such as software architectural design documents, audit reports, and software tested, should be

defined. It should be decided to reuse products produced in previous projects or to evaluate products that are ready on the shelf. If needed, which software will be procured, and if subcontractor use is required, the terms of use should be determined.

✓ Predictions

How much effort will be spent on a project or a particular part of the project can be estimated by comparing the experience of previous projects or by referring to expert knowledge, if available data? In predictive methods, the creation of the calendar, the start and end times, and the duration of the tasks are determined during the planning phase. In adaptive methods, calculations are made in the light of the information obtained for the beginning, and the estimated number of iterations to be applied and the labor capacity to be consumed are determined, but changes are possible throughout the project.

✓ Resource consecration

Equipment, personnel, and facilities should be allocated for specifically defined tasks, and responsibilities should be allocated for both the entire project and certain parts of it. Resource allocation is an important topic, especially for people's use according to their abilities and resource optimization.

✓ Risk management

Risk and uncertainty are related but different concepts. Uncertainty arises from a lack of information. Risk is the probability of an event occurring that will have a negative impact on the project. Risk management includes the activities of defining risk factors, analyzing the probability of occurrence and potential effects of each

risk factor, prioritizing risk factors, and developing risk reduction strategies to minimize probabilities and adverse effects.

✓ Quality management

Software quality requirements for a software project should be defined both qualitatively and quantitatively. Software quality assurance and quality improvement processes and the verification and validation processes of the products to be delivered should also be defined in the quality planning process. (SWEBOOK, p. 7-6)

✓ Plan Management

Plans should be managed in software projects where change is possible. The plans and processes chosen for software development should be constantly monitored, reviewed, reported, and renewed when necessary.

3.2.2.3. Conducting software project

During the execution phase of the software project, plans are implemented, and the processes created in the plans are implemented. In addition, the processes of the chosen software development methodology must be followed. Therefore, during the execution of the software project, the activities carried out are the implementation of plans, software procurement, supplier protocol management, implementation of measurement processes, monitoring process, control process, and reporting.

✓ Implementation of plans

Project activities should be carried out according to project plans, resource use and output production should be carried out in line with plans.

✓ Software procurement and supplier protocol management.

Software procurement and supplier protocol management include managing the protocols that the software development organization has made with the customers who buy the software products and their subcontractors who are the providers of software products and services. In addition, the contract specifies which products will be produced when delivery will be made and the sanctions to be applied in case of late delivery or no delivery. Therefore, the execution of the management of the project in line with the protocol conditions after the contract is put into practice should be followed carefully.

✓ Application of measurement processes

Throughout the execution of the project, relevant data to be quantified should be collected.

✓ Monitoring process

Throughout the project, commitment to the plans, outputs of sub-tasks, and completion criteria should be constantly monitored. Current effort expenditure, compliance with the schedule, and costs should be analyzed, and resource utilization should be examined. The project risk profile should be considered, and software quality requirements should be evaluated. Deviation analysis showing the difference between the current situation and the typical situation should be done.

(SWEBOK, p. 7-7)

✓ Control process

The outputs of the project monitoring processes are the basis for the decisions to be made. Changes can be made to the project when the possibilities and possible effects of risk factors require. These changes can be corrective actions or can be applied in the form of additional measures. The project plan and other project

documents can be revised to eliminate undesirable events and effects. In some cases, the control process may even result in the termination of the project.

✓ Reporting

Reports should be submitted to both internal and external stakeholders on the progress of the project up to that point at specified time intervals. These reports should address the information needs of the target audience rather than being detailed like internal team reports.

3.2.2.4. Reviewing and evaluating software project

At predetermined times and when needed, the level of achievement of project objectives and the status of stakeholder needs should be evaluated. The activities carried out in the review and evaluation process determine the level of meeting the needs and reviewing and evaluating the performance.

✓ Meeting the requirements

Since ensuring stakeholder satisfaction is one of the initial goals of the software project manager, the achievement of this goal should be evaluated regularly. This evaluation should be made on the products obtained at the critical stages of the project or the end of an iteration. If there are deviations in meeting the requirements, the measures mentioned in the control process in the previous section should be taken. The decisions taken should be documented and presented to the relevant stakeholders. (SWEBOK , p. 7-7)

✓ Review and evaluation of performance

Periodic performance evaluations of project members give an idea about possible problem areas such as their commitment to plans and processes and in-team conflict.

3.2.2.5. Closing software project

A project, a significant phase or an iteration, reaches the closing phase when all its plans and processes are completed. At this stage, success criteria should be evaluated. Archiving, retrospective review, and process development activities should be carried out while closing.

✓ Making the closing decision

Closing occurs when the specified tasks are completed, and the success criteria are met. For example, software requirements are approved as "met" or "not met" and the degree to which goals are met. Closing processes should be carried out with the participation of stakeholders, and the results should be documented with the approval of the stakeholders.

✓ Closing activities

After the closing decision is made, the method, place, and time approved by the stakeholders should be archived, and sensitive information, software, and devices should be destroyed. A retrospective analysis of the encountered problems, risks, and opportunities should be made. The lessons learned from the project should be drawn and transferred to the institutional learning and development departments. (SWEBOK, p. 7-8).

4. SOFTWARE DEVELOPMENT METHODOLOGIES

In order to do all software development activities, different development models have been developed over time. While each model can eliminate a deficiency encountered in the previous model, different methodologies can still be used according

to the size of the projects and the products to be produced, the structure of the customer and the business area, and a single model is not accepted as a standard.

4.1. Software Development Life Cycle

All stages that the software product goes through during both production and customer use are called the software development life cycle. The software development process consists of stages based on timing and divided into the content. Software development life cycle (SDLC) covers software processes that transform software requirements into software products. Software product lifecycle (SPLC) includes software configuration management and software quality assurance processes in addition to a software development lifecycle and “end-to-end” processes such as commissioning, maintenance, support, evaluation, and retirement of software. There is no temporal sequence between software development processes individually. Software lifecycle models (SDLC or SPLC) create the sequences between these processes. Life cycle models define critical processes according to their philosophies and determine the sequences and interactions between these processes. (SWEBOK, p. 8-4).

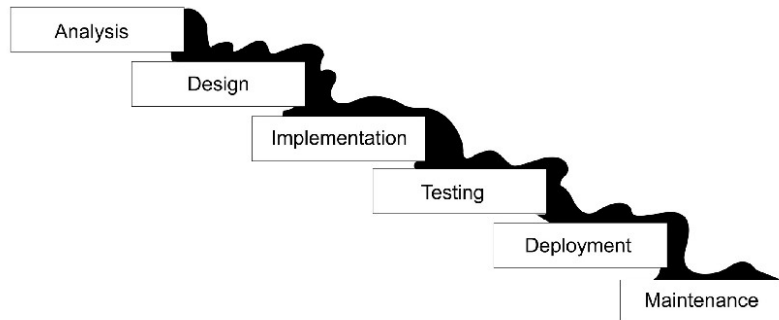
There are some basic steps in the software life cycle. It is possible to list these as Planning, Analysis, Design, Production, and Maintenance. Planning is the stage where the project plan is created, the feasibility study is carried out, and the requirements such as human resources and equipment are determined. In the analysis phase, existing systems are examined, and system and software requirements and, if any, fundamental problems in operation are determined in detail. Design is the stage where the basic structure of the software that will meet the requirements determined in the analysis phase is created. At this stage, all components included in the software are

defined and designed in detail. The production phase is the stage that includes processes such as developing, testing, and installing software components that have been designed. Maintenance is the activities of troubleshooting, making new additions, and making improvements after the software units are delivered.

4.2. Waterfall Model

The waterfall model is the oldest and basic model of software engineering in which the activities applied in software projects are performed in successive stages. This approach consists of completing the parts sequentially to form a whole. It is argued that one should not move on to the next one before completing a step. (Munassar et al.,2010, p. 95)

The customer provides all requirements at the beginning of the project. This method, which is suitable for projects with all requirements determined in detail, is unsuitable for projects that are likely to change requirements during project development. (Pressman et al., 2019, p. 25). In the waterfall approach, the entire software development process is divided into separate stages. In the waterfall model, typically, the result of one phase, in turn, provides input for the next phase. It consists of requirements analysis, design, coding and unit tests, integration tests, and maintenance - repair stages. (Sommerville, 2007, p. 94). In the waterfall picture below, each step would be completed before moving on to the next step: (Stanley et al., 2020, p. 21)

Figure 8*Waterfall Model Steps***4.2.1. Waterfall model steps**

Analysis of the requirements: All possible system requirements are captured at this stage, developed, and documented in a requirement specification document. In the first stage, the demands from the customer are analyzed. In order to understand the nature of the program to be created, the analyst must understand the knowledge area of the software, the required function, behavior, performance, and interface. Both system and software requirements are documented and reviewed with the customer. At the beginning of the software project, the customer needs are analyzed, documented, and approved. The applicability and feasibility of the customer's demands in the product are prioritized. Next, the time, cost, and similar factors required to implement the requirements are determined. At this stage, it is necessary to understand the requirements correctly and to produce a solution that will meet the need; returning and correcting the error will cause significant financial losses and delays in the project delivery time in the Waterfall model. (Pressman et al., 2019, p. 25)

Design: The first stage requirement features are examined at this stage, and system design is prepared. System Design helps identify the hardware and system and

helps define the overall system architecture. At this stage, rather than the technical details, the process of the software product that the customer needs is extracted. (Pressman et al., p. 26). Software design is essentially a multi-step process that focuses on different features of a program, such as data structure, software architecture, interface representations, and algorithmic details. The design process translates requirements into a software representation that can be evaluated for quality before coding begins. Like the requirements, the design is documented and becomes part of the software configuration. (Sommerville, 2007, p. 94)

Code generation: With the input from the system design, the system is integrated into the next stage. The design must be translated into a machine-readable form. This task is performed in the code generation step. If the design is done in detail, code generation can be done more efficiently. At this stage, the software development team will start the coding process. One of the essential points in this process is that the code writing process is based on the standard rules determined by the team. Thus, progress will be made in a controlled manner.

Performing unit and integration tests: After the code is created, the program test starts. The testing process focuses on the logical insides of the software and makes sure that all statements are tested. Tests are carried out in order to reveal errors and ensure that the defined input produces accurate results that match the required results. All units developed during the implementation phase are integrated into a system after the testing of each unit. After integration, the whole system is tested for any wrong or malfunction. After the development team completes its work, the test team gets involved. It tests the whole system with the developed demand. If a problem arises, it contacts the software development team. The testing continues until there is no problem

in the system. Then the process continues with customer testing, and this testing process ends with the customer's acceptance. (Özdemir, 2017)

Maintenance-support: After the software is delivered to the customer, there may be a request for development and change. The change will occur because errors were encountered. The software must be adapted to accommodate changes in the external environment, or the customer may want functional or performance improvements. Also, some much better versions could be released to improve the product. Maintenance is done to make these changes in the customer environment. Software support/maintenance reapplies each of the previous phases to an existing program rather than a new one. (Sommerville, 2007, p. 94)

4.2.2. Roles and responsibilities in the waterfall model

The waterfall model is generally accepted and applied by public institutions and large companies as the management standard of all kinds of large-scale projects. For this reason, the number of people in the team is mostly at least fifteen people. Each member has specific duties, and they should not take on the duties of other people in the team.

Project manager: The most fundamental responsible person in the team. The project manager should be contacted for the general course and planning of the project and possible problems.

Analyst: The person who analyzes the demands from the customer, directs them correctly, and designs for the desired job is the business analyst.

Software architect: Another essential role in the design process. Technically he is the person who makes the connections.

Software development officer: The person who realizes the design in the system. It is the person responsible for the coding required for the software.

Test supervisor: After the controls are made, he tests the whole system until the foundation is without any problems. It contacts the software development team to solve possible problems and follows the whole process until the problem is solved. (Sergeev, 2016)

4.2.3. Waterfall model principles and features

At the end of each phase, an output consisting of documentation is obtained, and each output constitutes the input of the following phase. Since the cost of possible changes due to the output-input relationship will be high for the project, skipping between phases or large-scale returns is not allowed. In the projects where the waterfall model has been applied, the phase endings are clearly defined, and the project is carried out under intensive supervision with strict examinations, intensive documentation, and management approval at each phase change. Therefore, its implementation requires strict discipline. This model adopts the "big front design" philosophy, and a substantial design is created before the coding phase (Farell, 2007, p. 11). System and software needs are determined at first by accepting the assumptions that everything necessary is known from the beginning and that there will be no change.

For this reason, it is more suitable for projects that are fully understood and unlikely to change. Because these assumptions constitute the biggest weakness of the waterfall model. The customer is in the project when the needs are defined, and it does not appear in the design and coding phases; it reappears in the test and integration phases. In the waterfall model, planning, timelines, target dates, budget, and putting the whole system into use at once are the most emphasized issues. The primary purpose of this approach

is to fix factors such as scope, time, and resources from the outset by not allowing changes during the project period and neutralizing change, which is a considerable risk factor (Vohra et al., 2013, p. 23-27). When the waterfall model emerged, and the conditions were evaluated, business processes were not as dependent on information systems as it is in our age, and projects did not have to be completed in a hurry. Technological development and change in business life were not as fast as our age, so the projects were not progressing at a pace as fast as in our age. Due to technology and resource constraints, information systems were hardware-centric, and the primary purpose of the software was to use limited hardware resources such as storage capacity and processing power in the most optimum way. (Munassar et al., 2010, p. 95). For these reasons, the waterfall model was seen more as a risk management method when it emerged, and the issues that were deemed necessary to manage this risk and which were the basis of the traditional project management world with the waterfall model were as follows:

- All requirements had to be defined early for planning and budgeting to be done correctly.
- Identifying requirements early and fixing them unchanged would prevent over-coverage.
- All design and decision-making processes had to be documented for future reference.
- Intensive examinations between phases would ensure that all stakeholders were aware of the process and early detection of potential problems.
- All requirements had to be documented before design activities started to ensure that the architecture was appropriate.

