March-2022

Agile Beeswax: Agile Development Methodology for Mobile Applications

UNIVERSIDAD DE GRANADA

Doctoral Program in Information and Communication Technologies

Doctoral thesis

Presented By

Hazem Abdelkarim Mahmoud Alrabaiah

Supervised By

Prof. Nuria Medina-Medina

Department of Languages and Computer Systems, School of Computer Science and Telecommunication Engineering,

Granada-Spain

UNIVERSIDAD DE GRANADA

Doctoral Program in Information and Communication Technologies

Agile Beeswax: Agile Development Methodology for Mobile Applications

Doctoral thesis

Presented By

Hazem Abdelkarim Mahmoud Alrabaiah

Supervised By

Prof. Nuria Medina-Medina

Department of Languages and Computer Systems, School of Computer Science and Telecommunication Engineering, Granada University

Granada-Spain

March 29, 2022

Editor: Universidad de Granada. Tesis Doctorales
Autor: Hazem Abdelkarim Mahmoud Alrabaiah
ISBN: 978-84-1117-357-5
URI: http://hdl.handle.net/10481/75443

Table of Contents

List of Figures

Chapter 1

Chapter2

Chapter 3

Chapter 4

Chapter 5

Chapter 6

List of Tables

Chapter 2

Chapter 3

Chapter 4

Chapter 5

Chapter 6

DEDICATION

To everyone I met and teach me something

ABSTRACT

Mobile applications have seen great development in recent years, and with this growth, tools, phones, and methods of developing these applications have evolved. However, the methods of developing these mobile applications have not seen the same growth as its usage. Software engineering research on mobile application development methodologies is not progressing at the same rate as the adoption of mobile applications. Only a few mobile application development methodologies have been presented in the scientific literature, especially agile ones. In addition, mobile application idea and concept workshops, requirements gathering, User Interface, User Experience, deployment, maintenance, complexity of testing, power consumption, and project assessment activities receive very little attention from existing methodologies. Moreover, in the current proposals are not sufficiently handling the special limitations for mobile applications, such as providing the necessary features to facilitate and support users' participation in the development process. After extensive study in academia and industry, in our opinion it indisputable that the research into the mobile application development process must continue to grow.

Mobile application development is a highly competitive environment, and in our opinion, agile methodologies can enable teams to provide value faster, with higher quality and predictability. The development of mobile applications has unique requirements, and agile methods can deal with some of these requirements, such as the continuous change in mobile applications requirements or the continuous participation of users. An efficient development process may assist increase competitive advantage and decreasing release cycles. For this reason, our objective has been to review the existing methodologies and models for developing mobile applications in the scientific literature and real methodologies adopted by experts in the development communities since this will help us address the main practices in the mobile application development process.

Based on a defined and appropriate frame, an analysis of these models and their usefulness to the industry has been performed to create a new methodology for developing mobile applications that suit academic and industry communities. This new methodological process based on agile methodologies for mobile application development has been named Agile Beeswax. Thus, Agile Beeswax is conceived after identifying the mobile development process's issues, challenges, and unique requirements. Agile Beeswax is defined as an integrated incremental, iterative development process for developing mobile applications.

One of its main strengths is that it has been created with academic and business perspectives to bring these two communities closer. Agile Beeswax tried to integrate different methodologies and practices in the development process to obtain an integrated method. We combined some scrum management practices, software engineering practices, and operational practices into one methodology. To achieve our purpose, the work has been divided into five main phases: Phase 1: A systematic literature review approach to review existing mobile application development methods. Phase 2: Interviews with mobile application developers working in small to medium software companies. Phase 3: Survey to a group of 35 experts, including academics and developers, to extract valuable knowledge about mobile development. Phase 4: Proposal of a new methodology for mobile application development. Phase 5: Validation of the proposed methodology using a second group of 35 experts, including mobile application developers and academic communities (some of them participated in the first survey).

Conclusion: We need an effective and practical methodology for mobile application development. An efficient development methodology may assist increase competitive advantage and decreasing release cycles, which is critical in the mobile application development process. The results in this thesis and the proposed methodology for developing mobile applications are intended to serve as support for mobile application developers.

RESUMEN

Las aplicaciones móviles han experimentado un gran desarrollo en los últimos años y, con este crecimiento, han evolucionado también las herramientas, los dispositivos y los métodos para desarrollar estas aplicaciones. Sin embargo, las metodologías de desarrollo de estas aplicaciones móviles no han experimentado el mismo crecimiento que su uso. La investigación en ingeniería de software sobre metodologías de desarrollo de aplicaciones móviles no ha avanzado al mismo ritmo que la adopción de aplicaciones móviles. Solo unas pocas metodologías de desarrollo de aplicaciones móviles se han presentado en la literatura científica, especialmente si nos centramos en metodologías ágiles específicas para desarrollo móvil. Además, los talleres de ideas y conceptos para crear la application, la recopilación de requisitos, el diseño de la interfaz de usuario y la experiencia del usuario, la implementación, el mantenimiento, la complejidad de las pruebas, el consumo de energía y las actividades de evaluación de proyectos de desarrollo móvil reciben poca atención en las metodologías existentes. Paralelamente, los métodos de desarrollo actuales no manejan suficientemente las limitaciones especiales que presetan las aplicaciones móviles, tales como facilitar las características necesarias para facilitar y soportar la participación de los usuarios en el proceso de desarrollo. Después, del exhaustivo estudio realizado en el mundo académico e industrial acerca del desarrollo de aplicaciones móviles, en nuestra opinión es indiscutible que la investigación en el proceso de desarrollo de aplicaciones móviles debe seguir creciendo.

El desarrollo de aplicaciones móviles es un entorno altamente competitivo y, en nuestra opinión, las metodologías ágiles pueden permitir que los equipos generen valor más rápido, con mayor calidad y previsibilidad. El desarrollo de aplicaciones móviles tiene requisitos únicos y los métodos ágiles pueden abordar algunos de estos requisitos, como el cambio continuo en los requisitos de las aplicaciones móviles o la integración de los usuarios durante todo el proceso. Un proceso de desarrollo eficiente puede ayudar a aumentar la ventaja competitiva de los productos móviles y disminuir sus ciclos de lanzamiento. Por ello, nuestro objetivo es revisar las metodologías y modelos existentes para el desarrollo de aplicaciones móviles en la literatura científica y metodologías reales adoptadas por expertos en las comunidades de desarrollo ya que esto nos ayudará a identificar y dirigir las principales prácticas en el proceso de desarrollo de aplicaciones móviles.

Basado en un marco previamente definido, se ha realizado un análisis de estos modelos y su utilidad para la industria para crear una nueva metodología para desarrollar

aplicaciones móviles que se adapten a las comunidades académicas y de la industria. Este nuevo proceso metodológico basado en metodologías ágiles para el desarrollo de aplicaciones se ha denominado Agile Beeswax. Por lo tanto, Agile Beeswax se concibe después de identificar los problemas, desafíos y requisitos únicos del proceso de desarrollo móvil, y se define como un proceso de desarrollo iterativo e incremental integrado para desarrollar aplicaciones móviles. Una de sus principales fortalezas es que ha sido creado con perspectivas académicas y empresariales para acercar a estas dos comunidades. Además, Agile Beeswax ha intentado integrar diferentes metodologías y prácticas en el proceso de desarrollo para obtener un método integrado. Concretamente, combinamos algunas prácticas de gestión de scrum, prácticas de ingeniería de software y prácticas operativas en una sola metodología. Para lograr nuestro propósito, el estudio se ha dividido en cinco fases principales: Fase 1: Un enfoque de revisión sistemática de la literatura para revisar los métodos de desarrollo de aplicaciones móviles existentes. Fase 2: Entrevistas con desarrolladores de aplicaciones móviles que trabajan en pequeñas y medianas empresas de software. Fase 3: Encuesta a un grupo de 35 expertos, incluidos académicos y desarrolladores, para extraer conocimientos valiosos sobre el desarrollo móvil. Fase 4: Propuesta de una nueva metodología para el desarrollo de aplicaciones móviles. Fase 5: Validación de la metodología propuesta utilizando un segundo grupo de 35 expertos, entre desarrolladores de aplicaciones móviles y comunidades académicas (algunos de ellos participaron en la primera encuesta).

Conclusión: Necesitamos una metodología efectiva y práctica para el desarrollo de aplicaciones móviles. Una metodología de desarrollo eficiente puede ayudar a aumentar la ventaja competitiva y disminuir los ciclos de lanzamiento, lo cual es fundamental en el proceso de desarrollo de aplicaciones móviles. Los resultados de esta tesis y la metodología propuesta para el desarrollo de aplicaciones móviles pretenden servir de apoyo a los desarrolladores de aplicaciones móviles.

ACKNOWLEDGEMENTS

To my parents Abdelkarim and Ayshah and my sisters and brothers Jumanah, Hussam, Ibtisam, Reem, and Muhammad. And thanks to my supervisor Prof. Nuria Medina-Medina.

AC RONYMS

A

AM: Agile Modeling

App: Mobile Application

App Dev: App Developer

ASD: Adaptive Software Development

AWS: Amazon Web Services

C

CMS: Content Management System

CSS: Cascading Style Sheets

D

DSDM: Dynamic Systems Development

DSM: Daily Scrum Meeting

DSSM: Daily Scrum of Scrums Meeting

E

ERP: systems Enterprise Resource Planning

F

FDD: Feature Driven Development

I

IT: Team Information Technology Team

L

LD: Lean Development

M

MAD: Mobile Application Development

MASAM: Mobile Application Software Agile Methodology

P

PHP: Hypertext Preprocessor

PM: Project Manager

PO: Product Owner

Q

QA: team Quality Assurance Team

R

RAD: Rapid Application Development

ROI: Return on Investment

ROI: Rational Unified Process

S

SDM: Software development methodology

SMART: (Simple, Meaningful, Adequate, Realistic, and Tractable)

SPM: Sprint Planning Meeting

SQL: Structured Query Language

SRM: Sprint Review Meeting

U

UCD: User Centre Design

UI: User Interface

UX: User Interface

X

XP: Extreme Programming

CHAPTER 1. INTRODUCTION

The availability of high-speed Internet, along with high-speed devices, and ease of use, has boosted the demand for the mobile application's development. There is a pressing need to build new mobile applications to make our everyday lives easier. We have mobile applications for advertising, communication, education, cooking, shopping, and so on at our fingertips. There are no limits to what mobile devices can perform. With the increase of these mobile applications, the need to increase the suitability of the development method of these mobile applications increases as well. This chapter will present an introduction to our thesis and research that covers the development process for mobile applications. This chapter will contain an introduction, problem statement, aims and objectives, research questions, the scope of the research, research methodology, research structure and phases, and the structure of the thesis.

1.1 Introduction to the Research

Smartphones, tablets, and apps have grown commonplace in our daily lives and have become part of our lexicon. Due to the opportunities offered by mobile phones and mobile software, often known as “apps”, it has become the focus of attention for the advantages that it offers to the users (Altaleb, & Gravell, 2019). According to Ceci (2021), there were 230 billion downloads of mobile apps worldwide in 2021, with 581 USD billion in sales in 2020 (Collado-Borrel et al.,2020). Furthermore, according to Ceci (2021), the Android Play Store had 3.48 million apps in the 1st quarter of 2021 and the Apple App Store nearly 2.22 million apps for iOS. Furthermore, (Cheney & Thompson, 2018) mentioned that by 2022, the total number of app downloads would reach 258 billion. The development and updating of mobile applications have led to the growth of mobile application developers and the establishment of new mobile development companies.

All these mobile applications, if developed by professionals, must follow a specific process of developing mobile applications. Mobile app development is becoming a practical topic in software engineering because apps are becoming more sophisticated and moving to more business-essential purposes. It will be critical to identify the most appropriate and relevant mobile software engineering processes, as the demand for mobile apps has pushed developers to use a variety of mobile software engineering methods (Corral et al., 2013).

Developers are prepared to install and operate mobile applications due to the growing popularity of mobile application development (Altuwaijri & Ferrario, 2021). However, as the number of methods and tools available increases, development teams must limit themselves to using best practices to develop their mobile applications to reduce development time, workload, and cost. The rise and growth of mobile applications and the specific features and requirements of mobile software development impose certain limitations on using existing software development methods. Because this type of software has unique characteristics and conditions (Rahimian et al., 2008), a new practical approach is needed. Currently, we believe that agile software development methods provide a good platform for developing mobile applications (Ghandi et al., 2017). In our opinion, however, the agile methods introduced so far are not enough to fully meet the unique requirements of mobile applications.

Furthermore, our literature review on mobile application development shows very little research on mobile application development methods based on practical research (Ghandi et al., 2017), including developer and academic communities (see section 2.5.13). The dearth of information and studies show that developers may not fully understand the development process, tools, and issues they confront (Kirmani, 2017). These challenges include user experience and user interface, time to market, planning requirements, development skills, complication testing, and intense competition with app competitors (Jabangwe et al., 2018). Moreover, the previous studies did not match the rapid development of mobile devices capabilities and the process of developing mobile apps (Corral et al., 2014) and (Jabangwe et al., 2018).

This lack of research projects makes sense that there is still a lack of understanding of the development process applied by mobile application developers or the real challenges and problems encountered during the mobile development process. Furthermore, it was clear that previous studies do not match its speed as fast as the evolution in mobile apps and devices. There is still a deficit in academic research and a lack of awareness of the in-depth difficulties and obstacles that arise while developing mobile apps.

1.2 Problem Statement

Very little is known about the industry development process for mobile applications (Flora et al., 2014). Software development technology offers various options to the software developers regarding mobile application development (Bazarov et al., 2017). Each of these technologies provides advantages as well as disadvantages to the users. However, there is little research in identifying the best approach while considering mobile application

development. This thesis carries out a study to define a new methodology for developing mobile applications based on agile methods, considering the mobile software solutions industry's development methodologies and an extensive literature review. This would minimize the gap between theoretical (academics) and practical (experts) or applied research in the mobile application development process. As a result, this thesis will propose a methodology that combines both academics and experts communities.

1.3 Aims and Objectives

Mobile application development is one of the booming industries of the 21st Century. Several trends have been noted in which mobile applications seek to provide consumers with simplicity of use and convenience. With the introduction of several new mobile application development technologies and firms that provide mobile application development to customers, the trend has continued (Martinez et al., 2020). This research compares various mobile development methodologies and practices that the mobile application development organisations conventionally use. In addition, the research layouts the foundation of the best mobile development methodology and practices amongst the commonly available software development methods. The thesis will lay the foundation using a theoretical framework and a literature review to guide mobile development experts and organisations working in the mobile application development sector.

In a more detailed way, aims and objectives are:

- O1: Point out the primary agile methodologies that suit the creation of mobile applications. Accordingly, this study will conduct a comprehensive review for the previous related works that were oriented for mobile application development processes. Then, and based on the performed papers review, this study will conduct a comparison for such selected methodologies to achieve a useful summary to conclude the weaknesses, strengths, and main components of each one (Chapter 2 and 3).
- O2: Identify best practices from different mobile development methodologies based on our interaction with academics and software development communities. This study will implement a descriptive-analytical approach and will include experts in developing mobile applications. Therefore, we will conduct an in-deepness investigation of mobile application development methodologies based on primary research works and interviews with experts' developers to design and

conduct a survey that contains the main components that should be present in a mobile application development methodology (Chapter 4).

- O3: Analyze the data collected from the survey from academic and industry communities to identify tasks mostly performed by experts during mobile development, gaps, possibilities for integration, points in common with the proposals in the scientific literature, etc. All this with the intention of establishing the basis for defining a new methodology (Chapter 5).
- O4: Propose and specify a new mobile application development methodology based on academic and industry practices, integrating the best characteristics and proposing solutions to the detected problems (Chapter 5).
- O5: Validate the proposed methodology by the academic and the industry communities. This work will measure the expert's adoption to use this methodology based on TAM model (Chapter 6).

The expected result is to propose a methodology which will help to provide a comprehensive understanding of the mobile app development process in the business and the academic communities and will provide guidance to inexperienced development teams to increase the chances of success in their mobile product.

1.4 Research Questions

This study seeks to identify the environment for developing mobile applications and the best methodology for its creation. In order to achieve this purpose, the study will attempt to answer the following research questions:

- Q1: What is the primary development process method used in the academic community or the industry to develop mobile applications?
- Q2: What are primary development practices used in either the academic community or the industry to develop mobile applications?
- Q3: Can we propose a new mobile application development methodology based on standard academic and industry practices?
- Q4: How does the proposed methodology compare with other current mobile application development methods in the general perspective?
- Q5: Can we validate our proposed methodology by the academic and the industry communities?

These questions will be answered throughout this thesis report, resolving the uncertainty surrounding the problem at hand. With this aim, each answer will be examined, reviewed and analyzed from a scientific perspective. This effort will permit to validate or refute our research hypothesis: that it is possible to define a new agile methodology for mobile development based on a theoretical-practical study that is suitable and susceptible to adoption by novice developers and experts in the field.

1.5 Scope of the Research

Over the years, new technologies have been created to meet the growing human needs. With the advancement of mobility, unique needs were not met by computers and notebooks until then. From the 2000s onwards, the Internet began to be used on cell phones, serving that demand (Pecorelli et al., 2020). A new requirement emerged with this new technology: specific applications from mobile devices. It was not enough to adapt what was developed for desktops and notebooks to mobile. It was essential to create complex software for this purpose.

In addition to creating specific software for mobile applications, modifying their development process was also necessary. Traditional software development models like the waterfall or even the most recent models were presented do not suitable for mobile development, given the characteristics of these devices concerning the desktop and notebook. Examples of this are the high volatility of requirements, large amounts, and diversity of devices on the market, each with different features and the need for quick deliveries. They are made in small functional modules to perform these deliveries as requested, which take less time to get ready than the entire software's development.

Furthermore, there is little research into the development of mobile applications. We need to integrate and explore other agile methods that can be developed or improved to be compatible with mobile app development. Most importantly, we need to determine how to integrate and integrate research into a real development process suitable for scientists and researchers. Integration can be achieved through interviews with mobile experts, flexible experts, and experimental research in academia and industry.

In addition, many mobile applications have failed in the development process due to cost issues, scheduling, insufficient technology, insufficient testing quality, user unfriendliness, personnel issues, or administration and management issues. To reduce the risk of failure, an agile approach can be used. This strategy for addressing this type of time-dependent challenges has proven be quite effective in other contexts and can be useful in

mobile development too. According to the Agile Manifesto (Krehbiel et al., 2017), its main features are:

- People and interactions, not processes and tools.
- Executable software, not large and crowded documents.
- Customers work together instead of constantly negotiating contracts.
- Respond quickly to changes instead of subsequent plans.

Our motivation is to create an integrated methodology based on agile practices that captures (and improve) the mobile software development experience from the industry and academic communities. This integrated methodology will generalize and enrich the results among all mobile application developers' societies.

1.6 Research Methodology

The research methodology is how to pass a study or science, contains detailed and methodically ordered knowledge, concerning a particular domain of expertise (Flick, 2015). In our particular case, the scientific understanding of the mobile application development process will be obtained through the literature review on agile methodologies in mobile development (Wynn, & Clarkson, 2018). Next, based on the literature, a survey will be carried out to understand the developers' opinions that fit this situation. Subsequently, the study's result will be analyzed to compare what is present in the literature.

Therefore, in order to achieve the objective of this work, this study will adopt a descriptive-analytical approach, and the data will be collected using three popular methods: interviews, extensive literature review, and surveys. The survey will be carried out with software developers who have experience in mobile development. Therefore, our research uses quantitative and qualitative methods and includes experts in developing mobile applications. Concerning the purpose of this research, we will conduct an in-depth analysis of mobile application development methods through preliminary research and interviews with experts who worked and developed these mobile applications, hoping to gain a deeper understanding of mobile application development, the process, problems, and challenges they face (Alrabaiah & Medina, 2019).

Consequently, the methodology used in this work is divided into five phases, as can be observed in Figure 1.1. In the first phase, an extensive literary review will be made on the subject of mobile application development and existing methodology will be analyzed. In the second phase, several informal interviews with experts will be conducted. In the third phase, starting with the information obtained in the literature review and the knowledge extracted

from the experts, a survey will be formulated and made available online for a set of 35 developers, who will respond to it. As a result, a detailed online questionnaire on the development of mobile applications will be designed in this phase. In the fourth phase, the analysis of the expert survey will be conducted, and results will be obtained. Based on the results, a definition of a new methodology called Agile Beeswax will be formulated. In the finale phase, we will validate the proposed methodology using a second population of 35 experts, including mobile application developers and academic communities.

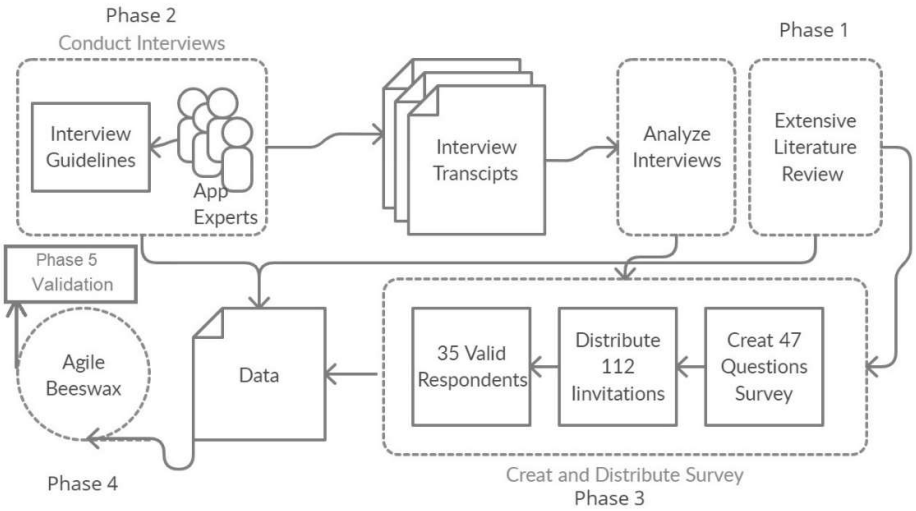


Figure 1. 1 The Applied Research Methodology

1.7 Research Structure and Phases

In this part will describe the main research phases that will be followed to achieve research objectives and conclude the main results. Consequently, this research is structured in four main phases as described below:

Phase 1: Extensive Literature Review

We will perform an extensive literature review, where books, articles, monographs, and websites will be consulted to build solid knowledge about the theme. The aim is to maintain a balance between the recommendations of academic researchers and the everyday reality of mobile development companies (chapter 2 and 3).

Phase 2: Initial Interviews

We will obtain useful information from five companies involved in the development of mobile devices during a set of initial interviews, which will be performed using a simple guide. We previously will conduct an in-depth review of the mobile application development processes outlined in earlier research to design this guide of the interviews to keeping it with

our research goals. These initial interviews with mobile app development experts will help us better understand the process, concerns, and problems of developing mobile apps (Alrabaiah & Medina, 2019) (section 4.1).

Phase 3: Extensive Online Questionnaire

Based on the knowledge obtained in phases 1 and 2, an extensive online questionnaire (survey) will be designed on the mobile applications development process. The survey will be available online for the participants. The survey will be divided into three parts. The first one will collect personal information and the types of projects developed by the respondents' companies. The second will be about using agile methodology, seeking to obtain the advantages and disadvantages in its use. Finally, part three will include questions about the impacts that the use of agile methods brought to the company (section 4.2).

After the end of the survey response period, an important task of analysing the results will be carried out. This work will analyze the obtained information in order to reach and complement the objectives of this research and reach the main components that must be present in a mobile application development process. After this analysis, a comparison with previous results will be made (section 4.2).

Phase 4: Development of Agile Beeswax

The main contributors to our research will be companies providing software applications for mobile devices, mobile professionals, scientists, and academics. Therefore, the survey will be explanatory and empirical, with significant conclusions drawn from the interviewed experts, which served as a basis for the proposal of a new method, namely Agile Beeswax. After reaching the final results of the main components in mobile application development, all these components were combined, structured and specified to form the presented methodology (chapter 5).

Phase 5: Validation of Agile Beeswax

After proposing this new methodology, it will be necessary to present it to the experts for validation, in order to evaluate it, improve its performance, or reject some of its components. Furthermore, knowing the level of their acceptance of adopting this methodology in developing mobile applications in their companies and businesses in line with our research hypothesis. A new survey will be made in order to present it to the experts to evaluate the proposed methodology. Specifically, we validate the proposed methodology with other population of 35 experts, including mobile application experts and academic or reserchers in the field (chapter 6).

1.8 Structure of the Thesis

This thesis consists of seven chapters. After this initiation chapter, this document presents the following structure:

- **Chapter 2 (Literature Review: Agile Software Development)**

  This chapter addresses the software development.

- **Chapter 3 (Literature Review: Mobile Applications Development)**

  This chapter addresses mobile applications, the use of agile approaches in the development mobile apps, and the main related methodologies for mobile application development.

- **Chapter 4 (Expert-Based Study in Mobile Application Development)**

  The chapter discusses the practical study performed with experts from mobile development companies on mobile development, which is performed by way of interviews and surveys.

- **Chapter 5 (The Proposed Methodology: Agile Beeswax)**

  This chapter discusses, proposes and specifies the Agile Beeswax methodology.

- **Chapter 6 (Validation of the Agile Beeswax Methodology)**

  This chapter describes the validation experience conducted with a new panel of experts about the proposed methodology, and a statistical analysis of the survey is performed to determine the amount to which this methodology is accepted or rejected.

- **Chapter 7 (Conclusions and Future Work)**

  The final chapter shows the last conclusions and perspectives for future work

CHAPTER 2. LITERATURE REVIEW: AGILE SOFTWARE DEVELOPMEN

This chapter will study software engineering and the software development process, and we will review the main agile methods that could be the base for the development process in mobile applications.

2.1 Software

Software is a computer program that contains documents and related configuration data necessary for the program to function properly (Sommerville, 2004). There are two main types of software: system software and application software (Yu, 2011). System software, such as operating systems, network software, and compilers, serve as tools to help create or support application software. Application software helps perform useful tasks such as information systems, office software, embedded systems, real-time systems, games, and scientific software (Peters, 2007). Within application software, this thesis focuses on mobile applications running on mobile phones, smartphones, or tablets. This software is also known as "apps".

2.2 Software Engineering

Software engineering is applying engineering to software and application software; developing, operating, and maintaining programs, that is, applying a structured, disciplined, and measurable approach (Sommerville, 2004). It had its origins in the late 1960s, and its main objective was to tackle the problem that was then described as the "Software Crisis" (Ebert, 2018).

Software engineering refers to the development of software systems. It focuses on: the precise specification of the system structure and the execution of these specifications; the objectives of the services provided by such systems; the activities necessary to develop confidence that the goals and specifications have been met; and the development of such systems over time (Motta et al., 2018).

Software engineering also addresses the methods, tools, and processes for developing software within time and budget (Finkelstein & Kramer, 2000). Frequently, businesses can have poor software for not using engineering. Poor software can be wrong, unreliable, insecure, backward, or expensive. If the company has a negative attitude, the top

management of the company will react badly to solving problems, including the company lacks formal documents and software design procedures; the development team lacks professionalism and discipline; will rarely uses software development tools: no system prototype; design from scratch; the process will take a lot of time; out of budget and improper resource compensation (Cooling, 2001), that is why it is important to apply software engineering principles in businesses.

A principle of software engineering is the contrast between a software product and a software process. The product is the result of a cycle of development. The process consists of a sequence or a series of steps for creating a product. The focus is on the outcome, product development. The process focuses on every step to be taken during the development of software products. In the words of Pressman (Pressman & Maxim, 2014), *"Computer software is a product designed and created by software engineers"*. It contains the running program and related documents, which contain not only text and numbers, but also graphical representations. It is several instructions, data or programs for computer operations and tasks. It is the opposite of hardware that describes a computer's physical aspects. Software is a generic term used for apps, scripts and programs running on a device.

To develop an application, you need a software development process; Pressman describes the software development process as a series of predictable steps in software development (Pressman & Maxim, 2014). In software engineering, the software development process provides a blueprint to follow.

In recent decades, many methodologies have been used in software processes. The software process and methodology seek to organize and manage how to solve software engineering project problems. Basically, this is a general guide that helps in developing software projects and determining who does what, where, why, how, and when it is needed. The software process is technical work with the application of tools, methods, and personnel to software tasks (AL-Allaf, 2003). Organisations need a familiar software development process that provides a consistent framework for performing and improving their work. When several people are working on a joint project, people need a way to coordinate their work. They can informally manage the flow of small tasks and projects, but the team needs more people or more formal mechanisms for more complex tasks (Humphrey & Kellner, 1989).

Software development methodology (SDM) refers to a form that follows the structure, planning, and methods of software development. Over the years, a good structure of this type has been developed, and each structure has its own recognized advantages and disadvantages. A software development method may not be suitable for every project. Each available

method is most suitable for a specific type of project and supports various technical, organisational, project and team considerations. Jayaratna (1994) states that more than a thousand branding methodologies are used worldwide and even more since then.

A recent classification of software processes divided these processes in two types: monumental (heavyweight process) and agile (lightweight process) (Barry & Lang, 2001). The next section will discuss this topic in detail.

2.3 Traditional Software Process Models (Heavyweight)

SDM is a planning strategy that performs the steps of the software program life cycle in a predictable, effective, and repeatable manner. Many software development methods have been developed to support the activities of the SDM. The most popular traditional software process models are Waterfall, Rational Unified Process (RUP), Build and Repair, Prototyping, Incremental, Spiral, and Rapid Application Development (RAD). Each model has its own methods, advantages and limitations (Pressman, 2005). The traditional software process model has many problems (Mahalakshmi & Sundararajan, 2013). Sometimes different stages of system development slow down the development process. In addition, demand specifications are changing because it is difficult for customers to determine their needs. The needs can be determined but as new technology is coming into the market the business world is also changing its structure constantly (Pressman, 2005).

We will take the waterfall model, prototyping and spiral development as examples to exhibit the basic features, usage, advantages, and disadvantages of this type of software process.

2.3.1 Waterfall Model

The waterfall model was introduced by Win Royce in Lockheed in the 1970s. It is named waterfall because it can be graphically represented as a waterfall from requirements definition to design creation, program implementation, system testing, and delivery to the customer (Jayaswal & Patton, 2006). The waterfall model is a software development method that is depicted to be sequential in nature whereby the development is seen to be steady flowing downwards like a waterfall through various phases as outlined in figure 2.1.

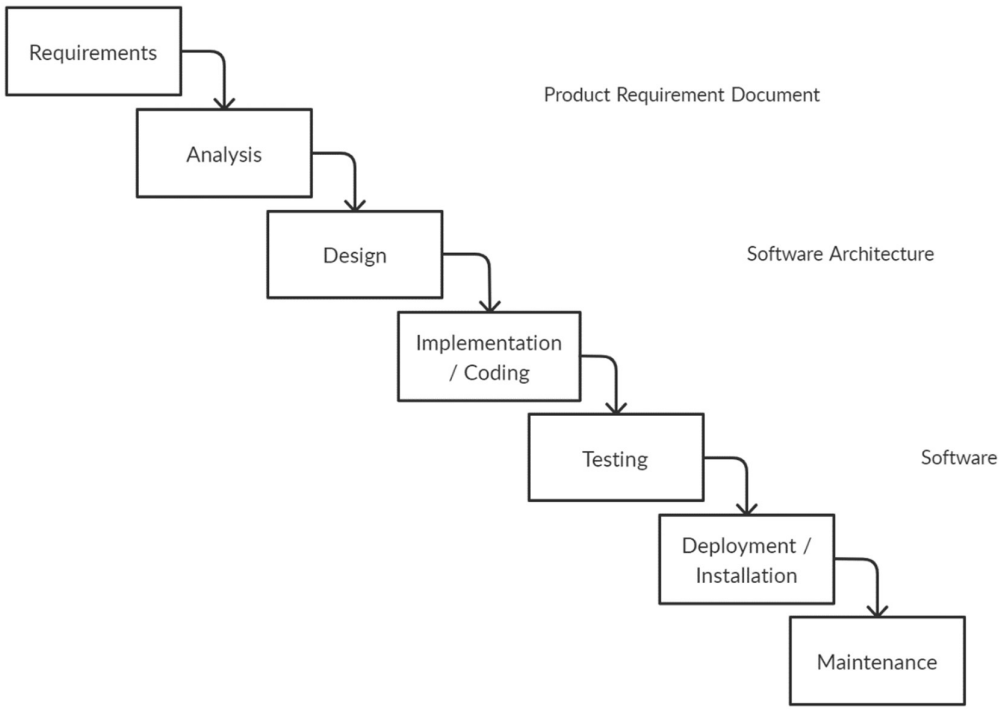


Figure 2. 1 Waterfall Methodology in Project Management (Andrei et al., 2019)

This method is described as a traditional engineering method used in software engineering. With this method, once the phase is started or completed, it is neither modified nor revised, which is the main disadvantage. Due to this inflexibility in a rigorous waterfall model, most studies have been criticized and therefore suggested more flexible alternatives (Wong et al., 2012). It is therefore not an attractive method, to some extent related to projects that exceed the budget and has been replaced by more flexible and versatile methods created specifically for software development.

The waterfall method is very suitable for supporting inexperienced project managers and teams or for creating volatile project teams. Progress in the development of the waterfall system is measurable. Thanks to its considerable structure and powerful controls, the waterfall is inflexible, slow, and expensive. The project moves forward and moves only slightly backwards. It is impossible to test the produced system until it is nearly totally coded and difficult to repair or respond to modifications. Changes made later in a person's life cycle are more pricey and so are not recommended. The waterfall is often difficult to read and understand.

2.3.2 Prototyping

A prototype is a type of modelling that looks and behaves like the target system. This methodology produces a program that performs a typical set of functions for the final product. Prototypes are created for judging, demonstrating, or experimenting. A good explanation of prototyping is given by (Lantz, 1997): "*Prototyping is about people. Users are not often able to articulate what they want an information system to do, and they cannot visualize it from written specifications. Prototyping enables them to see a system, play with it and modify it before it is implemented.*"

Proper use of prototypes can significantly improve the efficiency of the development process (Norgren, 2004). It's better to prototype first to understand the requirements instead of freezing requirements before design or development phase. The development of this prototype supports currently known requirements. This method is generally used when the purpose is to verify the acceptability of the implementation method, language, or end-user.

User participation in the development process allows designers and developers to receive direct feedback at the very beginning of the project. This provides a higher quality system as users have a natural inclination to change their opinions when identifying needs. Customers and developers can compare whether their software meets their specifications. It also allows the project manager to understand the accuracy of the initial project and whether the recommended deadlines and milestones have been successfully met. Quick user feedback can lead to better solutions (Matharu et al., 2015). In prototype design, you can easily modify and extend the model, for example, when plotting or simulating a planning software system, including its interface with input/output functions.

According to the research conducted by Ramathan (2007), prototyping also has certain disadvantages that will be discussed here as well one of those include insufficient analysis: when developers focus on limited prototypes, they prevent them from properly analyzing the entire design and even from finding better solutions or preparing incomplete specifications. Poorly designs are difficult to maintain. Due to the limited functionality of the prototype, it can also happen that the prototype is used as a starting point for the final product and that it will not scale well, and if the developer focuses on developing the prototype as a model, it may be forgotten. The prototype and the completed system can be misunderstood by the user: the user may start thinking that the prototype is indeed the final system to complete or improve. Developers can join the prototype they are trying to create; this can cause problems such as converting a limited prototype to the final system. Finally, prototyping requires excessive development time and costs to create a prototyping development team.

2.3.3 Spiral Development

Bary Boem, in 1988 came up with a formal software development system which was named the “spiral model” (Hamsini & Smitha, 2016). The model primarily combines some fundamental components of the rapid prototyping and waterfall model methodologies to combine the advantages of the bottom-up and top-down concepts. The spiral approach is a hybrid of waterfall and iterative development, with a greater emphasis on risk analysis. Its emphasis was on the significant part in which many pieces of research argued that it had been neglected in other methodologies. The spiral model is displayed in Figure 2.2 (Boehm, 1995).

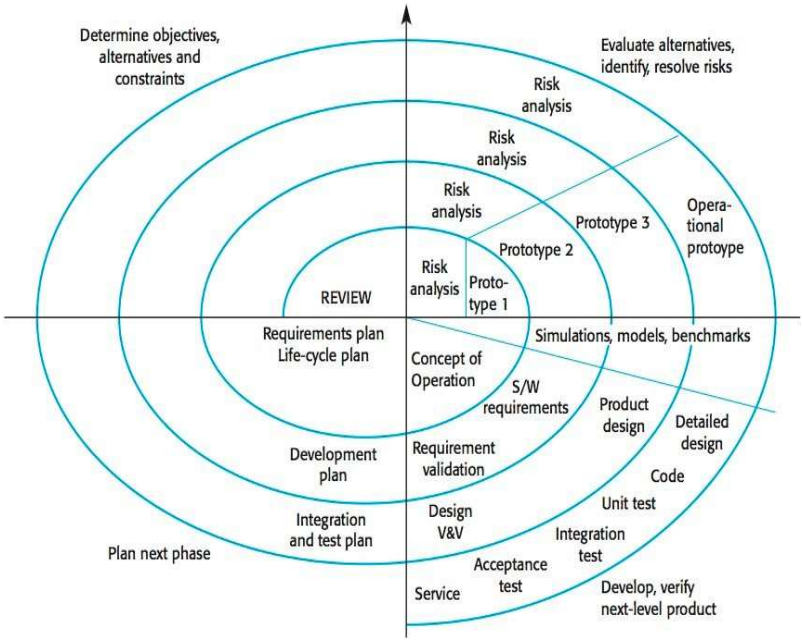


Figure 2. 2: The First Spiral Model (Boehm, 1995)

The spiral model includes planning, risk analysis, engineering, and assessment phases. Each phase starts with a design objective and ends with the client evaluating the work. It is an iterative method instead of a sequential method. The user sees the documentation very early instead of at the end of the process. Team members can be generic rather than experts. And it is a method that allows simple or complex modification of the program instead of a traditional inflexible tool. However, the spiral model is time-consuming and expensive to implement, and there's a chance to be able to stick to the budget or timeline, even it's really difficult to keep track of and manage.

2.3.4 Agile Software Development (ASD)

Agile software development is a group of software development methods that enhance iterative development, collaboration, and adaptability throughout the project lifecycle (Kumar et al., 2011). ASD makes things in small steps, with minimal plans rather than complete plans. This minimizes risk and allows the development lifecycle to adapt faster. It also focuses on stakeholder participation. This means that at the end of each iteration, stakeholders will be consulted about the product and given feedback.

Expression of agility can be defined as the ability to move quickly. Software development agility can be fast delivery and quick adaptation to changing needs. Agile means that the process is flexible enough and ready to adapt to changing supply requirements and schedules.

In the next section, we will review the Agile Software Development Manifesto and the main agile methods related to our research. Therefore, the following methods are included: Extreme Programming (XP) and Scrum.

2.3.5 Agile Software Development Manifesto

In February 2001, the Agile Software Development Manifesto was released, consisting of 17 methodologists (all from different backgrounds) who created the Agile Software Development Alliance to manage the challenges faced by software developers, known as the Agile Alliance (Beck et al., 2001). These authors are people who have published various software development methods with similar features; based on the expertise of professionals, they focus on the timely delivery of high-quality software (Andrei et al., 2019). This group of people defined the manifesto, defining four values and twelve principles to support better software development methods.

The values of the Agile Manifesto are (Beck et al., 2001):

- Individuals and interactions trump processes and tools.
- Functional software is better than complete documentation.
- Cooperation with the customer instead of negotiating a contract.
- Respond to changes instead of following a plan.

To better understand what agile software development is, the Agile Alliance refined twelve principles (Beck et al., 2001).

"*These principles are:*

- *Our highest priority is to satisfy the customer through early and continuous.*
- *Delivery of valuable software.*

- *Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.*
- *Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.*
- *Businesspeople and developers must work together daily throughout the project.*
- *Build projects around motivated individuals. Give them the environment and support they need and trust them to get the job done.*
- *The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.*
- *Working software is the primary measure of progress.*
- *Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.*
- *Continuous attention to technical excellence and good design enhances agility.*
- *Simplicity--the art of maximising the amount of work not done--is essential.*
- *The best architectures, requirements, and designs emerge from self-organizing teams.*
- *At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly."*

Agility can be applied to any software development process that changes requirements during the development process; effective communication between all stakeholders; attracting clients to the team; and organizing a team to monitor completed iterative software development and delivery.

2.3.6 Agile Methodologies

Agile development practices attempt to provide many opportunities for assessing the direction of a project in the developer's life cycle (Abrahamsson et al., 2003). This is achieved through close teamwork (visual domain who works together as a team). Close teamwork and regular development cycles can provide tangible benefits. Thus, by focusing on simple repetitive interactions at work and the products they provide, agile methods can be defined as iterations and increments. In a waterfall, the team only has one chance to fix each phase of the project. In the agile model, every aspect of design, development, etc., requirements will be continually re-analysed throughout the life cycle. The team may pause

every two weeks to redefine the direction of the project, and there is always time to shift it in a different direction.

These processes emphasize the coding method, including writing tests for the first time, doing "the simplest thing that might work". The Agile Manifesto and its four values provide a common basis for these processes (Beck et al., 2001): "Units and interactions transcend processes and tools; working software is better than complex documents; working with clients, not contracting and responding, next changes to the plan." If the software development process is in line with the above four values and principles, it is considered flexible (Ambler, 2005). The 12 principles can be summarised as follow: customer satisfaction; entrepreneurs and developers must collaborate on a daily basis during the project; acceptance of changing needs; frequent deliveries of working software; around positive people construction; personal dialogues in development teams; software operations represent a measure of progress; agile processes contribute to sustained development; constant emphasis on technical excellence; simplicity; team organisation and performance (Beck et al., 2001). Support for agile software development is provided through a range of agile software processes, including Agile Manifesto, XP, Scrum, Crystal Series (FDD), Lean Development (LD), Dynamic Systems Development (DSDM), Agile Modeling (AM), and Adaptive Software Development (ASD).

2.3.6.1 Extreme Programming (XP)

XP is one of the most widely used methods of agile software development and has been the subject of much discussion in the developer community (Newkirk, 2002). Kent Beck developed extreme programming because of the problems caused by the long development cycle of the traditional development model. This failure will have both economic and human consequences. It starts terminating without adding any functionality that is not explicitly required, which can slow down the process. This process has been driven by good practice in software development over the past several decades (Baker and Andres, 2004).

Extreme Programming addresses major software development problems such as schedule delays, project cancellations, system degradation, business misunderstandings, business changes, and staff turnover (Mahajan and Kaur, 2010). Extreme Programming consists of six phases (Beck & Andres, 2004): exploration, planning, iterations to release, production, maintenance, and death phase.

In the exploration phase, the concept of a user story is used to fulfil a set of requirements, and the customer writes a story card to be included in the first draft. Each story card describes a function that will be added to the program. The development team will be

familiar with the tools, techniques, and procedures they will use in the project. The key is to enable customers to get quick feedback in the first few stories so they can quickly figure out how to determine from the developer what they need and don't need. This phase lasts from weeks to months, depending on how well the developer knows the technology.

The planning phase prioritises stories and agreed dates, completes the most valuable set of streams and publishes the first minor revision before that date. The programmer estimates the amount of work involved in creating the story and agrees. Project speed is used to determine how many stories can be up to date or how long it will take to complete a set of stories. A few days are enough for the planning stage.

In the iterations to the first release phase, the work is structured in iterations. During these iterations, the performed tasks are concerned with the creation of usable software, and each iteration may include one or more sub-iterations, including modelling, programming, and testing.

The production phase requires further testing and performance verification before moving the system to the customer's environment. At the same time, due to changes at this stage, a decision should be made to add new features to the current version. The time taken for each iteration can be reduced from three weeks to one week.

In the maintenance phase, the project needs to keep the system running smoothly when developing new iterations; when starting the first version, customer support tasks are required. Therefore, after starting the system, the development speed may slow down. The maintenance phase may require new people on the team and changes in their structure.

The death phase occurs when the client has no other stories to include in the system, the system does not bring the expected benefits, or there is no budget for its maintenance. This requires meeting customer needs in other aspects such as system performance and reliability (Rocha & Guarda, 2018).

XP Process

An XP project is successful when the customer selects the business value to implement based on the team's ability to measure the functionality it can deliver over time. The development lifecycle contains the following steps (Jeffries et al., 2001):

1. The user defines the value to be implemented.
2. The programmer estimates the effort to implement it.
3. In accordance with their priorities and time constraints the user selects what to build.
4. This business value is built by the programmer.
5. Return to step 1.

In all iterations of this cycle, both the user and the programmer learn. The programmer should not be pressured to do more work than estimated, as software quality will be lost, or deadlines will not be met. In the same way, the user has the obligation to manage the scope of delivery of the product, to ensure that the system has the highest possible business value within each iteration.

Extreme Programming practices

Extreme programming's main objective is to produce software faster, more cost-effectively and in less time than using traditional techniques. These goals can be achieved by following values, principles and practices that differ from the methodologies used prior to its creation. The main objective in the XP process is customer satisfaction, to achieve it, instead of delivering everything desired by the customer at some date in the future distant, constant delivery of current customer needs is encouraged. All production teams are at the same level, as managers, developers, and customers, which makes the team highly productive.

XP is a set of ideas and practices derived from already existing methodologies (Beck & Andres, 2004). XP contains the following 12 practices. Figure 2.3 and Table 2.1 will summarize these practices.

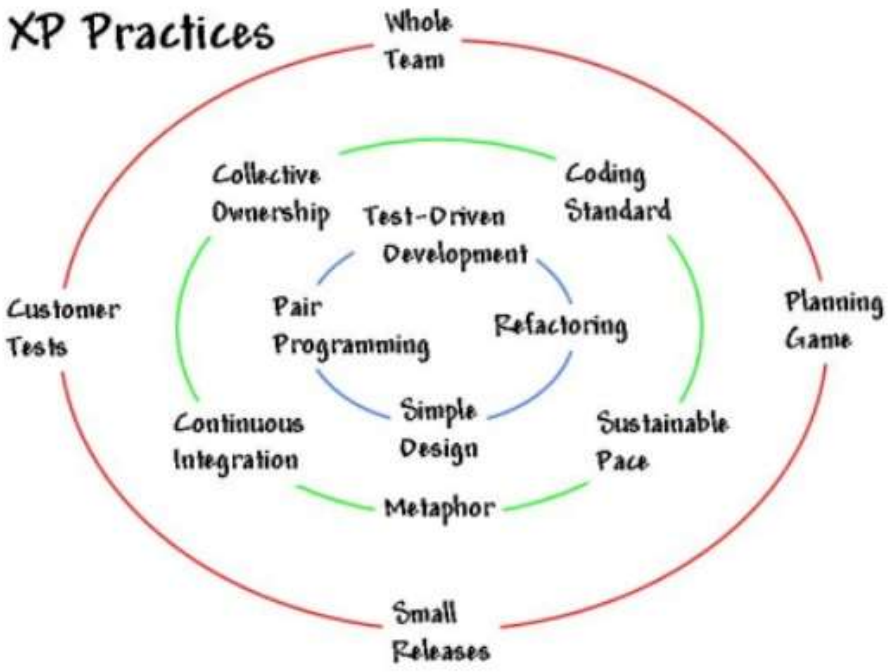


Figure 2. 3: XP Practices (Jeffries, 2003)

Table 2. 1: XP Practices (Jeffries, 2003)

XP Practices	
The planning game	Pair Programming
Small releases	In co-ownership
A metaphor	Continuous Integration
Simple structure	40 hours a week
Continuous testing	Customer on site
Refactoring	Coding standards

Exteme Programming practices in more detail (Jeffries, 2003):

1. The planning game implies frequent communication between clients and developers. The scope of the next revision is quickly defined. This means using the planning options available to match customer needs or scheduled tasks. Also, it is possible to install free capacity. The technical team estimates the amount of work required to implement the user story, and the customer determines the volume and delivery time and each iteration. Adding additional user stories or tasks to the current iteration is possible.

2. Small releases: The idea is to produce versions of the system that are operational quickly and then release a new version. In traditional approaches in software development, a release can take several years. But, customer requirements are subject to continuous change. The requirements may change during the release cycle (for example, because the market has changed). If the system is finally shipped using large versions, it could well be that the functionality does not meet the actual requirements.

3. A metaphor: Is a holistic view of a system that has commercial and technical implications and represents "what we are trying to do". XP does not emphasize the timely determination of stable system architecture. This architecture is considered evolutionary, and the inconvenience of not using it at the beginning of the project is solved by the presence of metaphors. The system is defined by a metaphor, or a setting of metaphors provided by the client and the development team. A metaphor is a general story

describing how a system is supposed to work. The software architecture metaphor is like a simple software development project.

4. Simple structure: The system should be as simple as possible, eliminating all the complexity. Develop the simplest solution that can be processed and implemented at a certain stage in the project. In traditional methods, a lot of work is devoted to system requirements. Requirements are fully worked out before coding. This opportunity is impossible to cope with the ever-changing requirements. It does not occur in XP . It is normal for XP to accept these changes. Besides simplicity, developers can implement the smallest project. The XP developers say this project is simple. Experience has shown that this is also the optimal design under normal conditions. Kent Beck said that the correct software design is always like this: as few classes and methods as possible, they pass all tests, clearly reflect the intentions of the programmer's implementation, and there is no duplication of logic (Saleh et al., 2019).

5. Continuous testing: Code generation is controlled by unit testing, programmers write unit tests, and clients write functional tests. Customers locally define functional tests that developers must perform. In XP, unit tests are created before the code is written and run continuously before each system modification. This is also one of the original intentions of front-end design. The XP team uses a test driver to make sure the system is working correctly after writing the code. They are delivered to customers in a short time frame, so they can immediately check if the implemented content matches what they want. This gives the developer immediate feedback.

6. Refactoring: Refactoring the system without changing the behavior. If the documentation for a software system is limited, the source code should be simple and straightforward, as well as more flexible for further changes. Refactoring improves the internal structure of the code without changing its external behavior. To achieve this goal, the developer must refactor the codebase. This means that anything requiring acceptance testing and complete unit testing should be removed from the system.

7. Pair Programming: All production code is written by two programmers working on the same machine. The person who writes is also called the "driver", and the other person is called the "navigator". Pair programming is an agile technique for software development where two programmers work on a single workstation together. One is the driver, the observer, or the navigator, which writes code, and checks each line of code as typed into it. Both programmers frequently change roles. In the process of programming, such a cycle is created: first, write a test case and then run the test. The

first test fails, then implement and program, then run the test again, if it falls back to execution, if it works, save the new test case. Now, when the driver is typing, the navigator reads what is being typed and can instantly identify many common coding errors. This method encourages developers to learn from each other and do things they have never done before, and newbies learn from experienced developers.

8. In co-ownership: Anyone can change the code anywhere and anytime, and team members are co-owners of the codebase. This is the traditional solution for every subsystem (or module or component) that a technician is responsible for. They are responsible themselves, but what happens when something happens, the technician trains or gets sick. The traditional solution is to back up technicians. However, the XP solution is that all design team members are jointly responsible for all their work, including source code and subsystems. An excellent suggestion to help team members is to arrange a separate meeting at least once a day.

9. Continuous integration: Integration and build is performed multiple times a day. An excellent way to do this is to use a dedicated machine and let a couple of developers take the time to integrate and test their code. If the test fails, discard the changes and continue until all tests are successful. In other words, the system used is immediately available and its functions can be successfully reproduced and tested. This is called continuous integration because the time interval between integrations is very short.

10. Forty hours a week: Team members should work a maximum of 40 hours a week at a balanced pace, with no overtime for two consecutive weeks. In line with the XP philosophy, no one can continuously work 60 hours a week without compromising product quality. Projects requiring additional timely work are often delayed. Instead, the team member can do planning to change the scope of the project or the delivery date.

11. Customer on-site: The customer must be present and available to the entire team. This is a big part of the success of an XP project as the customer continues to work to deliver maximum business value, and the developer can immediately resolve any associated issue. It also allows customers to change requirements at short notice, helping the team flexibly focus development on the most pressing needs.

12. Coding standards: Follow coding standards so team members can read the code and make changes easily. In XP, programmers and different team members program different parts of the system, so coding standards are required. The standard should be easily met and accepted on a voluntary basis. Standards such as code design, the use of tabs, brackets, comments, etc.

Weaknesses and challenges in Extreme Programming

XP methodology can result in well-organized, more flexible, more fun, and more predictable than other methodologies (Alexander, 2006). However, XP also has weaknesses, such as applying it when the team is large. In addition, XP requires a high degree of administration; as a result, the project can never finish, generates a bad documentation, etc. Predict the specific characteristics to be performed on a fixed time and budget is other of the common weaknesses that this type of iterative methods can present (Awad, 2005).

2.3.6.2 Scrum Method

Scrum is a set of working methods that were first adapted as a fast-programming method by Takeuchi and Nonak (1986) in Japan. The development method for Scrum software is a flexible, iterative procedure based on incisive methods which guide the development of complex software and products. He also described Scrum as a flexible and lightweight project management method, built on small, strong and self-organizing teams (Leffingwell, 2007).

Scrum is used to manage the system development process. This process only deals with how team members must flexibly build the system in an ever-changing environment (Abrahamsson, 2001), although it does not define specific software development practices. This ever-changing environment can include requirements, timelines, resources, and technologies. With the huge number of people using high-end advancement of technologies presented in both software and hardware and many people using them, we can see more and more the presence of software with greater complexity, causing the methodologies employed by the software development companies to become inadequate and not manage to adapt to these new requirements.

Scrum is built around Sprints. A Sprint is a short period of time during which a Scrum Team works to finish a specific amount of work.

Scrum Roles

There are three main Scrum roles: The Product Owner, the Scrum Master, and the Scrum Team. All management positions in a project are organized between these three roles, and it will be described in more detail as below:

Scrum Master

Scrum Master is a management role. The Scrum Master is to ensure that the project is conducted in accordance with and on schedule with Scrum’s procedures, values and laws.

The Scrum Master does not lead the team. The Scrum Master interacts with the project team, customer, and project management. The Scrum Master acts as a link between the owner of the product and the team (Mathew, 2019). The Scrum Master also ensures that the team works to eliminate constraints and changes in the process as effectively as possible. Scrum Master is responsible for promoting and adhering to the procedures of Scrum (Mathew, 2019). The Scrum Master is the motor of all Scrum processes.

The Scrum Master oversees the Scrum process, teaching Scrum to all project participants (Mathew, 2019). So that team members fit into the organisation's culture and continue to deliver the expected benefits to the individual. Furthermore, they take care to avoid any other obstacles that get in the way of the team, from achieving the Sprint goal. In short, this role keeps the team creative and productive while also providing visible team success for the Product Owner.

The Daily Scrum is a 15-minute standup meeting for the developers of the Scrum Team. The Scrum Master listens carefully to what each member of the team has to say. Then at the end of the week, Sprint planning is a timed work session that lasts around one hour for each week of a Sprint. The entire team agrees to fulfil Product Backlog items in Sprint planning. The Sprint backlog is defined by this agreement, which is based on the team's velocity or capability as well as the Sprint's length (Eloranta et al., 2016). The Scrum Master compares the required progress with the expected. After each Sprint, the Scrum Master organizes an assessment meeting with the Scrum Team, during which the lessons learned, and conclusions are analysed. The goal is to improve the team's knowledge and motivation for the next Sprint. The Scrum Master also advises the Product Owner on maximising the ROI (return on investment) for the team. Scrum Master organizing the final decision tasks for the Product Backlog helps assess the effort involved in developing Product Backlog items (Beck & Andres, 2004).

Product Owner

In Scrum, the Scrum Master and the customer choose the product's owner. An individual, not a group, must own the product. The main responsibility is to list, verify, maintain, manage and make late payments available to everyone in the project. First, the Product Owner lists the highest priority product backorders. It also must represent the interests of the customer with priority and requirements.

Since Scrum evaluates self-organisation between teams, the Product Owner should try not to control every detail. The Product Owner should also be available for team questions. The Product Owner is responsible for funding the project. The Product Owner creates general

design requirements, a commissioning plan, and a return on investment (ROI) (Schwaber, 2004). Return on investment is defined as the total amount earned or lost from investment in relation to the amount invested.

The Product Owner can be a business development manager, a project manager, or an internal user development manager. The Product Owner's authority and responsibility prevent him from finding the right balance between obligations.

Scrum Team

A Scrum Team is a project team that decides how to act and self-organizes to achieve Sprint's goals. In the Scrum methodology, the team is responsible for getting the job done. Ideally, a Scrum Team consists of seven cross-functional members. The Scrum Team creates and reviews a Sprint to-do list, evaluates workloads, and proposes any issues that need to be addressed from the project (Beck & Andres, 2004). To maximize the effectiveness of a team, it must manage itself and organize itself. The team is responsible for meeting the requirements and their functions. In a Sprint planning meeting, team members should decide which elements to implement in the next Sprint. The team must make its own decisions, do the necessary things, and demand the removal of obstacles. The team can decide how to proceed, and its decisions must be consistent with each operating organisation's charter, standards, structure, agreements, and technologies. However, as with the Product Owner, this freedom has to do with the responsibility for achieving Sprint's goals.

To achieve the goals of Sprint planning, a team will be formed of highly skilled members who will be experts in their field (Leffingwell, 2007). If the team size surpasses 8, Schwaber and Beadle (2002) advise separating them into several teams to reduce contact and dependence among team members. A typical team might be software engineers, analysts, user interface designers, architects, developers, and testers in a software project.

Scrum Practices and Artefacts

Scrum introduces some new artefacts. They are used during the Scrum process and will be described below:

Product Backlog

The Product Backlog contains requirements for a system or product that is being developed within a project. A Product Backlog is a set of functional and non-functional requirements that are ranked according to their importance to the company. The Product Owner is responsible for the content, priority, and availability of outstanding products. The

Product Backlog is dynamic, that is constantly changing to determine the needs of the project, and will never be completed as needed.

The Sprint Backlog

The Sprint Backlog defines team tasks to transform the Product Backlog selected for the Sprint into increments of potentially valuable products. Team members establish the Sprint Backlog in the Product Backlog and during the Sprint Planning Meeting, the top priority items in the backlog of the product are first selected and broken down into some smaller tasks (Schwalbe, K. 2012). If product items are split into small tasks, team members estimate the time it takes to complete each task. Team members try to maintain small tasks to finish every task within three days. In the second part of the Sprint planning meeting, the team creates an initial list of these tasks. Tasks should be separated; each task is from 4 to 16 hours. Tasks exceeding this time will be moved to a poorly defined list. Only the team can change the pending Sprint. The Sprint Backlog is a clear view of the real-time work that the team plans to do during the Sprint. The Sprint Backlog consists of these small tasks (Kayes et al., 2016).

The Daily Scrum Meeting (DSM)

DSM is a 15-minute status meeting to discuss the work done since the last meeting, projects to be completed before the next meeting, and obstacles faced by the developers. DSM supports communication, identifies and removes development barriers, emphasizes and supports rapid decision-making and increases transparency.

Daily Scrum of Scrums Meeting (DSSM)

It is the same short daily meeting synchronising the work between multi-Scrum teams (including all teams under different Scrum managers).

Sprint Planning Meeting (SPM)

This is a monthly meeting of the Product Owner with the team to discuss what will be done in the next Sprint. Each Sprint can last 30 days.

Sprint Review Meeting (SRM)

This is a monthly meeting at the end of each Sprint, usually four hours long. Team members share Sprint results with Product Owners and stakeholders.

Scrum Process

Scrum contains five main phases: Initiate, Plan and Estimate, Implement, Review and Retrospect, and Release (Satpathy, 2013), as seen in Figure 2.4 (4geeks, 2019).

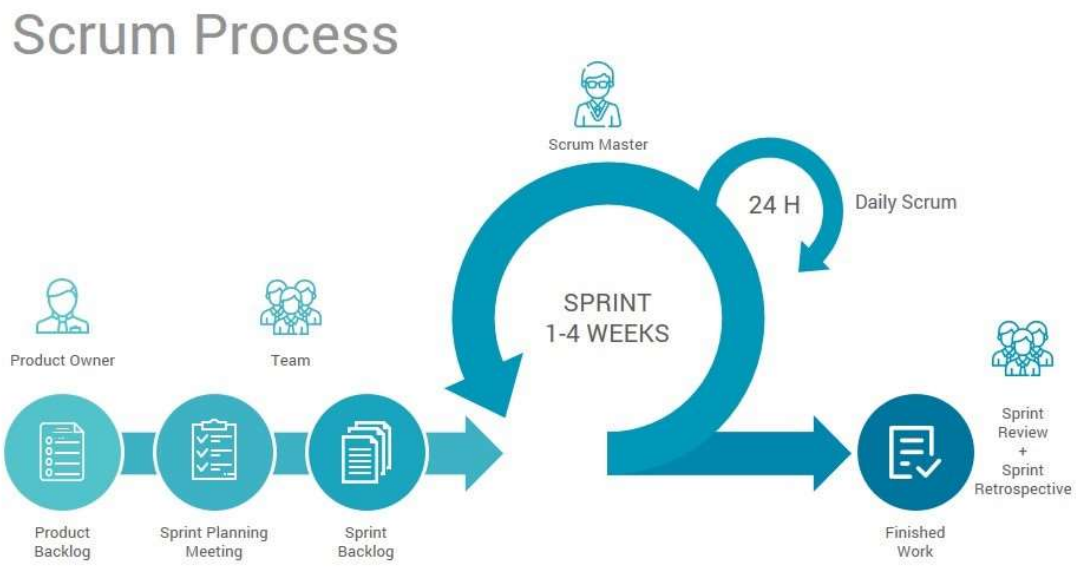


Figure 2. 4: Scrum Process (4geeks.io)

Scrum methodology contains five main phases:

1. Initiate

A Scrum project begins with a vision of the product of the developed system. The initial vision may be unclear, but it will become clearer as the project progresses. During this process, the Scrum rules will be established. Product Owner, Scrum Master, and stakeholders, including developers, designers, IT team, QA team, etc., will be identified. The Product Owner works with the Scrum Master to select the team members, then they create a priority product register .

2. Plan and estimate

The Product Owners tend to start writing user stories that clearly articulate the customer's needs and are understandable to all parties involved. These user stories are validated for Sprints, and the Scrum Master and Scrum Team are responsible for evaluating the work needed. This phase uses estimated work and effort to create schedules and to-do lists. After the next Sprint planning meeting, the development team create a to-do list that includes all the tasks that will be completed in that Sprint.

3. Implement

Scrum diagrams are often used to track work and activities. Although the Scrum implementation uses daily meetings, this is a poster meeting with a schedule. This is where the Scrum Team members keep each other informed of their progress and any obstacles they

may face. In this process, the team can consider maintaining and checking backlog on priority products.

4. Review and retrospect

During this phase, in a Sprint review meeting, the Scrum Team reviews the results of the Sprint for Product Owners and stakeholders. In addition, the Scrum Master meets with the Scrum Team to discuss the lessons learned during the Sprint; this information is recorded.

5. Release

In this phase, the relevant stakeholders get acceptable results.

Advantages and disadvantages of Scrum

Despite its advantages, this methodology has some inconveniences. We can be concise in analysing the Scrum development method by listing its advantages and disadvantages as shown in Table 2.2 (LITIC, 2010).

Table 2. 2: Advantages and Disadvantages of Scrum

Advantages	Disadvantages
Short 30-day Sprints to get work functioning	Less documentation than other methodologies
Motivated team	Deadlines cannot be ignored
A lot of owner involvement	Long planning is absent
Less overhead work	The team has no clear role
Cost-efficient	Sprints cannot exceed 30 days
Daily status meeting (usually 15 minutes)	The owner needs to dedicate time
Small team building up to 9 people	Project features may be dropped
Ability to make changes	
Success depends on software functionality	Distributed development is difficult
Easy judgment on project status	

CHAPTER 3. LITERATURE REVIEW: MOBILE APPLICATION DEVELOPMEN

Mobile application development is essentially a rising technology which deals with writing programs for smartphones and other portable devices, by executing different software engineering processes. This recent technology is extending in the market with new features, new possibilities as well as restrictions that were never in existence before.

The construction of mobile applications has initiated the development of new constraints, requirements, importance, and characteristics that have rarely been experienced in the past. This review of related literature seeks to present a detailed account of several issues. Some of the elements that will be discussed include the importance and application of mobile application development and agile methods in mobile application development, the common requirements needed in mobile software development, the reasons as to why we need methodologies in the development of the mobile application, and the process involved in mobile app development, among many other issues.

3.1 Mobile Apps Development and its Importance

The dramatic universal rise in the use of mobile phones has initiated mobile computing to be one of the dominating platforms in software engineering. As Falloon (2017) outlined, the mobile application seems to have revolutionized the mobile world, acquiring more power on different individuals of all cadres. Mobile applications development is at its peak within the changing technological industry. There exist several generalized reasons showcasing the logical relevance of mobile application development. A study by Wasserman (2010) outlined a good number of prudent reasons why there is wide adoption of mobile applications. According to Ceci (2021), it is estimated that over 5.7 million mobile applications are presently prevalent for mobile users in Android and Apple app stores, primarily in social networks, communication, travels and utility, games, and multimedia. The rise in the development of mobile applications in conjunction with the apps users, highlights the need for efficiency as well as a standardised mobile app development process. As argued by Wang

et al. (2018), mobile applications appear to be a technology depicted as new in software engineering emerging with new features, advantages, and restrictions that never existed before. Therefore, there is a need for reassessing alternatives concerned with building, planning, functioning as well as cost estimation of the mobile applications development.

3.2 Mobile Device Features

Mobile apps have been depicted as critical when utilising mobile devices such as contact lists, camera, GPS, compass and map, accelerometer, and many others. When such device features are employed within an application, there will be the creation of a much more interactive and fun environment (Wasserman, 2010). Furthermore, such features have been designed to significantly reduce the users' efforts. For example, a user who needs to complete a form on a banking application might need to send their photos to complete a given process. The app can allow the user to use the camera in the phone for capturing and submitting the photograph. Moreover, the device features can also help shorten the users' time to perform specific tasks in an app. In addition, development for mobile devices has several characteristics that make it unique and challenging, due to specific demands and technical limitations, such as: the rapid evolution of mobile devices, different types of standards, protocols and network technologies, need to operate on multiple platforms, specific needs of the users who use this type of device and short time to meet market requirements.

3.2.1 The World Has Gone to the Mobile Devices

It is undeniable that the universe has gone to the mobile devices and is impossible to turn back. Regarding the global statistics, the number of smartphone users has surpassed 6 billion, as pointed out by O'Dea (2021). Consumers employ their smartphones' usage to look for whatever they want in the universe. In contrast to the traditional websites, the mobile applications thrive as a browsing alternative and an intuitive purchaser making the mobile app presence inevitable for a business to develop (Hussain & Kutar, 2009).

In addition, in contemporary society, mobile applications are critical as far as leisure activities are concerned. The availability and emergence of mobile apps such as games, video, and music apps to phone users have proved that mobile apps are essential in business ventures and communication and for the relaxation of human beings. Kangas and Kinnunen (2005) stated that many video apps can be easily accessed and are far much convenient for users. Based on their uniqueness as a result of the revolution from childish games to more sophisticated ones, the games app is presently highly prevalent in contemporary society. On the other hand, for example those who value music, there also exist several music apps in

which users can enjoy using in listening to live music or playing the downloaded music. Consequently, reading apps are not left behind. They enable users to read academics in conjunction with other books of interest on their mobile phones since it is depicted to be relatively easy rather than moving from one place to another with heavy books (Charland & Leroux, 2011).

As most individuals continue being engaged by social media, the integration of features such as comments, likes as well as in-app messaging in mobile applications has effectively improved the social and business communication process. Moreover, users can share, review, and raise discussions of products or develop a community of customers (Falloon, 2017). The flexible accessibility of mobile applications is one of the significant reasons for its popularity in its utilization. It is undeniable that mobile apps can be available 24 hour on the mobile device at any place globally (Wang et al., 2018). Therefore, other importance of mobile app development includes gaining competitive in the first-time industry, enhancing customer engagement, and increasing convince for the customers.

3.2.2 Characteristics of Mobile Applications and How it is Different from Traditional Applications

As the performance of mobile platforms continues to improve, users expect their mobile devices to provide the same features as desktop applications. However, the mobile applications development process is different and requires adequate development technologies to be adopted (Dehlinger & Dixon , 2011). So as compared with the windows application process, the mobile application process is unique and requires a lot of time and strategies to be properly implemented for the development.

Dehlinger and Dixon (2011) ascertain that the mobile application software development process is entirely distinct from the traditional software development process in various aspects. In developing an outstanding mobile app, it is critical to understand and analyse the mobile application's underlying features and its practical application value. Fishbein et al. (2017) noted that many mobile platforms and tools could often cause developers to look at different features related to mobile application development and evaluate different new methods and new features that should be addressed in coding and design maintenance and application deployment. However, it seems that there is insufficient understanding of mobile applications' different categories, types, and characteristics and lack of research initiatives. In addition, mobile devices will therefore face potential attacks that

need to be addressed in a timely manner and require extensive research (Fishbein et al., 2017).

Jumaat and Tasir (2013) stated that there is a dramatic growth of the mobile app market. Due to the continued performance in mobile platforms with an increasingly growing demand and the number of users, there is a significant demand for mobile applications. Mobile applications are still being developed, and software is being developed for handheld devices such as tablets and mobile phones. This is usually downloaded by users of various platforms and stores with the distribution of mobile software or pre-installed software for mobile phones.

According to Muccini et al. (2012), the mobile app development process to some extent resembles the traditional software engineering process in some ways. Issues that are similar to both mobile and traditional software development include the following: traditional issues concerned with security, the integration with the device hardware, storage limitations, performance and reliability. Although it is clear that security aspects, space restrictions and other features become more critical in mobile systems. In addition, the fact remains that mobile applications present several additional requirements different from traditional software applications. This is summarized in Table 3.1 (Muccini et al., 2012).

Table 3. 1: Mobile Application Software Additional Requirements. Software's

No.	Mobile Requirements
1	Potential interaction with other applications
2	Sensor handling
3	Native and hybrid applications
4	Families of hardware and software platforms
5	Security
6	User interfaces
7	Complexity of testing
8	Power consumption

For example, mobile apps need to interact with other applications downloaded from several resources, which makes it necessary to manage this integration. In other hand, the mobile apps contain sensors and other devices (accelerometer, GPS, cameras, microphone, voice calls, and network protocols) that enrich the functionality of the app but complicate its

development. Furthermore, the mobile app type can be natively installed directly on the mobile device or use services across the internet (web-based hybrid apps), in each case requiring a different technology. In addition, a mobile app should be developed for several devices, encouraging various operating systems and security issues. Also, the app development should follow user interface guidelines, with the challenge to test apps in different platforms with several devices and web-based apps. All of that with limited device resources and power consumption.

Charland & Leroux (2011) argues that the software products, in conjunction with the ongoing demand for mobile devices, has made the software project teams adopt practices that inherently allow to support the underlying features of mobile applications. A combination of sensor handling, computing power, multi-modal user interface, location and security has propelled mobile devices to become a new platform for computing, particularly for business people and software developers. As a result, the increasing development and growth of the new computing platform need processes concerned with the engineering of the software to be tailored towards the development of mobile apps.

This is evidenced by the development settings, requirements of mobile apps in conjunction with the technologies supporting mobile apps, that mobile requirements are different from conventional desktop requirements (Charland & Leroux, 2011). In addition, it is undeniable that mobile application developers will ultimately experience the challenge concerning change requirements as dictated by the users, since the nature of mobile applications is more evolutionary than desktop applications. These customer expectations and the daily modifications make the system far more complex than how it used to be. As the complexity of the project advances, significant efforts in formalization and coordination are required in the mobile app development process.

Therefore, it seems undesirable to just apply and then implement the traditional software engineering processes in developing a mobile app without doing some modification or tailoring. The agile processes are regarded as essential with the fast-paced markets software, in which user satisfaction is administered by a continued and frequent generation of working software in which the user's requirements are very dynamic in the project and in a situation in which the delivery cycle is significantly short.

Furthermore, as a result of the advancement in communication technologies and mobile computing, there has been a dramatic increase in the demand and requirement of mobile applications (Fishbein et al., 2017). Developers of different hardware for tablet or mobile platforms force mobile developers to design a high number of applications

personalised for different devices. For this reason, it is recommended that the software can be reused in mobile apps development. Key features that define a successful and reusable mobile application include reliability, functionality, accessibility, portability, flexibility, usability, maintainability, efficiency and responsiveness, and repetition according to the qualitative characteristics and requirements of the user in accordance with the provisions of ISO 9126 (Kangas, & Kinnunen, 2005).

According to Kangas and Kinnunen (2005), the ten characteristics differentiating the mobile sites from the desktop sites on the basis of prioritisation of content primarily includes the following critical features and functions: Graphics and text, vertical rather than horizontal navigation; global and contextual navigation; hypertext, bars and tabs; breadcrumbs; footer; phone function integration; progress indicator; and the personalized and localized search.

Finally, according to Dehlinger and Dixon (2011), the software, hardware, environment, and the operating system determine the medium of communication for a specific mobile application. The present mobile app advancements, technological development and the varied market have imperatively benefited the mobile developers. It has created significant revenues as well as influenced new development opportunities as a result of the dramatically growing mobile applications portals (Fishbein et al., 2017). These new opportunities demand new capabilities and tools, as well as new software engineering methods to generate mobile applications with quality and efficiency.

3.2.3 Classification of Mobile Applications

Flora et al. (2014) found that mobile apps fall into three categories: native platform, web technologies, and hybrid platforms. Native applications are designed for specific operating systems. They are inherent in the device or platform. They use development tools and languages supported by their respective platforms, such as Objective C for iOS or Java for Android. Therefore, content created for the Android platform cannot be used for the iPhone on the iOS platform. Native apps are known for their high performance and speed, as well as high development costs. On the other hand, web applications can be accessed through a mobile browser; it is a responsive website that adapts to the user's hardware. As we all know, web applications can be accessed from almost any device, and their performance depends on your internet connection. Finally, cross-platform native or hybrid is a combination of native and mobile web apps. They behave exactly like native apps and have the same look and feel, but are based on cross-platform technology.

As is well known, a cross-platform native app is easy to build, but slower than native one. Its appearance is similar to native apps, although they have a different implementation which comes at a cost in performance. For its part, a mobile web app is not ideal because it runs through a browser. This is the reason for poor performance because they are hosted on the Internet, which also can lead to security problems. For its parts, native mobile applications are those that are installed on your device and can be accessed using icons. They are clearly designed for one platform and have the advantage of having all the functions of the device. They are very easily accessible to the hardware and also have a very high performance (Muccini et al. 2012). Due to these differences, the choice of application type is important from a methodological point of view.

Another way to classify mobile apps is based on their purpose. There are 33 categories of mobile applications in Google Play and 24 categories in the iOS App Store, for example: games (21.86%), business (10.11%), education (8.68%), lifestyle (8.62%), tools (6.12%), entertainment (5.79%), and travel (3.8%) (Guler, 2021). The purpose of the application may also influence the methodology to be used.

3.2.4 Issue and Common Requirement in Mobile Application Development

Mobile application software engineering shares the same practices with traditional applications. However, there are many prevalent issues that need to be addressed specifically in mobile development. Flora et al. (2014) provide several points that are distinct in mobile applications from traditional applications. One of the points is the potential of operating with each other. The mobile devices have several applications from different sources with the capacity to interact between them. Second, the aspect of sensor handling emerges. The typical touch screen, the response of the accelerometers to the movement of the device, microphones usable in application apart from cameras, voice calls, and global positioning systems in conjunction with multiple networking protocols are entirely inbuilt in one device. This could make the application development process more complex as it requires the developer to create alternative features for the application depending on the device configuration and its components (Freire-Obregón et al., 2019).

The next point is concerned with the families of the software and hardware platforms. Eler et al. (2018) argue that mobile applications are needed to support several devices with distinct hardware and typical screen sizes. Moreover, (Khan et al., 2021) ascertain that most aspects of the application would significantly affect the power consumption and the device's battery life. According to research by Wasserman (2010), mobile frameworks and

development tools are entirely concentrated on an individual developer with the intention of developing an application quite with ease. However, these applications are nowadays undertaken by small development teams due to their increasing functionalities.

For its part, Shmatko and Volkova (2019) stated that the first requirement for mobile development is the possession of technical skills. Basically, mobile android app development can be achieved using Windows PC, Mac, or Linux computers. Consequently, an mobile device or emulator such as Genymotion are required in the testing phase. Another crucial requirement, as outlined by Rosales-Morales et al. (2019), is the possession of programming languages, tools and mobile app platforms. In this case, the most prevalent mobile app platforms are iOS and Android. For Android, Kotlin and Java skills are needed. On the other hand, iOS app development will ask for Swift and Objective-C skills. Additionally, there is a need for the possession of C++, JavaScript, and HTMLS5 skills are prudent. Consequently, there is a need for having detailed knowledge in working with renowned tools and software inclusive of XCode and Android Studio.

Back-end computing is also one of the key requirements as far as mobile app development is concerned. Basing on the fact that mobile app deals with the allocation and the storage of data, it is essential for any mobile app developer to have comprehensive knowledge concerning the management of the databases, allocation of memory as well as an integration of hardware among many others (Heitkotter et al., 2012). Another requirement is the skills in User Experience (UX) and User Interface (UI). This requirement is critical in ensuring that the app developed lands in the hands of a wider audience as a result of the creation of far much enticing app design with an appealing user experience.

Awareness of cybersecurity is as well as a common requirement, a study by Al-Muhtadi et al. (2017) observed that the hackers are turning towards app data in fulfilling their evil motives such as the Facebook Cambridge Scandal and the Uber Data Breach. Therefore, it is undeniable that it is critical to have the right skills as far as the security of a mobile app is concerned. In a more appealing way, it is crucial to critically understand the guidelines regarding cybersecurity and their implementations in the process of developing a mobile app. Additionally, the understanding of the cross-platform app development technology is another essential requirement in mobile app development. This is based on the fact that most of the mobile app will ultimately run-on multiple devices and platforms based on demand (Ahmad et al., 2018).

3.2.5 The Need for a Methodology Specific for Mobile Application Development

Due to the progress made in improving mobile devices, there has been significant growth in the need for mobile application development. Javalas and Economo (2011) stated that it was a challenge for the development team if the mobile application had a long shelf life. As a result of this intense competition, the expectations and experiences of users have become a primary concern in deciding the success or failure of a mobile application. Therefore, there is a need to constantly adapt to the dynamic market trend in order to achieve a competitive advantage over projects, which can be gained by implementing an effective mobile application development methodology.

Charland and Leroux (2011) and Alsaqqa et al. (2020) argued that the need for mobile app development methodology could be attributed to the following assumptions: Mobile development methodology is critical in a situation where there is a need for the dramatic development of the production instead of the quality of the product. Mobile app methodology is needed to enhance flexibility for the client to have the capacity of changing the scope of the project at any time of need. In another situation, for instance, in agile methodology, the methodology is critical in understanding how the final product will appear. An effective mobile app methodology would be useful in enhancing adaptability, independence as well as detailed knowledge acquisition for the developers. Furthermore, having a detailed mobile app development methodology will ensure that the development process is streamlined and faster as well. It will promote flexibility in planning, customer involvement, continuous evaluation as well as management of the inherent risks associated with the mobile applications development (Duchting et al., 2007), (Wang, 2020).

Coinciding with these authors, and taking into account the characteristics, challenges and special requirements of mobile development discussed in the previous sections, we consider that it is essential to develop new methodologies specific to mobile development, and this is the purpose of this research.

3.2.6 Experienced Challenges while Developing Mobile Application

As ascertained by (Curran et al., 2019), it is undeniable that we are surrounded by technology all over more than ever before. Start-ups that are technically oriented are skyrocketing globally. Because smartphones are being used widely, many companies are investing a profound amount of money in mobile applications development to take advantage of the surging demand. Curran et al. (2019) compliments that as a result of the success of

several mobile applications in different areas ranging from gaming, lifestyle, social media related among many others, novel organisations are emerging. In these organisations, mobile app developers face a lot of challenges from developing and designing the app itself to making sure it is released and marketed effectively. The following analysis shows major problems app developers encounter while planning, developing and marketing mobile applications.

The first and greatest problem deciding on which app to develop. There is a stiff competition in the already overcrowded mobile application area, and it is even becoming more and tougher for an application to be noticed or stand out respectively. App developers are being pressured to develop more innovative apps that individuals will definitely acquire something from them (Mota et al., 2018). For this reason, generating a good idea for the app is one of the biggest challenges that should be covered in a development methodology for mobile applications.

Second, another core technical challenge in developing an app is confined to selecting the most appropriate development technology for the application and making decisions of making a hybrid, native or cross of a mobile application platform. A significant amount of research by Khaddage and Lattenman (2013) and (Biørn-Hansen et al., 2019) underpinned a detailed comparison with the developing technologies for mobile applications. Each technology has several shortcomings that most developers have no control over. This to date has not been addressed effectively and remains one of the prevalent challenges in mobile applications development. The mobile development methodology should not require a specific type of platform, but rather accommodate the particularities of the different technologies through its phases.

Dealing with distinct screen sizes and devices that mobile application needs to fit in possess a significant challenge for the mobile app developers as far as compatibility is concerned. Just designing and then developing a mobile app for a selected few screen sizes and devices remains not a good alternative. The underlying challenge is the development of an app that has the capacity of running fluidly across as most devices with as typical screen sizes as possible (Albuquerque Santos et al., 2021). The mobile development methodology should integrate tests with real devices to take this challenge into account.

On the other hand, (Tao et al., 2020) presented several security-related challenges emerging as a result of the wide utilization of mobile software applications in contemporary society. Moreover, the potential risk emerging from mobile devices as a result of the lack of best practices and software development principles were exposed. In the other side, enabling

the reuse of software across different platforms of mobile apps is quite difficult to design a context-aware application regarding hardware makers, OS platforms, computing platforms, and delivery methods. In addition, designing of mobile apps that are context-aware is another challenge. The context-aware is a process that uses any data relevant to the app such as user profile or user behaviour or even sensors data to improve the user's experience. The primary purpose of this process is to manage the complexities in providing applications across typical mobile platforms, and the underlying concept of the context in which changes are expected (Ennouamani et al., 2020).

The presented issues and challenges are crucial and should be taken into consideration in the different phases of the mobile development methodology to mitigate the impacts resulting from poor choices for the successful development of the mobile apps. Table 3.2 contains a summary of some of these and other challenges; which in our opinion have not been sufficiently addressed in the scientific literature.

Table 3. 2: Summary of the Challenges in Mobile Application Development

Encounters confronted by mobile developers while developing mobile applications.
Not being sure on which application is the right application to develop.
Inability to select the most suitable development technology.
The existence of different operating systems thus developers experiencing a challenge to select amongst them.
Incompatibility of the different devices and screen sizes for mobile apps.
The inability to clearly define the target market.
Security issues concerned with mobile app development such as malware and software or hardware fragmentation.
Inadequate funds in the mobile applications development.
Designing of mobile apps that are context-aware.
Reuse of software across different platforms of mobile apps.
Control of mobile application distribution when there is a lack of software and hardware platform configuration.

3.2.7 Why an Agile Software Methodology is Most Suitable for Mobile Apps Development

As stated by Morris and Maynard (2010) and Mollet (2021), making use of responsibility, self-organisation, the leadership of flexibility, and an agile mobile app development methodology enhance the development of high quality and suitable mobile applications. It significantly helps in the overcoming of a good number of the issues that the developers experiment while developing the mobile application as those concerned with seamless development, the demand of the users or the unexpected amendments from the customers in the course of the development. The fundamental advantage of utilising the agile methodology in mobile applications development is that it primarily involves continuous communication inside and between the groups. Agile methodologies depend on individual skills, so that they succeed. If these competencies do not exist, no process can make the project succeed. However, people working together with good communication and interaction perform tasks faster and better and have better ideas. Therefore, agile methodologies focus on increasing both the individual and collective capacity to obtain better results. Furthermore, agile processes are designed to prioritize each team's strengths, adapting processes to maximize their yield, extracting the maximum of each developer.

Agile teams are characterized by their own and intense organisational collaboration. Own organisation doesn't mean they do not have leadership, just that they have the freedom to organize themselves in different ways until they reach their goals. It is here the difference between communication and collaboration. The first is just about exchanging messages, sending and receiving some information, while the second is teamwork to deliver a product or make some decision. This collaboration facilitates the common form of disappointment about the app idea, the target market, or the technology used.

Flora (2018) stated that the continued interaction between the development team and the interaction between customers make the process more flexible and very much transparent. Therefore, at any given stage, if the customer raises the project demands, there is no impact on the remaining process.