- All coding processes had to be completed before testing to ensure that the capabilities of the entire system were evaluated.
- All coding processes had to be completed before testing to ensure that the capabilities of the entire system were evaluated.
- All requirements had to be met before the system was implemented. (Palmquist et al., 2013, p. 5)

4.2.4. Benefits of waterfall model

- Defining the requirements clearly, understandably, and precisely at the beginning of the project cuts down uncertainties in the project by providing more accurate time and cost estimation.
- Since it is a widely used method, it is generally well known in theory.
- It is easy to implement as it is a linear process.
- It is suitable for project staff, project managers, and project teams with fluctuating performance, as it is strictly defined and followed by who should do what and when.
- If a reasonable estimate of time, cost, and effort are made, the probability of overrun of allocated resources is minimal.
- Project progress is easy to track as the previous phase is completed before starting a phase. Besides, the absence of a simultaneous progressing phase facilitates resource tracking.
- The fact that documentation is a continuous activity prevents the effort of going back to the past for documentation purposes at the end of the project, as well as ensuring that the documentation is complete at any time of the project and allows the present step of the project to be evaluated through this documentation.

- The waterfall model is disciplined and stable and gives the project members the feeling that the project is well managed and everything is under control.
- It is clear when and how much the client will contribute to the project and is not expected to allocate time throughout the project.
- It prevents overlapping works of project staff who work on more than one project in functional organizations. Thus, an employee who continues to work on another project after the analysis is completed does not need to return.
- The detailed and comprehensive documentation prepared can be used as a communication tool and reduces the need for face-to-face communication, especially in large projects and projects where all the elements are not in the same environment. (Munassar et al., 2010, p. 96), (Balaji et al., 2012, p. 27), (Farrell, 2007, p. 10-14)

4.2.5.Constraints of waterfall model

- Flashbacks for change, improvement, or correction are likened to swimming upstream in a waterfall, and it is difficult and costly. Therefore, it cannot respond to changes, and requests that come in the process are often not accepted.
- Any negativity that has not been detected in any phase affects all subsequent phases.
- It has an inelastic, slow, and cumbersome nature due to its structural feature and tight controls.
- It may not be very realistic to expect the requirements to be defined fully and clearly at the beginning of the project, and failure to fully and accurately express the customers' needs may result in a defective or incorrect product at the termination of the project.

- Instead of understanding the requirements of the developers by meeting face to face with customers or users, reading from the documents prepared in the analysis and design stages makes it difficult to understand the requirements and does not allow the developers to input.
- Requirement inconsistencies, missing system components, and unexpected development needs can only be identified at the design or coding stage.
- System performance cannot be measured until all coding is complete, and errors cannot be detected until the entire system test.
- Intensive and up-to-date documentation is a time-consuming and labor-intensive activity and is generally seen as an unnecessary burden by developers.
- Written system descriptions are difficult to read by users or customers and to be fully understood and implemented.
- Even delays that do not concern all project staff, because the next phase does not start before completing a phase, may cause a waste of time for all project staff.
- High administrative expenses are unnecessary costs for small projects and small teams.
- Software development requires creativity, and the highly disciplined approach of the waterfall model negatively affects creativity. (Munassar et al., 2010, p. 96), (Farell, 2007, p. 10-14), (Awad, 2005, p. 3)

4.3. Agile Approach

Stanley et al.(2020) stated that “agile is a method of project management that focuses on dividing tasks into short phases of work, with frequent reviews of the project and adaption of planning mid-execution as needed.”

Agile software development is a cost-effective iterative and incremental (evolutionary) approach to the changing needs of stakeholders. It meets with a highly collaborative approach within a practical governance framework that includes adequate formalization by teams that self-organize and produce high-quality solutions. As a result, agile methods have emerged as alternative solutions for issues where traditional methods are considered to be insufficient. (Stanley et al. 2020, p. 41)

4.3.1. Brief history and principles of agile project management

The idea of Agile PM was first introduced in a paper in the Harvard Business Review by Takeuchi and Nonaka in 1986. However, according to Cervone (2011), the concept of Agile PM did not draw much attention until the use of agile methods in software development is firstly discussed in 1995. In 2001, seventeen software engineers, among the creators and leading practitioners of agile methods, came together to find an alternative to traditional software development processes based on documentation and be involved in a common ground based on the methodologies. These experts, who call themselves Agile Software Development Alliance, determined the basic principles and values of agile methods due to the study and published the Agile Manifesto. Agile Manifesto was created under the name of the Agile Software Development Alliance movement. Agile Software Development Alliance (2001) declared a manifesto with four values as follows: “We are uncovering better ways of developing software by doing it and helping others do it. Through this work, we have come to value:”

- “individuals and interactions over processes and tools”
- “working software over comprehensive documentation”
- “customer collaboration over contract negotiation”

- “responding to change over following a plan.”

Along with the Agile manifesto, 12 principles have also been produced and listed below:

- *“Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.”* It is done with continuous code submissions from the first stages of the project. The client is enabled to deliver value by starting the software as a whole before delivery. One of the main reasons for the widespread use of agile methods today is the rapid return on investments made.
- *“Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.”* The aim is to develop software that meets the customer's needs, benefits them, and adds real value. Changes in needs should be reflected in the software even in the later stages of the project. Thanks to applications such as test-driven development, comprehensive automatic tests, continuous integration, simple design, and code refactoring, the costs of changes are minimized, and the process is made more suitable for changes.
- *“Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.”* The software delivery interval is typically 1-4 weeks. In this way, continuous feedback is provided, and the software evolves and develops in line with the customer's wishes.
- *“Business people and developers must work together daily throughout the project.”* Continuous active communication is provided between different roles. Instead of role-based teams, teams are formed according to software features.

Testers, analysts, and software developers work together in the same team and are in constant communication.

- *“Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.”* Motivating means "to act, to provoke. As the owner or manager of the product, motivation, environment, and support will be created, and a motivated team will work in an environment of trust. Thus, it will provide transparency and honesty to the project.
- *“The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.”* Face-to-face meetings with the product owner or manager and between team members will create synergy and increase motivation.
- *“Working software is the primary measure of progress.”* Working software is not the same as complete software. The running software does not need to be finished to add value to the end-user. By introducing the running software gradually, the opportunity to provide more regular feedback to end-users is provided. Thus, the development team serves only the need without creating unnecessary features and functionality.
- *“Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.”* For the planning to be healthy, the work delivery speed of the team must be reliable. For example, it is not one of the preferred methods to temporarily increase the team's speed by methods such as overtime. Instead, the work should be divided into pieces considering the capacity of the team, and the expectation should be shaped in this way.

- *“Continuous attention to technical excellence and good design enhances agility.”* Care is taken to create technically perfect, simple solutions. The best design is a fast-developing design. Every feature prioritized by the customer must be created with technical excellence and sound design. One should not deviate from the absolute need for the sake of complexity. This may result in technical debt in the later stages of the project.
- *“Simplicity--the art of maximizing the amount of work not done--is essential.”* The concept of simplicity is not to apply the first evasive solution that comes to mind. It is to produce a solution that is easy to understand and change later, has the lowest cost, and meets the current needs.
- *“The best architectures, requirements, and designs emerge from self-organizing teams.”* Self-organization is the crucial point here. Teams supported by senior management will ensure their organization without feeling any pressure on them by taking risks when necessary. Thus, the risk of technical debt formation is also reduced.
- *“At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.”* To work more effectively and effectively, studies conducted with retrospective meetings are reviewed. (PMI, 2017, p. 18)

4.3.2. Features of agile approach

First introduced in the late 1980s, the agile project management methodology is characterized by a nonlinear approach. Requirements and solutions are developed throughout the process through collaboration between self-organizing, cross-functional teams rather than completing the "Planning" step.

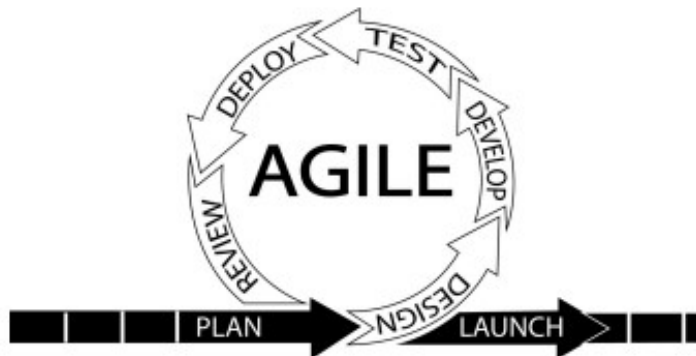
Project work is carried out continuously throughout the project period. Agile methods include adaptive planning, continuous improvement, and early delivery. It encourages agile, rapid, and flexible responses to change.

Iterations implemented in agile methods are formed by the experiences gained in previous iterations and the detected flaws. Assignments that require to be completed are prioritized according to the business evaluation they have. The project team decides how best job to do the work based on available resources and constraints. The project team must discharge determined tasks within a specified iteration time. It is the team, with its strengths and weaknesses, that is chargeable for forming the product to be delivered at the termination of the iteration. For this reason, in-team collaboration is essential. In this context, the basic principles on which agile methods are based can be listed as experimentality, prioritization, self-organization, time frame, cooperation. This method is becoming increasingly popular and is often used to manage the software creation process. It focuses on teamwork and finishes one feature of the software at once (exactly) before moving on to the next. Agile iterative flow is seen in the figure below.

(Stanley et al., 2020, p. 42).

Figure 9

Agile Iteration Flow



In agile methods, software development activities are applied in iterative phases. As a result of these iterations, usable products are created, and the development processes are repeated in the light of the feedback of the customers or users and line with the changing requirements if any. These iterative processes are continued until the exact product that the customer wants is produced.

In summary, Agile is a project management method that focuses on breaking down tasks into short work phases, frequently revising the project, and adapting planning in the middle of execution as needed. An agile approach is flexible and less rigid than other project management methods. (Stanley et al., 2020, p. 43)

4.3.3. Benefits of agile approach

- Due to the viewpoint of agile methods, changes are welcomed and even stimulated throughout the project. Moreover, thanks to the iterative process design, adaptability is high, and changing conditions are easy to manage.
- The confidence and inspiration of the undertaking staff increments and their efficiency level increments with the sharing of essential data and sentiments, on account of the short-foot gatherings and human and correspondence, focused cycles applied in coordinated strategies to think of up close and personal correspondence.
- The progress of the project is monitored more healthily, thanks to the working product, which is the result of short-term repetitions. Project risk is decreased, potential mistakes are detected earlier, delivery is more predictable, so the product reaches the market faster.
- The level of integration of business and software development processes raises, as customers or users can communicate face-to-face with the project personnel during the project process. As a result, the utilitarian quality of the software

increases, and customer fulfillment is ensured by fully expressing the desired features.

- Distinguishing issues with the input from the functional item transmitted towards the end of the cycle are addressed in the following stages, and the support and maintenance interaction of the product is made more straightforward, and maintenance support costs are reduced.
- Since agile methods are applicable for transitions and returns between the applied stages, retrospective problems identified during the interaction can be handily adjusted.
- In general, only the product features of the produced software are translated into documentation, and other processes are excluded. Therefore, no documentation should be perceived. With lightweight documentation, documentation reduces the workload.
- Performing the integration and testing activities in all iterations prevents the work of the employees working in this field from accumulating and ensures a more stable effort. (VersionOne Inc., 2020), (Floara et al., 2014, p.3626-3627)

4.3.4. Constraints of agile approach

- Products cannot be reused in other projects, as solutions are developed with agile methods that are sufficient to meet the requirements defined according to specific problems or functions.
- As communication and interaction will be less in crowded projects, the effectiveness of agile methods decreases.
- Large and complicated systems that require functional integrity may not be divided into repetitive parts.

- Requirements may not be adequately defined if the customer or end-user engagement is low due to lack of time or reluctance.
- The resource and scope planning of the project is difficult due to its openness to change.
- If the documentation prepared with the understanding of "as much documentation as necessary" is insufficient, the developers who join the project later, the legal regulation and certification processes, and maintenance-maintenance phases may be negatively affected.
- For systems that are not fault-tolerant, the testing and quality control procedures of agile methods may not be sufficient, because in these cases, software errors can endanger human life or cause significant financial losses.
- For the subcontractor to submit an offer and a contract to be made, what is required must be correct and unaltered. The uncertainty and volatile environment of agile methods make subcontracting applications difficult.
- Since teams are self-organizing, individuals should be skilled, experienced, and well-versed in agile processes. Unskilled and inexperienced personnel can reduce team performance, thus negatively affecting the project development process.
- Customers who cannot define their priorities precisely and who want to change their priorities on the worklist frequently can harm their plans.
- Customer benefit is prioritized in agile methods, but customer benefit generally meets the customer's functions. Since the philosophy of simplicity is applied as an inheritance of lean development, user satisfaction issues such as user interface and user experience can be ignored. (Awad, p. 95), (Floara et al., 2014, p.3626-3627)

4.3.5. Types of agile approach

Although there are Agile principle frameworks developed by applying different principles, several methods are widely used. The essence of all these different Agile methods; They focus on delivering a product that works at frequent intervals and focusing on business and problem solutions quickly (Rajagopalan et al., 2016, p. 40).

4.3.5.1. Lean software development

Lean software development is an iterative Agile software development method developed by Mary and Tom Poppendieck. LSD is the application of lean manufacturing procedures with lean thinking in software development. LSD provides a good framework for organizations with its values and rules that support agility and are shaped by professional practices. In addition, LSD focuses the development team on adding value to the customer. The main principles of this approach are as follows:

- a. Elimination of excess
- b. Increasing learning
- c. Do not decide as late as possible
- d. Delivery as fast as possible
- e. Strengthen the team
- f. Forming a union
- g. Seeing the whole (Poppendieck et al., 2013, p. 3-4).