As a summary, some of the fundamental advantages of agile app development methodology include the following:

- Proffers work conditions that are quite comfortable towards the team members.
- It is employed as a cohesive work approach.

- It provides regular updates for the customers concerning the elements of the projects.
- It follows an iterative approach that allows to address the impact of modifications throughout the entire life cycle of the application.
- It anticipates implementation updates and debugging to generate and maintain a satisfactory user experience.
- It enhances the design of sustainability and responsiveness in the usage of the app and the experience of the users.
- Each of the stages calls for testing in order to ensure that the product remains seamless and that it capacitates the launch of the product in a significantly short duration.
- It enhances flexibility and transparency of the process as a result of the constant interaction with the customers, the developers as well as the testers.

All these characteristics are of great value in mobile app development, where requirements are particularly subject to change, the design of the user interface is critical, and the user experience needs to be constantly analysed and updated

3.3 The Methodological Proposals Existing in the Scientific Community for Mobile App Development

As far as mobile app development approaches are concerned, Flora and Chande (2013) explained that the mobile explained that the software and mobile app development process primarily involves the process of dividing the work concerned with app development into different phases with the objective of making improvements on the design, management of the product as well as project management at large. For its part, Mitra, J. (2020) adds that the use of a mobile app development methodology could prevent the creation of vulnerabilities in mobile apps. With this aim, the methodology can support the complete software development cycle or focus on a part of this process. But, in any case, the methodology should involve predefinition of the artefacts and the variables developed and completed by a project team to maintain and develop the mobile application, as well as its main phases and tools.

The modern mobile app development process can be described as vaguely as agile. In the scientific literature we can find numerous works that present a mobile development, without mentioning the methodology followed, but presenting some characteristics from the agile methodologies. We also found proposals that offer software engineering methods, but which are not methodologies. For example, we can mention the Broke-implement method

(Bhatia, L., & Jain, B. (2019)), a simple method that is based on the agile methodology, as the project is divided into smaller parts. But instead of focusing on all modules simultaneously, the developer will deal with one module at a time.

Other proposed mobile app development methodologies as researched by Hamsini, Ganes, and Smitha (2016), include waterfall, incremental and iterative development, prototyping, extreme programming, rapid application as well as spiral development. Some individuals take into consideration a life cycle model as a more generalised term in categorising the methodologies in conjunction with the software development processes to mean the specific process selected by a certain organisation.

In the next section, we will explain agile approaches used in developing mobile apps more in detail, focusing on the few methodologies specific for mobile development and some non-specific methodologies that have been applied to this field with special relevance.

3.4 Different Approaches and Methodologies for Mobile App Development

When developing typical mobile applications using flexible and non-flexible technologies, researchers have proposed a number of agile methodologies. However, as is established in the systematic review of the scientific literature performed in (Jabangwe et. Al., 2018), the number of primary studies about software engineering process models for mobile app development is low and, generally, the rigor and relevance of these papers is low too. In this review a total of 20 primary studies were identified in which a complete engineering model for developing a mobile app was presented. After an analysis of the papers included in that systematic review of the literature, and our own review based on (Flora and Chande, 2013) (Flowers, 2017) among others, we selected the papers shown in the Table 3.3 based on their relevance and rigor.

Table 3. 3: Different Approaches and Methodologies in Mobile App Development

	Mobile Process	**Mobile development process description**	**Authors**	**Year**	**Agile**	**Not Agile**
1	Mobile D	An agile Approach for mobile app development	Abrahamsson et. al.	2004	XP, Crystal	RUP
2	RaPiD 7	Rapid production of documentation — 7 Steps	Dooms et al.	2005	AM	-

3	Hybrid: mE	Designing an agile methodology for mobile software development — A hybrid method engineering approach	Rahimian et. al.	2008	ASD	NPD
4	MASAM	Development process of mobile app SW based on agile methodology	Jeong et. al	2008	XP	RUP, SPEM
5	Scrum for app	Scrum to support mobile app dev projects in just in time learning context	Scharff et. al.	2010	Scrum	-
6	SLeSS	A Scrum and Lean Six Sigma integration approach for the development of software customization for mobile phones	Cunha et. al.	2011	Scrum	Lean Six Sigma
7	MADeM	Based on Mobile-D methodology, including UML modelling approaches	Alsabi & Dahanayake	2016	XP	RUP
8	Mobile Llities	Agile and Scrum methodology taking into account the specific needs, characteristics and problems of mobile development	Daniel Martinez et al.	2020	Scrum	-

3.4.1 Mobile-D

Mobile-D is depicted to be among the pioneering area of agile methodologies. The work was performed by Abrahamson et al. (2004), in which they suggested that agile development is an important part of the mobile application development environment. However, new agile approaches are needed to improve the mobile application software process. The authors claim that they also suggested a new approach which they called Mobile-D. This approach is basically confined to the developing practices (Programming XP), life cycle coverage (Rational Unified Process (RUP)) and Scalability (Crystal methodologies). The methodology inherently has five phases incorporated with different tasks, stages as well as practices. This includes Explore, Initialise, Productionise, Stabilise, System test and fix as outlined in Figure 3.1.

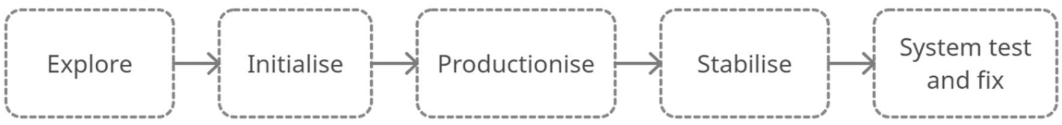


Figure 3. 1: The Five Phases of the Mobile-D Software Development Process

Mobile-D adopts some practices in the development process. The practices of the development phases are made up of nine main components. The following are the components (Abrahamson et al. (2004): 1.) Phasing and Pacing 2.) Architecture Line 3.) Mobile Test-Driven Development 4.) Continuous Integration 5.) Pair Programming 6.) Metrics 7.) Agile Software Process Improvement 8.) Off-Site Customer 9.) User-Centred Focus. Most of these components are common agile practices that have been customized to mobile app development.

3.4.2 RaPiD7

This method was proposed by Dooms et al. (2005). The Rapid Production of Documentation, seven steps (RaPiD7) primarily are believed to significantly improve work documentation without scarifying the quantity of documentation. The method primarily provides a description concerning the planned human interaction in all software projects thus initiating the development of documents within a facilitated workshop. Figure 3.2 outlines the seven steps involved in RaPidD7.

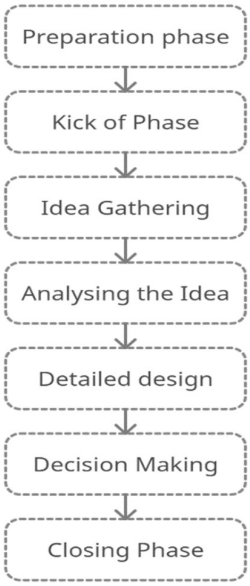


Figure 3. 2: The Seven Steps of the RaPidD7 Workshop Layer

Basically, the RaPiD7 methodology supports the entire software development project without taking into consideration if they are linked directly to mobile application development. Since the RaPiD7 was done in the Philips Digital Systems Laboratory, its development was initiated within the Nokia timeframe of 2002-2003. Despite its simplicity, since its main steps can be used in mobile application development, we added it to our research list. This decision is in line with the opinion of other authors (Flowers, 2017), who

also consider it relevant for mobile development, among other reasons because of its adoption in Nokia. Basically, the approach considers two agile practices: “Do the Simplest Thing That Will Work” and the “Whole Team.” It has been known to significantly improve the ancient approach in specifying work through the provision of a planned human collaboration in the initial phases in the software projects by allowing the means of decision making and joint document authorization with in-built quality assurance.

3.4.3 Hybrid Methodology Design

Hybrid methodology design was from the work of Rahimian et al. (2008). They presented a hybrid agile risk-based methodology that could provide a much more suitable method for mobile applications development. Hybrid Engineering Methodology contains seven phases 1.) Idea Generation. 2.) Project Initiation 3.) Analysis. 4.) Design 5.) Implementation. (Development Engine) 6.) Test.7.) Commercialization. As seen in Table 3.4.

In the Idea Generation phase, the idea discussion and production are concentrated. After that comes the Project Initiation phase, that focus on preliminary analysis. In phase three (Analysis) comes the detailed analysis and creation of a functional prototype. Later on, the team must start the architectural design and more detailed design (Design). In the fifth phase, they start the Implementation, where the adaptive cycle panning, concurrent component engineering, and updates to the component library. In the sixth phase comes the testing for quality review and market testing (Test). After all, they start the Commercialisation phase.

Table 3. 4: The Phases Involved in Hybrid Methodology Design

No.	Phases
1	Idea Generation
2	Project Initiation
3	Analysis
4	Design
5	Implementation (Development Engine)
6	Test
7	Commercialisation

The fundamental features of mobile software development for this methodology include software product line support, market consciousness, agility, architecture-based

development, and early physical architecture specification. The inclusion of learning and review sessions as well as support for reusability.

3.4.4 Mobile Application Software Agile Methodology (MASAM)

This methodology was proposed by Jeong et al. (2008), which was anticipated to provide the mobile app development process with the use of an agile approach swiftly. It is concerned with the Agile Unified Process, Software and System Process Engineering Meta-model (SPEM), Extreme Programming (XP) and Rational Unified Process (RUP). It is based on GUI architecture-centered having the capacity of using agile approaches in the utilization and dramatic development of the domain knowledge (Flora, 2018). MASAM methodology has four phases as outlined in Table 3.5 (Jeong et al. 2008).

Table 3. 5: The Four Phases of the MASAM Process

No.	Phases	Activity
1	Preparation Phase	Grasping Product Product Concept Sharing Project Setup
2	Embodiment Phase	User Need Understanding Architecting
3	Development Phase	Implementation and Preparation Release Cycle
4	Commercialization Phase	System Test Product Selling

The preparation phase contains three main activities: grasping product, product concept sharing, and project setup. The main tasks are to prepare a product summary and pre-planning for the project. Next, they do user definition and initial product analysis. While the project is set up, the development team should support development process coordination, project resource coordination, and pre-study the project Jeong et al. (2008). The second phase contains two main activities: user needs understanding and architecting. The main tasks are: story card workshop, UI design, Non-functional requirement analysis, architecture definition, and pattern management. Later on, the development phase contains two main activities: implementation and preparation with the release cycle. The main tasks are environment setup, development planning, release planning, iteration cycle, and release. Finally, the

commercialization phase contains two main activities: system test and product selling. The main tasks in this phase are acceptance test, user test, launching test, and product launching.

The MASAM methodology is recommended for small firms that fully concentrate on the development of mobile software applications. Nevertheless, the authors hardly provide a case study regarding the actual implementation of the method in an ideal environment (Jeong et al., 2008).

3.4.5 Scrum and Lean Six Sigma (SLeSS)

The SLeSS was developed by Cunha and his colleagues in 2011 (Cunha et al., 2011), which is an agile approach for mobile applications, that focuses on process improvement and management of projects (Flora, 2018). The method makes use of the Lean Six Sigma (LSS) product backload (improvement of processes) and the customization Product Backlog (customization of development projects). This approach is crucial to make easier the adaptation to changes in requirements in the final stages of a given project and with limited impact on the traditional approach. It significantly reduces overtime hours, assists in meeting deadlines, and shortens the development cycles. Moreover, the approach plays a significant role in the improvement of quality, achievement of quality targets and performance of the ideal software development project. The combination of Scrum and Len Six Sigma reduces the costs incurred in the production, enhances productivity, management process, development process overall as well as ensuring fewer failure and defects in the final outcome of the mobile app development process. The work is very interesting, but it is not a methodology, but the integration of existing methodologies. As indicated above, very little research has been done on specific methodologies for mobile development and in our opinion this effort is necessary.

3.4.6 Scrum for Mobile Application

Scrum for mobile application was the work of Schariff et al. (2010) in which the utilization of the Scrum in the class setting to develop a mobile app. This was studied at Pace University to develop a proposed novel model of working with Scrum methodology to develop a mobile app that involved an owner of the product in conjunction with a certified Scrum Master from the industry and students from the university.

The development team developed a mobile app that was designed specifically to study the emergent market in Africa regarding mobile use and the entire experience in implementing Scrum and its relevance in the development of mobile in detail. Therefore, it is not a new methodology, but rather the application of an existing methodology to mobile

development. Even so, it provides important conclusions, such as the suitability of the agile approach to develop this type of software.

3.4.7 MADeM

In 2016, Alsabi & Dahanayake (2016) proposed MADeM based on the Mobile-D methodology (Abrahamsson et al., 2004). MADeM is a methodology for SMART (Simple, Meaningful, Adequate, Realistic, and Tractable) modelling in lightweight mobile app development. MADeM tried to use a collection of specific models drawn from the SMART model and a methodology engineering approach in specific phases. These approaches include UML modelling approaches, activity diagrams, class diagrams, use case diagrams, and user view diagrams. Again, the proposal is based on the integration of two existing methodologies, whose union results positively in the development of a mobile application; but a new methodology is not defined, as is the purpose of this thesis, where all the phases and activities should be defined, structured and detailed, specifically thinking in the generation of a mobile product.

3.4.8 Mobile Ilities

Danielo Martinez et al. (2020) proposed the Mobile Ilities framework for mobile app development, which is built on Agile and Scrum methodologies. They considered mobile app development's unique needs, characteristics, and challenges. Connectivity, interoperability, flexibility, energy, heterogeneity of devices, platforms, and data Security are the aspects proposed by Mobile-Ilities as seen in Table 3.6. The authors used this agile Scrum framework as a work assignment for computer science students. They thought the results were promising and may serve as a guide for inexperienced developers during the app development process. However, generalizing these findings is difficult, especially given that the framework was used in an academic setting and the app developed was limited to a single university. Moreover, the developers had no experience with the iterative technique in general, had no connection with any clients, and had not conducted any user testing.

Table 3. 6: Mobile Ilities

Name	Description
Connectivity	The application should be accessible regardless of bandwidth limits or data connectivity.
Interoperability	The application should address resource sharing and

	communication between apps while using APIs.
Flexibility	The app should be adaptable to various mobile scenarios and user settings such as screen brightness, connection preferences, etc.
Energy	The application should lower the battery's power usage of the mobile device.
Heterogeneity of Devices	The application must work on both high-end and low-end devices.
Platforms	The app should ensure that users on various platforms have a consistent user experience.
Data Security	The program must handle internal and external threats, as well as weaknesses caused by user ignorance, device assaults, or purposeful abuse.

3.6 Pragmatic Approaches to Mobile Development

Very few proposals have been found of methodologies for mobile development, but a bigger gap exists if we focus on methodologies that consider a pragmatic approach to how companies/developers develop their apps. We only found a few experimental studies that have been exposed to the development of mobile applications in real contexts. Although none of them proposes a new methodology to develop these mobile applications, these studies helped us in developing our methodology, which will be presented later in this study.

For example, Flora et al. (2014) presented a survey to improve the mobile app development process. The mobile app development team members, agile experts, and researchers were the main applicants. This survey examined various agile approaches for successful mobile applications development. These agile approaches were XP, Scrum, and Lean. Furthermore, the survey was performed to determine the appropriate Agile practices used in mobile app development. Agile development practices include incremental iteration development, short development cycles, customer involvement, and adaptation to changes. The outcomes of the study illustrate how Agile naturally meets the demands of mobile app development and how agile methodologies have the potential to improve the efficiency and quality of mobile app development (Alrabaiah & Medina, 2021).

On the other hand, Kirmani (2017) shows that flexible strategies are suitable for mobile applications development. The challenges of the work were: flexible strategies are

reasonable for fast-growing industries, consumer loyalty is important, the transfer cycle is short, and there is cooperation appropriate between organisations and engineers. In this study, authors applied a survey that found 86% of respondents believe that agile methods and practices are appropriate for mobile application development. This study shows that obtaining these agile strategies is applicable to upgrading the speed and quality of mobile application development.

Furthermore, Kirmani M. (2017) indicates that agile methods are suitable for mobile apps development. The works issues were: agile techniques are ideal for fast-paced markets, customer satisfaction is crucial, frequent delivery is important, the delivery lifecycle is short, there is proper communication between companies and developers, and excellent design and simplicity are important too. According to the findings, 86% of survey respondents believe agile methodologies and practices are appropriate for app development. Adopting these agile practices can improve the time and quality of app development.

Finally, in an empirical study on mobile applications, Arshad et al. (2018) examined the main challenges during the development process of native, web or cross-platform hybrid mobile applications. The authors identify the main challenges through a systematic literature review as a first step. After that, they seek to find further issues and challenges as a second step by gathering data from mobile application developers through structured interviews, allowing practitioners to describe more challenges. According to this study, the most prevalent challenges in mobile application development are fragmentation, testing, user experience, compatibility, and change management. The participants identified critical difficulties in the interviews were fragmentation, testing, reuse of code, lack of tool support, lack of expertise, and change management. Interviews revealed additional issues not addressed in the literature review, such as a lack of training, a lack of teamwork and enrollment, a lack of communication, and a lack of knowledge management. The authors conclude that these challenges and problems should be considered successful and effective applications. They believe that identifying these difficulties will assist teams in implementing mobile applications. They suggest that practitioners should pay more attention to the issues that have been recognized in academics and business (Alrabaiah & Medina, 2021).

3.7 Summary

From the literature review, it is clear that mobile app development is critical in contemporary. It is the result of the increasing utilization of mobile applications. Many mobile applications are already in the market. However, as observed from the research

review, mobile app developers experience a good number of challenges when developing mobile apps. This has been discussed and reviewed in detail in the chapter. Consequently, several mobile app software development approaches have been presented and discussed in detail, with the various processes and steps outlined (Alrabaiah & Medina, 2021). With this review, the lack of works that present complete and rigorous methodologies for mobile development is evident.

Moreover, when these mobile applications are developed by professionals, all of these applications should follow a well-defined process of developing. With this aim, due to the growing popularity of smartphone application development, development teams must limit themselves to using best practices to develop their applications to reduce development time, workload, and cost. The rise and growth of mobile applications and the specific features and requirements of mobile software development impose certain limitations on using existing software development methods. Because this type of software has unique characteristics and conditions, new practical approaches are needed. At this time, we believe that agile software development methods provide a good platform for developing mobile applications. In our opinion, however, the agile methods introduced so far are not enough to fully meet the unique requirements of mobile applications.

CHAPTER 4. EXPERTS-BASED STUDY IN MOBILE APPLICATION DEVELOPMENT

According to our study and analysis of the literature review, few scientific papers address or identify organisations' unique challenges during the mobile app development process. Moreover, the studies based on the observation of the mobile application development processes followed by experts, academics and researchers in this field are fewer. This is consistent with the findings of both (Ghandi et al., 2017) and (Flora et al., 2014) that few studies cover this topic. Very few research papers identify the features of the mobile application development process, the issues and problems that arise during the application development process, and the tools used (Jabangwe et al., 2018); best practices to keep in mind while creating a mobile app, the benefits, drawbacks, and concerns of using Agile in mobile application development are also not sufficiently considered. These concerns lead us to the fact that one of the main objectives of this research is to generate a new methodology to systematise the development process of mobile applications.

This chapter will present the most important results of expert-based data analysis, which will be the basis to build our methodology for developing small to medium mobile applications. We begin our research to achieve our objectives by interviewing experts in developing mobile applications, which is covered in the first section of this chapter. Later on, in the second part, we will talk about the distributed questionnaire to experts and academics or researchers in developing mobile applications to extract valuable data about this process. Each section contains four parts: Study Population, Research Methodology, Main Results, and Data Analysis.

4.1 Study Through Interviews with a Group of Companies

In this section, we want to take a closer look at the process of developing mobile applications used in real companies, as well as the complications and problems associated with this process. To this end, we interviewed 14 mobile app development experts, focusing on the development process they followed, the tools used in the process, and general

questions and concerns. From here, we draw conclusions that may be of interest to bring the business world and the academic community closer.

4.1.1 Study Population

The interviews target were experts in building mobile apps. We contact with 70 mobile app development companies, mainly from Spain and Jordan. Finally, we interviewed 14 experts from 13 different companies.

4.1.2 Research Methodology

One of our study's objectives is to understand better the mobile apps development process in the industrial market. Therefore, this part will focus on the apps development process that some mobile app companies follow.

Conducting interviews is widely used during the assessment of the software development process (Carlsson et al., 2007). Technical interviews are an effective method as many companies are involved in interviewing software engineers and programmers that may help or improve the quality of the desired software. Considering our objective in this research, we have chosen to conduct a qualitative study by looking for companies in our area (Spain and Jordan) that specialise in developing mobile applications. Also, we used the semi-structured form of interviews as it is more prevalent in qualitative research (DeJonckheere and Vaughn 2019). This type of interview enhances more ideas and questions during the interview. Next, we asked the candidates in these companies to conduct a semi-structured interview. The interviewer had a list of questions covering some key points. Each interview session took about 25 to 40 minutes. A summary of participants' information can be found in Table 4.1, where PM is Project Manager, App Dev is App Developer, ES is Spain, and JO is Jordan.

Table 4.1 shows that most of the participants are project managers and application developers. It can also be noted that the experiences were between six to ten years in apps development, the size of the companies ranging from small to medium, and the location of these companies was mainly between Spain and Jordan.

We were satisfied in interviewing 14 experts in mobile apps development as the new interviews will add little input to our research. Furthermore, we will distribute a questionnaire to expand our research. In each interview, we wrote down everything we talked about during the sessions, recorded the audio during the interview, and saved it for future analysis. The interview's core topic was the methodology used in the company to develop a mobile application. Further, the interviews focused on mobile application development's main tools

and challenges. General questions were also asked about the company, the development team size, and experience. From the interview transcripts, we analysed each data and information we had. We listened many times to the audio record for best data analysis after each interview until no additional data was collected from the interviews or new knowledge. We break down interview transcripts into small units to generate subtitles for each company's main phases of developing mobile apps.

4.1.3 Main Results

In this section, we will present a summary of each interview. To maintain information and data privacy, we present the company name and the person we met just with the first alphabet in their names. In each interview, the objectives of this research were clearly stated, and all participants were accepted to be part of it by their choice and consent.

A. First Interview

We interviewed M.A.M.L (ID1) in the C.S. S.L.N.E company. He is a project manager and business owner. The company has eleven to twenty years of software development experience, including six to ten years of software app development experience. The company employs from twenty-one to fifty workers and has developed from 5 to 20 mobile applications.

In this company, mobile apps development has six phases: Planning phase, Prototype phase, Design phase, Development phase, Testing phase, and Launch and Deployment phase. Finally, support and maintenance will be considered. Each phase is briefly described below. This same structure will be used to describe the rest of the interviews.

The development phases in C.S. S.L.N.E company*:*

Planning: Make a roadmap for the customer's mobile products. It is used to plan the intended application as well as the time and cost requirements developed with the customer. Requirements are gathered directly from the customer's IT department. If they do not have, they will move on to the next phase, the “prototype phase,” to find out the requirements directly from the customer.

Prototyping: Gather customer needs by drawing, modelling, and creating wireframes using mock-up tools called UXPIN. They repeat this step until approved by the customer.

Design: Starts the design for screens, which helps better understand customer requirements until it gets approved by the customer. Create a home screen that allows to better understand the customer needs until their consent is obtained.

Table 4. 1: Participant's Information

Interviews Summery	**ID1**	**ID2**	**ID3**	**ID4**	**ID5**	**ID6**	**ID7**	**ID8**	**ID9**	**ID10**	**ID11**	**ID12**	**ID13**	**ID14**
Role	PM	PM	PM	PM	App Dev	App Dev	PM	App Dev	PM	PM	App Dev	PM & App Dev	PM	App Dev
Software Experience years	11-20	6-10	11-20	6-10	11-20	11-20	6-10	11-20	6-10	11-20	6-10	11-20	6-10	6-10
Mobile apps Experience	6-10	5	5	4	6-10	6-10	6-10	6-10	6-10	6-10	6-10	6-10	N. A	6-10
Company team size	21-50	21-50	11-20	11-20	6-10	5	11-20	6-10	6-10	6-10	11-20	6-10	6-10	1-5
Number of apps developed	5-20	5-20	5-20	5-20	20-30	20-30	5-20	20-30	20-30	20-30	20-30	20-30	10-20	5-20
Location	ES	ES	ES	ES	JO	JO	JO	JO	ES	JO	JO	JO	ES	JO

Development: After the customer approves the screens, they start building the Data Model and Rational Unified models and use the minimum documentation during all processes. Then they begin building databases, ORMs, business logic and web services using SOAP and REST. Build and code web apps with cross-platform tools like jQuery mobile, Intel XDK, HTML5, Angular, PhoneGap, Apache Cordova, etc. They will continue to provide customer feedback and suggestions during development until this step is completed.

Testing: In this phase, they use the simulator provided by the development tool used in the development tool. This company uses no special testing tools or simulates on a physical device or mobile phone.

Launch and Deployment: When the app is ready, completed and approved by the customer, the mobile product will be launched and published on the Google Paly, Apple Store, or both.

Support and Maintenance: This company provides free technical support for any application bugs during the warranty period, and each new update means new requirements, new contracts and deals as a new project.

B. Second Interview

We interviewed A.M (ID2) in the S.S company. He is the owner of the business and has the role of project manager. The company has six to ten years of software development experience, including five years in mobile application development. The company employs from twenty-one to fifty employees. They developed from five to twenty mobile applications. S.S uses project management tools: Kanbanchi, Task Management, and Project Management.

In this company, mobile app development is divided into four phases: Prototype phase, Development phase, Testing phase, and Launch and Deployment phase.

The development process at S.S. company:

Prototype: This phase is for the basic analysis and application design. They gather customer requirements by drawing and creating mock-ups using a modelling tool called UXPIN and start designing screens (this helps to understand customer needs better until they are approved). Sometimes interactive prototypes are made.

In addition, sometimes they use Adobe Illustrator to design all of their screens, get full-screen flowcharts, and print them in PDF format to get contextual descriptions of the screens. The team will then receive final approval from the client.

Development: After the client approves the screen, they will try to develop 80% of the application in 20% of the time required for the entire project. Starting with basic system

development (the system's core), most importantly, using native development tools like Xcode and Android Studio. Non-interactive prototypes or even interactive prototypes help develop the core of the system quickly, and working in parallel can help too. However, the development process is still based on customer feedback and suggestions prior to completing this phase.

Testing: They take advantage of the simulator provided by the built-in development tools. This company uses the simulators that are provided with the same development tool.

Launch and Deployment Phase: When the app is ready and complete, the customer will approve it. The mobile application will be launched in the Apple Store or Google Store.

Support & Maintenance: Free support for all application bugs.

C. Third Interview

We interviewed J.B (ID3), who works at C.N. He runs as a Scrum Manager and has the role of project manager. The company has eleven to twenty years of experience in software development, including five years of mobile application development. The company employs eleven to twenty employees, and five to twenty mobile applications have been developed. This company uses Scrum to build mobile apps.

C.N. company use Trello as a project management tool. Mobile app development is divided into five phases: Requirement Gathering, Product Backlog, Sprint Backlog, Start Sprint (including Prototype and Graphic Design, Development, Testing, and Customer or User Feedback), and Launch and Deployment.

The development phases in C.N company:

Requirements Gathering: Start by talking to the customer and writing a summary of all the requirements and tasks for the application, then tweak the settings until the customer signs up. According to this briefing, they can analyse the project time and cost and then start using Scrum as a development method.

Product Backlog: Create a priority list. Using project management tools like Trello, they can view a list of all tasks classified as A, B, C or D (most important to least important). Then he assigns 100 to 5 points to each question depending on the time needed (100 points needs more time to develop) and finally starts the first Sprint.

Sprint Backlog: The Sprint Backlog defines team tasks to transform the Product Backlog selected for the Sprint into increments of potentially valuable products, including incomplete team building tasks.

Start Sprint: Start with the basic application requirements. Each Sprint cycle lasts from 1 to 7 days. The first Sprint produced a beta version of the application. They will choose the main app features to provide the client with a beta app during the first Sprint. Each Sprint consists of four stages: Prototype and Graphic Design, Development, Testing, and Customer or User Feedback. There will be 10–15-minute meetings each morning to discuss daily projects and new tasks. Developers and designers do the same tasks together and make physical contact and communication via Google chat.

Launch and Deployment: After the application is completed and approved by the customer, the mobile product will release and get launched on its native store.

Support and Maintenance: Free support for any application errors during the warranty period.

D. Fourth Interview

At A.W, we interviewed R.S.R (ID4). He is the owner of the company and works as a project manager in his own company. The company has six to ten years of software development experience, including four years of mobile application development experience. The company employs between twenty-one and fifty people and has developed from five to twenty mobile applications. They use Asana as a project management tool, OwnCloud for accessing and sharing files, contacts, calendars, and chat, and Scrum as a management method. They have a Scrum Manager course.

In this company, mobile app development is divided into six phases: Planning phase, Prototype phase, Design phase, Development phase, Testing phase, and Launch and Deployment phase.

The development phases in A.W company:

Planning: Creating a plan for the client's mobile product and establishing the roadmap, planning the expected application, as well as the time and cost requirements to be developed with the client prior. This phase is the initial phase of collecting application requests directly from the client. In the end, the requirement gets signed by the customer.

Prototyping: In this phase, they seek to clarify and formalize the requirements for the application, then proceed to draw and create wireframes and mock-ups using tools such as Sketches and Zeppelin, prior to customer approval.

Design: Integration of screen design, system, and data architecture. Sketch, Zeppelin, Adobe Illustrator, and Dia model database are used to design the mobile application development. The development phase will not start until they get the client's consent.

Development: After the client approves the app screens, the client, system engineer, and data architect are ready, they begin to develop the mobile application. They use native tools like Swift, Xcode and Android Studio to build their apps. They send clients new versions every fifteen days to keep them in progress and allow them to make changes or provide feedback, which they can do now because it's much more difficult to make those changes at the end of the project. They receive feedback and suggestions from customers during the development process until the end of this phase.

Testing: This phase usually involves third-party testers. There is no dedicated testing tool or simulation on a physical device or mobile phone.

Launch and Deployment: After the completion and approval of the App by the customer, the App is launched in the Apple Store, Google Play or both.

Support and Maintenance: Free support is provided for any application errors during the warranty period. Each new update means new requirements and requires new contracts and deals, such as a new design.

E. Fifth Interview

At this company S.A, we interviewed A.J (ID5). He owns the company and works as a project manager. The company has eleven to twenty years of experience in software development, including six to ten years of experience in mobile application development. The company employs six to ten employees and has developed twenty to thirty mobile applications. According to respondents, the first problem they struggle with is that more than 80% of customers do not know what they want (they come just with a general vision, which makes requirement gathering difficult). On the other hand, the company uses project management tools like Trello. In this company, mobile app development consists of six phases: Planning phase, Prototype phase, Design phase, Development phase, Testing phase, and Launch and Deployment phase.

The development phases in S.A company:

Planning: Defines the roadmap for the customer's mobile products. This step begins with speaking directly to customers and filling out paperwork that reflects their needs. Subsequently, questions are answered at follow-up meetings regarding the requirements and expected conclusions. Once the acceptance conditions have been met, the customer must be signed. Thus, at this stage, the expected application and time requirements, as well as the cost planning developed with the client, are carried out until the client signs them.

Prototyping: In this phase, they try to clarify and formalize the application requirements, and then proceed to design and create wireframes and mock-ups until the customer approves. The use of interactive prototypes is discouraged as clients will feel that the application is almost complete or will not be able to create the same interactive prototype.

Design: Start designing the main screens, which consists of three to four screens that have the main app functionality, systems, and data architecture. They will not start programming until they get the client's consent. They can then move to other screens. They use Photoshop for design.

Development: After the customer approves the application screen and completes the system and data architecture, the application programming begins. They use native tools like Swift, Xcode, Android Studio and Android Kotlin to build their apps, but they use Unity3D for game apps. They do not recommend using cross platforms. Each time they send customers new demos to keep them updated and allow them to make changes or provide feedback. At the same time, they will send these samples for testing to the Quality Assurance (QA) department. They use a shared Excel spreadsheet to communicate with all teams during the bug fixing process (they have a small team with less than ten members). They receive feedback and suggestions from customers during the development process until the end of this phase.

Testing: During this phase, the quality assurance department is responsible for testing. There are no dedicated testing tools or simulations on a physical device or mobile phone.

Launch and Deployment: After the app is ready, completed and approved by the customer, the mobile product will be launched in the Google Play, Apple Store, or both.

Support and Maintenance: Provide free support for application errors during the warranty period. Each new update means new requirements and requires new contracts as a new project.

F. Sixth Interview

In IN. company, we interviewed R.M.M (ID6). He is a mobile app developer. This company has eleven to twenty years of experience in software development, including eight years of experience in mobile application development. The mobile apps development team contains five members. This company has between eleven to twenty workers and developed more than twenty mobile applications.

In R.M.M company, they use Trello tools for project and task management. This company uses Agile methodologies and Kanban in the mobile apps development process. IN.

company develops native and web app technology. In this company, mobile apps development has five phases: Prototype phase, Design phase, Development phase, Testing phase, and Launch and Deployment phase.

The development phases in S.S company:

Prototype: The application development process begins by meeting with the customer and getting a general idea of the application to be developed. The requirements are then divided into main sections, from most important to least important, and these sections are summarized in a Word file. The wireframe design is then drawn and discussed with the client. According to R.M.M., drawing the application's wireframe makes it easier for them to take customer requirements, especially when the customer does not complete the idea. Screen size, location, components, and priority are discussed with the customer in this session.

Design: After getting the application requirements and drawing the wireframe and screens, they are sent to the design department. There is a page-by-page or model-by-model app layout; each model can contain multiple pages. The Balsamiq and Photoshop tools are used to assist in developing the structure and final design of the application. After finishing the app design, they do a test before they send it to the development department. App architecture and logic are about the app data-flow, test data-flow, back-end calling, and models.

Development: The design is sent to the development department. App development starts with agile practices. The application has been developed page by page or model by model according to the importance of each model.

Testing: They perform a usability test to be as user-friendly as possible. They test model by model, then do a regression test to ensure that all the components of the application work together. They do the testing inside the company. The client's presence in the application development process is up to the client, as some clients do not want to cooperate or assist. In this case, each model is created and sent to the client to view and test using tools like TestFlight for Android or Beta app in the app store. They do a manual on-device test.

Launch and Deployment: After the app is ready, completed and approved by the customer, the mobile product will be launched and released on its native store.

Support and Maintenance: They provide one year of free support and create service plans for an additional fee. Each new update means new requirements for new contracts and transactions, as well as a new design like a new project.

G. Seventh Interview

In J.T. company, we interviewed S.N (ID7). He is a project manager and a business owner. This company has six to ten years of experience in software development, including eight years of mobile application development experience. The mobile apps development team contains five persons. This company has between eleven to twenty workers and developed more than twenty mobile applications.

In J.T. company, they use Trello tools for project and task management. They do not follow a specific methodology for building a mobile app but use Agile practices in the mobile app development process.

J.T. company develops hybrid cross-native platform and web app technology.

In this company, mobile apps development has five phases: Prototype phase, Design phase, Development phase, Testing phase, and Launch and Deployment phase.

The development phases in J.T. company:

Prototype: An initial meeting is held with a customer to hear their requirements and create a document that specifies what the app looks like and how much it will likely cost. After the client's initial approval on the estimated overview documents, the process of taking the client's requirements begins precisely with an attempt to draw some wireframes for the application to help understand the client's wishes and what we can implement to achieve the actual requirements. After that, the client signs the agreement.

In this phase, it is essential to know the app's objective, which helps the developers give the customer more suggestions.

Design: After getting the application requirements and drawing the wireframe and screens. They started to design every page using a free tool called Figma; sometimes, they used Photoshop to assist in developing the structure and final design of the application. They have a meeting with the client to discuss each app's pages to get final approval. After finishing the app design, they do a test before sending it to the development department.

Development: The design is sent to the development department. The application will be developed page by page, starting with the essential pages. They develop a hybrid cross-platform using Ionic. They use a CSS tool before changing it to a web app. If the project is large, they will divide it into smaller parts using Sprints in the development process. They use the Ionic tool in the development process.

Testing: They do not have a test team and do interlay tests in the company. The development team is responsible for testing, but that could affect quality. They perform functional tests. They test model by model (all the pages in this model), and then do a

regression test to ensure all the application components work together. They do the tests within the company. They ask the client to cooperate and assist in the testing phase; they create beta test versions and send them to the client to view and test them. They do a manual on-device test.

Launch and Deployment: After the app is ready, completed and approved by the customer, the mobile product will be launched and deployed at the native store.

Support and Maintenance: Free support for possible application errors is provided during the one-year warranty period, and a service plan is created with a 20-hour plan as a starting point. Customers may pay an additional fee for extending the service period or any new updates or maintenance. Furthermore, new updates mean new requirements for new contracts and deals, such as a new project.

H. Eighth Interview

In T.M. company, we interviewed M.A. (ID8). M.A is a business founder and native mobile app developer. This company has seven years of mobile application development experience. The mobile apps development team contains five persons. T.M. developed more than twenty mobile applications.

In T.M. company, they use Trello tools for project and task management.

This company does not follow a specific methodology for building a mobile app but uses Agile practices in the mobile apps development process. T.M. company develops a native app platform. In this company, mobile apps development has five phases: Concept phase, Graphic Design phase, Development phase, Testing and Quality Control phase, and Launch and Deployment phase.

The development phases in T.M. company:

Concept Phase: At this stage, they believe that the main objective is to identify the customer and the application's needs in general. An interview is created with the client about their needs and what they want the application to do. They usually have closed-end questions whose answer ends with yes or no, for example, if the client wants the payment to be through the application. Considering there is no boredom in this meeting, they believe to be careful not to lose the client. After that, a closed session is held between the business analysis and the project manager to discuss the client's meeting. They will try to get new ideas for the application and classify the essential requirements. They create wireframes for the core app pages. By the end of this phase, a proposal file is created, containing an overview of the company, its obligations (quality, ethics, and the use of the latest technology), a list of the

application's characteristics and wireframes, an administration panel (Content management system CMS) if required, time, and cost.

T.M. will start the development of the application after signing with the client on the proposal.

Graphic Design: This phase contains two sub-phases, the User Experience (UX) and the User Interface (UI). UX plays an important role in the success of the project. It covers the application's structure, the data flow, the buttons' location, the slide bar that exists or not, backward, or forwards. User Experience can give the user the first impression of the application. After finishing the UX, they start the UI phase. The application UI design process begins page by page with all customer needs in mind. They are considering the data flow that the client previously approved. The UI is designed with different resolutions to be compatible with potential phones. They sometimes use InVision tools to create a tappable prototype. They always share the app design with the customer and take their feedback. After finishing the app design, they do a test before sending it to the development apartment. They are using Photoshop and Illustrator as design tools.

Development: After getting the design's approval, the developers start to develop the app's back-end. The application will be developed page by page, beginning with the minimum essential pages. They are working in Sprints, one week to two weeks. After every Sprint, they send the result to the customer for review. They prefer to wait after finishing the design of all pages to start the development phase.

Testing and Quality Control: The application is submitted to the quality assurance department for the application testing process. Its main task is to search for all the bugs and errors in the application, and finally, a report is sent containing the errors found and how they occurred. This report will send back to the developer. They perform functional tests, unit tests, and regression tests. They do the tests within the company. Sometimes they do a manual on-device test. When all bugs and notes are solved, they will have a meeting with the customer to sign and deliver the app.

Launch and Deployment: Once the app is ready, finished, and approved by the client, the mobile product will be released and launched in its native store.

Support and Maintenance: They give free support for any errors in the apps during the guaranteed time for one year, and they do a maintenance plan will be with additional cost, and any new updates mean new requirements and need new contract and deal, like a new project.

J. Ninth Interview

In N.S. company, we interviewed M.T. (ID9). M.T. is a business founder and project manager. This company has seven years of mobile application development experience. The mobile apps development team contains five persons. M.T. developed more than twenty mobile applications.

This company develops parts of ERP systems (enterprise resource planning). ERP system helps businesses handle day-to-day tasks such as purchasing, accounting, employment, and project management (NA, 2022). They take one element from ERP, such as employment and convert it into an employee self-service application (employee system). For example, the employee department contains invoices and salaries of the employees, registration of the entry, exit, etc., or a manager's section contains (preparation of sales invoices, assistance during working hours, etc.). They will develop an app with just one ERP section (employee, managers, etc.). The requirement's gathering process is carried out differently since the system is already implemented. They only have to understand the comprehensive system and convert it into an app. This company does not follow a specific methodology for building a mobile app but uses Agile practices in the mobile apps development process. N.S. company develops a native app platform and hybrid native cross-platform using Ionic. In this company, mobile apps development has five phases: Data collection phase and System Analysis, Development phase, Testing phase, Launch and deployment phase, Marketing phase.

The development phases in N.S. company:

Data Collection: The application development process begins by collecting all the data taken from the ERP system, the basic needs, and the most used in this system.

System Analysis: Start to understand the system and its data-flow and architects profoundly, complete the business process design (e.g. how the employee departments are working in the company) and business unit design, and draw the data-flow for each business unit.

Development: The development has two main parts, the front-end development, and the back-end development. The front-end contains the UX and the UI design. After finishing the design of the app front-end, they are sent to the back-end developers.

Testing: They start applying Sprints between the quality assurance (QA) team and the back-end developers until they have the final product. A software quality analyst is responsible for using software quality assurance concepts and techniques to decrease software failure risk in the development process.

Launch and Deployment: Once the application is ready, completed, and approved by the customer, the mobile product will be launched and deployed in the native app store.

Marketing: After filling out the application, they try to find potential customers and give them away for a month for free.

Support and Maintenance: They provide one year of free support and create maintenance plans for additional costs. New updates mean new contracts and transactions requirements, such as new projects.

K. Tenth Interview

In N.S. company, we interviewed M.T. (ID10). M.T. is a business founder and project manager. This company has eight years of mobile application development experience. The mobile apps development team contains more than five persons. M.T. has developed more than twenty mobile applications. In N.S Company, they use the JIRA tool for project management and task management. This company does not follow a specific methodology for building a mobile app but uses Agile practices in the mobile apps development process. N.S. company develops a native app platform. In this company, mobile apps development has six phases: Data Collection phase, System Analysis phase, Development phase, Testing phase, and Launch and Deployment phase, and Marketing phase.

The development phases in N.S. company:

Data Collection: The phase objective is to identify the customer and the application's needs. An interview is created with the client about their needs and what they want the application to do. Usually, customer requirements are high and changeable. Consequently, the customer is asked that these requirements need a more profound analysis before submitting a project proposal. At this stage, sufficient time is given to understand and analyse the application requirements before moving on to the design phase and drawing the application pages. A closed meeting is held in the company to discuss the client's requirements. In some cases, the customer is offered to divide the application into phases; the first phase contains the main working part. In the next stages, a set of new services will be added to the application.

System Analysis: In this phase, the company start to understand the system and its data-flow and the system architects profoundly and complete the business process design and business unit design. And draw the data-flow for each business unit. They create the wireframes for the main app pages. By the end of this phase, a proposal file is created, containing a list of the application's characteristics and wireframe, time, and cost.

Development: The development has two main parts, the front-end development and the back-end development. The front-end contains the UX and the UI design. Sometimes they can start to develop the app based on the wireframe. They do not need to wait till all the UI is completed. They develop the main content of the app in the first phase. If the customer wants to add more service to the app (with delivery, tracking map), it can be added in the next phase. They use Sprints in the development phase. They use native development tools.

Testing: They start applying Sprints between the QA team and the developers until they have the final product. The customer is always included in the development and testing process.

Launch and Deployment: After the application is ready, completed and approved by the customer, the mobile product will be released as a beta version for three months. It will be released at the native store later.

Marketing: In some cases, they develop products that need to find potential customers and are responsible for finding the right customers.

Support and service: During the one-year warranty period, free support is provided in the event of possible application errors, and a service plan is created for additional costs. Each new update means new requirements for new contracts and transactions and a new design.

I. Eleventh Interview

In S.W company, we interviewed R.A (ID11). R.A. is a business founder and app developer. This company has ten years of mobile application development experience, with eleven to twenty workers. The mobile apps development team contains more than five persons. S.W. developed more than twenty mobile applications. This company does not follow a specific methodology for building a mobile app but uses Agile practices in the mobile apps development process. S.W company develops a native app platform, and hybrid cross-platform using React Native.

In this company, mobile apps development has six phases: Requirements Gathering phase, Design phase, Development and Structuring phase, Testing phase, and Launch and Deployment phase.

The development phases in S.W. Company:

Requirements Gathering: A meeting with the client is held to obtain the application's requirements and objectives. Often the customer has some requests, and their application is very expensive or unsuitable for the application's nature. Here the customer is oriented on

what is appropriate for the project and the customer himself, and the total cost of the application. Also, some applications need a parallel website. In this case, the application's development will be independent of each other, but there is coordination between the two teams. Subsequently, a kick-off meeting begins within the company. This meeting will identify the team and the team's readiness, present the application's idea to be developed, and distribute the most important tasks and tools used in the development process. To-do items are listed, described, and prioritized. Work is defined for the first Sprint.

Design: The application UI design process begins page by page with all customer needs in mind. They take into account the data flow previously approved by the client. After the end of each phase or Sprint, customer feedback is taken. After finishing the app design, they do a test before they send it to the development department. They are using Photoshop in UI Design. When they have the customer design approval, they send it to the development phase.

Development and Structuring: Small to mid-size apps does not need much structuring. They develop native apps and hybrid cross-platform using Facebook React Native. Back-end development and database are built. They started to develop the app by sections and sectors (could be pages). After completing each section, the result is sent to the quality assurance (QA) team for review. The development department will modify the app according to the recommendations of the quality assurance QA team. They use Sprints in the development phase. They use native development tools.

Testing: They start applying Sprints between the QA team and the developers until they have the final product. The customer is always in the development and testing process. They test on multiple devices (low specification devices, high specification devices, tablets). Sometimes they do automated testing; according to them, it is costly and takes time. They create a testing link with the customer in this phase.

Launch and Deployment: After the application is ready, completed and approved by the customer, they ask the customer to create a Google Play account and Apple Store account to deploy the application.

Support and maintenance: During the one-year warranty period, free support is provided in the event of possible application errors, and a service plan is created for additional costs. New updates mean new contracts and transactions requirements, such as new projects.

L. Twelfth Interview

At W.N.T company, we interviewed S.L and W.D (ID12). S.L. is the project manager and is the founder of W.N.T., while W.D is an application developer. The company has eleven to twenty years of experience in software development, including six to ten years of experience in mobile application development. The company employs six to ten employees and has developed from twenty to thirty mobile applications. This company uses Trello as a project management tool.

In this company, mobile app development is divided into six phases: Requirements Gathering, Design and Back-end phase, Development phase, Testing phase, and Launch and Deployment phase.

The development phases in W.N.T. company:

Requirements Gathering: Defines the roadmap of the customer mobile app. This phase starts with talking directly with the customer. Some customers have an existing business and want to build an application to serve this business. Or, some companies have a website and want to create an application for this page. Some companies have an information technology department. The development team can speak with the manager of this department to understand the application's requirements. Some clients come up with a general idea of what they want to put into an app. A meeting follows this for a more in-depth analysis of the requirements, in which many questions are presented about the application and some possibilities of some cases, and how the application should respond in case these cases occur. Questions about the application's goals, such as whether the client want to earn money through this application? Some features are added to the app through the responses, and some advice is given to the customer. This leads to improved trust with the customer. And, of course, the final choice for adding and removing features is for the customer.

After understanding the application's framework and scope, the proposal is presented, including the initial scope, time, and cost with the contract to be signed by the client. This gives the client an idea of what will be agreed and developed. The company is trying to clarify the scope of the application as much as possible, to avoid many changes in the future, not for the sake of changes, but to control the cost.

Design and Back-end: When the detailed scope is signed, the UX and UI design process begin, parallel with the back-end development, database, and API. In the UX and UI design, usually, two to three pages are prepared to agree with the client on the general application form. After approval of this design, they complete the rest of the app pages. They

use Photoshop and illustrator in UI Design. When they have customer design approval, they send it to do the development phase.

Development: They develop native apps and hybrid cross-platform. The process of choosing the platform's development depends on the desire, the needs or the client's financial capacity, since the native development is the highest cost.

Testing: They start applying Sprints between the QA team and the developers until they have the final product. The customer is always included in the development and testing process. Later, they ask the customer for testing the app. They do test on multiple devices and unit testing.

Launch and Deployment: After the application is ready, completed and approved by the customer, they ask the customer to create a Google Play account and Apple Store to deploy the application. They use the tools for monitoring the app's crashing.

M. Thirteenth Interview and N. Fourteenth Interview

The next two interviews were with specialists in developing games for mobile applications. Games are a special type of software that requires specific methodologies to integrate the ludic and narrative parts. Therefore, it will not be summarized in the same way as before. Only some of what is most useful to us will be taken.

In this company, O.G, we interviewed O.D (ID13). O.D. is a product manager. This company has six to ten years of experience in software development, including five years of experience in mobile games application development. This company has between six to ten workers and developed between ten to twenty mobile games applications.

In the other company, 3D.G, we interviewed W.Y. (ID14). W.Y. is a game app developer. This company has six to ten years of experience in mobile games application development. This company has just one worker and have developed between five to twenty mobile games applications.

In the two interviews, it was found that the ID13 and ID14 agreed on some things regarding the development of game applications, such as: mobile games' development and design differ because it is difficult to define the requirements since it may have started with one idea and ended with different ones. Sometimes a video of existing mobile games or previous designs is presented to communicate what the customer wants in the requirements. On the other part, the requirements gathering and graphic design phases overlapped and took more time than typical mobile applications.

It was emphasized in the interview with ID14 that when the design is clear, this greatly facilitates the development-coding process.

ID13 explained that the team's heterogeneity is a big part of the problem since each of the designers' and developers' compatibility helps facilitate and improve the development process's quality. That distinguishes game applications, the quality must be very high, as the user does not accept the existence of errors and bugs in the game.

Common development phases in O.G. and W.Y. companies:

Coding: Getting programmers for games is a complicated process, and developers love gaming as an additional burden. For this, external resources are sometimes used in game development and the purchase of pre-made libraries, and they are modified within the company. They work on Sprints between designers and programmers until the completion of the app game development (ID13).

Testing: In this phase, they do manual testing, on-devices, and virtual tools. The customer is always in the development and testing process (ID 13, ID14).

Launch and Deployment: After the application is ready, completed and approved by the customer (ID13), ID13 and ID14 ask the customer to create a Google Play and Apple Store account to deploy it. Marketing is essential in App game development.

Marketing: This phase is essential in App game development. The success of a game is often based on the fact that players have been engagement from the visualization of the promotional material (videos, demo, etc.), even before the development of the game has been completed.

4.1.4 Data analysis

After obtaining all the information gathered in the fourteenth interview, an analysis was carried out to fully understand the development process followed by experts and the tools they were using during the development process. In Table 4.2, five companies' mobile apps development processes are shown as a sample (ID1, ID2, ID3, ID8, and ID10).

A. Process Comparison

The following is a comparison of the tasks performed by the different companies, based on a set of phases that were identified, under a name or another, in almost all the companies.

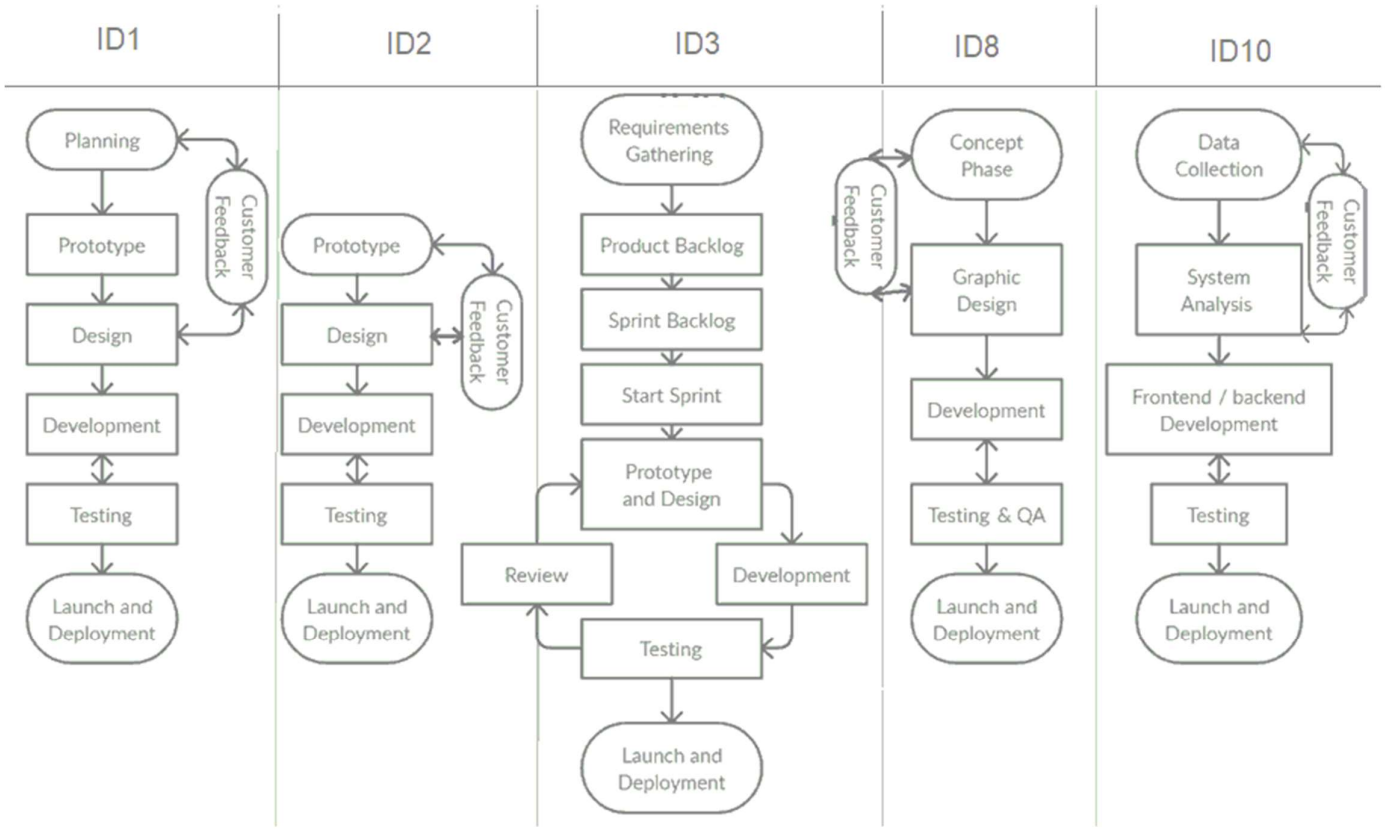
Requirements & Prototype: We found the most difference in developing mobile applications adopted by the interviewed companies was usually at the beginning of the process, before the Design phase. These phases were between Planning, Gathering Requirements, Developing the Idea, Prototype, Concept phase, and Data collection and System Analyst although with different names. The goal of these phases was to understand and collect the requirements and develop the idea of the mobile application. In this way, we note that all companies have relied on planning tasks to conceive the application's requirements and their understanding with the client to get to the desired application. In this line, only two companies have followed a different beginning using Scrum terms as it divided the requirements to the Product Backlog and developing ERP systems. On the other hand, all interviewed companies performed prototypes to facilitate communication with the customer.

Design: There is a design phase in all the companies. However, what we found different, is that some companies have integrated the User Experience (UX) with the requirements gathering phase to enhance the construction of the application idea (mainly by using wireframing). Many of these companies relied on interviews with the customer to determine the requirements. Some of these interviews are specific to questions in advance (filling a template), and others are open question interviews with the customer. The design phase is usually divided into two main sup-phases: User Experience and User Interface. Most companies start the design by designing a page by page with the customer’s review, and upon completion of the design of all the main app pages, a final test is done.

Development: It should be noted that all companies consider the development phase to be the main goal and necessary phase of mobile application development. Most companies wait until the customer approves the most important application page designs to start the development phase. Some companies call this phase the coding phase or the back-end development. Most companies produce an initial working version of the application, and the customer's opinion is taken on it. Many of these companies work in a Sprint in mobile application development, usually between one to two weeks, and end by sending a working part of the application to the customer. We found two companies working in parallel between developing the back-end and UI.

Some companies were part of the priority determinants of building the application is the customer's financial possibility, so they cannot expand the application capabilities if the budget is small, and this affects the final product.

Table 4. 2: Development Processes for ID1, ID2, ID3, ID8, and ID10 Companies



Testing: All companies test their applications as an important part of the development process. Many companies use simulators provided by built-in development tools. And many companies use third-party testers. Many companies have a QA department, and their main work is testing and sending the notes to the developers. Some companies use the customer as part of the testing process. Customer feedback is taken and sent to the development department after being checked by the project manager. Functional testing, regression testing, usability test, unit test, on-device test, automated test, and user acceptance tests are part of the Testing process used in many companies. We found that the Testing phase usually cycles or Sprints between the QA department and the developers until they have the final product.

Launch and Deployment: Note that all companies launch and deploy at the end of the project with obvious similarities. Some companies were using tools to monitor the application after the app release.

B. Tools Comparison

All companies use different tools in the development process, and there is no focus on a particular type of these tools. Some interesting data regarding the tools used that we have been able to extract are:

- Many companies are using UPXPIN to help in the design of the application.
- Many companies using Adobe illustrator and Photoshop in the design process.
- Many companies use native tools in developing mobile applications, while just one uses cross-platform.
- Companies use different methods and tools to connect the team among themselves, like Owncloud, Shared Excel cheat, or Google chat.
- Many companies used the same sources provided in the native tools for the testing phase.

C. Team Roles Comparison

After this research and the interview with the experts, we were able to identify the main roles in their companies. Most software teams use various Agile project management methods to manage and handle their work optimizing workflow and ensuring rapid app development without decreasing quality. Agile methods such as Scrum, Kanban, and DevOps focus on the vision and the app's idea, limiting the drifting away from its original aim during the development process. Agile and Scrum highlight customer collaboration and continuous feedback, small release, team collaboration, short Sprint iteration, high control, incremental, and iterative development processes.

We found essential team roles and how they work within the development environment through our research across companies. Of course, we find that some team members do more than one position. Each role identified is described below.

Project Manager: A project manager is the development team lead. Every development team needs a project manager. Project Manager is responsible for the success or failure of the project. This role works with the clients to define the objective, cost, scope, and schedule and fixes high-level troubleshoots issues.

Scrum Master: Scrum Master is a professional managerial position created by Scrum. Scrum Master is in charge of making sure that the project is completed following the guidelines, values, and principles of Scrum and according to the plan (Mathew, 2019). The Scrum Master does not lead the team. The Scrum Master communicates with the project team, clients, and management throughout the project. This role is in charge of the low-level

troubleshoots issues. The project manager takes a more general position; the same person can play two roles in some teams.

Product Owner: This role is unique to Scrum methodologies. The primary duty of this role is to be responsible for the Product Backlog list, control, maintain, manage, and make it visible to everyone in the project. The Product Owner manages the Product Backlog list with the highest priority first. They must also represent the client's interest through requirements and prioritisation.

Design Lead / Designers UX, UI: This role creates all look and feel and app's visual and audio assets .

Development Lead / Developers: Also called coders, this role is responsible for all coding, fixing bugs, back-end development, and all its needed for database, API, etc., developing the whole application architecture.

Analyst: Scrum Master or the Product Owner can fill this role. This role maintains requirements documentation and how it can fit the budget. The analyst is responsible as a communication facilitator between team members to ensure that the team builds the right product.

Quality Assurance Lead: Quality Assurance is not optional when creating a professional product. The main task of this role is to identify bugs and problems of the app, complete manual, and automated testing, create trackable reports on bugs, and create test cases.

One of the challenges of some companies regarding the development team was to maintain a stable development team in the company as better new jobs are always offered. One of the best solutions is that the team member feels that they are still learning and improving. The psychological dimension for the team is also important. Team members must feel comfortable and belong to their group.

D. Conclusions

One of the most important things to notice in any interview is that the first challenge for developers is to understand the client's requirements. We have noticed that they often use different methods to meet the needs of their customers. In this line, in most companies, they find that they tend to represent and show the requirement in a prototype model to the customer to meet or reach his/her requirements, taking into consideration that this prototype is not an interactive prototype, because they do not want the customer to imagine that the final product is about to end or that they may show to the customer features that in the future

they cannot comply. In addition, we can see that all companies are working hard to integrate customers into the development cycle and keep them updated to get together to the desired mobile application.

It should be noted that the customer can affect the quality of the mobile application development process. In many cases, it has been seen that the customer's cooperation with the development team facilitates the overall application development process, as the customer's cooperation in explaining what they want in the requirements gathering phase makes it easier for the team to understand the application. The customer familiarity with some technical and development skills also makes it easier for the team to understand the application requirements.

In some companies, they tried to fix the problem of the uncooperative customer, imposing penalties when it delayed responding to requests for communication with the company, such as a design opinion or some inquiries about requirements or application tests. Some companies have tried to limit the client's change of requirements, asking the client to sign the initial proposal and after the completion of the design of the application pages.

It was also noted that some companies are working to improve the application development process's quality if the customer has a higher financial ability. Some customers do not adhere to the application development team's expert advice, want to stick to their idea and opinion; this may affect the quality of the application, the UX, and the UI. It was also noted that the mobile application development process is affected by the company's size and experience. Some small companies try to reduce and facilitate the development process using free tools, and the development Sprints are short and fast. Furthermore, increasing the application development process's quality corresponds to an increase in the cost, and not every customer can pay this increase. Always asking the customer about the app's objectives facilitates understanding the app and the design process.

Regarding the cost of the application and its impact on the formation of the team members, we found that in some companies, the user experience designer is dispensed, gives his/her task to the graphical UI designer. Also, some companies were dispensing with the QA team as developers perform the testing phase with the customer to reduce the cost and match the customer's financial ability. These behaviours lead to a decrease in the quality of the mobile application. Indeed, some companies were reluctant to impoverish the way they develop applications. Indeed, some companies were reluctant to reduce the way they develop applications and reduce the quality to suit the customer's purchasing power, since this can damage the company's reputation.

On the other hand, most companies choose not to move on to the design stage until most of the client's requirements have been met. At the same time, all companies share information about the existence and sequence of prototyping, graphic design, development, and testing stages with customers who participate in most of these stages and provide feedback.

In addition, most companies, which are always trying to develop the most important part of a mobile application, this must be done initially for the application to run as quickly as possible and contain as many core requirements as possible. Another interesting finding is that all companies using mobile application development tools prefer to test on simulators provided with the same tools.

We observed that most companies have and follow a mobile application development method and repeat this method with each new project, incorporating some improvements such as using new tools or incorporating new members to the team. We also noticed that the technological level of the customer is becoming more understanding of the technology and applications, which makes it easier for the development team to gather requirements.

Most companies trust and work on an iterative design or method and incremental building model to develop mobile applications. We found iterations between the analyst and the customer, designer and developers, developers and QA, and designer with the customer. Companies break down projects and tasks into bite-sized tasks (page, model, or backlog) and then develop page by page or backlog by backlog. So, they start prioritizing it. Some companies only develop the most important requirements, and the less important requirements are developed in future updates. They will choose these features depending on the market trend and how users and customers behave. Future updates contain additional features or services in the application, such as adding a delivery service, communicating directly with the seller, determining the location, etc. Sometimes there are problems between designers and developers, these problems can be solved with cumulative experience.

On the other hand, the interviewed companies have adopted several testing methods, between manual and automated testing, with functional and usability testing. Any application must be manually tested before automated, manual scenario testing or case testing. Automated testing for mobile applications uses tools to find errors and defects. The QA team selects a set of tools, writes scripts, launches, and collects data. Functional testing is intended to ensure that the application is working as expected. In usability testing, QA focuses on ease of use and if it is well received by the user .

Finally, none of the interviewed companies have faithfully or integrally adopted existing development methods, much less reviewed the proposals that exist at the academic field to improve those processes. In the authors' opinion, it is necessary to make an effort to develop new methodologies that embrace the solid principles previously identified by the academic community as well as the latest advances and interesting proposals that the scientific environment has made and continues to do. At the same time, this new methodology should present the viability that the companies need and be enriched by the apps industry's experience. With this aim we have working in this research.

4.2 Study Through Online Surveys to Developers and Researchers in Mobile Applications

As mobile applications become more complex and important, companies and researchers must use the development process to adapt to more aspects of the process covered by modern agile methods. These methods require advanced documentation to fully understand the development process (Cunha et al., 2011). In response to this need, we are conducting this thesis to understand better the process of creating mobile applications for the industrial market, assess their similarities, and determine the similarities and differences between industrial processes and academic recommendations.

To achieve part of our goals, after initial interviews we will survey a broader pool of experts in the field of mobile application development in both academic and industry communities. This survey will allow us to identify milestones and trends in the application development process and determine the similarities and differences between industrial processes and academic recommendations. After all, our goal is to provide a unique methodology for mobile apps development based on and covering the academic and industrial communities.

This section analyses the questionnaire administered to software development specialists in mobile app development and to researchers or academics in the same field. The questionnaire focuses on the development process participants follow, tools used in the development process, main issues and challenges, and survey respondents' opinions, advice, and recommendations.

4.2.1 Study Population

Based on our research methodology, in the third phase, we did a questionnaire where the main contributors are mobile application companies, mobile experts, and academic researchers, as seen in Figure 4.1. Using this tool, we will describe the basic development process used by some companies and researchers in mobile applications to determine the similarities and opinions of researchers and scientists.

We surveyed many mobile software experts, developers and scientists. Specialists, scientists and researchers in the field of mobile application development also participated in the survey. We carefully select potencial participants, some of whom are finally survey respondents. More than 110 invitations to surveys were sent, and around 40 responses were

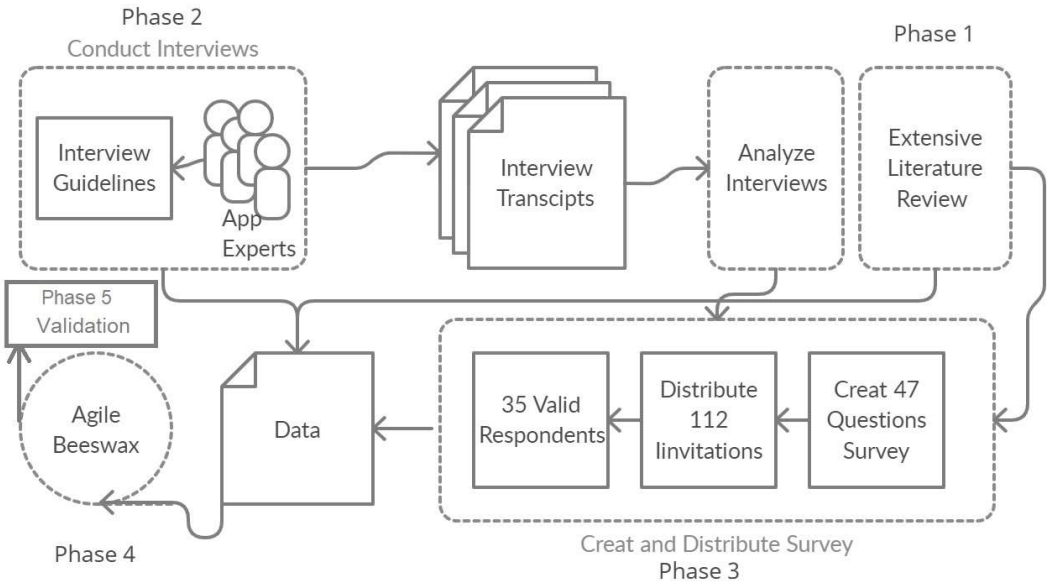


Figure 4. 1:The Applied Research Methodology

received, 5 of which were excluded. We went to the participants's place of work, contacted them by phone, e-mail or visiting him and encouraged him to answer the questionnaire. These people were invited to participate in the study as volunteers and all the surveys were anonymous. Finally, these surveys were a combination of closed and partially open-ended questions and we obtained 35 valid surveys.

4.2.2 Research Methodology

Surveys are widely used in general research activities (Julie Ponto, 2015) and software development research (Al-Zewairi et al., 2017, Punter et al., 2003). Our survey was validated before using it (Taylor et al., 2001) with the help of some experts and academics in

the field (Beecham et al. 2005, Mengual-Andrés et al., 2016) and in the software development process (Looks et al., 2021).

Based on everything learned (reading articles and interviews), the questionnaire was designed as part of our current research to identify the best agile methods followed by researchers and app developers and determine how to merge these methods to construct our new agile methodology.

The first part of the survey contains data and information about the participants; the second, organisational information from participants; the third, development process-related issues: idea and strategy, user experience design, user interface design, design technical decisions, development, testing, deployment, and monitoring. With more detail, the questionnaire consists of 46 questions divided into three main parts as follows:

The first part requires information about the participant, such as speciality, origin, and age. The second section involved the participant's organisation, including information such as whether the organisation developed apps, size of the company or organisation, number of apps developed in the organisation, whether the organisation used agile methods in app development, and whether they believe that agile systems are ideal for app development. The third part of the survey is also the main part, asking about the application development process. It consists of seven parts, each with 5 to 6 questions. These parts are:

1) Ideas and strategies: In this section, we will focus on the origins of application ideas, the tools used to develop ideas and management teams, whether the organisation conducted marketing research to evaluate and analyse competitors and users rating and review, and whether it is developing a roadmap and target group for the app.

2) User experience development: This section asks if an organisation is implementing a clickable User Experience (UX) prototype, workflows, or wireframe, and should iterate to improve UX. This section also asks about the tools used in this phase and user feedback.

3) User interface design: We asked how the developers move from the wireframe to the UI mock-up design, what tools they used, whether the prototype was clickable, whether they tested the workflow model, whether they performed iterations during the design of the UI, and whether they usually solicited user feedback during UI design. We also asked if they had full approval from customers before moving to the development stage.

4) Design technical decisions: This section asks about the choice of host environment, how the environment affects performance and scalability, what tools to use to build web APIs, whether to use SQL, what tools were used to develop a hybrid web-based platform, and which approaches were used to build the app—native or cross-platform native.

5) Development: We inquired about code reuse in the management and team-building process, user integration, and whether developers utilize specialised tools to assist with user integration. We concentrate on agility concepts, like how engineers use Sprints in agile methodologies. We also inquired about the Sprint plan as well as the tasks that must be completed during and after each Sprint. Did developers request feedback from the project manager or quality assurance for review?

6) Testing: This section asked whether developers used tools for testing (including specific automated tools) and whether they had a test plan and a checklist for verification and tests, testing app features, user-friendly testing, performance testing, user acceptance testing, and regression testing. We also asked if the designers reviewed the testing process to ensure their vision was implemented.

7) Implementation and monitoring: In this section, we ask developers if they performed monitoring for their apps, the tools they use, whether they track ratings and reviews in the app store, and how those ratings affect updates.

For more explanation, the whole questionnaire can be found in Appendix A. Next, we began analysing first the academics' and company experts' responses separately and then all responses together. Results are showed in the next section

4.2.3 Main Results

Every day, a lot of new applications for mobile devices are published. Professional developers need to follow some procedure and process while developing the app. The build of each mobile application is different, but we can always identify a standard process. We are here to focus on this standard process and this survey provides interesting results in this regard.

A. General information about the participants

Figures 4.2–4.8 and Table 4.3 below provide an overview of the demographics and roles of participants and organisations (Alrabaiah & Medina, 2021).

Most of the participants were male with 77% and 23% were female. Moreover, the age was less than 44 years, about 83% of the participants. Most of the participants were software mobile app developers in the top level of software management. They developed apps in native platforms (60%), some in hybrid cross-platforms (36%), and some in web platforms (33%). 90% of experts worked in companies that developed apps (see Figure 4.9); 40% of participants had experience, having developed more than ten apps; 17% had developed more than 20 apps. Fewer than 51% of participants worked at a small organisation

and about 40% at a medium-sized one (Figure 4.7). Table 4.3 represents the participants' information, including demographic characteristics, roles, organisation size, methodology, and tools usage.

Table 4. 3: The Main Participants' Information

Participants	Description
Gender	Male (77%) Female (23%)
Academic/Worker	Companies' experts (80%) Academic (20%)
Age	Under 44 years old (83%)
Country	Majority from Jordan and Spain
Experience	Develop apps (90%)
Company size	Less or equal five employees (51%) five to twenty employees (40%)
App Platform	Native platforms (60%) Hybrid cross-platforms (36%) Web platforms (33%)
Team management tools	JIRA (43%) Team management (40%) Teamwork project (33%)
Methodologies been Adopted	Agile (84%) Scrum (56%) XP (13%) Kanban (13%)

Detailed results can be seen in figures 4.2 to 4.11 emphasizing the following information (Alrabaiah & Medina, 2021):

- Most of the participants came from Jordan and Spain (see Figure 4.2).
- Most of the participants were male (77%); 23% were female (see Figure 4.3).
- The majority of participants (83%) were under 44 years of age (see Figure 4.4).

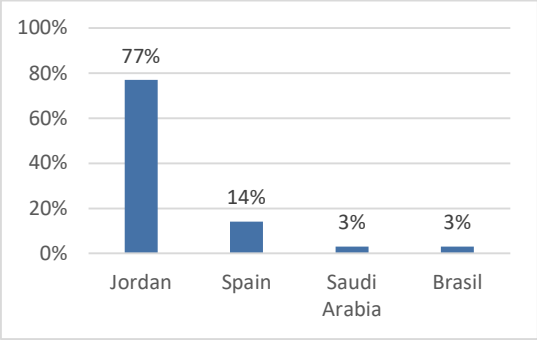


Figure 4. 2: Participants Country

- Most participants work in companies (80%); 20% are academics (see Figure 4.5)

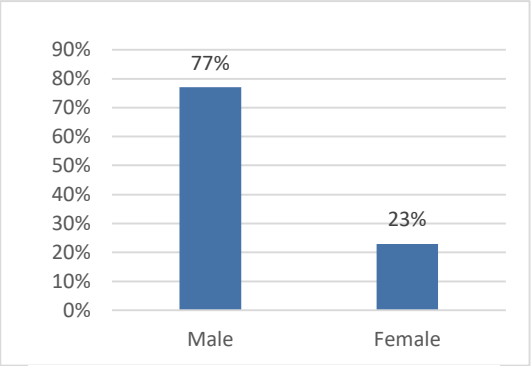


Figure 4. 3: Participants Gender

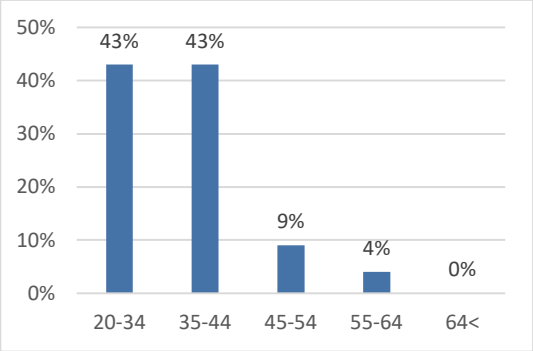


Figure 4.4: Participants Age

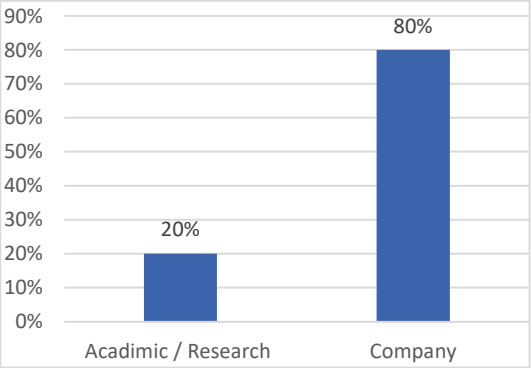


Figure 4.5: Participants Work

- About 90% of experts were working in companies that develop apps; 40% of participants had experience, having developed more than ten apps; 17% had developed more than 20 apps (see Figure 4.6 and 4.9).
- Most of the participants are mobile application developers at the highest level of software management (see Figure 4.7).

- Less than 51% of respondents work in small organisations (less than 5 employees), and about 40% work in medium-sized organisations (up to 50) (Figure 4.8).

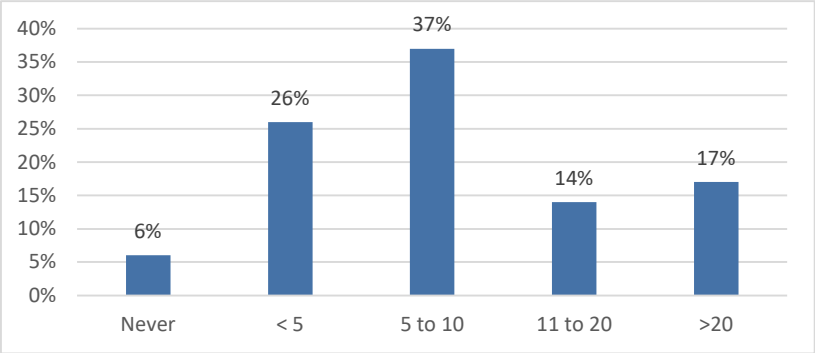


Figure 4.6: Participants Experience (Number of apps)

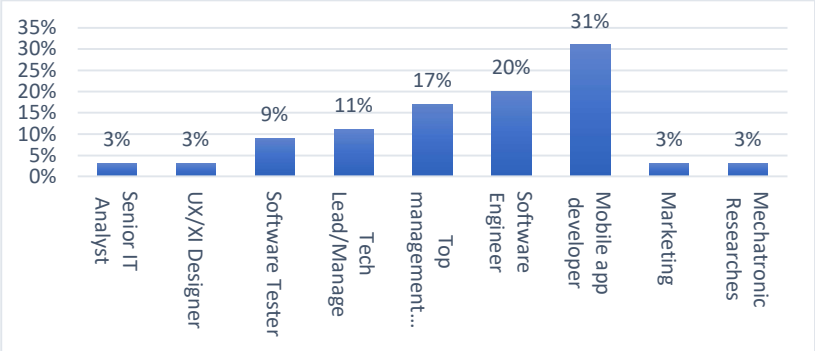


Figure 4. 7: Participants Role

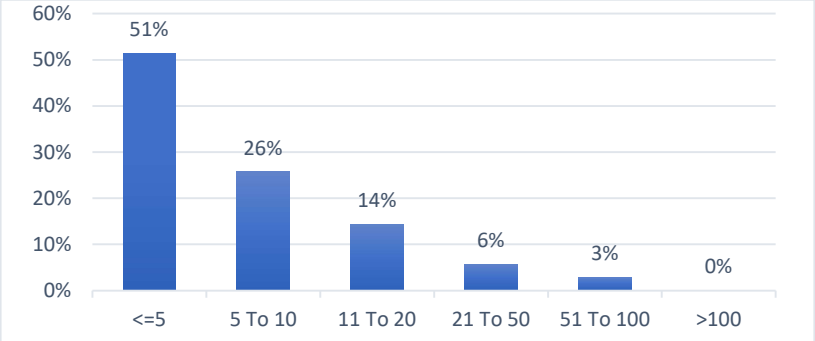


Figure 4. 8: Organization Size (Number of Employees)

- Most participants developed apps in native platforms (60%), some in native cross-platforms (36%), and some in hybrid-web platforms (33%).
- About 43% of respondents use JIRA as a team management tool, 40% use Trello, and 33% use the Teamwork Project.

- The majority of respondents (83%) use agile methods to create mobile applications (Figure 4.10); 56% use Scrum during development, about 13% use XP, and about 13% use Kanban (Figure 4.11).
- The majority of respondents (61%) believe that Agile is the best method for developing applications; approximately 36% of respondents said it happens sometimes.

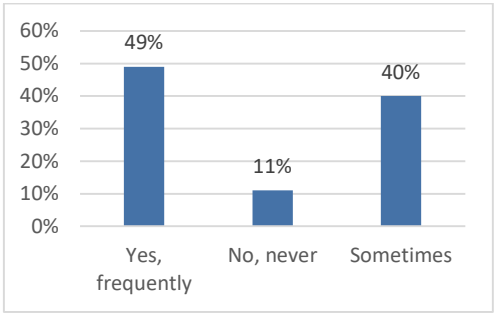


Figure 4. 9: Organisation Developing Mobile App

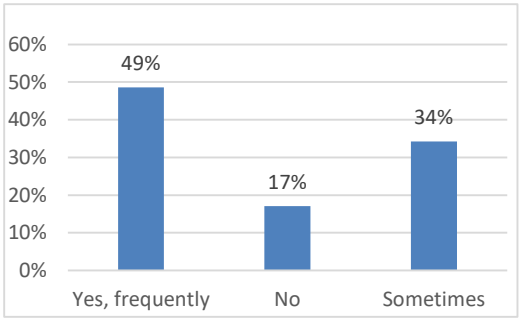


Figure 4. 10: Using Agile in the Development Process

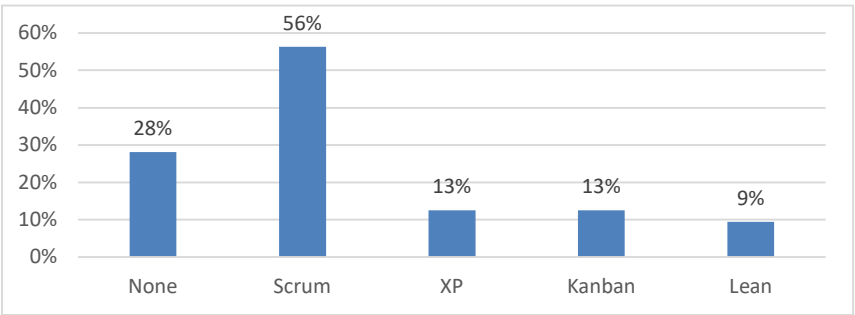


Figure 4. 11: Agile Approaches in the Organization

B. Development process

This study identified six key milestones in the participants' mobile app development cycle. These are the following steps: Ideas and Strategies, User Experience (UX) design, User Interface (UI) design, Design to Development handoff—issues and decisions, Development,

and Deployment and Monitoring. This section describes all the results obtained in these six main phases and their sub-phases during the survey, as described below (Alrabaiah & Medina, 2021). Important considerations about the ideal methodology for mobile development are derived and discussed from the survey results.

1) The idea and strategy

Based on our findings, this phase can be structured in three main sub-phases: Idea, Strategy, and Marketing campaign. Specific questions have been created for all sub-sections. This sub-phase is followed by the resulting phase output (Alrabaiah & Medina, 2021).

a) Idea: After the team comes up with an idea, they should start thinking about putting it into action. Studying the competition is an excellent place to start planning and presenting user feedback and ratings for similar or competing applications for the application to be developed. Almost all respondents indicated they started market research, 54% said they always did it; 43% said they did it sometimes (Figure 4.12). (This is seen in Appendix A, question 12: "*When you start thinking of developing an app, do you usually implement some market research -competitive audit-focusing on other apps that are carrying out the same idea and its user rating and reviews?").*

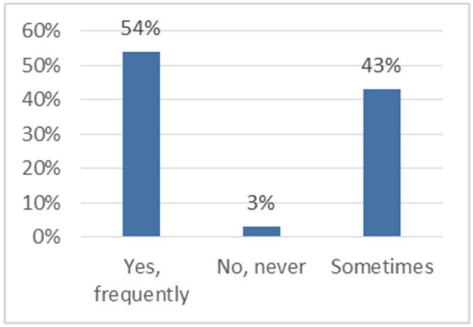


Figure 4. 12: Carry Out Market Research

All significant achievements begin with an idea. How can I adopt new ideas or improve current ones? For this, it is necessary to consider a problem that we might address. The team should talk to a professional about the issue to figure out why there is no solution or how to fix it. The team could ask about how the application fits into the overall solution. These questions can help to come up with new or better app ideas. Can the development team solve this problem with the new mobile application? Experiment with alternative approaches to meeting or adapting to customer needs. Start planning the application once we have come

up with an idea. The greatest place to begin preparing is conducting market research and identifying competitors. Survey responses to question 12 confirm that most experts perform this task in the early stages of the project.

b) Strategy: In our opinion, the team should find the problem while planning the application strategy. The scenario is critical. In line with this, about 59% of respondents always define application roadmaps, sometimes 26% (Figure 4.13). (This is seen in Appendix A, question 15: "*Do you usually define the roadmap for your app to be successful from day one?"*).