4.3.5.2. Kanban

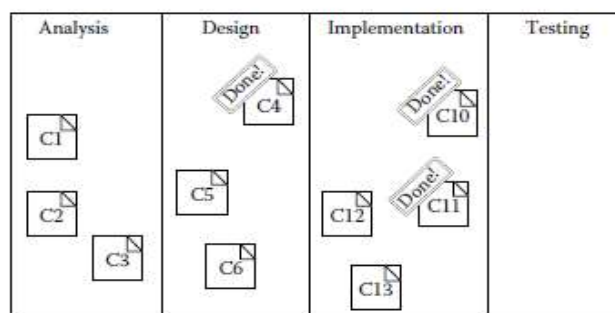
According to “Kanban is a lean-management practice that eliminates waste by using a just-in-time, pull-based production system.” Taiichi Ohno, working at the

automaker Toyota, had come up with an idea to increase efficiency in car production. Kanban means "Signboard" in Japanese. It comes from the fact that a sign has to be used to monitor the pulling process. The signage is arranged in different areas, each corresponding to a different stage of the production chain. Cards are used to represent different business items, and each card represents a different item assembled on the production line. As work items are lined up, cards move around the bulletin board, providing a simple way to track progress and needs. (Villafiorita, 2014, p. 225).

While the use of kanban started in the 1940s, the use of this method in the informatics industry dates back to 2004. David J. Anderson was helping a small and troubled team at Microsoft. The use of kanban started with visualization of current and future studies. (Ahmad et al., 2013, p. 2). This chart contains different columns showing different processes. Generally, these columns are named with specific statuses. For example; The statuses open, being worked on, reviewed, and done are mostly used. Information system departments where development processes are located are additionally used in addition to the statuses stated above, such as design and development. Thus, the current status of the card, that is, the task, can be quickly followed on this chart. The figure below illustrates the process by showing a Kanban board. (Villafiorita, 2014, p. 226)

Figure 10

Kanban Board



Kanban principles are based on three main pillars.

1. To increase transparency within the team and to show the status of the project or product being worked on, the tools that provide the use of charts in the form of a table that can be used in the physical environment or charts that can be followed on the internet are used. The team can decide together and name the columns to be shown on the sign. Visualization is one of the main principles.

2. It is the limit of the number of jobs worked, that is work in progress limit. Limit indicates the maximum working limit of the individuals in the team. The aim is to prevent the team member from getting work beyond his capacity. A team member whose continuing work in progress limit has expired should not start a new job before completing their current job. With the determination of the limit, better quality jobs will emerge, and the team members will be prevented from multitasking.

3. To increase the workflow and shorten the delivery time. For example, if the work is finished and the work in progress limit has not reached the maximum, the most important and priority job from the pending job list is withdrawn by the team member, and work begins on it.

Kanban principles are based on three main principles; The employee produces the most valuable output at the specified time by making the work visible, allowing team members to work within their boundaries. (Villafiorita, 2014, p. 226)

4.3.5.3. Extreme programming

It is one of the most well-known Agile software methods. The most well-known feature of this method is that it takes a "high quality," "fast," and "continuous" approach. Increasing the involvement of the customers in the process and focusing on their needs ensures that the decisions are made quickly, and the necessary feedback is given. While

doing this, the team works very closely, and continuous development and testing are supported. Each version transition can take approximately 1-3 weeks.

At the end of each release cycle, the compiled system is distributed to its end users, which can happen every 2 to 5 iterations. A requirements aggregation unit is a user-visible "user story" function that can be developed in a single iteration. Customers write stories on simple index cards to repeat. Then, the client and programmer discuss the actions in the next iteration as follows:

Developers estimate the completion time of each card.

Clients prioritize, change, and scope out as needed, most likely to complete the most valuable stories within the allotted time frame.

For each story, the programmer writes the task on a card on the wall or whiteboard and estimates the time required for each task.

Programmers can work in pairs. They adhere to the strict coding standards set at the beginning of the project. They run unit tests for every code they write and ensure that these tests run at 100% every time they check their code in mandatory version control and configuration management system. They evolve in small increments from 15 minutes to several hours, integrating their code several times a day. At the end of each of these integrations, the entire codebase has passed all unit tests. (Cockburn, 2000, p. 139). The table of applications belonging to XP is as follows. (PMI, 2017, p. 79)

Table 4

The Practises of XP

XP Practise Area	Primary	Secondary
------------------	---------	-----------

Organizational	• Sit together • Whole team • Informative workspace	• Real customer involvement • Team continuity • Sustainable pace
Technical	• Pair programming • Test-first programming • Incremental design	• Shared code/collective ownership • Documentation from code and tests • Refactoring
Planning	• User stories • Weekly cycle • Quarterly cycle • Slack	• Root cause analysis • Shrinking teams • Pay per use • Negotiated scope contract • Daily standups
Integration	• 10-minute build • Continuous integration • Test-first	• Single codebase • Incremental deployment • Daily deployment

Every day, the team organizes a stand-up meeting where they explain their work, what works out for them, and what can help. To keep the meeting short, it is held standing up. At the end of each iteration, they will schedule another meeting where they will review their outstanding performance and hope to do so next time. Finally, they publish this list for all to see in the next iteration.

There are four values that XP rewards. These; courage, communication, simplicity, and testing. The "courage" value is always seen as the courage to make progress and improve the system. In addition, one person on the team is designated as the "coach" of the team. This person checks with team members if they are using core practices: binary programming and testing, two spins, making the design simple, communicable. (Cockburn, 2000, p. 140)

4.3.5.4. Dynamic software development

Firstly, in 1995, Rapid Application Development was intended to be implemented to respond to the deficiency. It was introduced starting from three points: people are critical to the success of the project, change is inevitable, and no software is

perfect when it is first released. As the Agile model has developed, it has continued to develop in DSDM.

Unusual for an agile methodology, DSDM adopts a wide range of activities that convey issues such as governance, quality, and risk throughout the project lifecycle.

The pre-project phase is where a project starts, and all the activities required to create a project are carried out. The following two phases are the feasibility of the foundations, where the team investigates whether the goals can be achieved with the given constraints and the business value of the proposed project is analyzed, the requirements are prioritized, and the basic architecture is drawn. DSDM method attaches great importance to prioritization, and for this, it uses the MoSCoW method.

- a. Must Have (M)
- b. Should Have (S)
- c. Could Have (C)
- d. Will not have it (W)

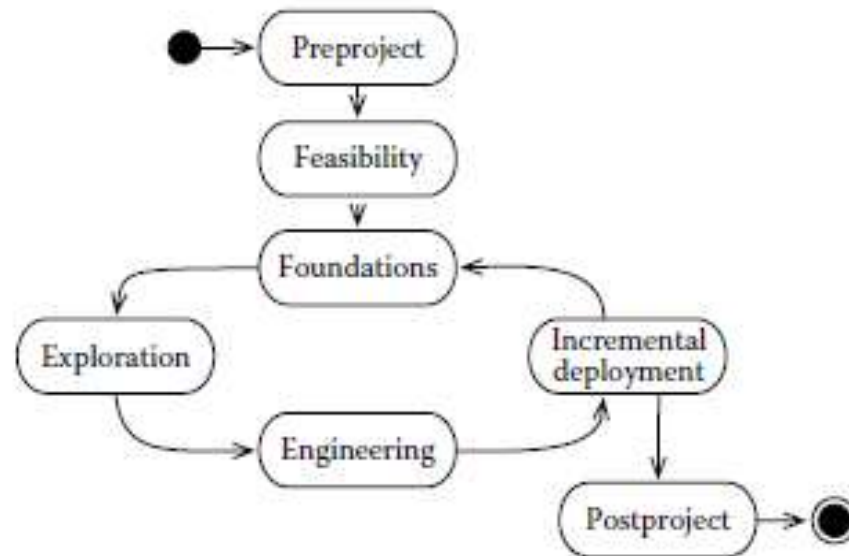
The iterative process starts at the end of this activity. The iterations are organized in three steps of fixed duration (time box). The discovery step generates a prototype that is used with the customer to improve requirements and priorities, identify non-functional requirements, and define an operational plan for the next activity. The engineering phase is where non-functional requirements are added to the prototype, and the prototype is tailored for delivery. Finally, the incremental distribution step is where the results of the current increment are published.

Finally, a post-project phase delivers the final solution to the customer and manages product maintenance. The method is well documented, and a variety of

resources are available on the Internet, including the official page of the DSDM consortium, which offers specific restrictions, materials, and templates for both practitioners and teachers. (Villaforita, 2014, p. 220-221). The figure below shows the iterative and organized DSDM development process in seven stages. (Villaforita, 2014, p. 220)

Figure 11

The DSDM process



DSDM differs from other Agile methods as it requires a standard reporting and follow-up; it is less applied in other Agile methods Provides a control framework for fast applications. It keeps time and resources constant and provides functionality. Functional requirements should be prioritized in this approach. At the same time, high quality and adaptation to changing requirements are essential. (Moran, 2015, p. 22). It is clear from the table below, that DSDM is very detailed in the guidance it offers. (Moran, 2015, p. 23)

Table 5*DSDM Methodological Dimensions*

DIMENSION	DESCRIPTION
Principles	Focus on the Business Need, Deliver on Time, Collaborate, Never Compromise Quality, Build Incrementally from Firm Foundations, Develop Iteratively, Communicate Continuously and Clearly, Demonstrate Control
Roles	Business Sponsor, Business Visionary, Project Manager, Technical Co-ordinator, Team Lead, Business Ambassador, Business Analyst, Business Advisor, Technical Advisor, Solution Developer, Solution Tester
Artefacts	Terms of Reference, Business Case, Prioritised Requirements List, Solution Architecture Definition, Development Approach Definition, Delivery Plan, Management Approach Definition, Feasibility Assessment, Foundation Summary, Timebox Plan, Timebox Review Record, Project Review Report, Benefits Assessment
Practices	Facilitated Workshops, Timeboxing, Iterative Development, Moscow Prioritisation, Modelling (incl. Prototyping)
Phases	Pre-Project, Feasibility, Foundations, Evolutionary Development, Deployment, Post-Project

4.3.5.5. Crystal

The basic Crystal philosophy is: Software development is beneficially seen as a collaborative invention and communication game, with the secondary purpose of delivering working software and setting it up for the next stage.

The two consequences of this philosophy are that different projects must be run differently, and the amount of modeling and communication people have to do is the amount they need to move the game together—the values that should be at the core of Crystal's methodologies: communication-centric people and high tolerance.

The first means that tools, work products, and processes are only there to support the human component. Second, it recognizes various human cultures. Third, within the

tolerance of the Crystal family, the team can choose to work in a highly disciplined manner, reducing intermediate work products as the team generates employee code more often and uses more prosperous communication channels between people.

Since every project is different and evolves, the set of rules adopted by the team must also be shaped and developed. In addition, changing bottlenecks in the system determine the use of the job. The two rules common to the Crystal family are:

1. The project should use incremental development at intervals of four months or less.
2. The team should organize pre- and post-increase reflection workshops (also prefers to organize medium increment reflection workshops). (Cockburn, 2000, p. 166).

In summary, the important thing within the framework of Kristal is the efficiency of teamwork, correct communication, simplicity, and improving the process by adding new features. It attaches great importance to having software that works like other Agile frameworks. It is a flexible and easily applicable method. (Rajagopalan et al., 2016, p. 42).

4.3.5.6. Adaptive software development

Highsmith (2000) states that adaptive software development is always a philosophy that argues that projects are in a continuous process of adaptation. Adaptive software development has three cycles for production: speculation, collaboration, and continuously sequential learning. Highsmith (2000) suggests that adaptive software development has five main goals:

- It offers an alternative to the perception that optimization is a unique solution for more complex problems.

- Provide an organization with a set of frameworks or models in the use of adaptive principles.
- Build collaboration and interaction between similar or careless people to find solutions for organizations that arise product development problems.
- Establish a procedure for organizations to use the adaptive approach in their large-scale projects.
- Suggest an effective management style for organizations (Highsmith, 2000)

4.3.5.7. Feature driven development

Feature Driven Development is a model-driven and iterative process. As the name suggests, it is focused on "feature" development.

“There are six primary roles on FDD;

- Project Manager,
- Chief architect,
- Development manager,
- Chief programmer,
- Class owner and/or
- Domain expert.”

FDD is a development process that focuses on the iterative and incremental delivery of working and valuable software. It focuses primarily on customer needs, so manufacturers prioritize the needs and features requested by the customer. There are five FDD processes;

- “Develop an overall model”: In the FDD method, first of all, the relevant team determines the model of the problem. Then, under the management of the chief

architect, the whole team works together to develop the model. Thus, the whole team will dominate the process.

- “Build a features list”: After the model is determined, the features to be developed are divided into small jobs. Developing a feature will take a maximum of two weeks.
- “Plan by feature”: The features that are decided to be developed are assigned to the developers.
- “Design by feature”: The developer makes the desired development regarding the particular feature.
- “Build by features”: At the end of the development, the chief architect makes the necessary controls and makes the approval process. (PMI, 2017, p. 83-84).

4.3.5.8. Scrum

According to Schwaber ve Sutherland (2017), “A framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value.”

The Scrum method, developed by Ken Schwaber and Jeff Sutherland, is applied in the development of vast and complex structures. Scrum is not an acronym, and it is a term taken from the sport of Rugby. It indicates a way to restart play in a fault situation or when the ball goes out of play. (Stanley et al., 2020, p. 48)

Scrum is a framework in which we can use types, processes, and techniques. While doing this, supporting concepts are used: Transparency; The work done should be shared with the whole team, all members of the team should accept the end of the work. Observation: To complete the work to be completed correctly, the work done must be followed frequently. Adaptation; If the person responsible for the observation predicts that the completion of the work is complex and unacceptable, it will make a

rapid change in the product, avoiding the deviation and difficulty of the development process (Schwaber et al., 2020, p. 3).

Core principles of Scrum methodology;

- **Self-Organizing Teams:** A strength of Agile is the self-organizing of multi-functional teams with team members constantly interacting while working on specific short-term goals. The Scrum team is a cross-functional and self-organized team.
- **Self-Managed Teams:** Scrum ensures reaching the target with regular feedback and short-term planning. Everyone on the team should understand what the goal is, knowing that the job is challenging but keeping in the right direction and focusing.
- **Information Sharing:** Information is shared by all team members. Communication is facilitated by locating team members in the same place, sometimes enhanced by "team rooms" containing informative charts such as burning charts showing remaining work.
- **Minimum Processes and Principles:** The Agile approach breaks down work by shrinking processes only to the minimum required to handle the current business.
- **Frequent Releases:** Most agile approaches release software more often than traditional methods. Agile Teams deliver code early and often, obtaining frequent checkpoints to verify that they understand the requirements.
- **Continuous Testing:** Agile Teams at the end of each iteration. It tests to make sure all planned features are fulfilled. This frequent testing helps reduce the risk. This way, more problems are detected early in the project life cycle.

- **Supporting Infrastructure:** Supporting infrastructure is needed to produce sufficiently stable software and work at the prescribed quality level, especially for large applications. For example, continuous builds validated by applications, automated unit-level testing framework, automated system-level regression testing, and an automated software release process.

Customer Collaboration: It makes it possible to place large stones in advance as it works in order of need and importance in the software development process. Nowadays, as large stones are shaped according to customer needs over time, it makes it possible to structure according to the regular feedback of the customer. It is the hallmark of the Scrum methodology, and face-to-face communication maximizes its effectiveness. However, as the complexity and size of the system grow, it can be not easy to find a single person who is a good communicator, works full time, and understands the entire system. (Gechman, 2019, p. 58)

4.3.5.8.1. Scrum team and roles

Teamwork is essential in Scrum. Each individual in the team has their responsibilities. Throughout the study period, team members must be completed in a result-oriented manner. How to do a job that is brought to the agenda and the ability to implement innovations are in the Scrum team. Four types of crews can be seen.

In a manager-led team structure, the team is a little more passive and needs direction.

In the self-organized team structure, the Scrum team members, their duties and roles are predetermined. Therefore, how to do the job belongs to the Scrum team.

The team that sets up its team acts more independently by working on incoming works, job tracking, and process management. The participation of the entire team determines scrum roles.

A self-managing team refers to a higher-level management structure. It emphasizes the meaning and purpose of the work to be done. It is responsible for the general state of affairs and looks from a broader perspective. (Gechman, 2019, p. 61)

Scrum Master

In Agile literature, the Scrum Master, also called Team Leader or Coach, performs some functions essentially similar to the Software Project Manager, including acting as the interface between the Agile Team and the outside world.

Some of Scrum Master duties include:

- It is the protector of the agile process, and it ensures the smooth running of the process.
- It removes barriers that affect productivity and facilitates critical meetings.
- Responsible for understanding Scrum well enough to educate and guide other roles and educate and assist other stakeholders.
- It provides constant awareness of the status of the project. However, he does not manage the team and cannot assign tasks to team members as task assignments are the internal responsibility of the team.
- Prevents new requirement changes from being made during the sprint. It allows the team to focus on performing their tasks without being distracted and take advantage of the resulting balancing environment. (Heldman, 2018, p. 42), (Gechman, 2019, p. 61)

Product Owner

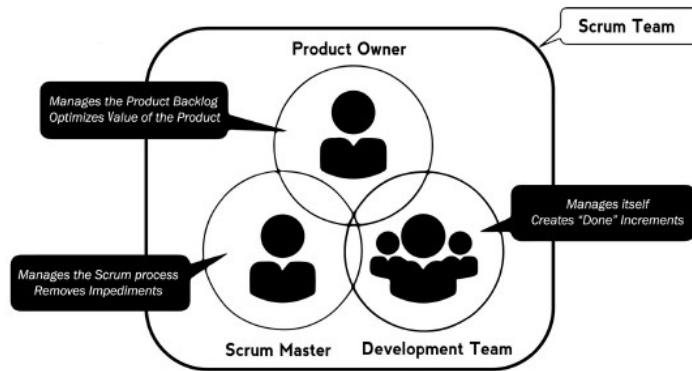
The Product Owner is the guardian of customer requirements and the only point of contact regarding all requirements and the planned implementation sequence. The Product Owner is the interface between the organization, the customer, the product-related needs, and the team. They are the "boss" of the project but allow members to set their own goals and manage themselves. Each project has only one Product Owner.

Some of Scrum Master duties include:

- Backlog expresses its tasks clearly.
- It frees the team from feature and bug fix requests from multiple sources.
- It lets everyone know what the priorities are and what to work on next.
- It makes the job batches visible and understandable to everyone. (Stanley et al., 2020, p. 52)

The Scrum Team

It is a self-organizing, cross-functional group of people who practically do the job of developing and testing the product. Since the team is responsible for the production of the product, it also has the authority to decide how the job will be done. Team members sign up for tasks based on the priority of the job and their skill set. They make predictions for the job and take on enough tasks to fill the iteration time. Agile teams are self-managing, self-organizing, and self-managing. Team size is usually kept in the range of 5–9 people. A Scrum Team means a team plus a Scrum Master and Product Owner. (Gechman, 2019, p. 61), (Heldman, 2018, p. 42). The figure below shows how all these roles work together in a project. . (Stanley et al., 2020, p. 64)

Figure 12*The Scrum team*

4.3.5.8.2. Scrum activities

Sprint and Sprint Planning

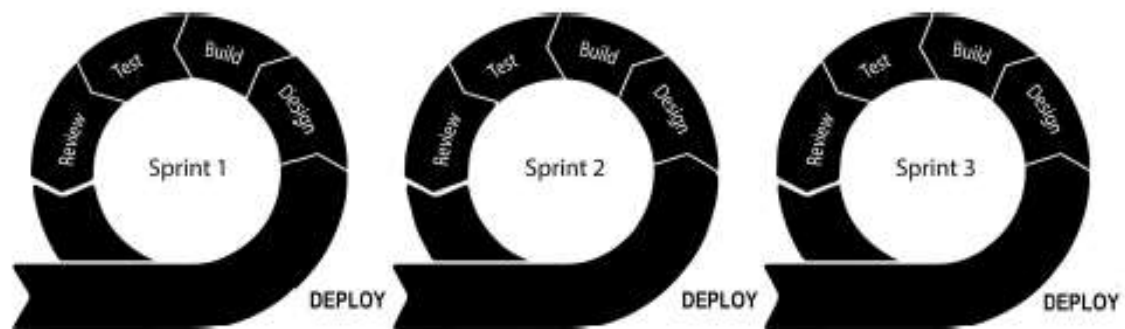
Sprint is the period in which the jobs selected from the worklist are completed in a limited time. The entire Scrum team participates in the sprint planning, and the Scrum Master manages the meeting. Sprint time is usually one week, two weeks, or 30 days. After the sprint period has been determined, no changes should be made regarding the time. Each Sprint end is a start for the other Sprint. For example, sprint planning for a one-month Sprint is approximately eight hours. The shorter the sprint time, the shorter this time. Schwaber and Sutherland (2020) Sprint Planning answered the following questions "What can be delivered as part of this Sprint employee product? How will the working piece of the product be delivered? " They explained in the form. About what will be delivered as part of the product, the product owner tells the entire Scrum team the prioritized work on the job list, informs the Scrum team why it is needed and adds value to the product when the job is completed. On the other hand, the development team shares its current capacity with the team and chooses the works

within the limits of capacity. Thus, the entire Scrum team determines the Sprint goal together. The completion of the current Sprint, on the other hand, takes place by meeting the definition of "Done" by all the planned works within the given time. The team creates the Sprint plan and the Sprint worklist after evaluating the worklist and determining how it will occur. Once the planning is done, the development team can break down the jobs on the Sprint backlog into smaller chunks.

Thus, it aims to work more efficiently. The development process is entirely the responsibility of the development team. (Stanley et al., 2020, p. 47)

Figure 13

Sprint planning



Sprint can be canceled if it is not successful and there is a situation that endangers the target. Cancellation is made only by the owner of the product. Cancellation is rare. The development efforts of the incomplete works within the canceled Sprint are re-evaluated, and these jobs are sent back to the backlog. Completed works are accepted by the product owner (Schwaber et al, 2020, p. 8)

Daily Scrum

Daily Scrum is a meeting with the development team that takes just 15 minutes. Thus, the state of the works in the Sprint within the team is known transparently by the

team. Every day, issues such as what will be done during the day, if any, which situations prevent the work are expressed briefly. Thus, obstacles that may occur in the Sprint target are determined in advance. Mistakes in understanding the content of daily meetings are pretty standard. First, team members should understand that this is not a reporting meeting but rather ensure transparency within the team and eliminate potential obstacles. (Heldman, 2018, p. 44)

This meeting can be managed in two ways. The first is that team members talk more and update the status. The second is done by the members of the development team answering the following questions, and these questions are stated by Schwaber and Sutherland (2017) as follows: “

- What did I do yesterday that helped the Development Team meet the Sprint Goal?
- What will I do today to help the Development Team meet the Sprint Goal?
- Do I see any impediment that prevents the Development Team or me from meeting the Sprint Goal? ”

Sprint Review

Sprint evaluation meeting is held at the end of each Sprint. What has been done with the product developed during the sprint is shown in this meeting. Thus, the value added by the work and what can be done afterward are evaluated in this meeting and shared with the participants. The owner organizes this meeting. The product owner invites the relevant stakeholders and the Scrum team. The owner explains what works within the current Sprint goal and lists and why not.

On the other hand, the development team conveys each work worked on in detail to the participants and answers their questions. At the end of the meeting, cooperation

is achieved within the framework of the questions and feedback received from the participants. At the meeting, customers, product owner, development team, Scrum Master, managers, managers, CEO, participants are found. (Heldman, 2018, p. 45)

Sprint Retrospective

Sprint Retrospective is a meeting that holds a mirror for the Scrum team to take the following steps more carefully. During the Sprint, which ends in the retrospective meeting, an evaluation is made by considering the criteria such as communication, openness, planning, and teamwork within the team. The inputs in this meeting become an improvement item for the next Sprint and create a more atmosphere. The meeting is held after the Sprint review meeting. A retrospective meeting of approximately 3 hours will be sufficient for a one-month Sprint. The retrospective meeting proceeds on two fundamental questions.

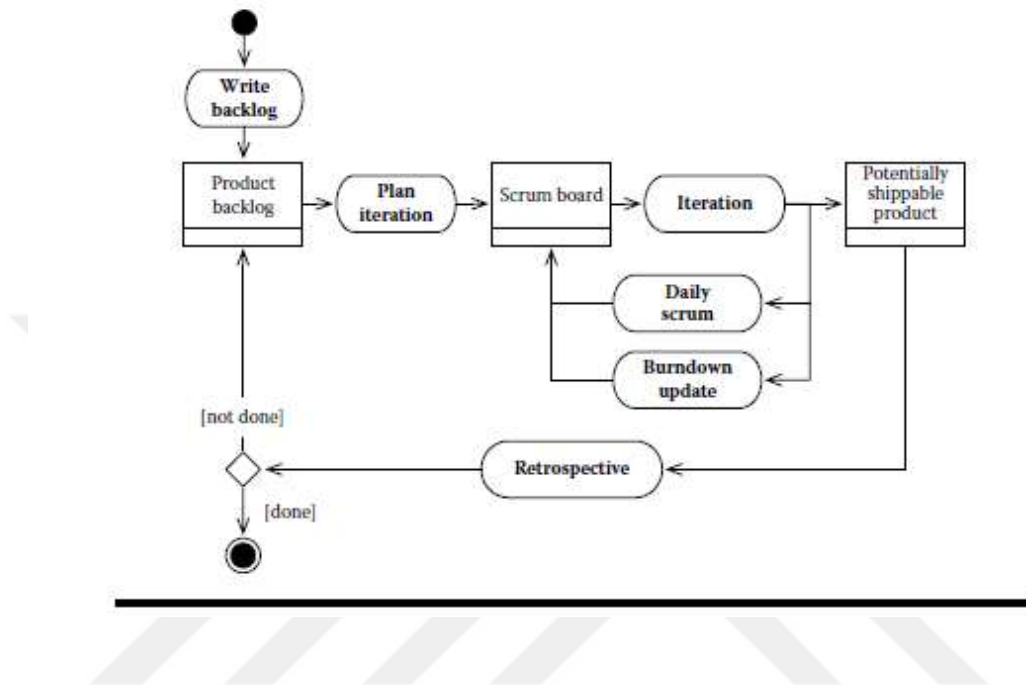
- a. What went well?
- b. What could be done better?

Based on these two main questions, a common denominator is tried to be formed with the team members' opinions, and these points are taken into consideration in the next Sprint. Thus, more teamwork and business planning are aimed.

A collaborative environment should be created at the beginning of the meeting, team values and operating agreement should not be ignored. After the retrospective, the high-priority actions to be taken according to the team decision are determined. Detailed planning is done for the decisions taken. Finally, new goals are set with measurable results. (Villaforita, 2014, p. 223-225). The figure below shows the scrum flow process. (Villaforita, 2014, p. 222)

Figure 14

The Scrum process



5. RESEARCH

In this section, the waterfall model and agile methods will be discussed after the literature review. A survey was conducted in order to receive the opinions of people working in the software industry and to present an experience-based comparison for these methodologies.