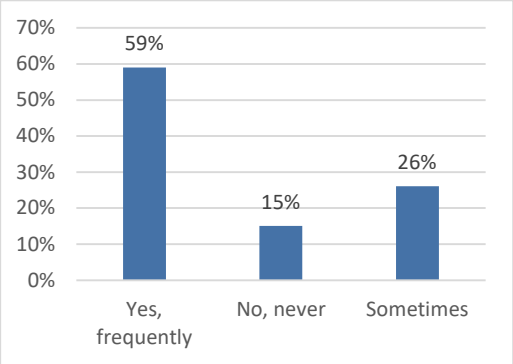


Figure 4.13: Define App Roadmap

It's critical to establish a setting for the app roadmap. What are the app specifications? What are the app plans? What do we hope the app will become in the future? Write what the app must do, consider its core functionality, and think about what we could add in the future.

C) Marketing campaign: This task is a part of the overall marketing plan and starts with marketing strategy. Participants agreed on the importance of marketing to the app's success. Approximately 53% of participants made a marketing plan; 29% did sometimes (Figure 4.14), unless the app was for internal or B2B use. (This is seen in Appendix A, question 14: "*Do you usually create a marketing campaign for your app?"*).

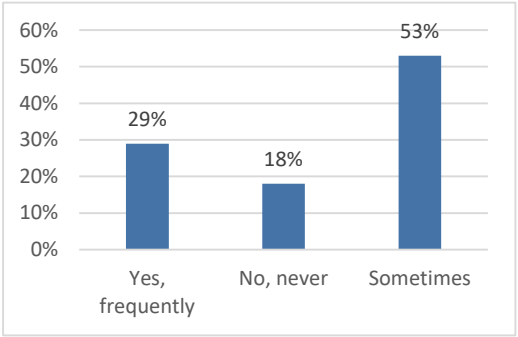


Figure 4.14: Applying Marketing Campaign

At the end of this phase, the team can write an internal report to describe the thoughts and strategies. About 44% of participants had done so, with 35% claiming to have completed a final report sometimes (Figure 4.15). (This is seen in Appendix A, question 13: "*Do you usually write a final report after finishing the planning-strategy stage?"*).

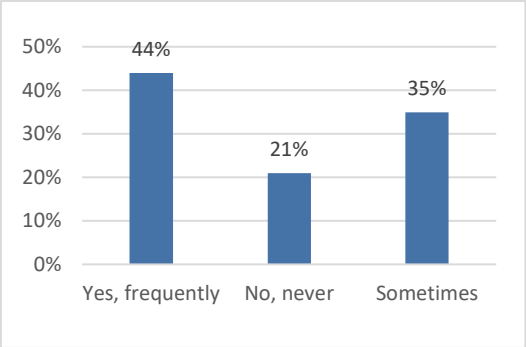


Figure 4.15: Writing a Final Report

Regarding to the tools, JIRA is used by 43% of respondents as a team management tool, while Trello is used by 40% and the Teamwork Projects is used by 33%. (Figure 4.16). After multiple iterations, the team will receive a final report at the end of this stage. (This is seen in Appendix A, question 16: "*Which tools are you using in a project and team management?"*).

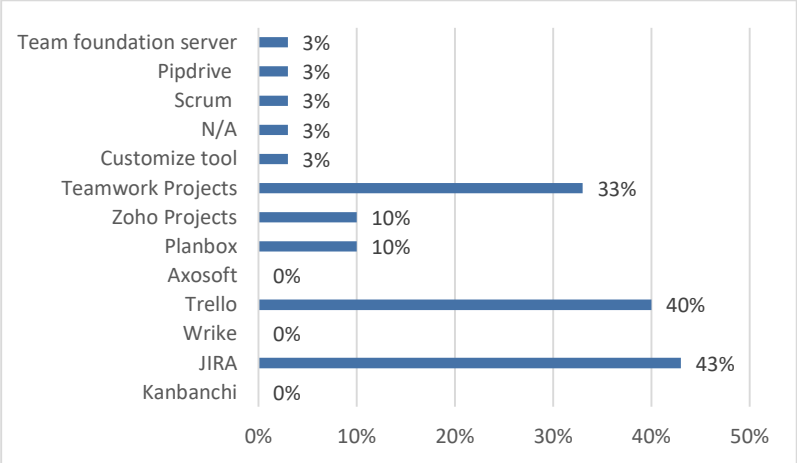


Figure 4.16: Tools Used in Project and Team Management

2) User experience design

After reviewing the goals and requirements with the user and the team, the team starts writing and creating blogs that contain the data and functions in the application and how to organize and present them. The development team can start with pencil and paper to sketch

the wireframe, representing what they need and visualising customer needs. As illustrated the survey to question 17 in Appendix A, over 94% of participants started making wireframes with a board or paper and pencil. Specifically, 64% answered yes frequently, and sometimes 30%, as seen in Figure 4.17. (This is seen in Appendix A, question 17: "*Do you usually start to collect your requirement and functionality using a Whiteboard and pencil and paper?"*).

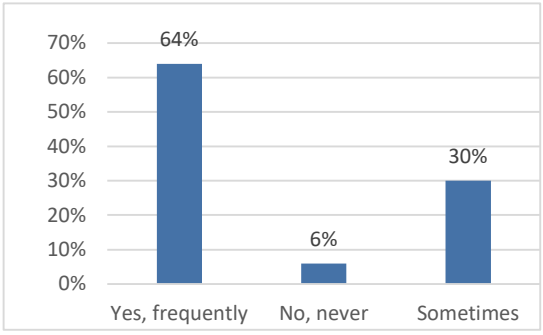


Figure 4.17: Using Tools (Whiteboard, Pencil, and aper) in Collecting

Based on our previous studies, wireframe, workflow, and testing can be identified as the primary sub-phases of the user experience phase (Alrabaiah & Medina, 2021). Specific questions have been created for all sub-sections.

a) Wireframe: A wireframe is a visual representation of projects and design content. Teams must first draw and create the screens and their functions and data in this step. As demonstrated in the survey, the majority of participants used whiteboards (43%); some used paper and pencil (47%); and a few others used other tools such as Sketch (23%) or InVision (23%) to construct wireframe (Figure 4.18). (This is seen in Appendix A, question 18: **"***If you usually draw a wireframe presenting the functionality and data, which tools are you using to draw your wireframe?"*).

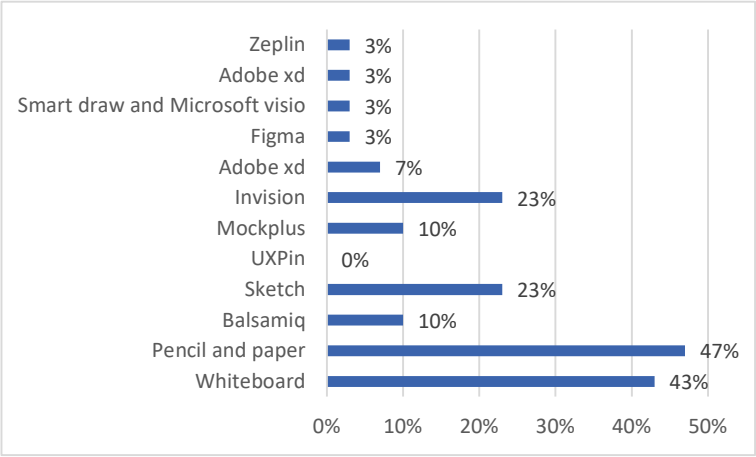


Figure 4.18: Tools Used in Wireframe Drawing

Therefore, we can conclude that it is critical to comprehend and consider the majority of the application's functionality at this point. The team needs to sketch or draw to alter things now, but they have to rewrite and redraw the code afterwards.

b) Workflow: The workflow specifies the path between the wireframe badges that users can navigate through the application. Most of the participants agreed to do the workflow with 91%. 53% said they do it frequently, while 38% said they sometimes complete the workflow (Figure 4.19), as indicated in the survey. (This is seen in Appendix A, question 19: "*Do you usually create workflows that present the pathways users can travel within the app?"*). In addition, we can conclude that it is necessary to be careful when designing the workflow.

Furthermore, based on our findings the app usability such as the number of clicks required to complete each task, shouldn't require many clicks. If one task requires many clicks, it is necessary to return to the wireframe and resolve the issue there. At each iteration, teams should double-check all functionality and usability.

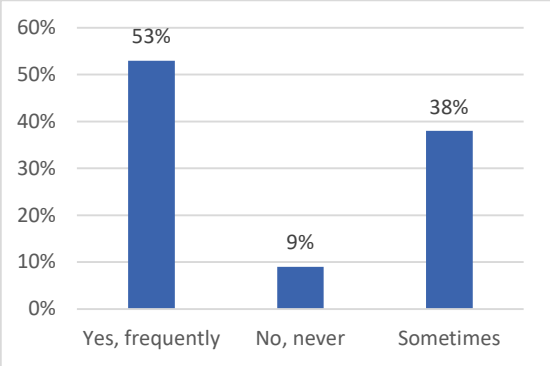


Figure 4.19: Create App Workflow

As demonstrated in Figure 4.20, the majority of participants perform iterations between wireframe and workflow (46% yes frequently, sometimes 36%). (This is seen in Appendix A, question 20: "*Do you usually do iterations between wireframe and workflows to improve the design?"*). For this task, the team can either draw the workflow on the whiteboard or paper with a pencil, or even InVision can help.

c) Testing workflow: In our opinion, creating a tappable click-through model can improve the design. However, participants were less interested in creating a tappable click-through in each development: 30% did not use it, 24% used it, and 46% only sometimes, as shown in Figure 4.21 below.

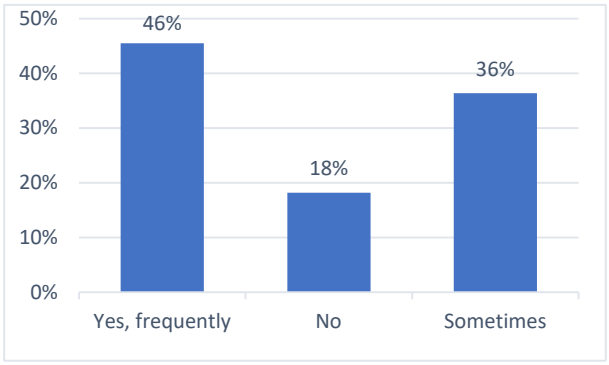


Figure 4.20: Iterations Between Wireframe and Workflow

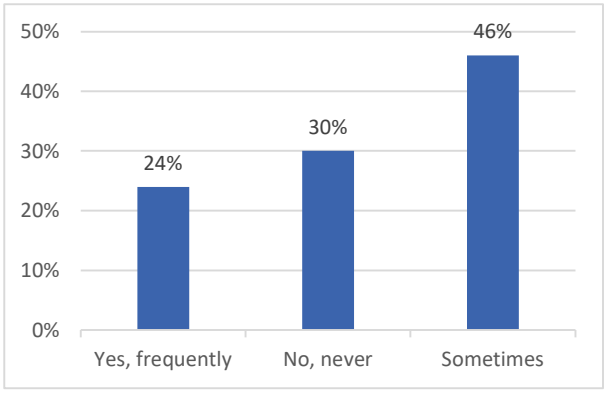


Figure 4.21: Building a Tappable Click-throw

As indicated in Figure 4.22, to build the tappable click-throw prototype mostly half of the participants (48%) in this model used Sketch, 22% used InVision, and 19% used Mockplus. (This is seen in Appendix A, question 21: **"***Do you usually test your wireframe and workflow on a tappable click-throw UX prototype? (We are not talking about UI Prototype- this will come later)?" and* question 22: **"***If yes; which tools do you use to make a tappable UX prototype? Whiteboard, Pencil and paper, Balsamiq, Sketch, UXPin, Mockplus, InVision, Other…"*). In our opinion, InVision can help to test the wireframe and workflow, after which the team can import the design. Even without a function, the model can be provided to the user for testing, implying no code in the phone. Teams should attempt to locate the issue, update it, and continue.

3) User interface design

The user interface (UI) phase comprises three key sub-phases: Style guideline, Replacement / Mock-up, and Testing (Alrabaiah & Medina, 2021). Specific questions have been created for all sub-sections.

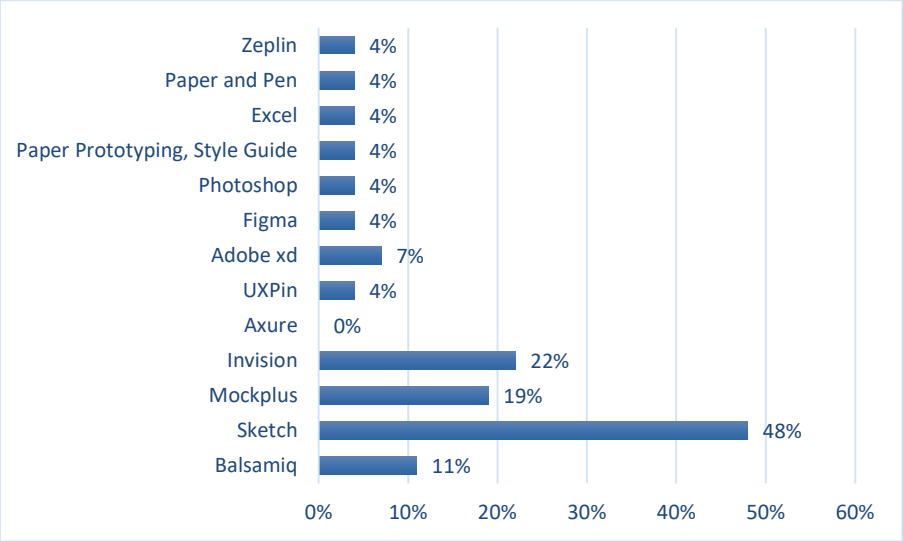


Figure 4. 22: Tools for Tappable Prototype

a) Style guideline: To increase the usability of the app and eliminate incompatible components and inconsistent templates, it is necessary to follow the suggestions in the style guidelines. For example, R. Baharuddin (2013) mentions that one of the most important aspects determining app success is usability. Usability and familiarity are very important property of good UI design. For its part, the ability to learn is a crucial aspect of excellent user interface design. If the appropriate colours and fonts are chosen, the application will be simple to use since it uses a consistent design language. Another consideration is to try to determine who the clients or the users are. Also, it is important to reduce clutter, and use the app day and night. A highly skilled team assists in developing apps that are acceptable for all users. It is critical to recognize that there is no one-size-fits-all solution. User Interface Style Guides are tailored to the application's exact requirements, and these design decisions will be implemented in the phases of the User Interface Design.

Based on our study, usually the user interface (UI) phase comprises two key sub-phases: Replacement / Mock-up, and Testing (Alrabaiah & Medina, 2021). Specific questions have been created for these sub-sections and respondents' answers are analyzed below.

b) Replacement / Mock-up: In this phase, teams start updating the wireframe of the app element's wireframe: colours, buttons, logos, photographs, and so on, to transition from wireframe to UI design. As seen in the graphic below, approximately 38% of participants used Sketch for this task (Figure 4.23). (This is seen in Appendix A, question 23:" *Which tools usually do you use to move from wireframe to UI Design elements? Whiteboard, Pencil and paper, Balsamiq, Sketch, None, Other…"*).

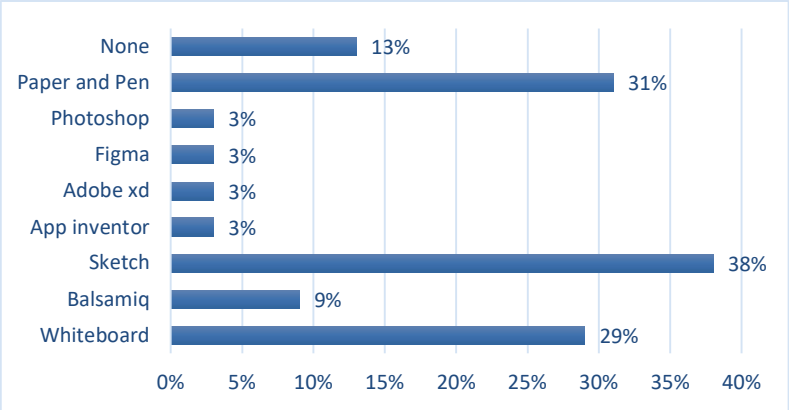


Figure 4. 23: Tools for Tappable Prototype

The tools for constructing an interactive UI prototype are shown in Figure 4.24. Teams can design the style in Balsamiq 30%, InVision 30%, then Mockplus 19% to construct a click-through model prototype. (This is seen in Appendix A, question 24: **"***Which tools usually do you use to move from wireframe to UI Design elements? Whiteboard, Pencil and paper, Balsamiq, Sketch, None, Other…"*).

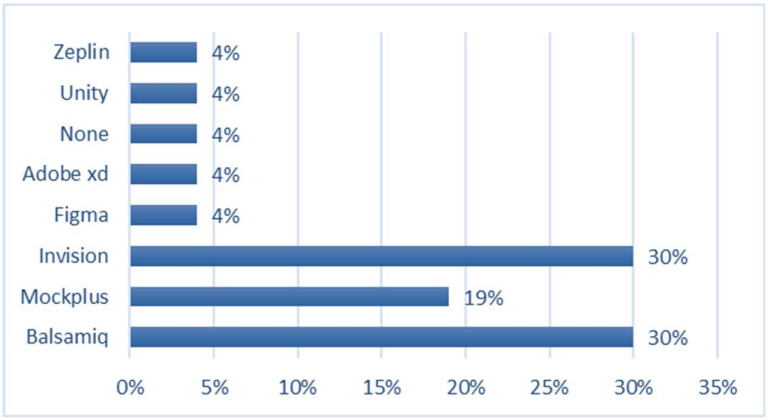


Figure 4. 24: Main Tools in Building a Tappable UI Prototype

c) Test again*:* As seen in Figure 4.25, the majority of participants re-tested their model, with over 58% saying yes frequently, 33% sometimes, and only 9% saying no. (This is seen in Appendix A, question 25: *"When you finish all UI screens, do you usually test again click-through -workflow- model to be sure that it still works and have correct data-flow?" and question 26: "When you finish UI design, do you usually ask for user feedback?"*). After building a user interface, 62% of respondents always, and sometimes 32% of respondents, ask for user feedback (see Figure 4.26). (This is seen in Appendix A, question 25: *"When you finish all UI screens, do you usually test again click-through "workflow-*

model to be sure that it still works and have correct data-flow?" and question 26: "When you finish UI design, do you usually ask for user feedback?").

Based on our findings, we believe it is critical to revisit the click-through model prototype, repeat the test, and devote the necessary time to providing design, usability, and user feedback. It is important to keep in mind that further adjustments can be pricey.

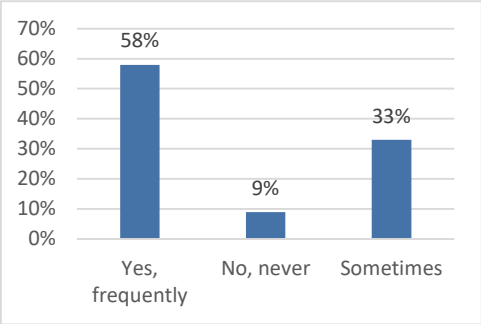


Figure 4. 25: Iteration Testing in UI Design

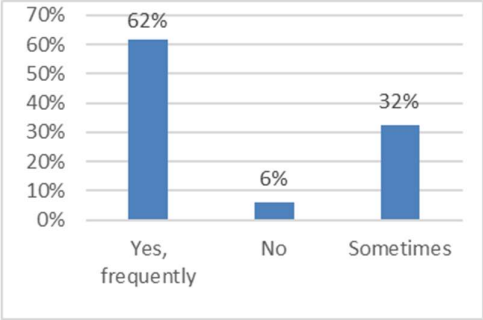


Figure 4. 26: User Feedback After UI Design

4) Design to development handoff- issues and decisions

The shift from design to development is one of the administrative problem issues in application development. Misunderstandings and poor communication between the design and development teams are the most common roadblocks to a successful transition. We suggest having the same team handle the design and the development phases. According to the surveyed experts, it's ideal to use tools like Sketch (27%), Photoshop (42%), or Zeplin (6%) to produce a smooth transition, as demonstrated in Figure 4.27 below. (This is seen in Appendix A, question 27: *"Do you usually use any tools to ensure a smooth transition from design to the development process? Which?").*

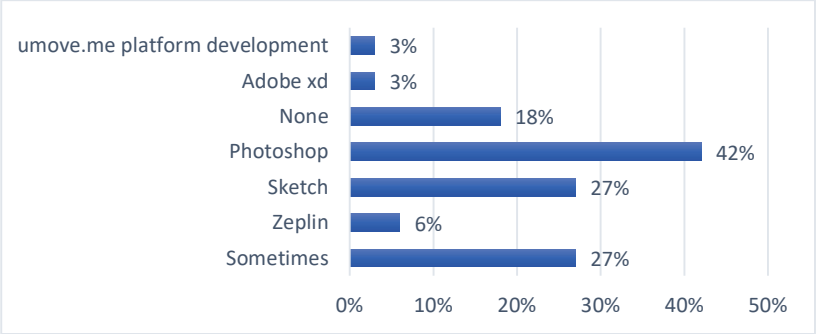


Figure 4. 27: Tools Used from Design to Development Transition

In our opinion, Customer approval, Front-end / Back-end, and Database are the three key sub-phases of the design to development handoff phase, which are based on the issues and technical decisions to be considered (Alrabaiah & Medina, 2021). Specific questions have been created for all sub-sections. We consequently organized the survey in these three phases.

a) Customer approval: Figure 4.28 shows that approximately 91% of respondents received final customer approval after completing the app design. (This is seen in Appendix A, question 28: *"Do you agree not to move to the development phase until you get mostly full approval from the customer that this is what he wants and needs?").* Therefore, when creating an application design, it is important to obtain the customer's approval so that the design meets their needs and wishes.

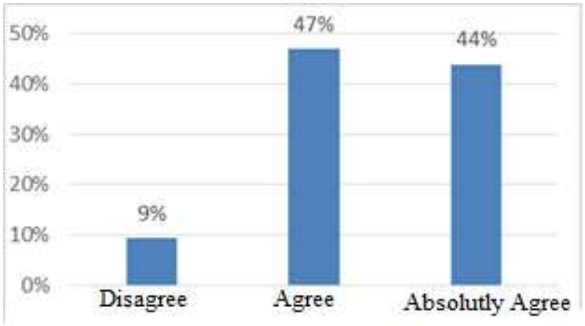


Figure 4. 28: Starts the Development Phase After the Customer Approval

b) Front-end: Teams can use various technologies and programming languages to design mobile applications. Every technology has its own set of benefits. Some are less expensive, while others are less effective. Teams must select a reliable technology.

Mobile applications can be developed in three ways: native platform, cross-platform native, and hybrid web technologies. It is nice to know that the majority of respondents build their apps on a native platform (61%), cross-platform native (36%), and hybrid (web technology) (33%), as shown in Figure 4.29 below. Participants used HTML (56%), JavaScript (47%), PHP (44%), Ioniq (30%), CSS (30%), and Jquery (24%) as seen in Figure 4.30. (This is seen in Appendix A, question 29: *"Which approaches do you usually use to build your app?",* and question 32: **"** *Which language or tool do you normally use when you are developing a web platform?").*

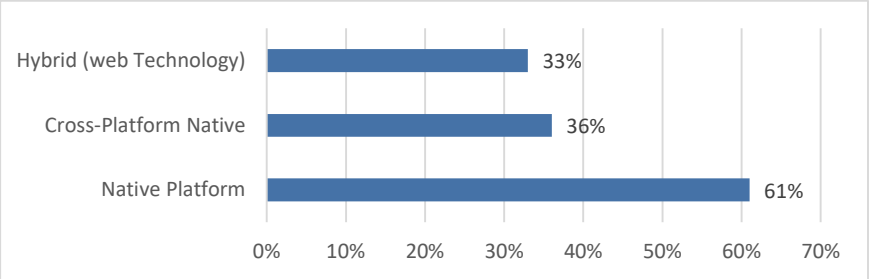


Figure 4. 29: App Platform Used

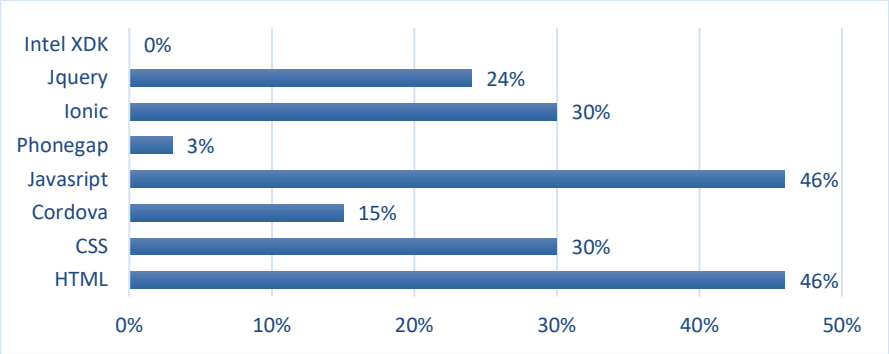


Figure 4. 30: App Web Technology Tools

*c) Back-end (*API / Web Server): App back-end is anything related to the app security, storage of the data, and business logic (Nandyal & Rafi, 2020). When creating a web application, teams must select the back-end development language. Participants used Java (56%), JavaScript (47%), PHP (44%), and C # (34%) as seen in Figure 4.31. (This is seen in Appendix A, question 30: *"Which languages do you usually use to build your Web API?").*

d) Database: Creating a reliable and structured database for the application is very important; most of the respondents use SQL (88%), and 12% do not (Figure 4.32). This was explained in answer to Questionnaire 31 in Appendix A. (This is seen in Appendix A, question 31:*"Do you usually use SQL for mobile apps databases?").*

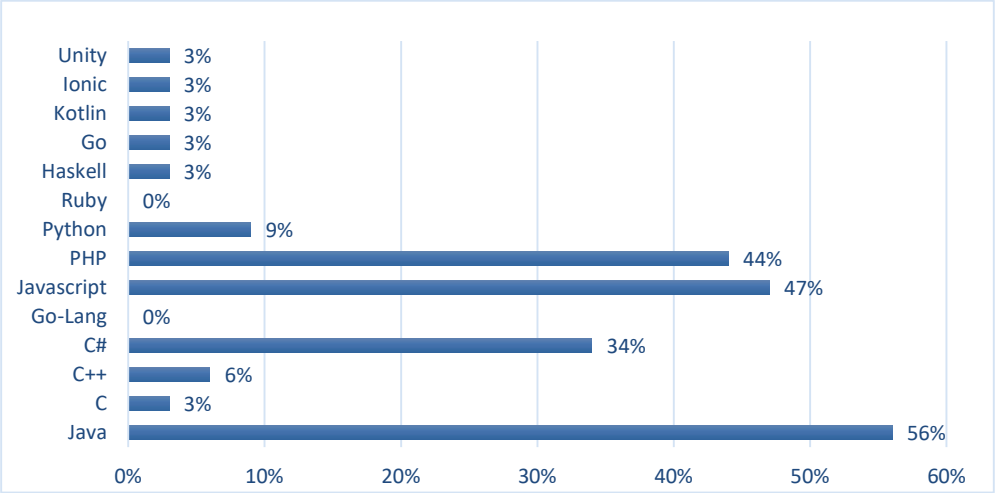


Figure 4. 31: Web API Language Used in App Development

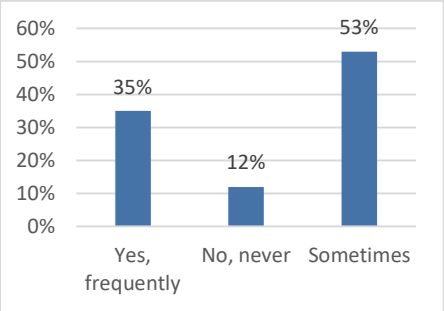


Figure 4. 32: Using SQL in App Development

In terms of the host infrastructure that hosts the database and API, 45% of respondents use Google Cloud, and 31% of respondents use Amazon AWS (Figure 4.33). (This is seen in Appendix A, question 34: *" Which hosting providers do you usually use for your APIs and Databases?")*.

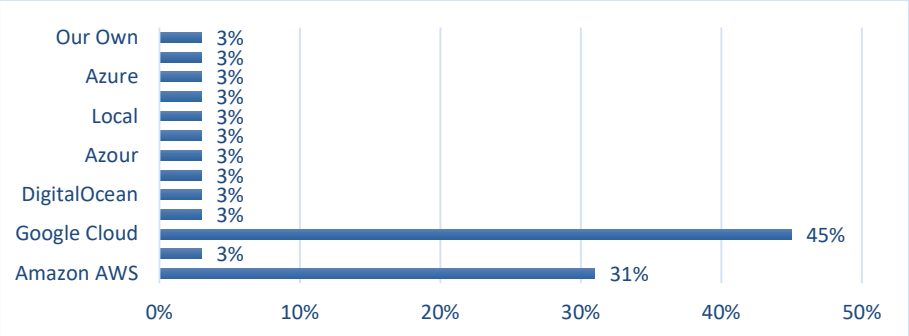


Figure 4. 33: App Hosting Provider

When choosing a hosting infrastructure, teams need to consider its impact on price, performance, scalability, and reliability. The hosting infrastructure should be shared with the customer. Most of the respondents shared this with their clients. About 53% answered in the yes frequently, and 31% said they sometimes speak (Figure 4.34). (See Response to Questionnaire 33, Appendix A). (This is seen in Appendix A, question 33: *" Do you share the importance of choosing the hosting environment with the customer, and how is it essential in performance and scalability?").*

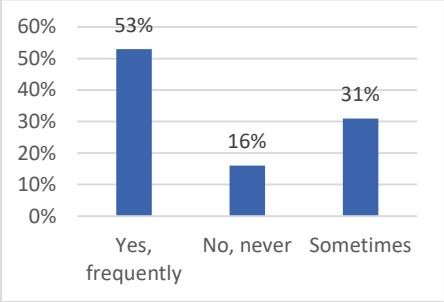


Figure 4. 34: Customer Participation in Choosing the App Hosting Environment

5) Development

Building mobile apps is an iterative process. Most participants use iteration or Sprints during the development process (approximately 50% answered yes, sometimes 35%, as shown in Figure 4.35). (This is seen in Appendix A, question 35: *"When you start the development process, do you use an iteration model "Sprints- in agile methodology?").*

In our opinion, the development phase contains three main sub-phases: Sprint planning, Development, and Testing (Alrabaiah & Medina, 2021). Specific questions have been created for all sub-sections. We consequently organized the survey in these three phases.

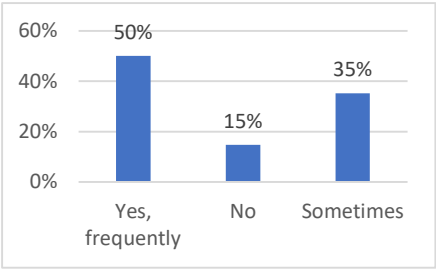


Figure 4. 35: Working in Iterations in the Development Phase

a) Sprint planning: 60% of participants start with a Sprint planning, 27% say sometimes they did (Figure 4.36). (This is seen in Appendix A, question 36:" *When you start planning for the Sprint, do you focus on the tasks to be implemented during this iteration and estimate the time needed to finish this task?").*

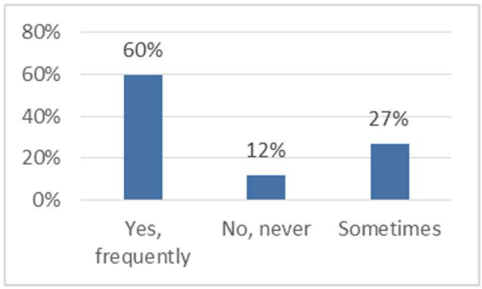


Figure 4. 36: Starting Development with Sprint Planning

Therefore, the Sprint starts with a Sprint planning session, consisting of a list of activities that must be completed in each iteration. Each task should explicitly explain the amount of time it will take to finish it as well as its requirements. To guarantee that developers completely comprehend each assignment, developers must be included in the Sprint planning. When developers begin planning a Sprint, they should think about how to reuse code. Only 9% of respondents did not reuse code throughout the development process, while 65% did always (Figure 4.37). (This is seen in Appendix A, question 37: *"Do you try to reuse code throw the development process?").*

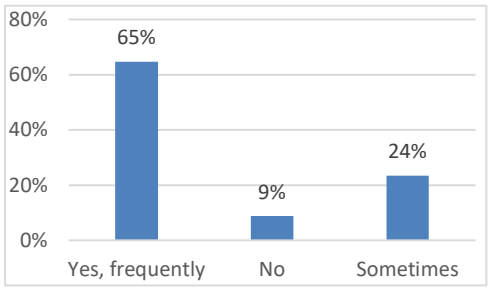


Figure 4. 37: Reuse Code

b) Development: After the Sprint, the team will start developing the application's functionality. It's critical to provide the results to the project manager or quality assurance for assessment. According to Figure 4.38, over 70% of respondents do this, with 15% saying they do it sometimes. (This is noted in Appendix A, question 38: "*When you complete a Sprint, do you send back the results to your project manager or quality assurance for review*?").

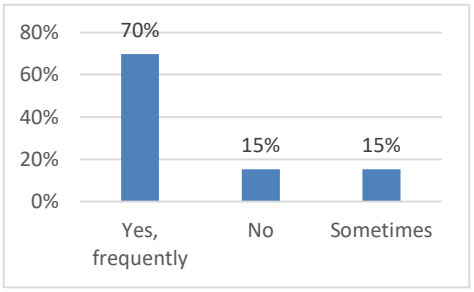


Figure 4. 38: PM and QA Review After Each Sprint

In our opinion, a detailed understanding of the requirements of the individual pending file is essential to the success of this phase. In the event of failure, the team should notify the project manager of the deficiencies. Agile principles in developing apps are a means to break down all requirements to Product Backlog milestones and start development by building the app Backlog by Backlog in iterations and cycles - Sprint backlog, using incremental development cycles. All these considerations derived from the study in this section and previous will be taken into account when defining our methodology to develop mobile applications.

c) Testing: The same app developers should not test the application to be more realistic. If the QA team is doing this work, they should start with functional tests, including a test plan and a checklist of activities to be verified. About 67% of the participants performed a functional test and made a list of operations to be checked, 27% said sometimes said (Figure 39). (This is seen in Appendix A, question 40: *"In functional testing, does the testing team have a test plan and list of actions to check?").* Functional testing mainly focuses on usability, performance testing, and responsiveness. In later Sprints, performance testing and app responsiveness are very important. 77% of respondents use and pay attention to user-friendly usability testing, 68% of respondents pay attention to user acceptance tests, and 55% of respondents pay attention to performance testing and Regression tests 39% (Figure 40).

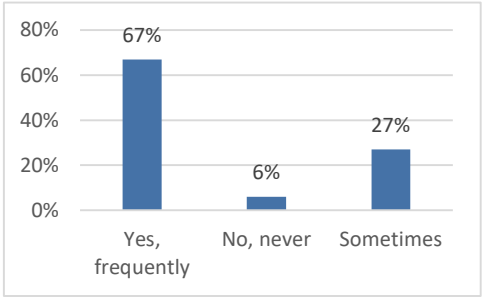


Figure 4. 39: Participants Performed Functional Testing with a List of Actions to Check

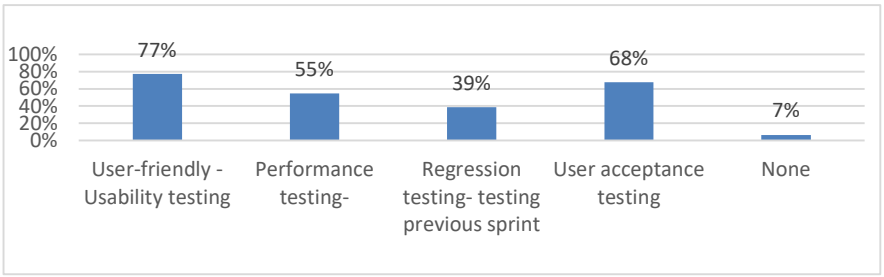


Figure 4. 40: App Testing

Based on the acquired knowledge, we recommend including designers in the testing process; ask them to verify if their described vision has been developed. Most of the respondents agree (70% yes frequently, 15% sometimes), as shown in Figure 4.41. (This is seen in Appendix A, question 41: *"Which one of these testing do you use to test app features?").* The team must complete the regression test. After the Sprint test is finished, the team must review the previous Sprint.

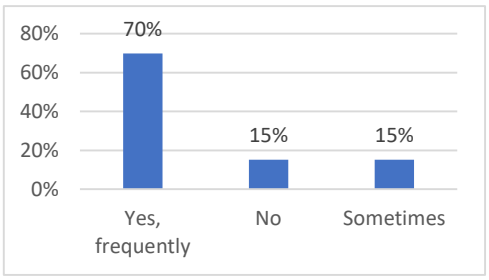


Figure 4. 41: Reuse Code

According to On-Device testing, we suggest testing the app on various screen sizes and versions. Google Firebase (28% of respondents) and native tools (28%) are examples of native or automated testing technologies used by the participants (Figure 4.42). (This is seen in Appendix A, question 43: *"If you are using automated specific device testing, which tools you are using?").* We believe it is critical to test the app on various physical devices.

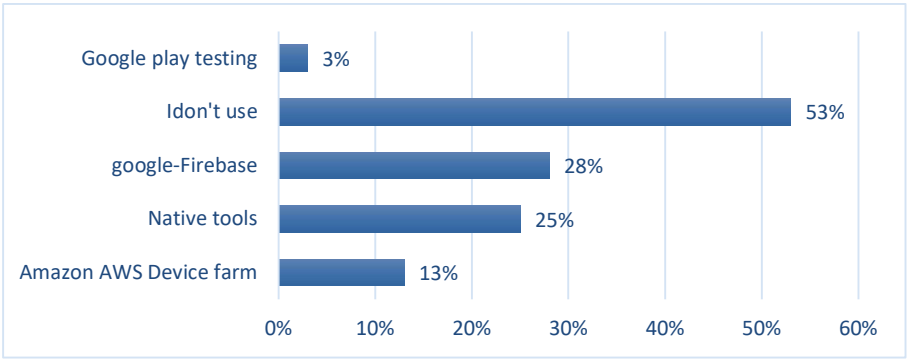


Figure 4. 42: Automated Specific Device Testing

In our opinion, the Security Tests and User Acceptance tests (Figure 4.40) are the most critical. The owner or user candidate will judge the application and decide if the app is accepted or not.

At the end of each Sprint, the development team should talk to stakeholders and learn from everything done in the Sprint. They should learn how to ensure continuous improvement in the next Sprint. The goal is to optimize the process at each iteration. Try to find potential users and other testers, listen to their opinions, and then decide. The team can run a beta test or cause the application available for testing by real users. After the review is complete, it is best to run a final development Sprint, which will fix any remaining issues. All these considerations will be studied in greater depth to determine their inclusion in the mobile development methodology to be defined.

6) Deployment and monitoring

In our opinion, the Deployment and Monitoring phase presents the las phases in the development process for the mobile app (Alrabaiah & Medina, 2021). In the deployment, if the app is a hybrid web application, it requires a server back-end to host the data. The team is necessary to select the server they want to use. Based on the obtained knowledge during our research, we recommend a server with a scalable environment such as Amazon Web Services. The app will not fail with such a server if it becomes popular. For the app to be deployed to the Apple Store or Google Play Store, it must meet the requirements of those companies.

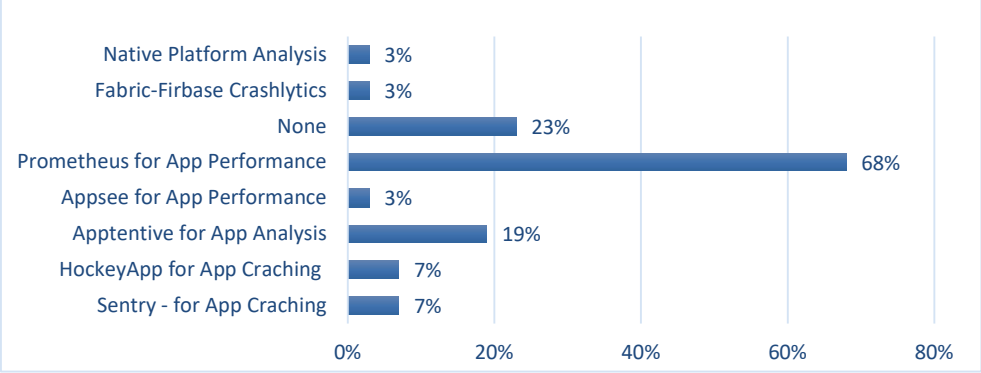


Figure 4. 43: App Monitoring Tools

In the other hand, in monitoring, the app's review always shows a long list with the history of app updates, including bug fixes, changes, new features, and performance updates. Some tools can help to monitor the apps, such as Prometheus for App Performance (68% of

participants use it) (Figure 4.43). (This is seen in Appendix A, question 45: " Which one of these tools do you use to help you in monitoring your app?").