5.1. Personal Characteristics of the Participants in the Research

Table 6

Distribution of Participants by Demographic Characteristics

Demographic	Group	n	%
Gender	Female	53	36,6
	Male	92	63,4

Educational Status	Associate and Bachelor's Degree	103	71,0
	Master's degree and above	42	29,0
	Total	145	100,0

When the distribution of the participants according to their demographic characteristics is examined; It was determined that 36.6% were female, 63.4% were male, 71.0% were associate degree and undergraduate graduates, and 29.0% had master's degrees or higher.

Table 7

Distribution of Participants by Work Experience

Work Experience	Group	n	%
Total Work Experience	0-5 years	34	23,4
	6-10 years	65	44,8
	11-15 years	34	23,4
	16 years and above	12	8,3
Work Experience in the Industry	0-5 years	43	29,7
	6-10 years	61	42,1
	11-15 years	31	21,4
	16 years and above	10	6,9
Work Experience in the Institution	0-5 years	105	72,4
	6-10 years	26	17,9
	11 years and above	14	9,7
	Total	145	100,0

When their distribution according to their work experience is examined; 23.4% had a total work experience of 0-5 years, 44.8% had 6-10 years, 23.4% had 11-15 years, 8.3% had 16 years or more, 29% 0-5 years of work experience in the sector, 42.1% of them 6-10 years, 21.4% of them 11-15 years, 6.9% of them 16 years and above, 72.4% It was determined that work experience in the institution was 0-5 years, 17.9% had 6-10 years, and 9.7% had 11 years or more.

Table 8*Distribution of Participants by Work Characteristics*

Operational Features	Group	n	%
Position	Business/Systems Analyst	42	29,0
	SoftwareDeveloping	61	42,1
	Expert		
	Manager	17	11,7
	Other	25	17,2
Worked Firm Sector	Banking/Finance	29	20,0
	Airline/Aviation	56	38,6
	Production	15	10,3
	Other	45	31,0
	Total	145	100,0

29.0% of the participants are business/system analysts, 42.1% are software development specialists, 11.7% are managers, 17.2% are in other positions, 20.0% are in the banking/finance sector, It has been determined that 38.6% of them work in the airline/aviation sector, 10.3% in the production sector and 31.0% in other sectors.

Table 9*Distribution of Participants by Agile Process Model Used*

	Group	n	%
	Scrum	128	88,3
	XP	17	11,7
	Total	145	100,0

When their distribution is analyzed according to the agile process model used; It was determined that 88.3% of the agile process model used was Scrum and 11.7% was XP.

Table 10*Distribution of Participants by Project Characteristics*

Project Features	Group	n	%
Average Duration of the Project	1-3 months	55	37,9
	4-6 months	45	31,0
	7 months and above	45	31,0
Average Number of Employees in a Software Project	1-7 people	69	47,6
	8-15 people	30	20,7
	16 people and above	46	31,7
Number of People Working on the Largest Software Project	1-10 people	58	40,0
	11-20 people	35	24,1
	21-30 people	32	22,1
	31 people and above	20	13,8
Number of Persons in the Software Project Team	7 people and below	62	42,8
	7 people and above	83	57,2
Total		145	100,0

When the distribution of the participants according to the average duration of the project is examined; When the distribution according to the number of people working in the average software project is examined, 37.9% of them are 1-3 months, 31.0% are 4-6 months, 31.0% are 7 months and more, and 47.6% are 1- When the distribution of 7 people, 20.7% 8-15 people, 31.7% 16 people or more, and the number of people working in the largest software project is analyzed; 1-10 people of 40.0%, 11-20 people of 24.1%, 21-30 people of 22.1%, 31 people and more of 13.8%, according to the number of people in the software project team When their distribution is examined; It was determined that 42.8% of them were 7 people or less, 57.2% were over 7 people.

5.2. Analysis Results Of The Waterfall Model And The Agile Approach In Software Projects

In this part of the research, the descriptive statistics of the waterfall model and agile approach in the software projects of the participants and the findings of the determination of the differences according to the demographic characteristics are included.

Table 11

Normality Test Results of the Waterfall Model and Agile Approach in Software Projects of the Participants

Sub-Dimensions	Kolmogrov-Smirnov			Central Tendency Measurements			
	Statistic	Sd	p	$\bar{X}$	Median	Skewness	Kurtosis
Teamwork and Motivation	0,120	145	0,000	3,78	3,80	-0,232	1,506
Project Requirements	0,181	145	0,000	3,78	4,00	-0,833	-0,254
Efficiency and Quality	0,178	145	0,000	4,11	4,00	-0,185	0,019
Team Competence	0,320	145	0,000	3,87	4,00	-1,052	0,327
Time and Time Constraint	0,194	145	0,000	3,55	3,50	-0,379	-0,381
Waterfall Model and Agile Approach in Software Projects	0,114	145	0,000	3,82	3,85	-0,778	1,246

In determining the distribution of the data obtained, arithmetic means, median, skewness, and kurtosis coefficients from central tendency measurements were used. It was determined that the distribution of the data obtained came from the normal distribution because the median and arithmetic mean values were close or equal to each other and the skewness and kurtosis values fell within ± 2 limits.

Table 12

Descriptive Findings of the Participants' Waterfall Model and Agile Approach in Software Projects

Scale and Sub-Dimensions	$\bar{X}$	s.s.
Teamwork and Motivation	3,78	0,49
Project Requirements	3,78	0,99
Efficiency and Quality	4,11	0,49
Team Competence	3,87	0,88
Time and Time Constraint	3,55	0,85
Waterfall Model and Agile Approach in Software Projects	3,82	0,37

When the descriptive findings of the waterfall model and agile approach in the software projects of the participants are examined, the participants; it was determined that the general average of the scale ($\bar{x}=3.82$) was high. When the sub-dimension averages of the scale are examined; teamwork and motivation ($\bar{x}=3.78$) are high, project requirements ($\bar{x}=3.78$) are high, productivity and quality ($\bar{x}=4.11$) are high, team competence ($\bar{x}=3.87$) is high, and time is high. and time constraint ($\bar{x}=3.55$).

Table 13

Independent Sample T-test Results of Determining the Differences of the Waterfall Model and Agile Approach in Software Projects of the Participants by Gender Groups

Sub-Dimensions	Gender	n	$\bar{X}$	s.s	t	sd	p
Teamwork and Motivation	Female	53	3,82	0,42	0,623	143	0,534
	Male	92	3,77	0,54			
Project Requirements	Female	53	3,93	1,03	1,284	143	0,201
	Male	92	3,71	0,97			
Efficiency and Quality	Female	53	4,22	0,42	1,964	143	0,052
	Male	92	4,05	0,52			
Team Competence	Female	53	3,87	0,90	-0,117	143	0,907
	Male	92	3,89	0,88			
Time and Time Constraint	Female	53	3,65	0,74	0,987	143	0,325

Waterfall Model and Agile Approach in Software Projects	Male	92	3,51	0,92	1,762	143	0,080
	Female	53	3,89	0,33			
	Male	92	3,78	0,40			

H₁: The waterfall model and agile approach of the participants in software projects differ according to gender groups.

According to the results of the independent sample t-test conducted to determine whether the differences between the waterfall model and agile approach in software projects of the participants according to gender groups are significant; It was determined that the waterfall model and agile approach in software projects of the participants did not show a statistically significant difference according to gender groups. ($p>0.05$).

Table 14

Independent Sample T-test Results of Determining the Differences of the Participants' Waterfall Model and Agile Approach in Software Projects According to Educational StatusGroups

Sub-Dimensions	Educational Status	n	$\bar{X}$	s.s	t	sd	p
Teamwork and Motivation	Associate and Bachelor's degree	103	3,79	0,50	0,133	143	0,895
	Master's degree and above	42	3,78	0,50			
Project Requirements	Associate and Bachelor's degree	103	3,84	1,00	0,903	143	0,364
	Master's degree and above	42	3,67	0,98			
Efficiency and Quality	Associate and Bachelor's degree	103	4,12	0,50	0,182	143	0,856
	Master's degree and above	42	4,10	0,47			
Team Competence	Associate and Bachelor's degree	103	3,85	0,94	-0,695	143	0,489

Time and Time Constraint	Master's degree and above	42	3,95	0,75	-1,463	143	0,147
	Associate and Bachelor's degree	103	3,50	0,91			
	Master's degree and above	42	3,70	0,68			
Waterfall Model and Agile Approach in Software Projects	Associate and Bachelor's degree	103	3,83	0,38	0,193	143	0,847
	Master's degree and above	42	3,81	0,39			

H₂: The waterfall model and agile approach of the participants in software projects differ according to the educational background groups.

According to the results of the independent sample, t-test to determine whether the differences between the waterfall model and agile approach in software projects of the participants are significant according to the educational status groups; It was determined that the waterfall model and agile approach in software projects of the participants did not show a statistically significant difference according to the educational status groups. ($p > 0.05$).

Table 15

Independent Sample T-test Results of Determining the Differentiation of the Participants' Waterfall Model and Agile Approach in Software Projects According to the Agile Process Model Groups Used

Sub-Dimensions	Agile Process Model	n	$\bar{X}$	s.s	t	sd	p
Teamwork and Motivation	Scrum	128	3,79	0,52	0,177	143	0,860
	XP	17	3,76	0,33			
Project Requirements	Scrum	128	3,78	0,99	-0,278	143	0,781
	XP	17	3,85	1,05			
Efficiency and Quality	Scrum	128	4,08	0,48	-2,505	143	0,013*
	XP	17	4,39	0,50			
Team Competence	Scrum	128	3,86	0,91	-0,740	143	0,460
	XP	17	4,03	0,70			

Time and Time Constraint	Scrum	128	3,52	0,88	-1,364	143	0,175
	XP	17	3,82	0,64			
Waterfall Model and Agile Approach in Software Projects	Scrum	128	3,81	0,39	-1,226	143	0,222
	XP	17	3,93	0,29			

p<0,05*

H₃: The waterfall model and agile approach of the participants in software projects differ according to the agile process model groups.

According to the results of the independent sample, t-test to determine whether the differences between the waterfall model and agile approach in software projects of the participants are significant according to the agile process model groups; It was determined that the difference between productivity and quality perceptions according to the agile process model groups was statistically significant at the 95% confidence level. ($t = -2.505$; $p = 0.013$; $p < 0.05$). Agile process model XP ($\bar{X} = 4.39$) has higher perceptions of efficiency and quality than Scrum ones ($\bar{X} = 4.08$). When the results were examined, it was determined that the H₃ hypothesis was accepted in the productivity and quality sub-dimension.

Table 16

Independent Sample T-test Results of Determining the Differences of the Participants' Waterfall Model and Agile Approach in Software Projects According to the Number of Persons in the Number of Software Projects

Sub-Dimensions	Number of Persons in Number of Software Projects	n	$\bar{X}$	s.s	t	sd	p
Teamwork and Motivation	7 people and below	62	3,77	0,51	-0,356	143	0,723
	Above 7 people	83	3,80	0,49			
Project Requirements	7 people and below	62	3,77	0,92	-0,161	143	0,872

Efficiency and Quality	Above people	7	83	3,80	1,05	0,067	143	0,947
	7 people and below		62	4,12	0,58			
Team Competence	Above people	7	83	4,11	0,42	-0,097	143	0,923
	7 people and below		62	3,87	0,85			
Time and Time Constraint	Above people	7	83	3,89	0,92	0,169	143	0,866
	7 people and below		62	3,57	0,89			
Waterfall Model and Agile Approach in Software Projects	Above people	7	83	3,55	0,84	-0,491	143	0,624
	7 people and below		62	3,80	0,39			
	Above people	7	83	3,83	0,38			

$p < 0,05^*$

H₄: The waterfall model and agile approach of the participants in software projects differ according to the number of people in the number of software projects.

According to the results of the independent sample, t-test to determine whether the differences of the waterfall model and agile approach in software projects of the participants are significant according to the number of people in the number of software projects; It was determined that the waterfall model and agile approach in software projects of the participants did not show a statistically significant difference according to the number of people in the number of software projects. ($p > 0.05$)

Table 17

Anova Test Results for Determining the Differences of the Participants' Waterfall Model and Agile Approach in Software Projects by Total Work Experience Groups

Sub-Dimensions	Total Work Experience	n	$\bar{X}$	s.s	F	p	Scheffe
Teamwork and Motivation	0-5 years	34	3,75	0,51	0,687	0,561	
	6-10 years	65	3,79	0,49			
	11-15 years	34	3,86	0,57			

	16 years and above	12	3,63	0,22			
Project Requirements	0-5 years	34	3,79	0,99	2,675	0,050	
	6-10 years	65	3,75	0,98			
	11-15 years	34	3,60	1,08			
	16 years and above	12	4,52	0,53			
Efficiency and Quality	0-5 years ⁽¹⁾	34	4,12	0,48	4,513	0,005*	(4-1)
	6-10 years ⁽²⁾	65	4,13	0,50			(4-2)
	11-15 years ⁽³⁾	34	3,94	0,49			(4-3)
	16 years and above ⁽⁴⁾	12	4,53	0,26			
Team Competence	0-5 years	34	3,90	0,81	2,119	0,101	
	6-10 years	65	4,05	0,81			
	11-15 years	34	3,59	0,97			
	16 years and above	12	3,75	1,10			
Time and Time Constraint	0-5 years	34	3,46	0,86	0,880	0,453	
	6-10 years	65	3,60	0,94			
	11-15 years	34	3,47	0,77			
	16 years and above	12	3,88	0,53			
Waterfall Model and Agile Approach in Software Projects	0-5 years ⁽¹⁾	34	3,82	0,36	2,790	0,043*	(3-2)
	6-10 years ⁽²⁾	65	3,85	0,39			(3-4)
	11-15 years ⁽³⁾	34	3,69	0,39			
	16 years and above ⁽⁴⁾	12	4,03	0,19			

p<0,05*

H₅: The waterfall model and agile approach of the participants in software projects differ according to the total work experience groups.

According to the results of the anova test to determine whether the differences between the waterfall model and agile approach in software projects of the participants are significant according to the total work experience groups; It was determined that the difference between productivity and quality perceptions according to the total work experience groups was statistically significant at the 95% confidence level. (F= 4.513; p=0.005; p<0.05). Perceptions of productivity and quality of those with a total work experience of 16 years or more ($\bar{X}=4.53$) 0-5 years ($\bar{X}=4.12$), 6-10 years ($\bar{X}=4.13$), and 11-15 years higher than the ones with ($\bar{X}=3.94$). It was determined that the difference between the perceptions of the waterfall model and agile approach in

software projects compared to the total work experience groups was statistically significant at the 95% confidence level. ($F= 2.790$; $p=0.043$; $p<0.05$). The perception of the waterfall model and agile approach in software projects of those with 11-15 years of total work experience ($\bar{X}=3.69$) and those with 16 years or more ($\bar{X}=4.03$) is lower than. When the results were examined, it was determined that the H5 hypothesis was accepted in the productivity and quality sub-dimension and scale. The source of the difference between the groups was determined by the Scheffe test, one of the posthoc tests.

Table 18

Anova Test Results for Determining the Differences of the Waterfall Model and Agile Approach of the Participants in Software Projects by Work Experience Groups in the Sector

Sub-Dimensions	Work Experience in the Industry	n	$\bar{X}$	s.s	F	p	Scheffe
Teamwork and Motivation	0-5 years	43	3,67	0,55	1,750	0,160	
	6-10 years	61	3,84	0,48			
	11-15 years	31	3,89	0,49			
	16 years and above	10	3,64	0,25			
Project Requirements	0-5 years	43	3,84	0,97	2,403	0,070	
	6-10 years	61	3,67	1,00			
	11-15 years	31	3,72	1,05			
	16 years and above	10	4,55	0,55			
Efficiency and Quality	0-5 years ⁽¹⁾	43	4,10	0,50	2,959	0,035*	(4-1) (4-2) (4-3)
	6-10 years ⁽²⁾	61	4,13	0,51			
	11-15 years ⁽³⁾	31	3,98	0,48			
	16 years and above ⁽⁴⁾	10	4,50	0,24			
Team Competence	0-5 years	43	3,90	0,81	2,060	0,108	
	6-10 years	61	4,05	0,83			
	11-15 years	31	3,60	0,96			
	16 years and above	10	3,65	1,18			
	0-5 years	43	3,52	0,83	0,563	0,640	

Time and Constraint	Time	6-10 years	61	3,61	0,94	1,689	0,172
		11-15 years	31	3,44	0,79		
		16 years and above	10	3,80	0,54		
Waterfall Model and Agile Approach in Software Projects	Time	0-5 years	43	3,82	0,38		
		6-10 years	61	3,84	0,41		
		11-15 years	31	3,72	0,35		
		16 years and above	10	4,01	0,18		

p<0,05*

H6: The waterfall model and agile approach of the participants in software projects differ according to the work experience groups in the sector.

According to the results of the anova test conducted to determine whether the differences between the waterfall model and agile approach in software projects of the participants are significant according to the work experience groups in the sector; It was determined that the difference between productivity and quality perceptions according to the work experience groups in the sector was statistically significant at the 95% confidence level. (F= 2.959; p=0.035; p<0.05). Perceptions of productivity and quality of those with 16 years or more of work experience in the sector ($\bar{X}=4.50$) 0-5 years ($\bar{X}=4.10$), 6-10 years ($\bar{X}=4.13$), and 11-15 years are higher than those with ($\bar{X}=3.98$). When the results were examined, it was determined that the H6 hypothesis was accepted in the productivity and quality sub-dimension. The source of the difference between the groups was determined by the Scheffe test, one of the posthoc tests.

Table 19

Anova Test Results for Determining the Differences of the Waterfall Model and Agile Approach of the Participants in Software Projects by Work Experience Groups in the Sector

Sub-Dimensions	Work Experience in the Institution	n	$\bar{X}$	s.s	F	p
----------------	------------------------------------	---	-----------	-----	---	---

Teamwork Motivation	and	0-5 years	105	3,75	0,48	0,940	0,393
		6-10 years	26	3,90	0,53		
		11 years and above	14	3,81	0,59		
Project Requirements		0-5 years	105	3,80	1,02	1,457	0,236
		6-10 years	26	3,97	0,76		
		11 years and above	14	3,41	1,18		
Efficiency and Quality		0-5 years	105	4,10	0,53	0,926	0,399
		6-10 years	26	4,23	0,28		
		11 years and above	14	4,05	0,52		
Team Competence		0-5 years	105	3,94	0,89	0,882	0,416
		6-10 years	26	3,69	0,83		
		11 years and above	14	3,79	1,01		
Time and Constraint	Time	0-5 years	105	3,51	0,88	0,720	0,489
		6-10 years	26	3,73	0,82		
		11 years and above	14	3,61	0,74		
Waterfall Model Agile Approach Software Projects	and in	0-5 years	105	3,81	0,39	1,688	0,189
		6-10 years	26	3,92	0,22		
		11 years and above	14	3,70	0,50		

H₇: The waterfall model and agile approach of the participants in software projects differ according to the work experience groups in the institution.

According to the results of the anova test to determine whether the differences between the waterfall model and agile approach in software projects of the participants are significant according to the work experience groups in the institution; It was determined that the waterfall model and agile approach in software projects of the participants did not show a statistically significant difference according to the work experience groups in the institution. ($p > 0.05$).

Table 20

Anova Test Results for Determining the Differences of the Participants' Waterfall Model and Agile Approach in Software Projects According to the Position Groups Worked

Sub-Dimensions	Position	n	$\bar{X}$	s.s	F	p
Teamwork Motivation and	Business/Systems Analyst	42	3,78	0,51	0,358	0,784
	Software Expert	61	3,75	0,51		
	Manager	17	3,82	0,48		
	Other	25	3,86	0,48		
Project Requirements	Business/Systems Analyst	42	3,89	1,02	0,226	0,879
	Software Expert	61	3,75	1,06		
	Manager	17	3,69	1,11		
	Other	25	3,80	0,69		
Efficiency Quality and	Business/Systems Analyst	42	4,25	0,41	1,597	0,193
	Software Expert	61	4,03	0,57		
	Manager	17	4,08	0,40		
	Other	25	4,12	0,46		
Team Competence	Business/Systems Analyst	42	3,98	0,96	0,882	0,452
	Software Expert	61	3,75	0,96		
	Manager	17	4,09	0,64		
	Other	25	3,88	0,70		
Time and Time Constraint	Business/Systems Analyst	42	3,62	0,79	0,211	0,888
	Software Expert	61	3,53	0,92		
	Manager	17	3,44	1,06		
	Other	25	3,60	0,65		
Waterfall Model and Agile Approach in Software Projects	Business/Systems Analyst	42	3,88	0,37	0,736	0,532
	Software Expert	61	3,77	0,42		
	Manager	17	3,80	0,40		
	Other	25	3,85	0,25		

H₈: The waterfall model and agile approach of the participants in software projects differ according to the position groups worked.

According to the results of the anova test to determine whether the differences between the waterfall model and agile approach in the software projects of the participants are significant according to the position groups; It was determined that the difference between the position groups in which the waterfall model and agile approach were studied in the software projects of the participants did not show statistical significance. ($p>0.05$).

Table 21

Anova Test Results for Determining the Differentiation of the Waterfall Model and Agile Approach of the Participants in Software Projects by Firm Sector Groups

Sub-Dimensions	Company Sector	n	$\bar{X}$	s.s	F	p	Scheffe
Teamwork and Motivation	Banking/Finance	29	3,76	0,47	0,729	0,536	
	Airline/Aviation	56	3,74	0,53			
	Production	15	3,73	0,28			
	Other	45	3,88	0,53			
Project Requirements	Banking/Finance	29	4,00	0,96	1,856	0,140	
	Airline/Aviation	56	3,77	0,85			
	Production	15	4,13	0,94			
	Other	45	3,56	1,16			
Efficiency and Quality	Banking/Finance ⁽¹⁾	29	4,15	0,52	4,676	0,004*	(3-1) (3-2) (3-4)
	Airline/Aviation ⁽²⁾	56	4,04	0,50			
	Production ⁽³⁾	15	4,53	0,48			
	Other ⁽⁴⁾	45	4,04	0,41			
Team Competence	Banking/Finance	29	3,84	0,98	0,426	0,735	
	Airline/Aviation	56	3,92	0,85			
	Production	15	4,07	0,86			
	Other	45	3,79	0,90			
Time and Time Constraint	Banking/Finance	29	3,53	0,84	0,392	0,759	
	Airline/Aviation	56	3,48	0,94			
	Production	15	3,57	0,82			
	Other	45	3,67	0,78			
Waterfall Model and Agile Approach in	Banking/Finance	29	3,84	0,44	1,051	0,372	
	Airline/Aviation	56	3,79	0,37			
	Production	15	3,97	0,34			

Software Projects	Other	45	3,79	0,36
-------------------	-------	----	------	------

p<0,05*

H₉: The waterfall model and agile approach of the participants in software projects differ according to company sector groups.

According to the results of the anova test to determine whether the differences between the waterfall model and agile approach in the software projects of the participants are significant according to the company sector groups; It was determined that the difference between productivity and quality perceptions according to company sector groups was statistically significant at the 95% confidence level. (F= 4.676; p=0.004; p<0.05). The productivity and quality perceptions of those whose firm sector is production ($\bar{X}=4.53$) are compared to banking/finance ($\bar{X}=4.15$), airline/aviation ($\bar{X}=4.04$), and other sectors ($\bar{X}=4.04$).) is higher than When the results were examined, it was determined that the H₉ hypothesis was accepted in the productivity and quality sub-dimension. The source of the difference between the groups was determined by the Scheffe test, one of the posthoc tests.

Table 22

Anova Test Reslts for Determining the Differences of the Participants' Waterfall Model and Agile Approach in Software Projects According to Average Project Duration Group

Sub-Dimensions	Average Duration	Project n	$\bar{X}$	s.s	F	p	Scheffe
Teamwork and Motivation	1-3 months	55	3,75	0,59	0,310	0,734	
	4-6 months	45	3,83	0,44			
	7 months and above	45	3,78	0,43			
Project Requirements	1-3 months	55	3,78	0,87	0,275	0,760	
	4-6 months	45	3,87	0,99			
	7 months and above	45	3,72	1,15			
	1-3 months	55	4,12	0,51	0,002	0,998	

Efficiency and Quality	4-6 months	45	4,12	0,54		
	7 months and above	45	4,11	0,43		
Team Competence	1-3 months	55	3,99	0,76	1,563	0,213
	4-6 months	45	3,69	1,02		
Time and Time Constraint	7 months and above	45	3,93	0,88	5,368	0,006* (1-2)
	1-3 ay ⁽¹⁾	55	3,83	0,75		
	4-6 ay ⁽²⁾	45	3,29	0,88		
	7 months and above ⁽³⁾	45	3,50	0,87		
Waterfall Model and Agile Approach in Software Projects	1-3 months	55	3,85	0,41	0,380	0,685
	4-6 months	45	3,79	0,37		
	7 months and above	45	3,82	0,35		

p<0,05*

H₁₀: The waterfall model and agile approach of the participants in software projects differ according to the average project duration groups.

According to the results of the anova test to determine whether the differences between the waterfall model and agile approach in software projects of the participants are significant according to the average project duration groups; It was determined that the difference of time and time constraint perceptions according to the average project duration groups was statistically significant at the 95% confidence level. (F= 5.368; p=0.006; p<0.05). Time and time constraint perceptions of those whose average project sector is 1-3 months ($\bar{X}=3.83$) are higher than those whose average project sector is 4-6 months ($\bar{X}=3.29$). When the results were examined, it was determined that the H₁₀ hypothesis was accepted in the time and time constraint sub-dimension. The source of the difference between the groups was determined by the Scheffe test, one of the posthoc tests.

Table 23

Anova Test Results for Determining the Differences of the Participants' Waterfall Model and Agile Approach in Software Projects According to the Average Number of People Working on a Software Project Groups

Sub-Dimensions		Average Number of People Working on a Software Project	n	$\bar{X}$	s.s	F	p
Teamwork and Motivation		1-7 people	69	3,73	0,51	0,847	0,431
		8-15 people	30	3,79	0,40		
		16 people and above	46	3,86	0,54		
Project Requirements		1-7 people	69	3,84	0,94	0,213	0,809
		8-15 people	30	3,77	1,06		
		16 people and above	46	3,72	1,05		
Efficiency and Quality		1-7 people	69	4,14	0,52	0,176	0,839
		8-15 people	30	4,12	0,49		
		16 people and above	46	4,08	0,47		
Team Competence		1-7 people	69	3,87	0,84	0,199	0,820
		8-15 people	30	3,97	0,94		
		16 people and above	46	3,84	0,94		
Time and Time Constraint		1-7 people	69	3,63	0,87	0,570	0,567
		8-15 people	30	3,55	0,72		
		16 people and above	46	3,46	0,91		
Waterfall Model and Agile Approach in Software Projects		1-7 people	69	3,83	0,36	0,111	0,895
		8-15 people	30	3,82	0,41		
		16 people and above	46	3,80	0,40		

H₁₁: The waterfall model and agile approach of the participants in software projects differ according to the Average Number of People Working on a Software Project.