C. Development Tools Distribution

According to the survey, we found that all participants use different tools in the development process. At the same time, they do not focus on a particular type of tool (Alrabaiah & Medina, 2019), but they focus more on some tools than others. Figure 4.44 summarizes the tools used by participants in each phase and sub-phase of the development process (Alrabaiah & Medina, 2021). The majority of respondents use native app development platforms (61%), cross-platform native (36%), and hybrid web technology apps (33%). In more detail, some interesting considerations about the tools used are:

- In Project Management, where 43% of participants use JIRA as a project management tool, 40% use Trello, and 33% use Project Teamwork.
- In the UX phase, 43% of participants use the whiteboard to draw the app wireframe, and 47% use paper and pencil. Furthermore, 22% of participants use tools like Sketch and Invision, while 1o% are using Mockplus.
- To move from UX design elements to UI design elements, 23% of participants use Skitch, 31% use paper and pencil, and 28% use whiteboard. Furthermore, 9% of participants use Balsamiq tools to design a tappable UX prototype.
- In the UI phase, 30% of participants use InVision and Balsamiq for each, to design a tappable UI prototype, and 19% use Mockplus
- In the design to development hands-off phase, 42% of participants use Photoshop to control design to development hands-of, 27% use Skitch, and 6% use Zeplin. While 18% of participants do not use any tools (None).
- In the design to development hands-off phase, 56% of participants are using Java to develop WEB API, 47% are using JavaScript, 44% are using PHP, 34% are using C#, Python 9%, While 88% of participants are using SQL to develop web API.
- In the Development phase (hybrid, web, and native app), 46% of participants use HTML to develop hybrid web app technology, 46% JavaScript, 30% CSS, and 30% use Ionic.
- In the UI phase, 30% of participants use InVision and Balsamiq for each, to design a tappable UI prototype, and 19% use Mockplus

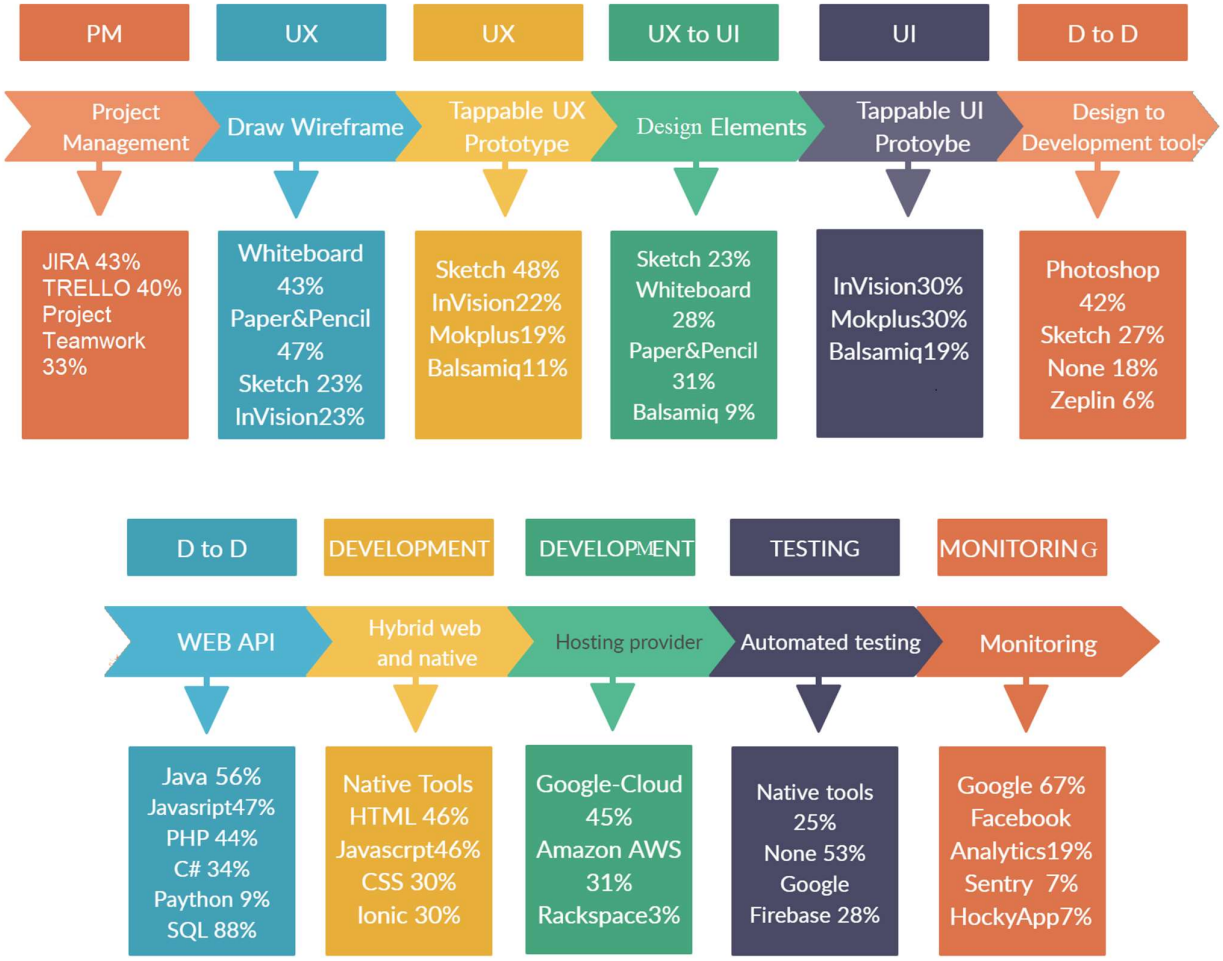


Figure 4. 44: Tools Used During the Application

- In The UI phase, 30% of participants use InVision and Balsamiq for each, to design a tappable UI prototype, and 19% use Mockplus
- In the design to development hands-off phase, 42% of participants use Photoshop to control design to development hands-of, 27% use Skitch, and 6% use Zeplin. While 18% of participants do not use any tools (None).
- In the design to development hands-off phase, 56% of participants are using Java to develop WEB API, 47% are using JavaScript, 44% are using PHP, 34% are using C#, Python 9%, While 88% of participants are using SQL to develop web API.
- In the Development phase (hybrid, web, and native app), 46% of participants use HTML to develop hybrid web app technology, 46% JavaScript, 30% CSS, and 30% use Ionic.

- In the Development phase (hosting provider), 45% of participants use Google Cloud as a hosting provider, 31% Amazon AWS, and 3% use Rackspace.
- In the Testing phase, 25% of participants test their app using native tools, 28% Google Firebase. In comparison, 53% of participants do not use any automated testing (None).
- In the Monitoring phase, 67% of participants monitor mobile apps using Google, 19% Facebook analytics. While 7% of participants are using Sentry, and 7% HockeyApp to monitor mobile apps.

4.2.4 Conclusions

In this chapter we have discussed the main phases and procedures adopted by mobile app developers in a real environment and by researchers in academia, as well as the tools used in the development process.

The literature review and related work section in chapters 2 and 3 identified agile methods for mobile app development. We also explained how agile methods are naturally suited to app development and how apps mapped onto agile themes. Accordingly, we conducted fourteen semi-structured interviews with experts and sent a structured survey to a total of thirty-five experts. This study sought to understand a mix of agile techniques and methods based on what we have learned and the recommendations of experts and researchers in app development to construct a general model for app development. We found no studies that analysed a mixture of all of the above (Alrabaiah & Medina, 2021). For that, we think this work is very important.

In our study, we found six main phases adopted by participants in the mobile application development lifecycle, which are: The idea and strategy, user experience design (UX), user interface design (UI), design to development handoff- issues and decisions, development, and deployment and monitoring. In addition, the agile approach was preferred. Most participants adopt that; 83% used agile methods to develop mobile apps, while 49% believed agile methods were the best way to develop apps, and 34% believed they sometimes were. Most participants (56%) used Scrum methodology in mobile app development, and about 13% used XP and Kanban (Alrabaiah & Medina, 2021) (Figures 4.10 and 4.11).

These results are in line with some research studies concluding that agile methods are ideal, a natural fit for their mobile app development practices, with Scrum as the most-used method. In particular, (Martinez et al. 2020), (Flora et al., 2018), (Ghandi et al., 2017), (Holler, 2006), and (A. Ashishdeep et al., 2016) agree that the agile methodology is best for

mobile apps because it follows a set of iterative incremental approaches that help the project adapt to changes (Alrabaiah & Medina, 2021).

When we adopt agile approaches in our organisation, we know that one point in the Agile Manifesto is customer collaboration over contract negotiation. User-Centred Design (UCD) is the core of the development process. UCD is a design philosophy and iterative process that aims to understand user needs and create products that meet end-users' needs. UCD focuses on understanding users and their framework through all design and development stages to achieve the greatest satisfaction and best user experience possible (Rubin & Chisnell, 2008). In this line, this study shows that 62% of participants request user feedback during and after UI; 32% said they sometimes do (Figure 4.26). Most participants (90%) requested user feedback and obtained final approval after completing the design phase. About 53% of participants shared and sought user feedback by choosing the infrastructure to host applications with clients and considering how this choice will affect cost, performance, scalability, and reliability; 38% said they sometimes perform these tasks (Figure 4.23) (Alrabaiah & Medina, 2021). According with this, Aguilar and Zapata (2016) find it helpful to coordinate UCD and Agile methodologies in software development for more useful programming elements. Pratt and Nunes (2013) suggest improving user satisfaction until we reach the highest user satisfaction rate. We need to put users at the centre of the design process. As Rubin and Chisnel (2008) pointed out, UCD is valuable for achieving greater usability in software (Alrabaiah & Medina, 2021).

On the other hand, one of the main challenges that developers face is the transition from design to development phase. Our findings on best handoff practices suggest using the same team for design and development services in the project and using some of the tools available to manage the handoff. We found that most of the respondents tried to adapt to the transition from design to development using certain tools (Alrabaiah & Medina, 2021). These considerations and the rest of issues raised of this study will be taking into account to define our own methodology in the following chapter.

On the other hand, when the development team starts planning for the Sprint, determine how developers can reuse code. Approximately 65% of participants reuse code throughout the development process; only roughly 8% said no. These figures are consistent with many studies and reports. In a report by Mojica et al. (2014), the practice of re-using software among mobile application developers is described. Ruiz et al. (2012) measured that 61% of Android app classes appeared across two or even more different apps (Alrabaiah & Medina, 2021).

Regarding to the iterative and incremental practices and Sprints, most of the participants (48%) indicated that they use an iterative design between the framework and the workflow; 36% said they sometimes do. The development of mobile applications is an iterative cycle process; most participants used iterations in development processes. Most participants (50%) answered that they performed Sprints and iterations in the development process; 35% reported that they sometimes do (Alrabaiah & Medina, 2021).

As Rubin and Chisel (2008) pointed out, designing, modifying, and testing items should be an iterative process. The Agile software development manifesto (Beck et al., 2001) states that agile methodologies are iterative and incremental, focusing on short, frequent cycles and the functional product (Alrabaiah & Medina, 2021). Once the Sprint is complete, the results should be returned to the project manager or quality assurance department for review. About 70% of participants doing this, and 15% sometimes do. Team collaboration is the key to success (Alrabaiah & Medina, 2021).

The iterative model demands a new process for product development and testing. Testing in an iterative model means that each iteration goes through unit tests, component testing, and integration testing (Choodi, 2006). The product goes through system integration testing and acceptance testing during the final iterations. Functional testing starts with the test plan and a list of actions to test. Approximately 67% of the participants perform testing; 27% sometimes do. Performance testing and app responsiveness are necessary for late iterations - Sprints. 77% of participants use and focus on usability testing, 67% use and focus on user acceptance testing, and 55% focus on performance testing. As Mascheroni and Irrazabal (2018) point out, several manufacturers have confirmed that continuous testing is a way to solve quality problems in continuous delivery based on continuous testing. An attempt to include the designer in the testing cycle (Balaziuk et al. 2020), we suggest involving designers in the testing process, they will let them see whether their vision was developed as they described it. Most participants agreed on using this phase (47% said yes; 50% said sometimes). Rubin and Chisnell (2008) recommend that users and the design team be in direct contact throughout product development, including the testing phase. Therefore, in order to ensure a more genuine experience, the developers should not perform testing (Alrabaiah & Medina, 2021).

Finally, it is evident that none of the companies included in our previous study tool (Alrabaiah & Medina, 2019) has faithfully or integrally adopted any existing development methodology, and this corresponds to the majority of questionnaire participants, much less reviewed existing academic proposals to improve these processes. The authors believe that

more work is needed to develop new methodologies that embrace the solid principles previously identified by the academic community and the latest advances and interesting proposals developed and developing in the scientific community. While at the same time presenting the feasibility the company needs, enriched by the industry's experience. The next chapter will present our new methodology for developing mobile applications (Alrabaiah & Medina, 2021).

CHAPTER 5. THE PROPOSED METHODOLOGY: AGILE BEESWAX

The main contribution of this research is the Agile Beeswax methodology for developing mobile apps based on the knowledge of mobile application experts and academic/researchers communities (chapter 4) and an extensive literature review (chapters 2 and 3). This chapter will propose our Agile Beeswax model and its main components. We will start with an introduction, Agile Beeswax practices, the proposed methodology, phases, Agile Beeswax general rules, team roles, and applicability and limitations. Later, we will compare the proposed model with the other reviewed methodologies in chapter 3.

5.1 Introduction

The process of creating software for smartphones and digital assistants is known as mobile application development. There are games, social networking applications, and eCommerce apps among these mobile apps. To develop these apps usually a methodical mobile app development process should be followed. Since the beginning of app development, a lot has changed, and it feels like developers and designers have been on a roller-coaster of evolution. "How do you get started developing apps?" is a common concern from new customers and developers. The look and feel of apps, the tools utilised, the application design process, and how we go through the many stages have all changed. For this reason, our objective is to present a new agile-based methodology for app development that we call Agile Beeswax. Agile Beeswax was conceived after identifying the mobile development process's challenges and unique requirements.

According to Abrahamsson (2003), the requirements for a good software development process are the following: The development process should methodically support and enable quality software development (production frame), be easy to understand, clear, straightforward, and adaptable. This model should be intended for small (and medium) development teams (most teams are small), aligned with the company's strategic planning, and produce observable results early (early validation). According to Abrahamsson (2002), a software development method is agile when it is incremental, small software releases, rapid

iteration, good communication and cooperative with customers, and developers working together. Furthermore, it must be straightforward, efficient, and easy to learn and modify. Also, it must be adaptive, and can make last-moment requirements changes with quick response.

Based on this discussion, we can extract a set of characteristics that a suitable methodology for mobile development should have. For example, this new methodology should take into account the main phases in the mobile development process identified from the expert knowledge and literature review described in chapters 2, 3, and 4. These are: idea and strategy, user experience design (UX), user interface design (UI), design to development handoff (issues and decisions), development, and deployment and monitoring. In addition, in our opinion the new methodology should be based on agile and Scrum management practices and project management practices. This methodology should support small releases, customer involvement and feedback, self-organisation, fixed meetings, and short Sprint iterations. In addition, a useful methodology for mobile development should add more operational practices, manage the mobile app as a product, and control the process to move from the design to the development phases effectively.

Therefore, an iteration and incremental methodology should be the core in the mobile app development process. Finally, we cannot forget how important engineering practices are to develop an app. Thus, the new methodology should contemplate code reuse, continuous integration, continuous delivery, and small releases. The next section will explain the main practices adopted by the proposed methodology Agile Beeswax, following these considerations and reflecting how the obtained results during the performed studies are incorporated into our proposal.

5.2 Agile Beeswax Practices

Agile Beeswax practices are based on three main components: Agile and Scrum practices, software engineering practices, and operational practices, as shown in Figure 5.1. These three components are working together to develop a mobile application. It is designed to fulfil the unique requirements of mobile software development as well as industry quality standards. Additionally, it is designed for a group of up to ten team members who operate in (or near) co-located office space, but it is ready to adapt for a multi-site team. All these are based on our previous chapters' findings since most team members in the app development industry were small teams in co-located office spaces.

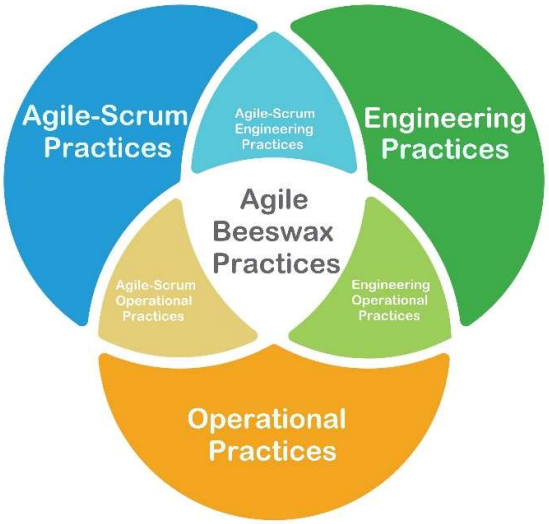


Figure 5. 1: Agile Beeswax Practices

Each component will be discussed with more detail in Figure 5.1 and the following sections.

5.2.1 Agile and Scrum Practices

In the first place, some Agile and Scrum practices are adopted in Agile Beeswax because they have demonstrated effective methods for developing mobile apps (Flora, 2018).

As Figure 4.11 in chapter 4 shows, most mobile development experts use Scrum in developing mobile apps (56%). Scrum is an agile project management framework that uses a small-to-medium-sized team. The team is managed by a Scrum Master, whose main job is to remove all team restrictions on completing the tasks. Scrum's approach is to direct and manage iterative cycles at the project level. With relatively lightweight and time-boxed, Scrum provides fixed meetings and an all-time project overview, details of delays, to-do lists, and a completed tasks list. Scrum is also well known, widely used, and easy to understand. Schwaber (2004) describes Scrum as iteration-based, quick to adopt changes, offering short Sprints, and matching the requirements changing of mobile projects.

Agile and Scrum practices include, for example, project management practices, Sprint planning, definition of Product Backlog, definition of Sprint Backlog, small release, customer involvement and feedback, self-organisation, fixed meetings, short Sprint iterations, Sprint retrospective meetings, high control process, backlog creation, retrospective and review meeting, definition of roles such as the Scrum Master, the Product Owner and the Scrum Team, etc. Agile Beeswax uses some of these practices, not all of them (we will explain it in more detail in section 5.2.4 in this chapter). However, since Scrum is a management framework and does not say much about specific engineering or technical practices for

software development, it must be supplemented with engineering and technical practices essential to mobile app development. Those practices are the second part of Agile Beeswax practices.

5.2.2 Engineering and Technical Practices

In the second place, the app development process requires some core engineering and technical practices borrowed from XP and software engineering and other approaches to support functional and non-functional capabilities such as reliability, performance, and security. Engineering technical practices include behaviour-driven development, continuous integration, continuous delivery, small release, refactoring, accepting changes in iteration at any time, test-driven development, requirement prioritisation, unit testing, coding reviews, and automated acceptance tests. Agile Beeswax includes some technical engineering practices (we will explain it in more detail in section 5.2.4 in this chapter), as shown in Figure 5.1.

5.2.3 Operational Practices

In third place, although Agile and Scrum provide a project management framework, and we have specified engineering practices for software and mobile development, we need another hand to manage the full process that guides every step of a mobile app: "the product" lifecycle from start to end. There, operational practices are included in Agile Beeswax. Operational practices focus on evaluating the development process, project management, eliminating waste, performing continual learning & improvement, generating continuous delivery, planning for unplanned work, flexible processes, accepting changes in iteration at any time, ready to adapt to changing priorities, customer involvement, and identification of issues that require immediate response and cannot wait for the next Sprint Planning session. Furthermore, since the participants used Kanban and Lean (13%, 9% respectively), we decided to incorporate some of their operational practices into Agile Beeswax.

5.2.4 Integration of Practices for Agile Beeswax

In conclusion, the development of mobile apps requires intersecting all these three types of practices. Therefore, Agile Beeswax practices are an intersection of Agile and Scrum practices, Technical Engineering practices, and Operational practices (see Figures. 5.1). Agile Beeswax practices include techniques and practices used by experts in app development. Major adaptation practices are: working in short iterations in the design and development phases, fixed meetings, Sprint review, Product Backlog, test-driven development, continuous integration, design improvement, small releases, and incremental design and development,

coding standards, simple design, user centre design, control of design to development handoff, continuous testing, regression testing, waste elimination, amplification of learning, delivery as-fast-as-possible, team empowerment, and continuous learning and improvement.

Table 5.1 summarises the practices adopted in Agile Beeswax (these practices have been categorized in major adoption, partial adoption, and non-adopted practices depending on if they are adopted or not and the frequency of their use in affirmative case).

From Figure 5.1 we can find that there are intersecting parts between these sections. These intersecting parts are shared with the same practices. Operational and Engineering practices intersect in practices such as accepting changes in iteration at any time, customer involvement, or ongoing delivery.

Table 5. 1: Practices Adoption in Agile Beeswax Methodology

Major Adoption	Partial Adoption	Non-Adoption
Agile and Management Practices		
• working in short iterations in the design and development phases • fixed meetings • Sprint review • Product Backlog	• project management practices • self-organisation • Sprint and Sprint planning • backlog creation • Scrum Master • Product Owner	• Sprint retrospective meetings • high control process • retrospective and review meeting • Scrum Team
Software Engineering Practices		
• test-driven development • continuous-integration • design improvement • small releases • incremental design and development • coding standards • simple design • user centre design	• continuous delivery • accept changes in iteration at any time • requirement prioritisation • unit-testing • coding reviews • automated acceptance tests	• behaviour-driven development • refactoring

• control the design to development handoff • continuous testing • regression testing		
Operational Practices		
• waste elimination • amplification of learning, • delivery as-fast-as-possible • team empowerment • continuous learning and improvement	• continual learning & improvement • continuous delivery • planning for unplanned work • flexible processes • accepting changes in iteration at any time • ready to adapt to changing priorities • customer involvement	• project management practice

As we can see in Table 5.1 Agile Beeswax does not adopt all practices. These not adopted practices mostly dos not fit the mobile application characteristics. Furthermore, we found that these practices are not adopted by the experts.

5.3 The Proposed Methodology: Agile Beeswax

When Agile Beeswax was designed, the requirements for a good development process described above were considered. Agile Beeswax is an incremental, iterative development process composed of two main iterative loops (Sprints), the incremental design and development loops, and one bridge connecting these two Sprints. The core of the methodology comes from Scrum management practices. However, since Scrum is a management framework, we add the engineering or technical practices for software development and continuous learning and improvement operational practices to manage the full process that guides every step of a mobile app s development process: "the product" lifecycle from start to end, as seen in Figure 5.2. The Agile Beeswax core process consists of six phases, and it is numbered from 1 to 6.2. Furthermore, Figure 5.2 shows the major software practices and operational practices. We choose the most practices that match the development phase during the development process and the controls from the beginning till the end of the app development process.

Next, we will explain more the core of the Agile Beeswax methodology, and we will start by describing its functionality, rules, team roles, and the main phases in the next sections.

5.4 Agile Beeswax Functionality

There are two main iterative loops in Agile Beeswax, the iterative, incremental design loop and the iterative, incremental development loop. Furthermore, one phase connects these two loops (Sprints), this phase is design to development handoff and technical decisions, as seen in Figure 5.2. The tasks that are performed in each loop are described below, and we will talk about it more in section 5.7.

1) At the beginning of the project and after kickoff meeting, development team will discuss the app idea and its strategy, until they define the main requirements. This will allow the development team to create the initial concept Minimum Viable Product (MVP), which present the most basic running form of the app, as we will describe in more detail in section 5.7 in this chapter.
2) First loop: Design Sprint
    - Incremental UX, incremental UI: The goal of incremental UX and UI is to create an iterations component that can be built piece by piece, based on Minimum Viable Product.
    - The first design iteration must satisfy the basic user needs.
    - The team start creating the wireframes, then add user stories for each screen.
    - The team should start the design Sprint by defining the requirements and tasks to be implemented during the next Sprint.
    - Before crossing the bridge to the development process, the development team should review and test the output if it matches the requirement of the app.
3) When the team completes one Sprint in the design phase and has approved the front-end, the team should send it to the second loop, crossing the bridge (design to development handoff and technical decisions).

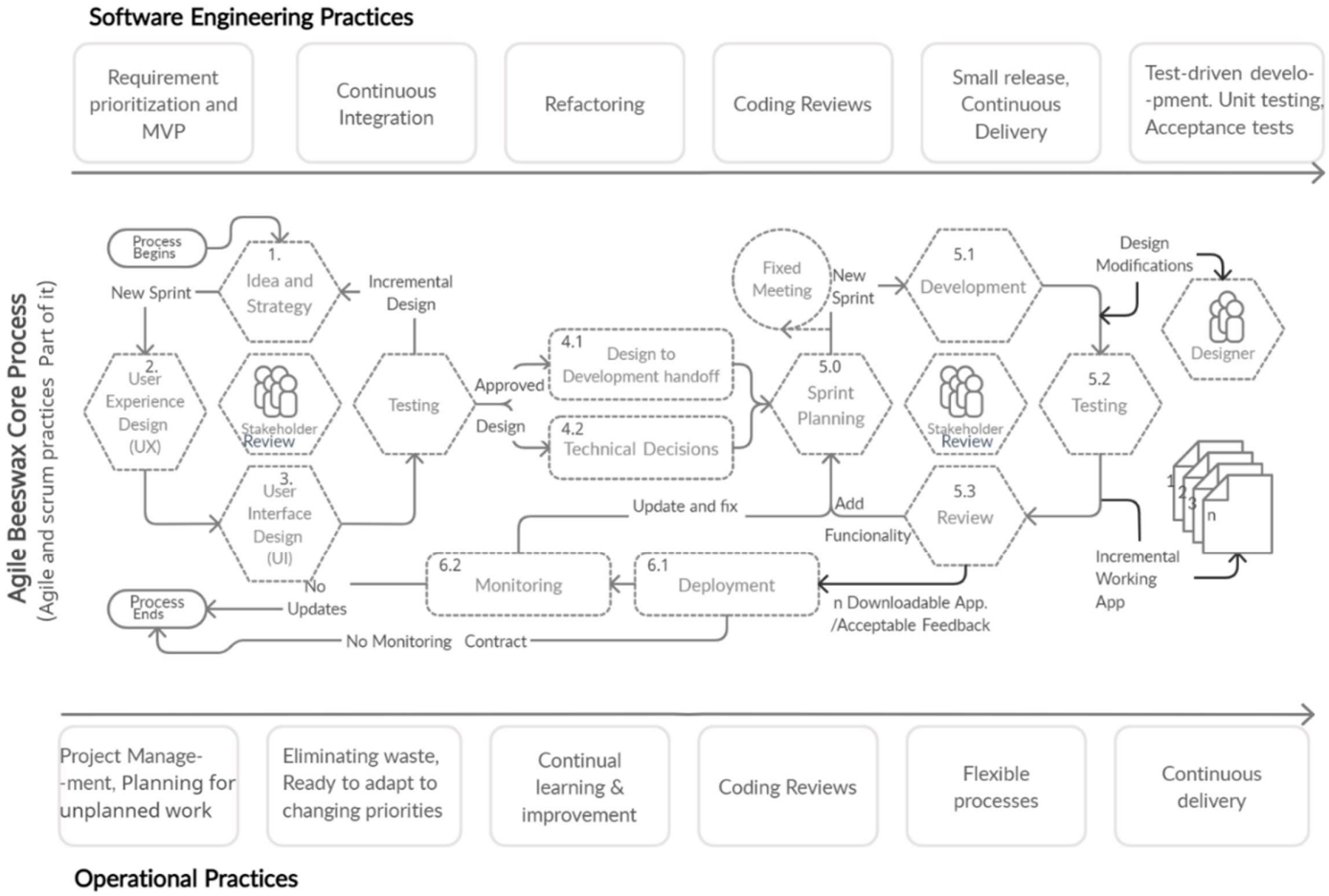


Figure 5. 2: Agile Beeswax Methodology with Main Practices

4) Second loop: Development

- We apply some Scrum practices to the development Sprint phase as described in section 5.2.1 and 5.2.4.
- The Product Owner defines the main requirement of the app and begins to break the high-level requirements into smaller user stories for each app screen in discussion with customers and other stakeholders.
- The Product Owner writes user stories as a Scrum Product Backlog and initiates a prioritisation session with the architect and developers.
- Each Sprint typically lasts 2-4 weeks.
- In Sprint 1, Day 1, the team starts with the Sprint planning and Sprint planning meeting. The Product Owner presents the Scrum Backlog items prioritised from highest to lowest necessary. The team discusses which items and stories will be completed by the end of this Sprint.
- Testing requires some engineering practices, such as test-driven development, continuous testing, continuous integration, continuous delivery, and automated acceptance tests. Testing should be performed regularly, as it will significantly reduce the financial costs incurred at each stage. As we progress in the development process, the costs of fixing bugs increase. It is often important to improve the planning and original design documents when creating different test cases.
- The team complete the Sprint by reviewing the results of this Sprint and adding new tasks and functionalities for the next Sprint if needed.
- After reaching the last Sprint and receiving feedback on it, the team move to the deployment phase.
- After completing the development process, the team apply for the deployment and monitoring. This is the last phase in the app lifecycle, where the development team publish their app to the app stores and starts monitoring it for any future bug fixing or any updates. Some companies do not make a maintenance contract with the customer. If this is the case, we have reached the end of the process. But if they have a maintenance contract with the customer, which is usually about one-year errors fix contract (based on our findings), the bugs or updates can be sent to the design team (the first loop) or the development team (the second loop) to be solved.

5.5 Agile Beeswax General Rules

For the best adoption of Agile Beeswax, we need some general rules. The development team can adopt these rules. These rules are still flexible and adaptable to their needs. With practical experience, these rules are changeable to their needs while using the methodology.

In Agile Beeswax, the development team should focus on continuous user/customer feedback in the all-project lifecycle. Furthermore, a fixed meeting between the development team is important for the project to succeed. Meeting sessions are not daily, but meetings should be on a fixed schedule. Mobile app development team sizes are often small and work in the same location. It gives the team the ability to communicate when they need it quickly. Another important thing while building the app is to create the app incrementally, block by block, until we reach the final product using Sprint. At the end of each Sprint, some operational practices, such as continuous learning and improvement, are needed. Sprints must be flexible to incorporate some modifications during the Sprint's life as an operational practice. Agile Beeswax requires an immediate response during the same Sprint without waiting for the next Sprint Planning session (only in critical situations). Note that one of the rules in Scrum management, is that the development team cannot add new requirements during the Sprint, they have to wait till the end of the sprint. Additionally, we believe that designers should be always in the development phase, working on updates and Sprint reviews.

5.6 Agile Beeswax Team Roles

Most software teams use various Agile project management methods to manage and handle their work optimising workflow and ensuring rapid app development without decreasing quality. Agile methods such as Scrum, Kanban, and DevOps focus on the vision and the app's idea, limiting the drifting away from its original aim during the development process. Agile and Scrum highlight customer collaboration and continuous feedback, small release, team collaboration, short Sprint iteration, high control, incremental, and iterative development. Getting a suitable team size is still a challenge for Project Managers. A large team can divide the work evenly, but small enough can communicate and be easy to adapt (Pendharkar & Rodger, 2009).

Agile Beeswax supports small to medium team sizes. Vijayasarathy & Butler (2015) mention in their research that agile and iterative teams were more likely to have small teams (Agile 69.8 % and iterative 80.0% teams); keep in mind that small teams are less than 10

persons. Accordingly, 10 is the maximum number of members stablished for the Agile Beeswax teams.

The mobile application development team roles were discussed in the 4.1.4 section to analyse the data collected from experts in companies, academic researchers, and literature review. Accordingly, the presented Agile Beeswax methodology shows its configuration of the team composition based on this information. We believe creating a high-quality application requires dividing the work into several people to specialise in a particular job. Furthermore, as we mentioned in chapter 3, that making use of responsibility, self-organisation, the leadership of flexibility, and an agile mobile app development methodology enhance the development of high quality and suitable mobile application. Of course, we find that some team members do more than one role. Agile Beeswax development team core roles are Project Manager, Product Owner, UX and UI designer, Developers, QA lead.

We will inform the names of all those team roles as follow:

Project Manager (PM): The Project Manager is the development team lead. Every development team needs a Project Manager. PM is responsible for the success or failure of the project. PM works with the clients to define the objective, scope, cost, and schedule. Other responsibility of this role is to fix high-level troubleshoots issues (integration of development, project testing, and deployment).

Scrum Master (SM): The Scrum Master is a management role responsible for ensuring that the project is carried out according to the procedures, values, and principles of Scrum and the plan. The Scrum Master does not lead the team. The Scrum Master communicates with the project team, clients, and management throughout the project. Scrum Master is more responsible for low-level troubleshoots issues (fewer responsibilities than the Project Manager). The Project Manager takes a more general position; the same person can play two roles in some teams.

Product Owner (PO): The most important responsibilities for the Product Owner are to be responsible for the Product Backlog list, control, maintain, manage, and make the Product Backlog visible to everyone in the project. Managing the Product Backlog list with the highest priority first is another responsibility of this role. They must also represent the client's interest through the requirements and their prioritisation.

Design Lead / Designers UX, UI: they are responsible to create all look and feel and all the app's visual and audio assets.

Development Lead / Developers: Also called coders, this role is responsible for all coding, fixing bugs, back-end development, API, and all required for the database. This role

is also responsible for developing the whole application architecture, deployment, unit testing, and continuous monitoring. Another of their responsibilities is in creating a secure app that is resistant to all types of technical attacks.

Analyst: This role is responsible for maintaining requirements documentation and how it can fit the budget. They are also a communication facilitator making sure that we are building the right product. Scrum Master or the Product Owner can fill this role.

Quality Assurance Lead (QA): Quality Assurance is not optional when creating a professional product. The main tasks of the QA are to identify bugs and problems of the app, complete manual, automated testing, create trackable reports on bugs, and create test cases.

Information App Architects: The architect presents a secondary role in Agile Beeswax methodology. Application architects supervise the design and development process. We need the architect to connect the design and the development teams, which will help in the app documentation. Application architects can estimate and recommend software and tools technologies used in the development process.

One of the challenges of some companies regarding the development team we found in our study was maintaining a stable development team in the company as better new jobs are always offered to the team members. One of the best solutions is that the team member feels that they are still learning and improving. The psychological dimension for the team is also essential. Agile Beeswax proposes taking into account these considerations during the team construction during the development process.

5.7 Agile Beeswax Phases

As seen in Figure 5.3, Agile Beeswax is structured in six phases,

1. Idea and strategy,

2. User experience design,

3. User interface design,

4.1 Design to development handoff and

4.2 Technical decisions,

5. Development (5.0 Sprint planning, 5.1 Development, 5.2 Testing, 5.3 Review),

6.1 Deployment and

6.2 Monitoring.

One of the main strengths of Agile Beeswax is that it has been created with academic and business perspectives to bring these two communities closer. Both perspectives have been integrated in each phase.

As we mentioned before in chapter 3, about the main issues and challenges during the development process of mobile application, they are subject to being taken into consideration in the initial phases of the mobile app development to mitigate the impacts resulting from poor choices for the successful development of the mobile apps. We will explain these six phases in more detail.

Phase 1: Idea and Strategy

This phase contains three main subphases: Idea workshop, Strategy, and Marketing campaign, as seen in Figure 5.4.

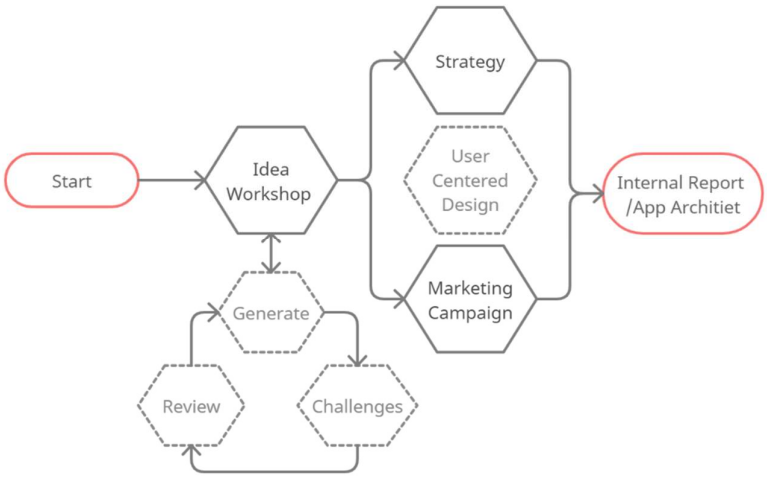


Figure 5. 3: Idea and Strategy Phase

Phase 1.1: Idea workshop, apps start from an idea. This idea may be a solution to an existing problem, a way to cover some of the users' needs, etc. We can develop ideas based on the ideas of others. The idea could be a product (app) one day, so it is essential to handle this process. The objective of the idea workshop is to come up with the best version of the app idea. Idea workshop is an iterative looped process. It could start from "create," "discuss," and "review." (Khalid et al., 2015).

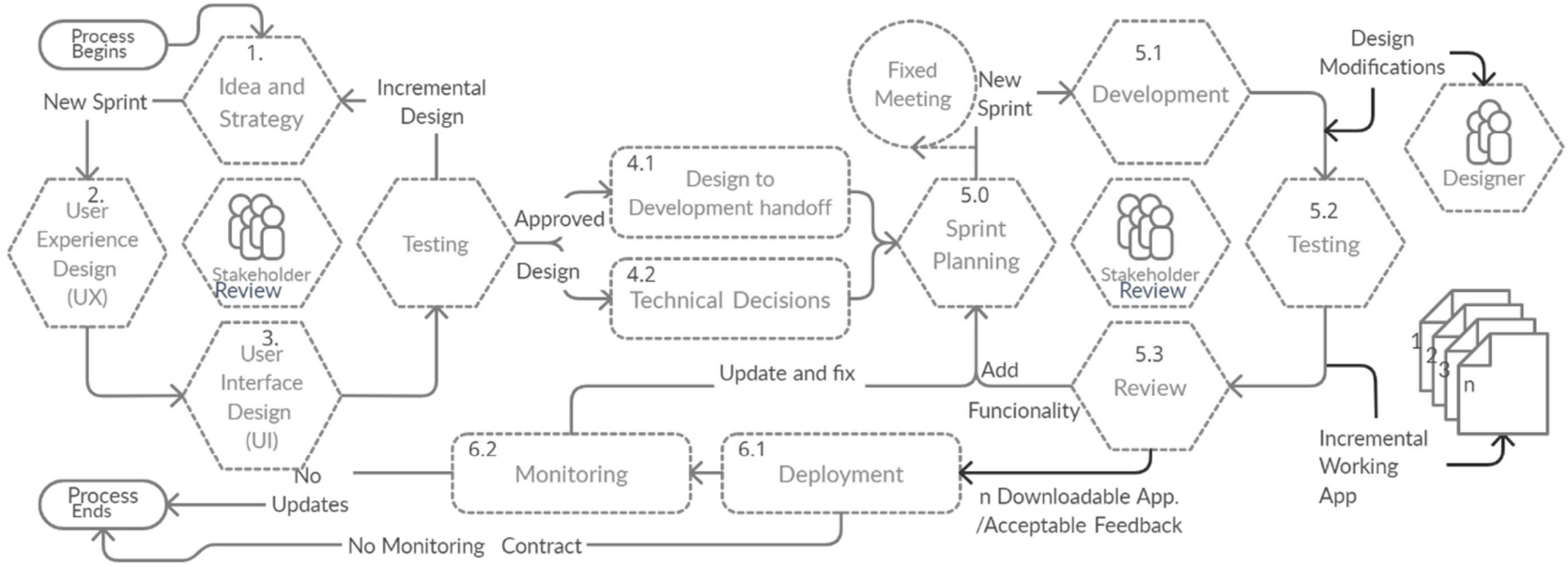


Figure 5. 4: Agile Beeswax Development Process and Phases

In the specific cycle of mobile development where users have an infinite number of apps at their fingertips, the realization of a good idea is essential. To create an idea, we suggest starting by asking several questions, such as "Is there a better way to do things using an app?" or "Can we solve this case in a better way?" "Can this app solve a problem?" (Green et al., 2020). Next, the development team starts discussing the idea and the problem and why no one else creates an app to solve this problem. What characteristics of this idea set it apart from others? What are our plans for bringing something different to the market? (Green et al., 2020). Software development needs technical expertise and creativity (Pham et al., 2018). Review the idea, discuss if this idea is a viable concept from a technical standpoint. Is it possible to make it? Who should oversee it? How would we go about putting it together? What kind of APIs, data, and tools do we require? What are the challenges to placing the idea into action?

Idea workshop is a method to drive ideas to evolve. When asked to put everything down on paper, the development team will be surprised at how much the concept idea has been developed, modified, or refined. Uncertainty is inevitably highlighted, and new questions are asked. Collaborate with others is essential to generate and revise ideas. It is time to put the ideas down on paper after we are pleased with them. Thus, it is not enough to have the idea and vision in our heads when they are clear. The team should list them and discuss them with others, such as the company team, workers, or even friends. We think the development team will be aware of the project goals through the definition of the idea.

On another part, when the client has the idea, a meeting is held to obtain the application's requirements and objectives. Usually, customer requirements are high and changeable. Consequently, the customer is asked that these requirements need a more profound analysis before submitting a project proposal. Here the customer is oriented on what is appropriate for them and the total cost of the application. In the meeting, the development team can have closed-end questions whose answer ends with yes or no. Subsequently, a kick-off meeting begins within the company. This meeting will identify the team, present the application's idea to be developed, and distribute the most important tasks and tools used in the development process. They will try to get new features for the mobile application and classify the essential requirements.

Software like Pages and Microsoft Word or any simple markup software will help in this phase. Using Trello software to track the idea and prioritise the process can help too.

Trello is an example tool to track the project and ideas cards, as seen in Figure 5.5, where a Trello board is showed.

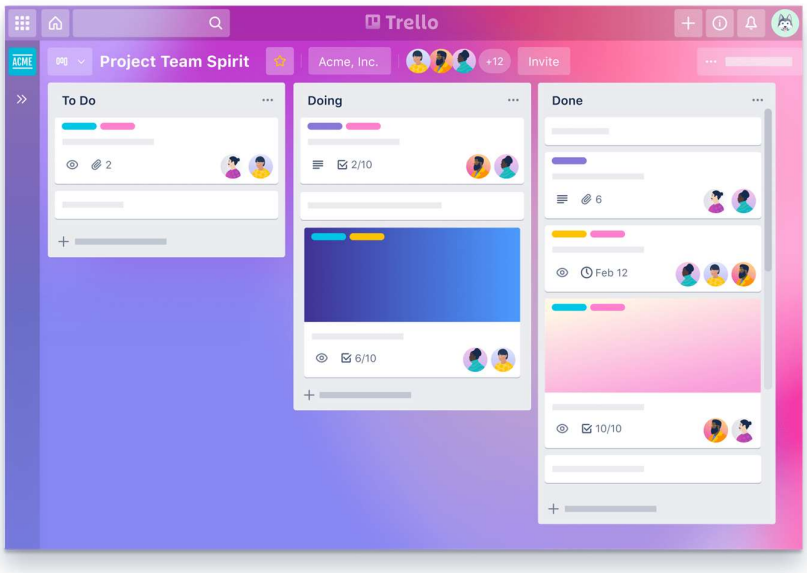


Figure 5. 5: Trello Board

On a Trello board, lists and cards are basic blocks to organise the development tasks. Task assignments, timeframes, productivity metrics, calendars, and more features can be added later. Cards can be moved from the To-Do list to another list easy. This tool can be good to organise the idea, but, of course, the Beeswax methodology does not impose its use; the development team can choose the best tool for them.

Phase 1.2: Strategy, the team can evolve the idea into a successful app by defining the strategy. Identifying challenges, a roadmap, potential users, legal and security issues, and the app's future are the main tasks in this phase. In this phase, the team should write out what the app must do and its core functionality. Both functional and non-functional requirements should be listed. App strategy involves identifying what the app is, what it is for, what we want from it, and what we hope it will become someday. The best place to start the strategy is to identify competing apps that have similar ideas. We can have free data from competitors, the number of installs, ratings and reviews, and app history, which will save some iterations in the development process.