According to the results of the anova test to determine whether the differences between the waterfall model and agile approach in software projects of the participants are significant according to the Average Number of People Working on a Software

Project; It was determined that the waterfall model and agile approach in software projects of the participants did not show statistically significant difference according to the Average Number of People Working on a Software Project. ($p>0.05$).

Table 24

Anova Test Results for Determining the Differences of the Participants' Waterfall Model and Agile Approach in Software Projects According to the Number of Employees of the Largest Software Project

Sub-Dimensions		Largest Software Project Number of Employees	n	$\bar{X}$	s.s	F	p
Teamwork and Motivation		1-10 people	58	3,75	0,49	0,968	0,410
		11-20 people	35	3,89	0,54		
		21-30 people	32	3,81	0,42		
Project Requirements		31 people and above	20	3,67	0,55	1,800	0,150
		1-10 people	58	3,78	0,99		
		11-20 people	35	4,01	0,79		
		21-30 people	32	3,83	1,05		
Efficiency and Quality		31 people and above	20	3,38	1,18	0,508	0,677
		1-10 people	58	4,07	0,53		
		11-20 people	35	4,18	0,49		
Team Competence		21-30 people	32	4,16	0,45	0,704	0,551
		31 people and above	20	4,07	0,50		
		1-10 people	58	3,93	0,85		
Time and Time Constraint		11-20 people	35	3,80	1,05	0,225	0,879
		21-30 people	32	4,00	0,75		
		31 people and above	20	3,68	0,91		
Waterfall Model and Agile Approach in Software Projects		1-10 people	58	3,59	0,88	2,488	0,063
		11-20 people	35	3,61	0,75		
		21-30 people	32	3,81	0,42		

H₁₂: The waterfall model and agile approach of the participants in software projects differ according to the largest software project employee number groups.

According to the results of the anova test to determine whether the differences between the waterfall model and agile approach in software projects of the participants are significant according to the largest software project employee number groups; It was determined that the waterfall model and agile approach in software projects of the participants did not show statistically significant difference according to the number of employees in the largest software project. ($p>0.05$).

Table 25

Customer Satisfaction Normal Distribution Analysis Results of Participants

Sub-Dimensions	Kolmogrov-Smirnov			Central Tendency Measurements			
	Statistic	Sd	p	$\bar{X}$	Median	Skewness	Kurtosis
Customer Satisfaction	0,274	145	0,000	4,17	4,00	-1,424	2,310

As a result of the non-normal distribution analysis, it was determined that the customer satisfaction perceptions did not come from the normal distribution ($p<0.05$). The analysis was continued with non-parametric tests.

Table 26

Descriptive Findings of Participants' Customer Satisfaction

Sub-Dimensions	$\bar{X}$	s.s.
Customer Satisfaction	4,17	0,74

When the descriptive findings of the participants' customer satisfaction are examined, the participants; sub-dimension mean ($\bar{x}=4.17$) was determined to be high.

Table 27

Mann-Whitney U Analysis Results of the Differences in Customer Satisfaction of the Participants by Gender Variable

Sub-Dimensions	Gender	n	Mean Rank	U	p
Customer Satisfaction	Female	53	72,77	2426,0	0,959
	Male	92	73,13		

H₁₃: Customer satisfaction of the participants differs according to gender groups.

According to the results of the Mann-Whitney U test, which was conducted to test whether the difference in the customer satisfaction perceptions of the participants according to the gender variable was statistically significant, it was determined that the difference in the customer satisfaction perceptions according to the gender variable was not statistically significant at the 95% confidence level ($p > 0.05$).

Table 28

Mann-Whitney U Analysis Results of the Differences in Customer Satisfaction According to the Variable of Educational Status of the Participants

Sub-Dimensions	Educational Status	n	Mean Rank	U	p
Customer Satisfaction	Associate and Bachelor's degree	103	77,68	1680,50	0,028*
	Master's degree and above	42	61,51		

$p < 0,05^*$

H₁₄: The customer satisfaction of the participants differs according to the educational background groups.

According to the results of the Mann-Whitney U test to test whether the difference is statistically significant according to the educational status variable of the customer satisfaction of the participants; It was determined that the difference of

customer satisfaction according to the educational status groups was statistically significant at the 95% confidence level. ($U= 1680.5$; $p=0.028$; $p<0.05$). Customer satisfaction perceptions of those with an associate degree or undergraduate degree (Mean Rank=77.68) are higher than those with a master's degree or higher (Mean Rank=61.51). When the results were examined, it was determined that the H14 hypothesis was accepted in the customer satisfaction scale subject.

Table 29

Mann-Whitney U Analysis Results of the Differences in Customer Satisfaction of the Participants according to the Agile Process Model Variable Used

Sub-Dimensions	Agile Model	Process n	Mean Rank	U	p
Customer Satisfaction	Scrum	128	71,03	836,00	0,105
	XP	17	87,82		

H15: Customer satisfaction of the participants differs according to the agile process model groups used.

According to the results of the Mann-Whitney U test, which was conducted to test whether the difference in customer satisfaction of the participants was statistically significant according to the variable of the agile process model used, it was determined that the difference in customer satisfaction according to the variable of the agile process model used was not statistically significant at the 95% confidence level ($p>0.05$).

Table 30

Mann-Whitney U Analysis Results of the Differences in Customer Satisfaction of the Participants according to the Variable of the Number of Persons Working in the Software Project Team

Sub-Dimensions	Number of People Working in the Software Project Team	n	Mean Rank	U	p
Customer Satisfaction	7 people and below	62	70,10	2393,5	0,453
	7 people and above	83	75,16	0	

H₁₆: The customer satisfaction of the participants differs according to the number of people working in the software project team.

According to the results of the Mann-Whitney U test, which was conducted to test whether the difference of the participants' customer satisfaction according to the variable of the number of people working in the software project team is statistically significant, it was determined that the difference of customer satisfaction according to the variable of the number of people working in the software project team was not statistically significant at the 95% confidence level ($p > 0.05$).

Table 31

Kruskal-Wallis H Analysis Results of the Differences in Customer Satisfaction of Participants by Total Work Experience Variable. Non-Normal Answer Of Anova Test

Sub-Dimensions	Total Work Experience	n	Mean Rank	χ^2	p	U
Customer Satisfaction	0-5 years ⁽¹⁾	34	80,72	13,155	0,004*	(1-3)
	6-10 years ⁽²⁾	65	78,31			
	11-15 years ⁽³⁾	34	51,35			
	16 years and above ⁽⁴⁾	12	83,71			

* $p < 0,05$

H₁₇: The customer satisfaction of the participants differs according to the total work experience groups.

According to the results of the Kruskal-Wallis H test to test whether the difference is statistically significant according to the total work experience variable of the participants' customer satisfaction; The difference between customer satisfaction

and total work experience was statistically significant at the 95% confidence level ($\chi^2=13.155$; $p=0.004$, $p<0.05$). According to the results of the Mann-Whitney U test, which was conducted to determine between which groups the perception of customer satisfaction differs, those with a total work experience of 16 years or more (Mean Rank=83,71) compared to those with a customer satisfaction perception of 0-5 years (Mean Rank=80,72). higher than those. When the results were examined, it was determined that the H17 hypothesis was accepted in the sub-dimension of customer satisfaction. The difference between the groups was determined by the Mann-Whitney U test.

Table 32

Kruskal-Wallis H Analysis Results of the Differences in Customer Satisfaction of the Participants according to the Variable of Work Experience in the Industry

Sub-Dimensions	Work Experience in the Industry	n	Mean Rank	χ^2	p	U
Customer Satisfaction	0-5 years ⁽¹⁾	43	77,69	12,588	0,006*	(1-3)
	6-10 years ⁽²⁾	61	79,54			
	11-15 years ⁽³⁾	31	50,47			
	16 years and above ⁽⁴⁾	10	82,8			

* $p<0,05$

H18: The customer satisfaction of the participants differs according to the work experience groups in the sector.

According to the results of the Kruskal-Wallis H test to test whether the difference is statistically significant according to the variable of the participants' customer satisfaction and work experience in the sector; The difference between customer satisfaction and work experience in the sector was statistically significant at the 95% confidence level ($\chi^2=12.588$; $p=0.006$, $p<0.05$). According to the results of the Mann-Whitney U test, which was conducted to determine between which groups the

perception of customer satisfaction differs, those with 0-5 years of work experience in the sector (Mean Rank=77.69) compared to those with 11-15 years of customer satisfaction perception (Mean Rank=50.47). higher than those. When the results were examined, it was determined that the H18 hypothesis was accepted in the customer satisfaction scale subject.

Table 33

Kruskal-Wallis H Analysis Results of the Differences in Customer Satisfaction of the Participants according to the Variable of Work Experience in the Institution

Sub-Dimensions	Work Experience in the Institution	n	Mean Rank	χ^2	p
Customer Satisfaction	0-5 years	105	73,68	0,759	0,684
	6-10 years	26	74,98		
	11 years and above	14	64,25		

H₁₉: The customer satisfaction of the participants differs according to the work experience groups in the institution.

According to the results of the Kruskal-Wallis H test, which was conducted to test whether the difference in customer satisfaction of the participants is statistically significant according to the work experience variable in the institution; It was determined that the difference between the customer satisfaction of the participants according to the work experience groups in the institution was not statistically significant. ($p>0.05$).

Table 34

Kruskal-Wallis H Analysis Results of the Differences in Customer Satisfaction of the Participants according to the Variable of Position

Sub-Dimensions	Position	n	Mean Rank	χ^2	p
Customer Satisfaction	Business/Systems Analyst	42	77,71	1,321	0,724
	Software Developing Expert	61	72,30		
	Manager	17	64,76		
	Other	25	72,38		

H₂₀: The customer satisfaction of the participants differs according to the position groups worked.

According to the results of the Kruskal-Wallis H test, which was conducted to test whether the difference in customer satisfaction of the participants is statistically significant according to the position variable; It was determined that the customer satisfaction of the participants did not show statistically significant difference according to the position groups worked. ($p > 0.05$).

Table 35

Kruskal-Wallis H Analysis Results of the Differences in Customer Satisfaction of the Participants according to the Variable of the Firm Sector

Sub-Dimensions	Worked Firm Sector	n	Mean Rank	χ^2	p	U
Customer Satisfaction	Banking/Finance ⁽¹⁾	29	55,36	15,254	0,002*	(1-4)
	Airline/Aviation ⁽²⁾	56	79,54			
	Production ⁽³⁾	15	100,70			
	Other ⁽⁴⁾	45	67,00			

* $p < 0,05$

H₂₁: The customer satisfaction of the participants differs according to the company sector groups.

According to the results of the Kruskal-Wallis H test, which was carried out to test whether the difference in customer satisfaction of the participants is statistically significant according to the variable of the firm sector; It was determined that the

difference in customer satisfaction according to the sector of the company was statistically significant at the 95% confidence level ($\chi^2=15.254$; $p=0.002$, $p<0.05$). According to the results of the Mann-Whitney U test, which was conducted to determine between which groups the perception of customer satisfaction differs, customer satisfaction perceptions of those whose company sector is production (Mean Rank = 100.70) are higher than those with banking/finance (Mean Rank = 55.36). When the results were examined, it was determined that the H21 hypothesis was accepted in the customer satisfaction scale subject.

Table 36

Kruskal-Wallis H Analysis Results of the Differences in Customer Satisfaction of the Participants by Average Project Duration Variable

Sub-Dimensions	Average Duration	Project n	Mean Rank	χ^2	p
Customer Satisfaction	1-3 months	55	75,88	1,016	0,602
	4-6 months	45	68,08		
	7 months and above	45	74,40		

H22: The customer satisfaction of the participants differs according to the average project duration groups.

According to the results of the Kruskal-Wallis H test, which was conducted to test whether the difference of the participants' customer satisfaction is statistically significant according to the average project duration variable; It was determined that the difference between the customer satisfaction of the participants according to the average project duration groups was not statistically significant. ($p>0.05$).

Table 37

Kruskal-Wallis H Analysis Results of the Differences in Customer Satisfaction of the Participants according to the Variable of the Average Number of Persons Working in the Software Project

Sub-Dimensions	Number of People Working on Average Software Project	n	Mean Rank	χ^2	p
Customer Satisfaction	1-7 people	69	72,77	0,209	0,901
	8-15 people	30	70,63		
	16 people and above	46	74,89		

H₂₃: The customer satisfaction of the participants differs according to the number of people working in the average software project.

According to the results of the Kruskal-Wallis H test, which was conducted to test whether the difference in customer satisfaction of the participants according to the variable of the average number of people working in the software project is statistically significant; It was determined that the difference between the customer satisfaction of the participants and the number of people working in the average software project was not statistically significant. ($p > 0.05$).

Table 38

Kruskal-Wallis H Analysis Results of the Differences in Customer Satisfaction of the Participants according to the Variable of the Number of Employees in the Largest Software Project

Sub-Dimensions	Number of People Working on Largest Software Project	n	Mean Rank	χ^2	p
Customer Satisfaction	1-10 people	58	70,01	5,062	0,167
	11-20 people	35	81,16		

21-30 people	32	78,64
31 people and above	20	58,38

H24: The customer satisfaction of the participants differs according to the number of people working in the largest software project.

According to the results of the Kruskal-Wallis H test, which was conducted to test whether the difference was statistically significant according to the variable of the number of people working in the largest software project in the customer satisfaction of the participants; It was determined that the difference in the customer satisfaction of the participants according to the number of people working in the largest software project was not statistically significant. ($p>0.05$).

Table 39

Reliability Analysis Based on Waterfall Model and Agile Approach Scale and Its Sub-Dimensions in Software Projects

Scale and Sub-Dimensions	Cronbach's Alpha	Madde Sayısı
Teamwork and Motivation	0,828	5
Project Requirements	0,961	4
Efficiency and Quality	0,793	3
Team Competence	0,915	2
Time and Time Constraint	0,862	2
Customer Satisfaction	0,871	2
Waterfall Model and Agile Approach in Software Projects	0,793	21

Participants; When the reliability coefficients of the waterfall model and agile approach scale in software projects consisting of 21 items are examined, it is seen that the Cronbach's Alpha value of the scale was calculated as 0.793. This value indicates that the reliability of the scale is high. When the reliability coefficients of the 5-item

teamwork and motivation sub-dimension are examined, it is seen that the Cronbach's Alpha value of the sub-dimension is calculated as 0.828. This value indicates that the reliability of the sub-dimension is very high. When the reliability coefficients of the project requirements sub-dimension consisting of 4 items are examined, it is seen that the Cronbach's Alpha value of the sub-dimension is calculated as 0.961. This value indicates that the reliability of the sub-dimension is very high. When the reliability coefficients of the productivity and quality sub-dimension consisting of 3 items are examined, it is seen that the Cronbach's Alpha value of the sub-dimension was calculated as 0.793. This value indicates that the reliability of the sub-dimension is high. When the reliability coefficients of the 2-item team competence sub-dimension are examined, it is seen that the Cronbach's Alpha value of the sub-dimension is calculated as 0.915. This value indicates that the reliability of the sub-dimension is very high. When the reliability coefficients of the time and time constraint sub-dimension consisting of 2 items are examined, it is seen that the Cronbach's Alpha value of the sub-dimension was calculated as 0.862. This value indicates that the reliability of the sub-dimension is very high. When the reliability coefficients of the customer satisfaction sub-dimension consisting of 2 items are examined, it is seen that the Cronbach's Alpha value of the sub-dimension is calculated as 0.871. This value indicates that the reliability of the sub-dimension is very high.

Table 40

The Relationship Between The Waterfall Model and Agile Approach Scale and Its Sub-Dimensions in Software Projects

Scale and Sub-Dimensions		(1)	(2)	(3)	(4)	(5)	(6)	(7)
Teamwork and Motivation ⁽¹⁾	r	1	0,071	-0,002	0,117	0,139	,310*	,484*
Project Requirements ⁽²⁾	r		1	,445*	0,077	0,139	,355*	,739*

Efficiency and Quality ⁽³⁾	r	1	0,032	,252*	,419*	,578*
Team Competence ⁽⁴⁾	r		1	0,076	,437*	,381*
Time and Time Constraint ⁽⁵⁾	r			1	,292*	,408*
Customer Satisfaction ⁽⁶⁾	r				1	,730*
Waterfall Model and Agile Approach in Software Projects ⁽⁷⁾	r					1

p<0,05*

Pearson correlation analysis was performed to examine the relationship between the variables, and according to the results of the correlation analysis; In software projects, there is a positive relationship between the waterfall model and perceptions of team competence ($r=0.381$), while there is a positive relationship between teamwork and motivation ($r=0.484$), time and time constraints ($r=0.408$), and productivity and quality ($r=0.578$). moderate, there is a high level of positive correlation between project requirements ($r=0.739$) and customer satisfaction ($r=0.730$). There was a positive correlation between customer satisfaction and teamwork and motivation ($r=0.310$), project requirements ($r=0.355$), and time and time constraints ($r=0.292$). 0.437), there is a positive moderate relationship between their perceptions. There is a low positive correlation between time and time constraints and perceptions of productivity and quality ($r=0.252$). There is a moderately positive relationship between the perceptions of productivity and quality and project requirements ($r=0.445$).

6. CONCLUSION

Looking at the history of civilization from past to present, projects, planning, and management of projects are encountered. The construction of buildings, developments in the health sector, changes in information technologies can be considered separate project studies.

In the globalizing world, the more difficult competition conditions require a flexible organizational structure and the management of all units in the business in coordination. In this way, it is aimed to increase efficiency. Project management offers a significant advantage to companies in this regard. Thanks to project management, businesses are able to work with a target focus, provide high motivation, facilitate internal control and enable a significant increase in quality.

Software projects are specialized projects and carry the basic features of project management. However, there are important differences that distinguish software projects from other projects. The most important difference is that the product produced in other engineering processes is physically tangible, while the software is intangible as a manifestation of intellectual processes. Due to this intangibility, the traceability of project management processes is interrupted and the uncertainty of the project increases. Therefore, software projects require special management techniques due to their content.

The waterfall model is the oldest and fundamental model of software engineering. Today, it is accepted as the management standard of all kinds of projects by public institutions and large companies. (Munassar et al.,2010, p. 95)

Agile methods have emerged as alternative solutions for the issues that traditional methods are considered insufficient. Today, both in Turkey and around the world, agile methods are preferred in software project management. (Hewlett-Packard Enterprise,2017)

Within the scope of the study, a comparison of the agile methods and the waterfall model was made and it was tried to determine how software developers perceived agile approaches. In order to compare the result obtained with the

experimental experience, a questionnaire was applied with the participation of 145 employees from the sectors who took part in agile projects.

The survey consists of two main parts. The first part was prepared to collect demographic information about the participants. In the second part, questions were asked about the perception of agile software development. According to the survey results, SCRUM comes to the fore as the commonly used approach. The results significantly overlap with the results of the study carried out by Agile Turkey in 2019. (Agile Turkey,2019)

The attitudes of the software developers participating in the study on agile approaches are quite positive. They think that agile approaches increase efficiency, quality, and customer satisfaction. Participants with total experience and industry experience of 16 years or more have higher perceptions of Agile approach productivity and quality. In addition, according to the survey results, those who think that the Agile approach is suitable for small teams are in the majority.

Since the waterfall model is carried out depending on the past phases of the project, it is retrospective in understanding and is an order in which all project elements are expected to do the work defined for them. The waterfall model, which makes use of this layout, is more suitable for projects where the requirements and goals are clear, the solutions and the work to be done are known, the requirements will not change, the software to be produced has no fault tolerance and has functional integrity. Defense systems and embedded systems can be given as the best usage area.

Agile methods, on the other hand, have a forward-looking understanding due to their structure that encourages change and feedback. Autonomous teams are expected to do everything by themselves and benefit from cross-functionality and

experimentation in this process. This requires an approach to creativity rather than order. Therefore, agile methods are more suitable for projects where requirements and goals are uncertain and uncertain, change is frequent and value generation is desired in the short term. The best use for agile methods is web-based workflow management systems. In this direction, according to the results of the survey, the productivity and quality perceptions of those whose firm sector is the production of agile methods are higher than those who are in the banking/finance, airline/aviation, and other sectors.

The survey result evaluations were made according to the statistical findings determined in order to maintain impartiality; However, as in many experimental studies, there are important limitations in this study. First of all, the participants who answered the questionnaire could not be selected by random sampling across Turkey.

Thus, the results cannot be generalized to the whole situation in Turkey, but they can provide software developers and researchers with a perspective on general trends in Turkey. Within the scope of future studies, dissemination of the research over a longer period of time to cover a larger sample or re-application of the survey at periodic intervals will be beneficial in terms of expanding the findings.

REFERENCES

- Agile Turkey. (2019). 9th Annual Agility Report Retrieved from:
<https://www.agileturkey.org/reportopt.pdf>
- Ahmad, O. M., Markkula, J. & Ovio, M., (2013). Kanban in software development: *A systematic literature review*. Finland, 39th Euromicro Conference Series on Software Engineering and Advanced Applications. doi:10.1109/seaa.2013.28
- Ahmed, A. (2012). *Software Project Management A Process-Driven Approach*, Florida: CRC Press
- Awad, M. A. (2005). *A Comparison between Agile and Traditional Software Development Methodologies*, Published Dissertation: the University of Western Australia.
- Balaji, S., Murugaiyan, M. S. (2012), Waterfall vs. V-Model vs. Agile: A comparative study on SDLC. *International Journal of Information Technology and Business Management*. Retrieved from: www.jitbm.com
- Butler Jr. A. G. (1973). Project Management: A Study in Organizational Conflict. *The Academy of Management Journal*, 16(1), 84-101. doi:10.5465/255045
- Cervone, H. F. (2011), Understanding agile project management methods using Scrum. *OCLC Systems & Services: International Digital Library Perspectives*, 27(1), 18-22. doi: 10.1108/10650751111106528
- Cleland, D. I., Gareis R. (2006), *Global Project Management Handbook: Planning, Organizing, and Controlling International Project*, (Twelfth Edition), the United States of America: The McGraw-Hill Companies

- Cockburn A.(2000-2001). *Agile Software Development* (Draft version:3b).
Highsmith Series Editors.
- Farrell, A. (2007). Selecting a Software Development Methodology based on
Organizational Characteristics: *An Essay Submitted in Partial Fulfillment
of the Requirements for the Degree of “Master of Science in Information
Systems”*, Athabasca University, Alberta
- Flora, H. K., Chande, S. V. (2014). A Systematic Study on Agile Software
Development Methodologies and Practices. *International Journal of
Computer Science and Information Technologies*, 5(3), ISSN: 0975-9646
- Gechman M. (2019). *Project Management of Large Software-Intensive Systems
Controlling the Software Development Process*, Florida: CRC Press
- Heldman K. (2018). *PMP Project Management Professional Exam Study Guide*,
Indiana: Sybex
- Hewlett-Packard Enterprise. (2017). Agile is the new normal: *Adopting Agile
project management*. Retrieved from:
<https://softwaretestinggenius.com/docs/4aa5-7619.pdf>
- Highsmith, J. (2000). *Adaptive Software Development: A Collaborative Approach
to Managing Complex Systems*, Dorset House.
- Horine, G.(2009). *Absolute Beginner’s Guide to Project Management*, (Second
Edition), Indiana: Que Publishing
- IEEE Computer Society. (2014). *SWEBOK: Guide to the Software Engineering
Body of Knowledge*, USA: IEEE
- Kerzner, H. (2017). *Project Management a Systems Approach to Planning,
Scheduling, and Controlling*, (Twelfth Edition), Hoboken, New Jersey:
Wiley

- Lock, D. (2007). *Project Management*, (Ninth Edition), Hampshire: Gower Publishing
- Moran, A. (2015). *Managing Agile. Strategy, Implementation, Organisation, and People*, Zurich: Springer
- N.M.A., Govardhan, A. (September 01, 2010). A Comparison Between Five Models Of Software Engineering. *International Journal of Computer Science Issues*, 7, 5, 94-101. Packard Enterprise, U.S.A.
- Özdemir, E., (2017). Waterfall Yazılım Geliştirme. Retrieved from: <http://erenozdemir.net/waterfall-yazilim-gelistirme/>
- Palmquist, M. S., Lapham, M. A., Miller, S., Chick, T., & Ozkaya, I. (2013). Parallel worlds: Agile and waterfall differences and similarities. *Carnegie-Mellon University Pittsburgh Pa Software Engineering Inst.* (No. CMU/SEI-2013-TN-021).
- Poppendieck M., Poppendieck T. (2013). *The Lean Mindset Ask the Right Questions*, Massachusetts: Addison-Wesley Professional
- Pressman R. S., Maxim B. R. (2019). *Software Engineering: A Practitioner's Approach* (9th Edition), New York: Mc Graw Hill
- Project Management Institute (PMI). (2000), *A Guide To The Project Management Body Of Knowledge (PMBOK GUIDE)* (2000 edition), Pennsylvania: PMI
- Project Management Institute (PMI). (2013), *Software Extens, on to the PMBOK Guide* (6th Edition), Pennsylvania: PMI

- Project Management Institute (PMI). (2017), *A Guide To The Project Management Body Of Knowledge (PMBOK GUIDE)*(6th Edition), Chicago: PMI
- Project Management Institute (PMI). (2017). Agile Practice Guide (PMBOK GUIDE), (First Edition), Pennsylvania: PMI
- Rajagopalan , S.,Mathew, S. K. (2016). Choice of Agile Methodologies in Software Development. *International Information Management Association*, 25(ISSN:1941-6679-Online Copy)
- Ruhe, G.,Wohlin, C. (2014). *Software Project Management in a Changing World*, Berlin:Springer
- Sergeev, A., (2016), Team Roles in Waterfall Methodology. Retrieved from: <https://hygger.io/blog/team-roles-in-waterfall-methodology/>
- Schwaber K., Sutherland J. (2017), The Scrum Guide, The Definitive Guide to Scrum: The Rules of the Game. Retrieved from: www.scrum.org
- Schwaber K., Sutherland J. (2020), The Scrum Guide, The Definitive Guide to Scrum: The Rules of the Game. Retrieved from: www.scrum.org
- Slack, N., Chambers, S., Johnston, R. (2007). *Operations Management*, (Fifth Edition), London: Financial Times/Prentice Times
- Sommerville I. (2007). *Software Engineering 8*, Harlow: Pearson Education Limited, Eighth edition
- Stanley, J. C., Gross, E. D. (2020). *Project Management Handbook: An Introduction to Scrum, Agile and DevOps for Absolute Beginners*, Prosper Consulting Inc.

- Stepanek, G. (2005). *Software Project Secrets: Why Software Projects Fail*, Berkeley: Apress.
- VersionOne Inc. (2020). 14th Annual State of Agile Report, U.S.A.:Analysis.Net Research. Retrieved from: <https://digital.ai/resource-center/analyst-reports/state-of-agile-report>
- Villafiorita A. (2014). *Introduction to Software Project Management*, Florida: CRC Press
- Vohra, P., Singh, A. (2013). A Contrast and Comparison of Modern Software Process Models. *International Conference on Advances in Management and Technology*. (Icamt 2013). 23-27.
- Weiss, J. W. , Wysocki, R. K.(1992). *5-Phase Project Management A Practical Planning&Implementation Guide*, The United States of America: Perseus Books Publishing