Many companies avoid entering the strategy stage. They can certainly do this, but they must remember that prevention is better and cheaper than cure and correction, and this is

cheaper than failure. This phase could cost nothing if we compared it with the failed application.

After deciding on the best solution, the development team should prioritise the app features and map out the Minimum Viable Product (MVP). MVP is to obtain the necessary to and the most basic form of the app **(**Lee, & Geum, 2020). By identifying the app's road map, we could locate the app's MVP. The development team could start writing all the app's basic functionality on a whiteboard, and they can add these functionalities in the future.

In addition, the Agile Beeswax methodology proposes to use a visual representation of the strategy since keeping everything in one's mind is difficult. Therefore, all the critical elements of the app development process should be summarised and held in one place. The proposal of Beeswax is using the application development canvas, whose aim is to put all the major app components in one place (Date, Version, Notes by, App name, Team, App icon, App Concept, MVP, Platforms, and Languages). The canvas will be beneficial for anybody participating in the app development process. For this purpose, software like Miro will serve. Miro gives templates that we can use, but it also allows the use of a blank canvas (Pigrum, 2021). Figure 5.6 is an example of an app canvas (Miro, 2021), which can be useful to structure all elements in the strategy and the app.

There are 21 boxes in the App Development Canvas. According to the explication in (Miro, 2021), let's take a closer look at each of them:

1. **Date** – The date in this box.
2. **Version** – Specify the version of the app.
3. **Notes by** – The name of the person taking notes on the Canvas.
4. **App name** – The name of the app.
5. Team – The names and roles of everyone in the team participating in the app development process.
6. **App icon** – Download and add the app icon to the canvas.
7. **App concept** –What is the purpose of the app? What should it be doing right now and in the future?

Figure 5. 6: Miro Canvas Template

8. **Question** – What problem are you trying to solve, and why is it so urgent? Every app should be a solution to a problem (or a set of problems).

9. **MVP** – Consider the most basic form of the app and describe it here.

10. **Platforms** – The team should make a list of the platforms that the team will be working with.

11. **Languages** – The team should list the languages they will be using in the app.

12. **Price** – The team should add up all the costs and put a number here, salaries, software and content costs, and all possible expenses.

13. **Budget** – The maximum amount of money to spend (the clients or the development team).

14. **Target group** – The audience that we want to reach is referred to as a target group.

15. **Characteristics** – The team should write down all the app's features in this area. (More than the MVP).

16. **Existing applications** – A list of competitors and similar apps.

17. **Success factors** – A list of the things that will aid app success.

18. **Core functions** – A list of the app's most important features and prioritise them.

19. **App discovery** – How potential users will find the app. (App Store, Google Play, referrals, social media, and search).

20. **Discards –** App or solutions that users will abandon in favour of the new app.

21. **Vision –** How do the team see the app's future?

Phase 1.3: Marketing campaign, this phase starts with a marketing strategy, identifying a target market, figuring out how to contact them, communicating with them, developing a marketing plan, defining the app marketing challenges, and driving app adoption. App marketing is critical to a mobile app's success or failure (N Inukollu et al., 2014). In the app store, there is a much-unused app. Teams should determine the marketing budget; but they do not need marketing if the app is for internal use. For example, in this

phase the development team can employ App Store Optimization, Instagram Ads, Twitter Ads, Facebook Ads, and other marketing channels. The development team might use a mobile advertising company specialising in mobile media buying, such as Moburst, REPLUG, or Zoomd.

Outputs of the idea workshop phase: Once the development team has the idea and strategy, it is time to write an internal report to record all these elements. With some iterations, a proposal file is created, containing an overview of the company, its obligations (quality, ethics, and the use of the latest technology), a list of the application's characteristics, and the app canvas. Furthermore, the proposal could contain a marketing strategy, administration panel (Content management system CMS: is a software that allows teams to manage the creation and alteration of digital material, this functionality enable to manage content and increase productivity easily (Adithela et al., 2018)) if required, time, and cost. To-do items are listed, described, and prioritised. Mobile architecture is a step-by-step strategy that must be finished before the development process can begin and complete its improvement during the development process. Mobile architecture offers a diagram of how the application's various components should be structured and related to one another. It outlines the procedures to be followed throughout the development process. This will be one of the major reasons for obtaining a well-developed application that is well-written and easy to test, update, and maintain.

At this stage, sufficient time is given to understand and analyse the application requirements before moving to the design phase and drawing the application pages. Even if the team has a great app idea, they may still not effectively present it to the clients. This is what we will discuss in the next section.

Phase 2: User experience design

After reviewing the goals and requirements with the customer and the team, the team starts writing and creating blogs containing the application's data and functions and organising and structuring them. This phase is generally handled by the app architect and user experience designers, but the whole team must discuss the app's structure. The first blogs contain the app architect, a list of features, and what we want to deliver in the app and where.

Agile Beeswax proposes to use a 2X2 matrix prioritisation chart to prioritise the app features (Figure 5.7) (Rivera, 2020). Prioritisation chart is a strong management tool that assists the development team in making the most use of development time. It helps the

development team determine which features are the most important in developing the MVP so the team can first focus on the most urgent needs, depending on the value and complexity of each feature. Thus, the development team choose the app's main features and add new features in the future version of the app. Therefore, the team must focus on developing the (MVP)

Figure 5. 7: Prioritisation Chart (Rivera, 2020)

The user experience (UX) phase contains four main subphases: Wireframe, Workflow, Test workflow, and Review followed by the phase output resulting from this phase. Figure 5.8 shows the UX phase diagram.

As seen in Figure 5.8, the UX phase is an iterative, incremental process, starting from the app requirements, app architect, and app goals. UX Phase is ready for any future updates or editing throws the development process.

Always the user is part of the development process, including the UX phase. In the last subphase (the Review), if the UX app is approved by the development team and the customer, they can proceed to the next phase (UI). If not, start a new iteration. Now we will explain the UX subphases in more detail.

Phase 2.1: Wireframe, when the app information architect creates a list of all app features and where to present them, the user experience designers start to draw the app wireframe. A wireframe is a visual representation of projects and design content. Teams must first draw and create the screens and their functions and data in this step. Wireframes are the

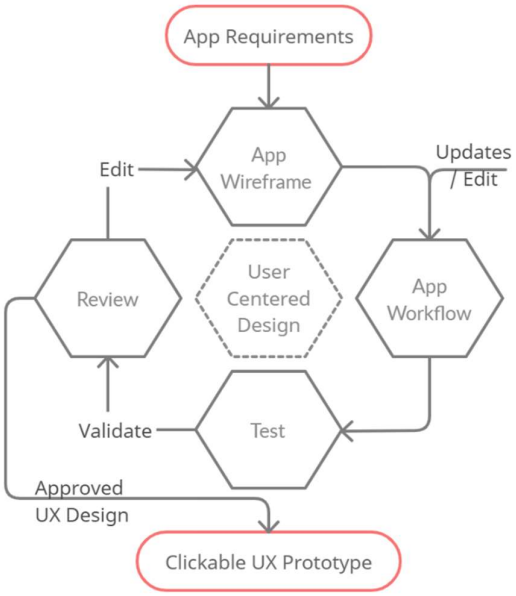


Figure 5. 8: User Experience Phase

project's initial planned design. Anything that was not being spotted in the idea and strategy phase generally shows up during this phase. The development team can start with pencil and paper to sketch the wireframe, which represents the requirements and visualises customer needs, based on app architect and app blogs lists. Even the development team can use digital technology such as Sketch or InVision to construct wireframe badges. According to Hamm, M. J. (2014), there is a possibility that using digital technologies for wireframing can speed up the process.

In addition, it is critical to comprehend and consider the majority of the application's functionality at this point. The team needs to sketch or draw to alter things now, but they must rewrite and redraw the code afterwards. UX wireframing aims to describe the app's flow and the app's front-end, including the number of windows, buttons, where each button takes the user, the registration process, and the login page. Figure 5.9 (Cacoo, 2021) presents an example of an app wireframe where everything is still general, the place of photos, text, buttons, and what the user will see on every page.

Phase2.2: Workflow, the workflow specifies the path between the wireframe badges that users can navigate through the application. Workflow diagrams aid in the identification of every probable interaction a user has with the app, as well as the navigation structure of the app. When designing a workflow, we need to pay attention to the number of clicks required to complete each task; it shouldn't require many clicks. If the task requires many clicks, the development team must return to the framework to solve this problem; designers should

make the mobile application more user-friendly. In addition, on each iteration, the team should always re-check all functionalities. Therefore, teams are also recommended to check the functionality and usability in this phase (Cacoo, 2021).

Figure 5. 9: An Example of an App Wireframe

Sometimes, several iterations between wireframe and workflow are needed. The team can draw the workflow on the whiteboard or paper with a pencil. Agile Beeswax adapts to any tools, but we recommended InVision which could be helpful to simplify the task for the next phases ("InVision," 2021). Figure 5.10 (Cacoo, 2021) is an example of a wireframe workflow using a pencil.

Phase 2.3: Testing, creating a tappable click-through model can improve the design. and it is important in order to validate the wireframe and the workflow with the users. Agile Beeswax adapts to any tool, but we recommended Sketch or InVision to test the wireframe and workflow, after which the team can import the design. In this level of design even without functionality, the click-through model can be provided to the user for testing and review. The team attempted to locate the issue, update it, and continue.

Phase 2.4: Review, at the end of this phase, we need to review the results of this phase between the team members and the customer. We will continue to the next phase (UI) if it gets approved.

Outputs of the user experience phase, after multiple iterations and many prototypes have been tested and finalised, the team will have a tappable workflow user experience (UX) prototype at the end of this phase. The team can draw the workflow on the whiteboard or paper. InVision can help too. Based on our findings, several experts advise adopting low-fidelity prototypes (ow-fidelity prototypes are frequently made on paper and do not allow users to interact with them).

Figure 5. 10: Wireframe Workflow (Cacoo, 2021)

Spend money on app functionality, features, and code instead of spending money on pricey prototypes. The Agile Beeswax methodology allows teams to use the tools they want, but we recommend these tools that were presented earlier.

Phase 3: User Interface Design

The UX wireframing and prototyping are concerned with the app functionality and how it works, whereas the User Interface (UI) design is concerned with how the app looks. The development team can start the UI designing phase when the UX design has been tested

and modified, it implies that many prototypes have been tested and finalised before designing the user interface

The user interface (UI) phase contains five subphases: UI Style guidance, Replacement / Mockup, Click-through model, Test again and Review, followed by the phase's output.

As seen in Figure 5.11, the UI phase is an iterative, incremental process, starting from the output of the UX phase (UX prototype). Always the user is part of the development process, including the UI phase, where clickeable mockups and testing are performed. In the last subphase (the Review), if the app UI gets approved by the development team and the customer, they can proceed to the following two phases. If not, a new iteration for editing and updating starts. However, UI Phase is ready for future updates or editing throws the development process.

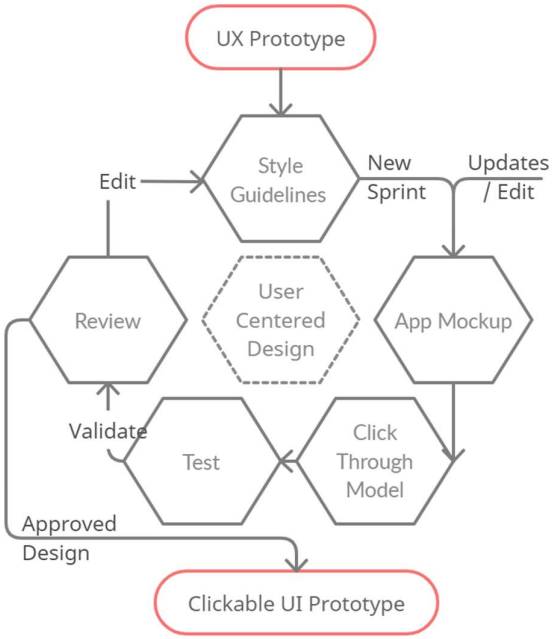


Figure 5. 11: User Interface Phase

Now we will explain the UI subphases in more detail.

Phase 3.1: UI style guideline, a style guide is a part of a design system that consists of a collection of guidelines for branding and the product visual style. It is a design language with a low degree of abstraction. It aids in the identification of aspects such as font, colours, layout, and components that have been approved for usage in accordance with brand rules. Good UI style guidelines will improve the usability of the app. To increase the app's usability and eliminate incompatible components and inconsistent templates, designers must see the

suggestions in the chosen style guidelines. R. Baharuddin (2013) mentions that one of the most important aspects determining app success is usability. Familiarity is an important property of good UI design. The ability to learn is a crucial aspect of excellent user interface design. Mobile applications are generally simple to use since they use a consistent design language. The development team should determine who the clients or the users are, which helps choose the best style guideline for the app. Furthermore, to reduce clutter, the development team should design the app for day and night. A highly skilled team assists in developing apps that are acceptable for all users. It is critical to recognise that there is no one-size-fits-all solution. Thus, user interface style guides are custom-made to the app's specific needs. Figure 5.12 is an example of Novus app UI style guidelines. It is very important to use a certain style for the mobile application, giving it a specific style.

Phase 3.2: Replacement/Mockup, in this phase, teams start updating the app element's (colours, buttons, logos, photographs, etc.) to transition from wireframe to UI design. This makes the app different from others. Agile Beeswax allows the use of various tools to perform this task, for example, the development team could use Skitch. Skitch is very helpful in adjusting apps screen elements that have various sizes. Figure 5.13 is an example of the Novus app Mockup (Bellatti, 2021), and we can see how the development team used the same UI Style guidelines and design elements in Figure 5.12 (Bellatti, 2021).

Figure 5. 12: Novus App UI Style Guidelines

Phase 3.3: Click-through model, provides a prototype for app UI projects. This phase starts by putting all the screens together, then it builds. Represents a screen navigation flow diagram and explains the purpose of each screen to the user. Teams can design the app style using Sketch and then import it into InVision to construct a click-through model prototype. This model let the development team share the colours and fonts with the customers and the

way of navigating between apps screens. Another option is to start with current iOS and Android UI components then adjust the design. Figure 5.14 shows the click-thorough model of the Novus app (Bellatti, 2021). The Agile Beeswax methodology allows teams to use the tools they want, but we recommend these tools that were presented earlier.

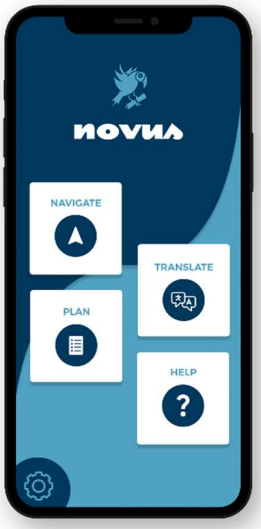


Figure 5. 13: Novus Mockup Home's Screen with its Four Main Features

Phase 3.4: Test again*:* At this point, it is necessary to re-test the model to verify that the user interface represents the app vision parallel with user feedback. We believe it is critical to review the click-through model prototype, repeat the test, and devote the necessary time to providing design, usability, and user feedback. It is important to keep in mind that further adjustments can be pricey. This is one of the main reasons why this phase is important for the Agile Beeswax methodology.

Phase 3.5: Review, in the review phase, the development team should test if the UI meets expectations. The team must do a detailed review of what has been achieved in one Sprint (iteration). The development team should talk to users or customers to get final approval and learn from everything done in the Sprint.

The outputs of the user interface phase: At the end of this phase, with some iterations, we will have a tappable user interface prototype. Furthermore, at the end of the UI design, the team should test and review all output results and see if they match the planned product

Figure 5. 14: Click-Through Model of Novus App

In Agile Beeswax, the work of UI designers does not end when the project is handed over to the developers. It is a never-ending, constantly evolving process. Up to the app's deployment, designers may find themselves providing and redelivering resources or adjusting interactions.

Based on the interviews with the experts, the app will not succeed even if the app idea is brilliant if the UI and UX design are low quality. Agile Beeswax adopts the engagement of the Product Owner at each level of the app design process to modify the final designs to meet the team's expectations. Finally, teams should remember to keep the app's users at the core of their thoughts. As stated in ISO 9241-210, 'Human-centred design for interactive systems: 1) user needs and context of usage; 2) user participation; 3) user evaluation; 4) iteration of

development solutions; 5) considering the entire user experience; 6) multi-disciplinary design team. UCD techniques vary widely, depending on the goal and stage of design (Preece, et al.,2001) (Gulliksen, et al., 2003), but they must be considerate within Agile Beeswax.

Phase 4: Design to Development Handoff - Issues and Decisions

The shift from design to development is one of the administrative problems/issues in application development. Misunderstandings and poor communication between the design and development teams are the most common roadblocks to a successful transition. We suggest having the same team to handle the design and development phases in the Agile Beeswax methodology. It is ideal to use tools like Sketch, Photoshop, or Zeplin, to produce a smooth transition. The Agile Beeswax allows teams to use the tools they want, but we recommend these tools that were presented earlier.

The design to development handoff phase contains five main subphases based on the issues and technical decisions to be considered: Customer approval, Technical design, Front-end, Back-end, and Database.

Phase 4.1: Customer Approval, as indicated in chapter 4, approximately 91% of respondents in our study received final customer approval after completing the app design. In Agile Beeswax, it is important to obtain the customer's approval to meet their needs and wishes when creating an application design. For this reason, Agile Beeswax adopts this phase.

Phase 4.2: Technical Design, teams can use various technologies and programming languages to design mobile applications, but teams must select a reliable technology. Each technology has its own set of benefits, some are less expensive, while others are less effective. Based on our findings from the interviews (Alrabaiah & Medina, 2019), many experts suggested adopting reliable technology during app development, such as native app tools, SQL, Google Cloud, Amazon AWS, Adobe. The proposed methodology does not require the use of any specific technology, but it does require a technological design process, where different alternatives must be analysed and compared and the technological choice must be justified.

Phase 4.3: Front-end, mobile applications can be developed in three ways: native platform, cross-platform native, and hybrid web technologies. There is a wide range of tools to develop mobile app. JavaScript, HTML, Ionic, React, and Ionic for example are tools used in web API app development. For APIs and database hosting, teams could utilize Google Cloud and Amazon AWS. Agile Beeswax does not limit the use of any language or tool in

the app development process. The development team has complete liberty to choose the appropriate development languages and tools for the front-end, since the diversity of applications makes it impossible to make a recommendation that fits all of them.

*Phase 4.4: Back-end, (*API / Web Server): back-end is the server-side development for processes, store, and secures data. When creating a web application, teams must select the back-end development language. Development team can use JavaScript, PHP, and C#. The Agile Beeswax allows teams to use the development language they want, but we recommend these tools that were presented earlier.

Phase 4.5: Database, creating a reliable and structured database for the application is very important. With this aim, SQL for example is widely used. Regarding the host infrastructure that hosts the database and API, Google Cloud and Amazon AWS could be used. Agile Beeswax considers that teams need to be concerned about its impact on price, performance, scalability, and reliability when choosing a hosting infrastructure. In addition, the hosting infrastructure should be shared with the customer. The Agile Beeswax methodology allows teams to use the development language they want, but we recommend these tools that were presented earlier.

Phase 5: Development

Building mobile apps is an iterative process. Most experts in our study use iteration or Sprints during the development process (Chapter 4). The Agile Beeswax methodology proposes some of the Scrum practices in the development phase. Since the mobile development process can be complex and challenging, adopting one methodology or framework will be helpful. We choose Scrum in the development phase too because it provides the development team with a high degree of transparency, showing the status of the project and the amount of time spent on it. Scrum can provide a functioning product with every 1-2 weeks and flexibility when adjusting the requirements. Scrum supports the encourages stakeholders and end-users in the development process to provide quick feedback on the app. The Sprint is the core of Scrum. Sprint is a specific time-box for a specific work to be completed and ready to review. Based on it, the development phase in Agile Beeswax contains four main subphases: Sprint planning, Development, Testing, and Review, followed by the phase output as seen in Figure 5.15.

Figure 5.15 shows the Development phase diagram. The output of phase 3 and phase 2 is the input of the development phase and phase 4 present the base of this phase. We can see that the development phase is an iterative and incremental process, based on Sprints to

implement an app. The process is welcoming to any edits or updates during the Sprint or the development phase. At the end of each Sprint, we have a working app or part of the app (from 1 to n) until the team reach the MPV or the complete app. The components are delivered in fast iteration by the team. Now we will explain the development subphases in more detail.

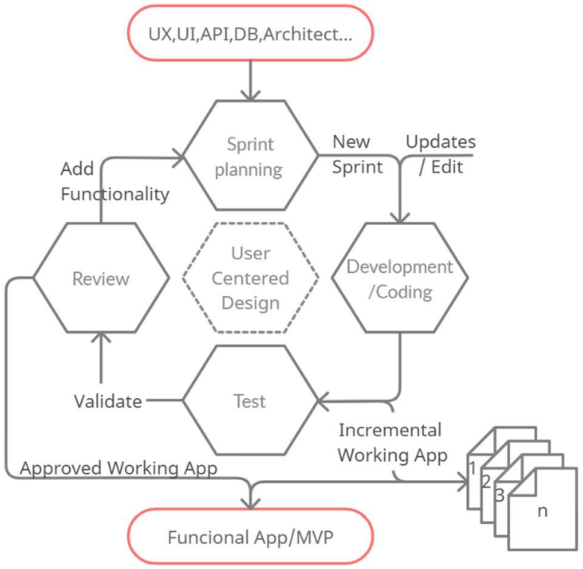


Figure 5. 15: Development Phase

Phase 5.1: Sprint planning, a Sprint begins with Sprint planning, which is a meeting limited to two hours for one week for one Sprint. Sprint planning is a list of tasks to be completed in one iteration. Sprint planning's purpose is to determine which tasks should be included in the next Sprint. The team discusses the tasks, ensuring that everyone understands and agrees on the acceptance criteria for each task. The Product Owner attends this meeting (for example, using online meeting tools) to plan the next Sprint actively. Each task should explicitly explain the amount of time it will take to finish it as well as its requirements. Developers must be included in the Sprint planning to comprehend each assignment. Each Sprint is structured in: Sprint planning, development, testing, review, and customer or user feedback. When developers begin planning a Sprint, they should think about how they can reuse code. Thus, following the Agile Beeswax, the team can now review the design and make the required adjustments. It is much better to update the design than to update the code. During development, a well-designed application causes fewer issues and adjustments.

Phase 5.2: Development, after the Sprint planning, the team will start developing the application's functionality. It is critical to provide the results to the Project Manager or

Quality Assurance for assessment. A detailed understanding of the requirements is essential to the success of this phase. In the event of failure or misunderstood, the team should notify the Project Manager of the deficiencies. In Agile Beeswax, we adopt to break down all requirements and app features into a Product Backlog (a Product Backlog is a list of prioritised features to be implemented as part of application development). Next, teams start implementing Product Backlog by Product Backlog using incremental development cycles. The components are delivered in fast iteration by the team.

Phase 5.3: Testing, to be more realistic, developers should not test the application. If the QA team is doing this work, they start with functional tests, including a test plan and a checklist of activities to be verified. Agile Beeswax proposes and performs functional testing mainly focusing on usability, performance testing, and responsiveness. In later Sprints, performance testing and app responsiveness are very important. In Agile Beeswax, we recommend including designers in the testing process; the development team should ask them to verify if their described vision has been developed. Furthermore, the development team must complete the regression test. The regression test is to test and review the previous Sprints.

In addition, since it is being developed for mobile, it will be necessary to apply on-device testing. Agile Beeswax requires testing the app on various screen sizes and versions. Google Firebase and native tools are examples of native or automated testing technologies which can be used. Figure 5.16 presents an example of using Google Fifbase to run Ropo test on multi-devices. Moreover, we believe it is critical to test the app on a physical device. The Security Tests and User Acceptance tests are the most critical.

Agile Beeswax adopts the result of the study of Shivageeta et al. (2020) to structure the testing process. Shivageeta et al. (2020) describe the testing in the iterative model as means that each iteration goes through unit testing, component testing, and integration testing. During the final iterations, the product goes through system integration testing and acceptance testing. Functional testing starts with the test plan and a list of actions to test.

Phase 5.4: Review, at the end of each Sprint, the development team should talk to stakeholders and learn from everything done in the Sprint. The development team should learn to ensure continuous improvement in the next Sprint. The goal is to optimise the process at each iteration. The development team should find potential users and other testers, listen to their opinions, and decide on the new Sprint.

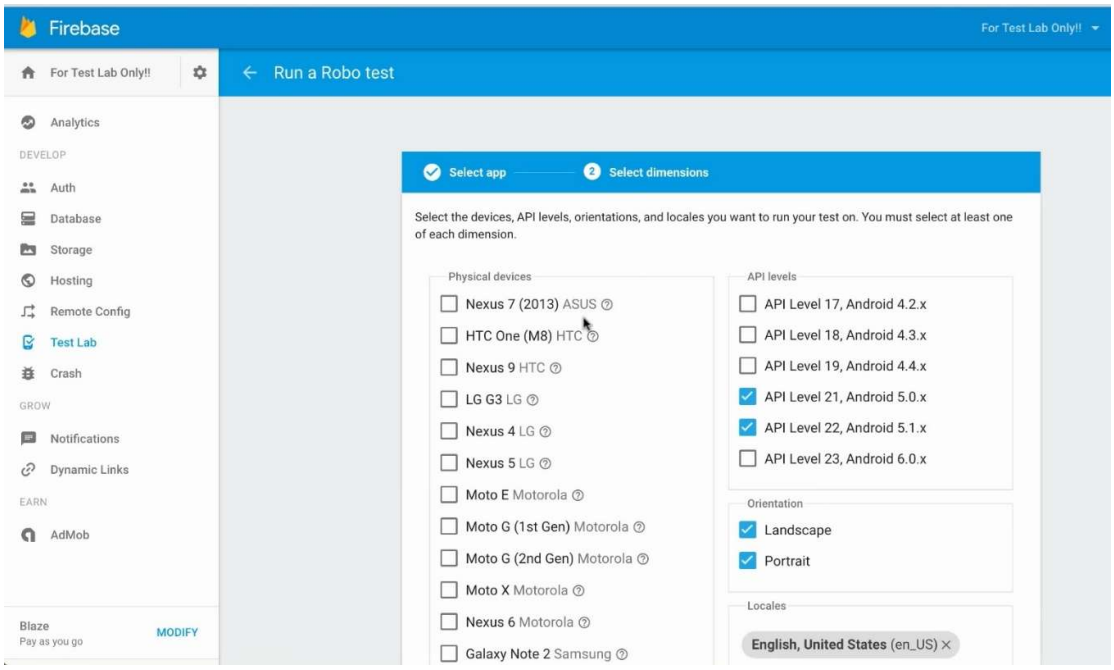


Figure 5. 16: Google Firebase to Run Ropo Test on Multi-Devices

In the Agile Beeswax review phase, the team should examine if the app demo meets expectations, do a detailed review of achieved, and identify any challenges or extra work completed. Furthermore, in this phase, the team should determine how many hours the team spent on the project during the Sprint. The team can run a beta test or make the application available for testing by real users. After the review is complete, it is best to run a final development Sprint to fix any remaining issues.

The outputs of the development phase: After each iteration, the development will have a beta version of the application. At the end of this phase and after several iterations, a fully functional app is achieved.

Phase 6: Deployment and Monitoring

It is time to make the app available to consumers once the development team has finished developing it. The most popular places for publishing the app are Google Play and Apple's App Store. The Deployment and Monitoring phase is an operational practice and consists of two main subphases: Deployment and Monitoring, as shown in Figure 5.17.

As seen in Figure 5.17, this phase starts since the team has a working app and is ready for publishing. This phase is prepared for any future update or fixing any problems during the monitoring subphase when errors will be sent to the development team for review. We will talk about the two subphases in more detail.

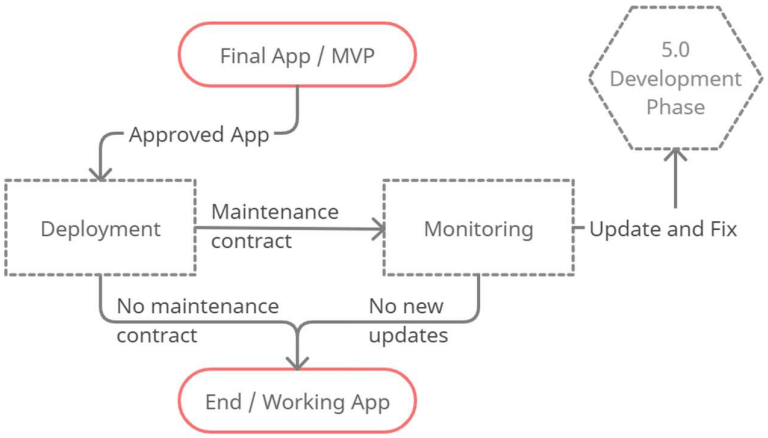


Figure 5. 17: Deployment and Monitoring Phase

Phase 6.1 Deployment, developers must publish the app in the app stores. App stores can be classified according to their accessibility, compatibility, simplicity of use, diversity, and user-generated content (Martin et al., 2017). Users can download apps published by developers in the App Store. Application stores have internal audit and accreditation rules applicable to this deployment process. Although in general, app and app upgrades can be made available quickly, in the case of a hybrid web application a server back-end to host the data is required. The team is necessary to select the server they want to use. In our study, the experts recommend a server with a scalable environment such as Amazon Web Services. Therefore, Agile Beeswax recommends Amazon Web Services and Google-Cloud for API and database hosting. With such a server, the app will not fail if it becomes popular. For the app to be deployed to the Apple Store or Google Play Store, it must meet the requirements of those app stores.

Phase 6.2: Monitoring, after the app's deployment, the app is now available to a huge audience of people. The development team can now monitor the performance of the app in real-time. According to Agile Beeswax, the development team carefully should monitor the app's reviews. Using various tools available, the development team can monitor user behaviours, app ratings, written comments, user acquisition, retention, and usage trends which are all key aspects to consider. Furthermore, specific analytics tools should be used to obtain comprehensive user crash reports. Agile Beeswax recommends Google Analytics, Firebase, Facebook Analytics for app analytics, and MetricFire or ScoutAPM to monitor app performance. Figure 5.18 shows an example of app monitoring using Google analytics.

As shown in Figure 5.18, we can monitor and analyse the app using a screen view tracking system. Developers can track the new users, in which country they are, what devices using, and much more.

Resume of Agile Beeswax phases

A deep analysis from the conducted study allows us to mix some agile techniques and methods based on what we have learned with experts and researchers' recommendations to obtain a solid base to construct a general methodology for app development, which we called Agile Beeswax. Table 5.2 present a summary of the Agile Beeswax phases and its subphases, indicating the roles and tools implied in each one.

In our previous study, we detected several phases to implement mobile apps performed by most of the experts who participated in our study. From this abstraction, we proposed six phases in Agile Beeswax that integrate this expert knowledge: The idea and strategy, user experience design (UX), user interface design (UI), design to development handoff- issues and decisions, development, and deployment and monitoring. Beeswax methodology follows the principles of agile development and of the User-Centered Design; emphasing the significance of collaborating between the customer and the development team. Therefore, Beeswax proposes an iterative cycle process for the development of the mobile application, where reusing code is a essential part to optimize the process. In addition, according to the performed studies, Beeswax suggests using the same team for design and development services in the project and using some of the tools available to manage the handoff. Additionally, the proposed methodology requires a solid process of testing and integrates designers in that process, letting them see whether their vision was developed as they described it. In general terms, the proposed methodology satisfies the characteristics identified as desirable in the performed analysis during our research and summarized in the section 4.2.4. The methodology Agile Beeswax therefore offers a systematized way of approaching the development of a mobile application with greater guarantees of success.

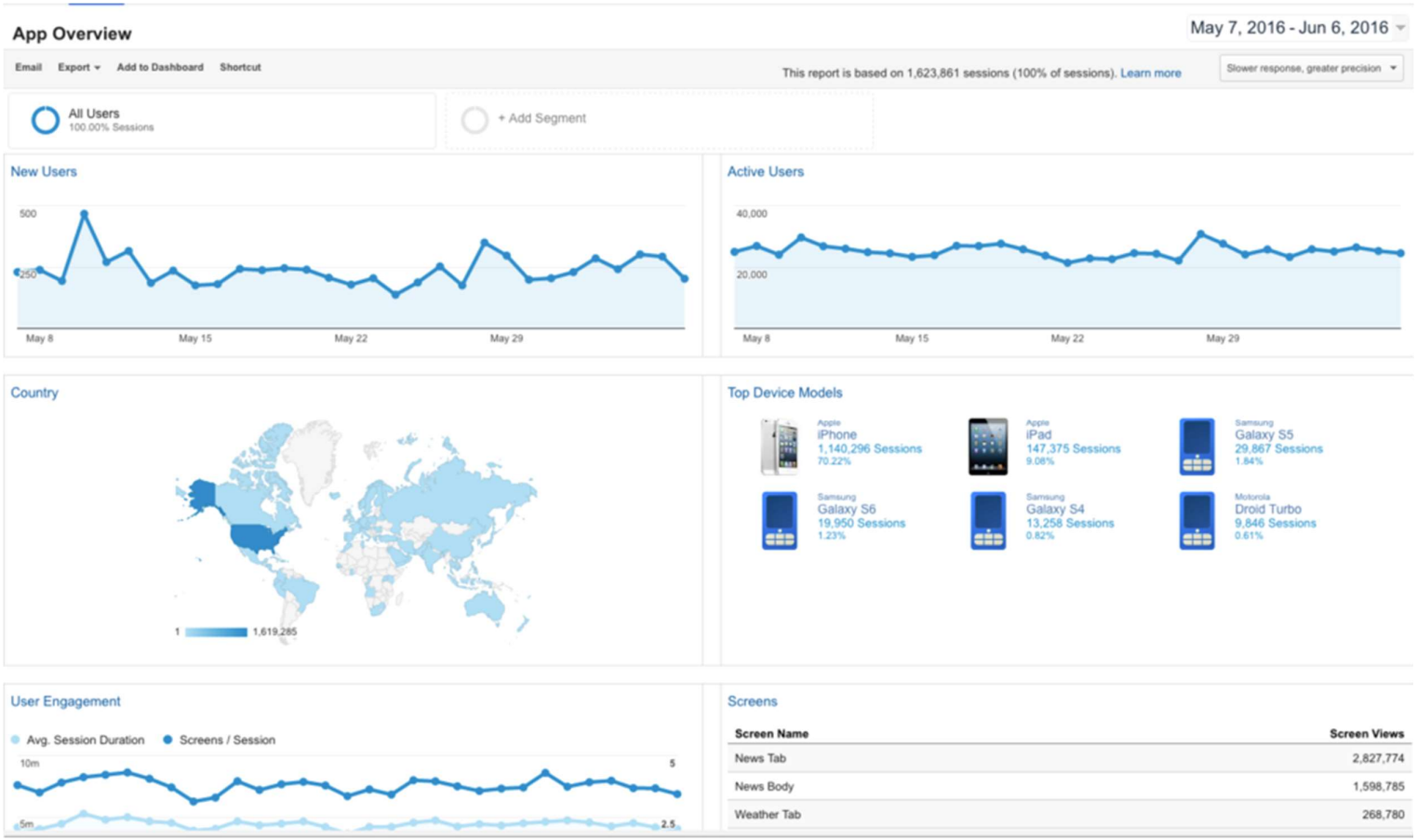


Figure 5. 18: App Monitoring Using Google Analytics

Table 5. 2: Summary of the Agile Beeswax Phases

ID	Phases	Subphases	Team Roles	Tools
1	Idea and strategy	Idea workshop Strategy Marketing campaign	Customer/ Product Owner App Architect Project Manager	Whiteboard Paper and pencil Pages, MS word, Trello
2	User experience design	Wireframe Workflow Testing workflow	PM, PO, App Architect Design Lead /UX Designers	Whiteboard Paper and pencil Sketch or InVision
3	User interface design	UI style guideline Replacement/Mockup Click-through model Test Review	PO, Design Lead /UI Designers	Sketch InVision iOS and Android UI Photoshop
4	Design to development handoff	Customer Approval Technical Design Front-end Back-end Database	PM, PO, UX/UI Designers Development Lead / Developers	Sketch, Photoshop, or Zeplin Google Cloud, Amazon AWS
5	Development	Sprint planning Development Testing Review	PM, PO, Development Lead / Developers Quality Assurance Lead (QA) UX/UI Designers	Native app tools, SQL, Google Cloud, Amazon AWS, Adobe, JavaScript, HTML, Ionic, React, and Ionic, SQL

6	Deployment and Monitoring	Deployment Monitoring	Development Lead / Developers Assurance Lead (QA)	Google Analytics, Firebase, Facebook Analytics, MetricFire or ScoutAPM

5.8 Comparison with the Reviewed Methodologies

Table 5.3 compares our methodology, Agile Beeswax, with other methodologies proposed by other researchers who have made significant advances in understanding relevant factors that influence the mobile development processes, as we covered in chapter 3. The table summarises the main methods proposed for developing mobile applications to perform the comparison. We have selected the most common characteristics used to evaluate each contribution. Consequently, the proposals are collated in terms of: used agile methods, used non-agile methods, its experimental condition during the creation of the proposal (or not), if the methodology is based on the knowledge of expert developers (or not), if it is based on academic knowledge (or not), if it implements a management approach (or not), if it applies a technical engineering approach (or not), if it uses an operational approach (or not), and the structure of the methodology (depending on its weight).

All methodologies included in the table could be used as mobile application development processes. However, selecting which one is more suitable in general terms is problematic since they have different issues in different contexts (academic or industrial). For this reason, our research goal has been to provide a single comprehensive methodology that integrates all characteristics mentioned in Table 5.3 into one methodology.

The proposed methodology is therefore the most complete of all those reviewed. Agile Beeswax is a set of well-structured agile mobile development processes that would be used in different contexts and which was born from an experimental approach (interviews, surveys, etc.) involving academic and industry and including a multiple approach: management, engineering and operational.

Table 5. 3: Comparing the Reviewed Methodologies and Our Methodology (Agile Beeswax)

Mobile Process	**Mobile D**	**RaPiD7**	**Hybrid**	**MASAM**	**Scrum**	**SleSS**	**MADeM**	**Mobile Ilities**	**Agile Besswax**
Year	2004	2005	2008	2008	2010	2011	2016	2020	2021
Agile	XP, Crystal	AM	ASD	XP	Scrum	Scrum, Lean	XP, Crystal	Scrum	Scrum
Non-Agile	RUP	-	NPD	RUP, SPEM	_	Lean Six Sigma	RUP	_	Lean, Kanban
Experimental	Yes	Yes	No	No	Yes	Yes	No	Yes	Yes
Based on Experts	Yes	Yes	No	Yes	No	Yes	No	No	Yes
Based on Academic	No	No	No	No	No	No	No	No	Yes
Management Approach	No	Yes	Yes	No	Yes	Yes	No	Yes	Yes
Technical Engineering Approach	Yes	No	No	Yes	No	No	Yes	No	Yes
Operational Approach	No	No	Yes	Yes	Yes	Yes	No	Yes	Yes
Methodology Structure	Heavy Weight	Heavy Weight	Light-weight	Heavy Weight	Light-weight	Light-weight	Heavy Weight	Lightweight	Lightweight

CHAPTER 6. VALIDATION OF AGILE BEESWAX METHODOLOGY

This chapter shows the analysis and validation of the proposed Agile Beeswax methodology. As we mentioned in chapter 5, the Agile Beeswax's main six components are: Development Phases, Management Practices, Software Engineering Practices, Operational Practices, Agile Beeswax General Rules, and Team Roles. In addition, two properties that are essential in Agile Beeswax are "Adaptive to Changes" and learning during the development process, which we will abbreviate as "Learning". Therefore, we must study these eight variables in the validation process to our presented methodology, Agile Beeswax. With this aim, we asked 35 experts in mobile app development to validate the Agile Beeswax methodology based on these eight variables and measure the extent to which these experts accept this methodology, and we will abbreviate it as “Adoption”.

This section aims to know the significant relationship between this eight-dimensional independent variable and the dependent variable "Adoption" of Agile Beeswax's methodology by experts. The validation of the methodology will be the extent to which participants accept and adopt this methodology. With this purpose, this chapter contains the: Research Methodology, Population and Sample, Validation Research Model and Hypothesis of the Validation Study, Data Analysis Methods, Research Instrument, Instrument Validity, Instrument Reliability, Data Analysis, Descriptive Analysis, and Discussion and Conclusion.

6.1 Research Methodology

This study explores the impact of the eight Agile Beeswax components (independent variable) on adopting Agile Beeswax (dependent variable) in small and medium businesses that develop mobile applications. Thus, a quantitative research method will be adopted to investigate the relationships between independent and dependent variables.

The Technology Acceptance Model (TAM) is an efficient tool to evaluate how people adopt and utilise new technology (Davis, 1986). According to Davis, user acceptance and adoption of new technology are affected by two key variables: Perceived usefulness and Perceived ease of use; Perceived usefulness such as "*Using [the product], would make it*

easier to do my job", and Perceived ease of use such as "*Learning to operate [the product], would be easy for me*". Therefore, our instantiation of the TAM model has used the Agile Beeswax components as an eight-dimension independent variable to assess their impact on adopting the Agile Beeswax methodology in small and medium businesses that develop mobile applications.

Moreover, the survey method is adopted to collect research data which are used in the validation process of the proposed methodology (Appendix B). O'Leary (1990) identified various potential validators, including the same expert who gathered the information from them previously, a different expert than the one who collected the knowledge, the knowledge engineer, or an independent validator. Accordingly, we validate our methodology by experts from whom we collected the information to build our methodology before, and other experts in developing mobile apps and academics and researchers.

6.2 Population and Sample

The study population was a sample selected from small and medium-size apps development companies most of them in Jordan and Spain. We surveyed many mobile software experts, developers, specialists, and researchers in mobile application development. We carefully selected the participants, more than 114 invitations to surveys were sent, and thirty-five responses were received. The participants were apps developers, top management, graphic designers, mobile app development experts, and academic/researcher in the field. More details about the participants will be discussed in section 6.8. Specifically, about 30% of participants were selected from the first survey that built the Agile Beeswax methodology (chapter 4), while 70% were new participants. According to company security issues, all participants were kept nameless.

6.3 The Validation of the Research Methodology and Hypothesis of the Validation Study

The validation methodology of this study search for a relationship between the eight dimensions of the independent variables and the adoption of Agile Beeswax by experts (Table 6.1). Therefore, we developed the main hypothesis (H01) to test this research validation methodology (Figure 6.1) as follow:

H01: There is a positive and significant relationship between the eight dimensions of the independent variable (Phases, Management Practices, Software Engineering Practices, Operational Practices, Agile Beeswax General Rules, Team Roles, Adaptation to Changes,

and Learning) and the dependent variable (Agile Beeswax Adoption). That is, these characteristics affect positively the adoption of the methodology.

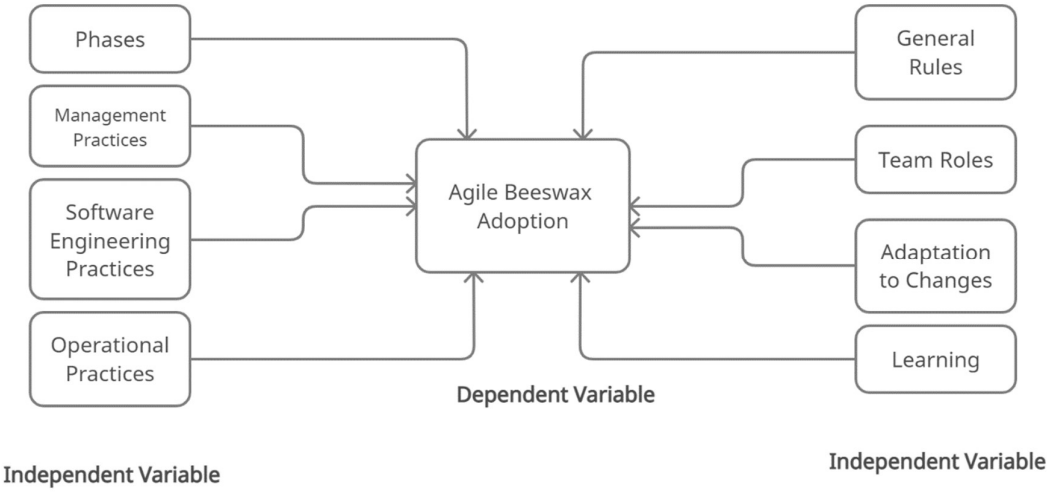


Figure 6. 1: The Research Model and Hypothesis of the Study

Table 6.1 summaries the definition of the study variables. Section 6.3.1 will explain the study variables in more detail.

Table 6. 1: Study Variables

Variable/Dimension	Description
Phases	App development's six phases: idea and strategy, UX, UI, design to development handoff, development, deployment and monitoring.
Management Practices	Agile and Scrum management practices.
Software Engineering Practices	Practices borrowed from XP and software engineering and other approaches to support functional and non-functional capabilities.
Operational Practices	Practices to manage the entire process that guides every step of the mobile app development: "the product" lifecycle from start to end.
General Rules	Rules such as: continuous integration, continuous deployment, continuous feedback in the all-project lifecycle,

	fixed team meetings, and working in sprints.
Team Roles	Team members such as: Project Manager, Scrum Master, Product Owner, Design Lead / Designers, and Development Lead / Developers.
Adaptation to Changes	Requirement management practices such as: reviewing the user and the team's goals, creating blogs, wireframes, workflow, test workflow, and user centre design.
Learning	The importance of learning and improving in each development process to improve the team and the process itself.
Adoption	The extent to which participants accept and adopt the Agile Beeswax methodology.

The Independent Variable

Agile Beeswax's main components: Phases, Management Practices, Software Engineering Practices, Operational Practices, Agile Beeswax General Rules, and Team Roles Adaptation to changes and Learning are presented and described below (for more description see chapter 5).

Phases: Agile Beeswax is a new agile-based methodology for mobile app development. Agile Beeswax is an incremental, iterative development process composed of two main iterative loops (Sprints): the incremental design loop and the incremental development loop. One bridge connects these two Sprints. One of its main strengths is that it has been created with academic and business perspectives to bring these two communities closer. Agile Beeswax is structured in six phases:

1. Idea and strategy: To come up with the best version of the app idea and identify challenges, a roadmap, potential users, legal and security issues, and the app's future.
2. User experience design: The UX wireframing and prototyping are concerned with the app functionality and how it works. UX phase is an iterative, incremental process, starting from the app requirements, app architect, and app goals.
3. User interface design: This phase is concerned with how the app looks.

4. Design to development handoff and technical decisions: The shift from design to development is one of the administrative problems/issues in application development. This phase is concerned with all these issues.

5. Development: Building mobile apps is an iterative process. Scrum and its best practices are adopted in the development phase.

6. Deployment and monitoring: The app is available to consumers once the development team has developed it. This phase takes care of the app deployment and monitoring while the app running.

Agile Beeswax Practices: We believe developing a mobile application needs more than adopting just the management practices (Agile and Scrum). We need to adopt more practices. Agile Beeswax practices intersect Agile and Scrum practices, technical software engineering practices, and operational practices. Agile Beeswax has major and important practices adoption, partial practices adoption, and non-adoption practices (Table 4.1).

Management practices: Agile and Scrum practices are adopted in Agile Beeswax because they have demonstrated be effective methods for developing mobile apps. Scrum's approach is to direct and manage iterative cycles at the project level. With relatively lightweight and time-boxed, Scrum provides fixed meetings and an all-time project overview, details of delays, to-do lists, and a completed tasks list. Agile and Scrum practices include, for example, project management practices, small release, customer involvement and feedback, self-organisation, fixed meetings, short Sprint. (Table 5.1).

Technical software engineering practices: The app development process requires some core engineering and technical practices borrowed from XP and software engineering and other approaches to support functional and non-functional capabilities such as reliability, performance, and security. (Table 4.1).

Operational practices: These practices manage the complete process that guides every step of the mobile app development lifecycle from start to end. Agile Beeswax adopts operational practices from Kanban and Lean. (Table 4.1).

General Rules: These rules define the general philosophy of the methodology and guide the team throughout the process. According to these general rules, in Agile Beeswax, the development team should focus on continuous user/customer feedback in the all-project lifecycle. Furthermore, a fixed meeting is important for the project to succeed. Meeting sessions are not daily, but they must be fixed schedule meetings. Mobile app development team sizes mostly are small size and working on the same site. This gives them the ability to

be in contact when they need it quickly. Another important thing while building the app is to create the app incrementally, block by block, until we reach the final product using Sprint.

Team Roles: The Agile Beeswax team's structure is based on Scrum and agile management approaches. We incorporate some highly specific roles for team members to develop apps quickly and efficiently without losing quality. The mobile app development team structure is always flexible according to the needs of the project and capabilities of the team. The main Beeswax roles are: Project Manager, Scrum Master, Product Owner, Design Lead / Designers, Development Lead / Developers, Analyst or the same Scrum Master or the Product Owner, and Quality Assurance Lead. Adopting agile and Scrum management approaches helps team communication and marketing decisions easier and organise the development process's main operation.

Adaptation to changes: For the ability of requirements changes, Agile Beeswax follows many practices. For example, after reviewing the user and the team's goals and requirements, it will be necessary to start writing and creating blogs containing the data and functions in the mobile app and ways to organise and present them. With just a pencil and paper, the team can start to make the wireframe, which explains what the team or the customers need visually. It is possible to start with a medium-sized mobile screen when wireframing. In the UX phase, there are main sub-phases: wireframe, workflow, and test workflow; this helps define the main requirement for the user.

Additionally, the Agile Beeswax also asks the user to be present in the development process (user centre design). When the team finish the main requirements, visualise them in the wireframe, and are tested. Agile Beeswax recommends asking the customer for confirmation and signature. Additionally, break down the project into small blocks. Thus, the team can control the requirements more, developing it page by page or backlog by backlog. Similarly, the main practices help developers maintain changing requirements, such as working in iterations in the design and development phases. In parallel with this, Agile Beeswax recommends testing the app after each phase, don't wait till the end of the project. Therefore, as designs or developments are completed, users are consulted to identify and integrate possible changes in these products. If these practices are considered, the predominant change in requirements will only occur in the early application development stages, and there is not much to redo or deal with changes in requirements.

Learning during the development process: Agile Beeswax considers the importance of learning and improving in each development process to improve the team and the process itself. Agile Beeswax working on some Scrum practices, Sprints, kickoff

meetings, fixed meetings during the development process, and Sprint review are practices to learn from each Sprint to improve the development process. At the end of each Sprint, the development team must talk with stakeholders to learn from everything they did. The goal is to improve the process in each iteration. They also should try to find testers such as potential users, listen to their feedback, and then make decisions. The development team can launch a beta test or give the app to a real user to test. After the team finishes the reviews, the development team should perform one final development Sprint to fix any remaining problems.

On the other hand, Agile Beeswax believes that keeping up with the latest trends is an integral part of any software specialist's job, and this will animate the team members that they are continually learning and improving. The team should discover the right resources to keep the team with the latest technology, join a software development community, watch or listen to numerous podcasts, and attend conferences and meetings, these are excellent ways to follow recent trends. In addition, Agile Beeswax believes it seems very important to know how applications we create are operated. This gives a significant advantage to learn and improve.

6.4 Research Instrument

A questionnaire was used to collect the research data as one of the most used methods to collect the primary data. The developed questionnaire is divided into two parts; the first part has 10 questions that include demographic and information about participants. The second part consists of 48 independent and dependent variables questions. The dimensions of the independent variable were selected from the proposed Agile Beeswax methodology, and its questions were developed based on related works (Nur & Joviando, 2021). At the same time, the dependent variable questions were selected from the TAM Model, as we will explain later in this section. Therefore, a 5- Likert scale was used to answer each question that helps analyse data using SPSS software, as shown in Table 6.2. The questions for every variable can be found in Appendix B.

Measure the study's variables besides facilitating the participant's process of answering the variables questions as shown in Table 6.3. The Likert scale has three levels based on (5-1) / 3 =1.33.

Table 6. 2: Five-Likert Scale

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
1	2	3	4	5

Table 6. 3: Likert Average Scale

Average	Level of Agreement
1 - 2.33	Low
2.34 - 3.66	Medium
3.67 – 5	High

The questionnaire was prepared online using Google Forms (Appendix B) that started with a cover letter page to clarify the study's purpose. Also, this cover letter explains that all data will be kept in the strictest confidentially. Before every subsection and before asking questions related to this section, there was a description of the corresponding Agile Beeswax components, that help the participants to understand the proposed methodology. Furthermore, we add a link for a full description of our proposed methodology that they can access online.

The first sector consists of the five questions focused on the demographic data, gender, age, working in a company or academics, and specialist. We asked five questions about the organisation, such as whether or not the organisation developed apps, size of the company or organisation, number of apps developed in the organisation, and whether the organisation used agile methods in app development.

The second sector of the questionnaire discussed the independent variable, consisting of the eight dimensions (Phases, Management Practices, Software Engineering Practices, Operational Practices, Agile Beeswax General Rules, Team Roles, Adaptive to Changes, and Learning). Each dimension will be discussed in detailed questions.

Four items were used to measure the adoption of the Agile Beeswax methodology that was derived from the TAM Model. These are the four items we used to measure the adoption.

- I believe that Agile Beeswax will be helpful for my job.
- I believe that I want my company to carry on adopting the Agile Beeswax method.
- I think that the adoption of Agile Beeswax is not needed much effort.

- I believe that Agile Beeswax is easy to adopt.

6.5 Data Analysis Methods

The study has used the SPSS 24 (Statistical Package for Social Sciences) software to analyse and interpret the data gathered through the research study questionnaire. SPSS allows verification and test the integrity and reliability of the questionnaire and then examine the impact of each independent variable on the dependent variable dimensions. Accordingly, the following statistical methods have been used:

Variable instrument test:

1. The Descriptive Statistical Test: this test describes the sample, standard deviation, frequencies, percentage and means .
2. The Cronbach's Alpha test: this test is one of the most popular measures of internal consistency (reliability of the instrument). This technique is mostly used when we have Likert scale questions in such a questionnaire that represent each variable. So, in this thesis Cronbach's Alpha will measure the reliability of all questions that represent each variable.
3. The Variance inflation test (VIF): this test is used to measure the interference between the independent variable dimensions. In other words, this test is to find the similarity between independent variables that represent the Agile Beeswax methodology. That means if there is a similarity between independent variables will lead to a strong correlation between them that will affect the relationship to the dependent variable.

Hypothesis test:

4. The Multiple Linear Regression test: this test is used to explore the relationships between independent variables and dependent variables.
5. F-test and T-test: F-test and T-test are the two statistical methods used to test the main hypothesis and the impact of each independent variable on the dependent variable dimensions. In this study will support the research study to decide whether to accept or reject the hypothesis.

6.6 Instrument Validity:

Before sending the questionnaire to our participants, it has been sent to five academics who are specialists and experts in software engineering. They were professors and associates in different specialists of software engineering area. The researcher took their opinions and comments to enhance the questionnaire.

6.7 Instrument Reliability

The Cronbach's Alpha test was used to study the instrument reliability and internal consistency to ensure the validity of the questionnaire as a tool for collecting data necessary for the current study and that it can measure what we want to measure. The reliability of the tool is intended to obtain the same data when re-studying using the same study tool on the same individuals under similar conditions.

The results of this test were categorised as follow (Sekaran, 2003):

1- Good: Cronbach's Alpha value is greater than (0.80).
2- Acceptance: Cronbach's Alpha value is greater than (0.70) and less than (0.79).
3- Poor: Cronbach's Alpha value is greater than (0.60) and less than (0.69).

The Cronbach's Alpha test results are presented in Table 6.4 for all study variables and its related items (questions), indicating major instrument reliability is good %95.2 (Cronbach's Alpha is over %80) and variables 1 and 6 are acceptable (%72.3 and %71.4 respectively). These results indicate that the questions that were used to measure the study variables have high reliability.

Table 6. 4: Cronbach's Alpha Test Results

No.	Variables	Cronbach's Alpha	Number of Items	Questions
1	Phases	0.723	4	Q 1 to Q 4
2	Management Practices	0.853	10	Q 5 to Q14
3	Software Engineering Practices	0.838	6	Q15 to Q20
4	Operational Practices	0.844	6	Q21 to Q26
5	Agile Beeswax General Rules	0.801	6	Q27 to Q32
6	Team Roles	0.714	3	Q33 to Q35
7	Adaptation to Changes	0.836	6	Q36 to Q41
8	Learning	0.810	3	Q42 to Q44

9	Adoption	0.875	4	Q45 to Q48
	Total	**0.952**	**48**	

6.8 Data Analysis

In this part, the collected data were analysed using Statistical Package for Social software (SPSS 24). The following tests were performed:

- **Descriptive Statistics:** It includes a description of the data and variables of the study such as percentages, frequencies, means, and standard deviation values of the data related to the study variables. The percentage is the repetition of each category of demographic information of the participants divided by the total number of all categories repetition. Frequencies are the repetition of each category of demographic information of the participants**.** Mean is calculated by summation all the data for each paragraph of the questionnaire and then divided them among the number of respondents. Finally, standard deviation shows the variation of each response on the paragraph from the mean.
- **Variance Inflation Factor (VIF):** The VIF test was used to ensure if there is no multicollinearity problem between independent variables. Multicollinearity is the appearance of high intercorrelations between two or more independent variables in a multiple regression model. When a researcher or analyst attempts to evaluate how well each independent variable can be used most effectively to predict or comprehend the dependent variable in a statistical model, a multicollinearity problem can lead to skewed or misleading conclusions.
- **Pearson Correlation Test:** This is a statistical measure of the strength of the association between two variables' relative movements. The values range from -1 to 1. A correlation of -1.0 indicates that there is a perfect negative connection, whereas a correlation of 1.0 indicates that there is a perfect positive correlation. A correlation of 0.0 indicates that there is no linear link between the two variables' movements. This test shows the interrelationship between variables but does not indicate the cause and effect relationship which we need to examine, so the study used the Multiple Regression analysis to examine the statistical effect of independent variables on the dependent variable.
- **Multiple Regression Analysis:** This is a statistical method for examining the relationship between a single dependent variable and multiple independent factors.

The goal of multiple regression analysis is to use known independent variables to predict the value of a single dependent variable.

Each predictor value is weighted to indicate its proportional contribution to the overall forecast.

- **Hypothesis Testing:**
  - The F-test was used to test every single hypothesis. Section 6.9.6 will discuss the F-test in more detail. F-test is a statistics tool used when comparing statistical models fitted to a data set based on the F statistic. The F statistic is defined as the ratio of two independent chi-squares (a value can be measured using a statistic equation) variates divided by their degrees of freedom. Researchers employ the F-test to test the equivalence of two population variances. The F-test is commonly used by researchers to determine whether or not two independent samples were selected from a normal population with the same variability, hence, this test can be used to examine the significance of several regression coefficients at the same time.
  - The T-test was used to test the impact of Agile Beeswax methodology's Components (Phases, Management Practices, Software Engineering Practices, Operational Practices, Agile Beeswax General Rules, Team Roles, Adaptive to Changes, and Learning) on the dependent variable Agile Beeswax Adoption. T-tests of regression are used to determine whether a predictor is significantly linearly connected to the response after adjusting for the effects of all other predictors in the model. Section 6.9.6 will discuss the T-test in more detail.

6.9 Descriptive Analysis

This part uses SPSS (Statistical Package for Social Sciences) to study and understand the participants' answers through the following statistical methods, namely standard deviation and means.

6.9.1 Personal and Demographic Data

This part focuses on the personal and demographic analysis that includes gender, age, and the participants working. The analysis of the demographic data serves to understand the study sample's individuality. Tables 6.5 and 6.6 summarises the primary information about the participants. Figure 6.2 summarises the specialist of the participants.

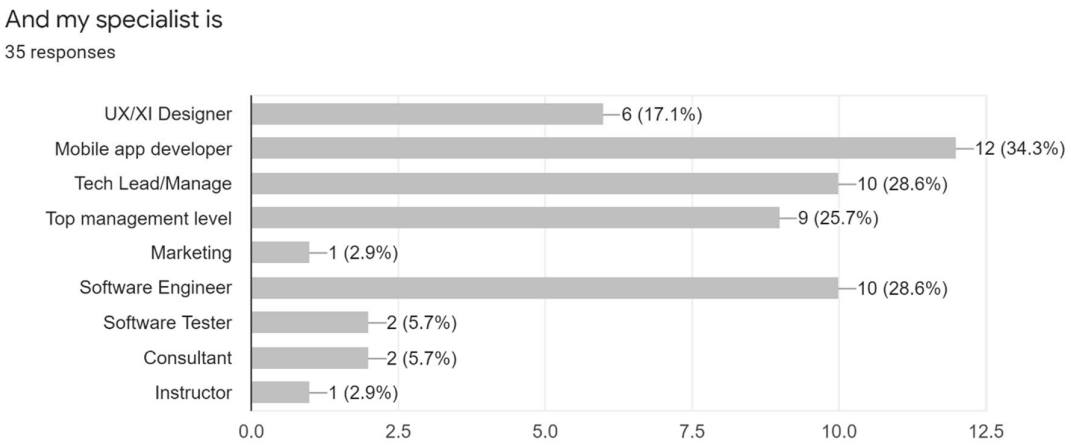


Figure 6. 2: Participants Specialist

Table 6. 5: Demographic Information of the Participants

Statement	**Categories**	**Frequency**	**Percent**
Gender	Male	24	69%
	Female	11	31%
	Total	35	100%
Age	20-34	13	37%
	35-44	15	43%
	45-54	5	14%
	55-64	2	6%
	Total	35	100%
Works in`	Company	27	77%
	Academics	8	23%
	Total	35	100%

Table 6. 6: General information About the Participants

Participants	Description
Gender	Male (69%) Female (31%)
Academic/Worker	Companies' experts (77%) Academic (23%)
Age	Under 45 years old (80%)
Country	From Jordan and Spain
Experience	Develop apps (80%) yes, and (20%) sometimes
Company size	Less or equal five employees (23%) five to twenty employees (77%)
Number of apps developed in the last five years	5 to 10 apps (49%), 11 to 20 (31%)
Using the Agile method in the development process	Yes (89%), Sometimes (11%)

Most participants (80%) were under 45 years. Also, most of the participants were male (69%); 31% were female. Nearly all the participants work in companies (77%), while 23% are academics. This indicates that all the study samples are experts and can assess the impact of the eight components of the independent variable on adopting the Agile Beeswax methodology. About 80% of experts have developed apps; 80% of participants had experience developing five to twenty apps. This shows that workers in executive positions are more able to validate and assess the impact of the independent variables on adopting the Agile Beeswax methodology. Less than 23% of respondents work in small organisations (less than five employees), and about 43% work in medium-sized organisations (five to ten employees). Most respondents (89%) use agile methods to create mobile applications. Therefore, this sample is valid for a methodology validation experiment.

6.9.2 Independent Variable: Agile Beeswax Methodology Components

This section presents the descriptive statistics for the eight-dimensional independent variable: Phases, Management Practices, Software Engineering Practices, Operational Practices, Agile Beeswax General Rules, Team Roles, Adaptive to Changes, and Learning.

The means and standard deviations were extracted to identify the responses of the study sample members to the interrelationships and quality of information. Standard

deviation is a calculation of how the respondents' data are distributed over the mean. A low standard deviation indicates that all responses were closed to the mean, while a high standard deviation led to a high spread out to the mean. The value of standard deviation close to zero indicates that data are closed to the mean value. The following is a presentation of the results of the descriptive analysis of the study variables based on the extraction of means and standard deviations.

Table 6.7 shows the mean scores, standard deviations, mean rank, and level of agreement of the Phases variable. The mean score for the Phases variable was relatively high. For individual items, Q2 and Q3 (Q2*: I believe that mobile application development phases presented at Agile Beeswax are easy to understand,* Q3: *I believe that none of the mentioned application development phases can be eliminated or waived if we want a high-quality mobile application*) gained the highest mean score with 4.26 for each and Standard Deviations 0.852, 0.741, respectively. These results (items from Q1 to Q4) showed that experts' satisfaction with the Phases variable is high, as they all indicate high relative importance for all the questions belonging to this section (listed below).

- Q1: I believe that the Agile Beeswax phases cover all the steps and phases of mobile application development.
- Q2: I believe that mobile application development phases presented at Agile Beeswax are easy to understand.
- Q3: I believe that none of the mentioned application development phases can be eliminated or waived if we want a high-quality mobile application.
- Q4: I believe that the order and sequence of phases correspond to the mobile application development methods.

Table 6. 7: Phases Descriptive Statistics

Statement	Mean	Std. Deviation	Rank	Level of Agreement
Q1	4.11	0.867	4	Hight
Q2	4.26	0.852	1	Hight
Q3	4.26	0.741	2	Hight
Q4	4.14	0.912	3	Hight

Table 6.8 shows the mean scores, standard deviations, mean rank, and level of agreement of the Management Practices variable. The mean score for the Management Practices variable was relatively high. For individual items, Q12 (*I believe that to develop a mobile application, Agile and Scrum Management practices are required, with Delivery as-fast-as-possible practice*) gained the highest mean score with 4.51 and a standard deviation of 0.612. These results (items from Q5 to Q14) showed that experts see that their satisfaction with the Management Practices variable is high, as they all indicate high relative importance for all the questions belonging to this section (listed below).

I believe that to develop a mobile application, Agile and Scrum Management practices are required, in particular…:

- Q5: Working in short iterations in the design and development phases.
- Q6: Fixed meetings.
- Q7: Sprint review.
- Q8: Product Backlog.
- Q9: Small releases.
- Q10: Incremental design and development.
- Q11: User centre design.
- Q12: Delivery as-fast-as-possible.
- Q13: Team empowerment.
- Q14: Continuous integration.

Table 6. 8: Management Practices Descriptive Statistics

Statement	Mean	Std. Deviation	Rank	Level of Agreement
Q5	4.40	0.914	3	Hight
Q6	4.37	0.877	4	Hight
Q7	4.37	0.690	5	Hight
Q8	4.26	0.919	8	Hight
Q9	4.43	0.655	2	Hight
Q10	4.34	0.684	7	Hight
Q11	4.37	0.598	6	Hight
Q12	4.51	0.612	1	Hight

Q13	4.11	0.583	10	Hight
Q14	4.14	0.772	9	Hight

Table 6.9 shows the Software Engineering Practices variable's mean scores, standard deviations, mean rank, and level of agreement. The mean score for the Software Engineering Practices variable was relatively high. For individual items, Q16 (*I believe that to develop a mobile application, Technical and Software Engineering practices are required, with Design Improvement practices*) gained the highest mean score with 4.29 and a standard deviation of 0.710. These results (items from Q15 to Q20) showed that experts see that their satisfaction with the Software Engineering Practices variable is high, as they all indicate high relative importance. For all the questions belonging to this section, we can list them as follows:

I believe that to develop a mobile application, Technical and Software Engineering practices are required, in particular…:

- Q15: Test-driven development.
- Q16: Design improvement.
- Q17: Coding standards.
- Q18: Control of design to development handoff.
- Q19: Continuous testing.
- Q20: Regression testing.

Table 6. 9: Software Engineering Practices Descriptive Statistics

Statement	Mean	Std. Deviation	Rank	Level of Agreement
Q15	4.09	0.702	4	Hight
Q16	4.29	0.710	1	Hight
Q17	4.23	0.547	2	Hight
Q18	3.89	0.900	5	Hight
Q19	3.86	0.772	6	Hight
Q20	4.09	0.742	3	Hight

Table 6.10 shows the Operational Practices variable's mean scores, standard deviations, mean rank, and level of agreement. The mean score for the Operational Practices variable was relatively high. For individual items, Q25 (*I believe that to develop a mobile*

application, Operational Practices are required, with accept changes in iteration at any time) gained the highest mean score with 4.37 and a standard deviation of 0.730. These results (items from Q21 to Q26) showed that experts' satisfaction with the Operational Practices variable is high, as they all indicate high relative importance, for all the questions belonging to this section /listed below).

I believe that to develop a mobile application Operational Practices are required, in particular…:

- Q21: Simple design.
- Q22: Eliminate waste.
- Q23: Amplification of learning.
- Q24: Continuous learning and improvement.
- Q25: Accept changes in iteration at any time.
- Q26: Planning for unplanned work.

Table 6. 10: Operational Practices Descriptive Statistics

Statement	Mean	Std. Deviation	Rank	Level of Agreement
Q21	4.03	0.857	6	Hight
Q22	4.14	0.879	5	Hight
Q23	4.20	0.797	4	Hight
Q24	4.26	0.852	3	Hight
Q25	4.37	0.731	1	Hight
Q26	4.34	0.838	2	Hight

Table 6.11 shows the mean scores, standard deviations, mean rank, and level of agreement of the Agile Beeswax General Rules variable. The mean score for the Agile Beeswax General Rules variable was relatively high. For individual items, Q27 (*I believe that the first design iteration must satisfy basic user needs*) gained the highest mean score with 4.63 and a standard deviation of 0.690. These results (items from Q27 to Q32) showed that experts' satisfaction with the Agile Beeswax General Rules variable is high, as they all indicate high relative importance for all questions in this section (listed below).

- Q27: I believe that the first design iteration must satisfy basic user needs.

- Q28: I believe that once the design Sprint has been completed and confirmed, it should be sent to the development phase (crossing the bridge, design to development handoff and technical decisions).
- Q29: I believe that the Product Owner must split the requirements into smaller user stories for each application screen in discussion with customers and other stakeholders.
- Q30: I believe that after writing the user stories (by the Product Owner) as a Scrum Product Backlog, a prioritization session with the architect and some developers must be initiated.
- Q31: I believe that testing should be performed regularly, as it will significantly reduce the financial costs incurred at each stage
- Q32: I believe that Agile Beeswax certainly favors the rapid development of mobile applications.

Table 6. 11: Agile Beeswax General Rules Descriptive Statistics

Statement	Mean	Std. Deviation	Rank	Level of Agreement
Q27	4.63	0.690	1	Hight
Q28	4.31	0.631	5	Hight
Q29	4.57	0.655	2	Hight
Q30	4.26	1.010	6	Hight
Q31	4.57	0.655	3	Hight
Q32	4.40	0.604	4	Hight

Table 6.12 shows the mean scores, standard deviations, mean rank, and level of agreement of the Team Roles variable (in the sense of how these roles make it easier for the team to understand each other and make good decisions). The mean score for the Team Roles variable was relatively high. For individual items, Q35 (*I believe that with Agile Beeswax, we will have good communication between project members*) gained the highest mean score with 4.60 and a standard deviation of 0.604. These results (items from Q33 to Q35) showed that experts see that their satisfaction with the Team Roles variable is high, as they all indicate high relative importance, for all the questions belonging to this section (listed below).

- Q33: I believe that with the using of Agile Beeswax, you will have good communication between project members.
- Q34: I believe that with the using of Agile Beeswax will improve team decisions.
- Q35: I believe that with the using of Agile Beeswax will enhance team organising.

Table 6. 12: Team Roles Descriptive Statistics

Statement	Mean	Std. Deviation	Rank	Level of Agreement
Q33	4.09	0.919	3	Hight
Q34	4.23	0.843	2	Hight
Q35	4.60	0.604	1	Hight

Table 6.13 shows the Adaptation to Changes variable's mean scores, standard deviations, mean rank, and level of agreement. The mean score for the Adaptation to Changes variable was relatively high. For individual items, Q41 (*I believe that with Agile Beeswax, developers are not very concerned about changing customer needs*) gained the highest mean score with 4.66 and a standard deviation of 0.684. These results (items from Q36 to Q41) showed that experts see that their satisfaction with the Adaptation to Changes variable is high, as they all indicate high relative importance for all the questions belonging to this section (listed below).

- Q36: I believe that using wireframing design in the early stages will help in understanding the application's requirements.
- Q37: I believe that designing the application workflow in the early stages will help in understanding the application's requirements.
- Q38: I believe that with Agile Beeswax, it will be easier to deal with changing requirements.
- Q39: I believe that dividing the app project to Sprint backlog (divide a task into smaller ones) and prioritizing it, will help manage core app requirements.
- Q40: I believe that with Agile Beeswax, most changes in requirements occur in the early stages of the development process.

- Q41: I believe that with Agile Beeswax, developers are not very concerned about changing customer needs.

Table 6. 13: Adaptation to Changes Descriptive Statistics

Statement	Mean	Std. Deviation	Rank	Level of Agreement
Q36	4.37	0.547	6	Hight
Q37	4.43	0.558	5	Hight
Q38	4.60	0.497	2	Hight
Q39	4.43	0.698	4	Hight
Q40	4.57	0.558	3	Hight
Q41	4.66	0.684	1	Hight

Table 6.14 shows the mean scores, standard deviations, mean rank, and level of agreement of the Learning variable. The mean score for the Learning variable was relatively high. For individual items, Q44 (*Using Agile Beeswax will motivate me during the development process*) gained the highest mean score with 4.66 and a standard deviation of 0.639. These results (items from Q42 to Q 44) showed that experts see that their satisfaction with the Learning variable is high, as they all indicate high relative importance for all the questions belonging to this section (listed below):

- Q42: I believe that it is important to learn from each Sprint in the development process for the next Sprints or another project. We can discuss that in the Sprint Review.
- Q43: Agile Beeswax believes that keeping up with the latest trends is an integral part of any software specialist's job.
- Q44: Using Agile Beeswax will motivate me during the development process.

Table 6. 14: Learning Descriptive Statistics

Statement	Mean	Std. Deviation	Rank	Level of Agreement
Q42	4.51	0.818	2	Hight
Q43	4.37	0.910	3	Hight
Q44	4.66	0.639	1	Hight

Finally, Table 6.15 summarises the independent variables' mean scores, scores, standard deviations, mean rank, and level of agreement. We can see that the independent variable's dimensions 8 and 7 (Learning and Adaptation to Changes) have the highest mean with 4.51 and a standard deviation of 0.678 and 0.441, as all variables indicate high relative importance.

Table 6. 15: Mean Scores and Standard Deviations for all Variables

Variable ID	Variables	Total Mean	Rank	Standard Deviation
1	Phases	4.19	7	0.625
2	Management Practices	4.33	4	0.486
3	Software Engineering Practices	4.07	8	0.547
4	Operational Practices	4.22	6	0.620
5	Agile Beeswax General Rules	4.46	3	0.510
6	Team Roles	4.30	5	0.638
7	Adaptation to Changes	4.51	2	0.441
8	Learning	4.51	1	0.678
DEP	Adoption	4.17		0.675

6.9.3 The Dependent Variable: Adoption of Agile Beeswax Methodology

After presenting all the results of the eight independent of the independent variable (sometimes referred to as independent variables for simplicity), in this section, we will discuss the dependent variable (Adoption) in more detail.

Table 6.16 represents the descriptive statistics of the Agile Beeswax methodology Adoption.

Table 6. 16: Agile Beeswax Methodology Adoption Descriptive Statistics

Statement	Mean	Std. Deviation	Rank	Level of Agreement
Q45	4.40	0.812	1	Hight
Q46	4.17	0.954	2	Hight
Q47	4.03	0.664	4	Hight

Q48	4.09	0.702	3	Hight

Table 6.16 shows the mean scores, standard deviations, mean rank, and level of agreement of the Adoption. The mean score for the Adoption variable was relatively high. For individual items, Q45 (*I believe that Agile Beeswax will be helpful for my job*) gained the highest mean score with 4.40 and a standard deviation of 0.812. These results (items from Q45 to Q48) showed that experts are highly satisfied with the Agile Beeswax Adoption variable. For all the questions belonging to this section, we can list them as follows:

- Q45: I believe that Agile Beeswax will be helpful for your job.
- Q46: I believe that I would like my company to carry on adopting the Agile Beeswax methodology.
- Q47: I believe that the adoption of Agile Beeswax is not needed many efforts.
- Q48: I believe that Agile beeswax is easy to adopt.

In addition, the dependent variable (Adoption) have a high mean value of 4.17 and a standard deviation of 0.675. This indicates a high relative acceptance.

6.9.4 The Variance Inflation Test (VIF)

The VIF was used to ensure non-interference (not overlap) between independent variables before further analysis to study the hypothesis. In this test, we can be sure that, as an example, the Phases variable is fully independent of another variable. This assesses the strength of correlation and connection between the variables in a model. A high VIF shows that the linked independent variable has a high degree of collinearity with the model's other variables. VIF has been used to test if the independent variables are truly independent of each other. The variable must pass this test in order to ensure that it can be included in the test process at this study model. Another term associated with the VIF test is Tolerance. Tolerance is the reciprocal of VIF (1/VIF). If the VIF is less than ten or the Tolerance is less than one (1.0) and more than (0.01) (Hair et al. 2009), this indicates a non-interference between the independent variables. If a variable meets the tolerance conditions test, it is considered suitable for inclusion using the current model. According to Table 6.17, the Tolerance test for all dimensions of the independent variable was ranged between 0.313 and 0.653. Therefore, all results were accepted because it was less than one (1.0) and more than (0.01). Moreover, the VIF results for independent variables were less than ten. Hence, these

results indicate a non-interference between the components of Beeswax considered during the study of adoption.

Table 6. 17: The Variance Inflation Test (VIF)

Independent ID	Independent Variable	VIF	Tolerance 1/VIF
IND1	Phases	1.928	0.519
IND2	Management Practices	1.986	0.504
IND3	Software Engineering Practices	3.196	0.313 lower
IND4	Operational Practices	3.162	0.316
IND5	Agile Beeswax General Rules	2.704	0.370
IND6	Team Roles	1.801	0.555
IND7	Adaptation to Changes	1.532	0.653 higher
IND8	Learning	2.850	0.351

6.9.5 Pearson Correlation Test

The Pearson Correlation test was used to examine the appearance intercorrelations between the eight dimensions of the independent variable and applied to ensure no high correlation between them, which justifies their independent existence. Therefore, this test enhances the degree of certainty of the independence of the variables and their non-interference with each other, and thus the validity of the data for performing Regression Analyses. Referring to Table 6.18, we note that the degree of correlation of each variable with the other variables of the study is less than 70%. Since 0.7 is the maximum permissible limit of correlation between two independent variables (Hair et al. 2009), Therefore, the results presented in Table 6.18 confirm the independence of the study dimensions from each other and reinforce the previous independence tests shown in Table 6.17 before. Accordingly, and depending on the results of the data validity tests, regression analyses can now be used to test the study hypothesis.

Table 6. 18: Pearson Correlation Test

Ind. Num.	Independent Name	IND1	IND2	IND3	IND4	IND5	IND6	IND7	IND8
IND1	Phases	1							
IND2	Management Practices	.272	1						
IND3	Software Engineering Practices	.349	.671	1					
IND4	Operational Practices	.515	.547	.657	1				
IND5	Agile Beeswax General Rules	.569	.444	.418	.654	1			
IND6	Team Roles	.242	.350	.492	.331	.493	1		
IND7	Adaptation to Changes	.327	.337	.390	.312	.380	.465	1	
IND8	Learning	.558	.494	.664	.663	.387	.269	.442	1

The Pearson correlation coefficient, r, is a number that ranges from +1 to -1. If the value is 0, this means there is no relationship between these two variables. If the value is bigger than 0, this is a positive relationship; that is, when one variable's value rises, the other variable's value rises as well. If the value is less than 0, its negative relationship; that is, When one variable's value increases, the value of the other variable decreases.

6.9.6 Regression Analysis and Hypothesis Testing

The Multi Regression test was used to test the hypothesis. Since we have a multidimensional independent variable, we do a multi regression test (not a simple regression). F- test was used to determine the impact of the eight dimensions of the independent variable as a whole on the dependent variable to ensure that the study model is

appropriate to use. The T-test was used to test the impact of Agile Beeswax methodology components separately (Phases, Management Practices, Software Engineering Practices, Operational Practices, Agile Beeswax General Rules, Team Roles, Adaptive to Changes, and Learning) on the dependent variable Agile Beeswax Adoption to determine the single impact of each of the dimensions of the independent variable effect on the dependent variable.

The main output of Multi Regression is the F value (>0), T value (<4), R square(<1), and Sig. α (significant). Significant (Sig.) or Alpha (α) is the notion that a result from data acquired by testing or experimentation is not likely to occur randomly or by chance, but is instead likely to be traceable to a certain cause is referred to as statistical significance.

Statistical significance is essential for academic fields or practitioners who rely largely on data analysis and research. Typically, studies that employ surveys as data collection tools reject the null hypothesis at the level of significance ($\alpha \leq 0.05$), implying that the level of confidence in the results is 95% or greater; the lower the threshold of significance, the higher the level of confidence. Thus, we can re-define our hyphotesis as follows:

R square value is used to measure when those the independent variables are taken together as a set or as a group, do they predict and affect the Adoption (dependent variable) which indicates fitness of the model used to reach the study purposes. As for the value of Sig. α, to measure when those the independent variables are taken separately if they do predict and affect the Adoption (dependent variable).

H01: There is a positive and significant relationship at (Sig. $\alpha \leq 0.05$) between the independent variable (Phases, Management Practices, Software Engineering Practices, Operational Practices, Agile Beeswax General Rules, Team Roles, Adaptation to Changes, and Learning) and dependent variable (Agile Beeswax Adoption).

Multiple regression analysis was used to test the main hypothesis, as we mentioned above, and Table 6.19 shows the obtained results.

Table 6. 19: Multiple Regression Analysis

Methodology components	**T value T-test**	**Sig. α**
Phases	2.956	.007
Management Practices	2.206	.036
Software Engineering Practices	2.255	.033

Operational Practices		2.709	.012
Agile Beeswax General Rules		1.337	.193
Team Roles		-2.466	.021
Adaptation to Changes		-1.577	.127
Learning		3.360	.002
R Square	.918		
F (F-test)	36.190		
Sig.	.000		

The value of F in the F-Test is 36.190, and its significance is .000, which is less than ($\alpha \leq 0.05$). This means that the main hypothesis was accepted. In other words, the group of independent variables (Phases, Management Practices, Software Engineering Practices, Operational Practices, Agile Beeswax General Rules, Team Roles, Adaptation to Changes, and Learning) shows a statically significant relationship with the dependent variable (Agile Beeswax adoption).

The R Square for the main hypothesis is equals (0.918), which means that about 92% of Agile Beeswax components explain its impact on the Agile Beeswax adoption in small to medium-sized organisations that develop apps in Jordan and Spain. In other words, all the independent variables account for 92% of the variance in Adoption. Furthermore, about 8% refers to other factors affecting the Agile Beeswax Adoption. Thus, the set of components selected to validate the model covers almost all the issues to be taken into account in order to adopt the methodology, which validates and strengthens the evaluation experience conducted and its positive results.

The T-test was used to test each dimension of the independent variable and its impact (significant) on Agile Beeswax Adoption (dependent variable).). A positive value of T implies a positive statistically significant between the independent variable and the dependent variable, while a negative value of T implies a negative statistically significant between both variables.

Table 6.19 illustrates the following:

1. It is clear from Table 6.19, and by following the values of the T-test, that the relationship represented by the Phases have a positive statistically significant effect on the adoption of the Agile Beeswax methodology, as the calculated T-value amounted

to 2.965, which are significant values at a significance level of **Sig.** α = 0.007 (α≤0.05). Accordingly, there is a statistically significant impact of the Phases of the methodology, and the Adoption of this methodology at the level of significance is α≤ 0.05.

2. By following the T-test values (Table 6.19), the relationship represented by the Management Practices has a positive statistically significant effect on the adoption of the Agile Beeswax methodology, as the calculated T-value amounted to 2.206, which are significant values at a significance level of **Sig**. α = 0.036 (α≤0.05). Accordingly, it rejects the null hypothesis and accepts the alternative that there is a statistically significant impact of the Management Practices of the methodology, and the adoption of this methodology at the level of significance is α≤ 0.05.

3. By following the T-test values (Table 6.19), the relationship represented by the Software Engineering Practices has a positive statistically significant effect on the adoption of the Agile Beeswax methodology, as the calculated T-value amounted to 2.255, which are significant values at a significance level of **Sig.** α = 0.033 (α≤0.05). Accordingly, there is a statistically significant impact of the Software Engineering Practices of the methodology, and the adoption of this methodology at the level of significance is α≤ 0.05.

4. By following the T-test values (Table 6.19), the relationships represented by the Operational Practices have a positive statistically significant effect on the adoption of the Agile Beeswax methodology, as the calculated T-value amounted to 2.709, which are significant values at a significance level of **Sig.** α = 0.012 (α≤0.05). Accordingly, there is a statistically significant impact of the Operational Practices of the methodology, and the adoption of this methodology at the level of significance is α≤ 0.05.

5. By following the T-test values (Table 6.19), the relationship represented by the Agile Beeswax General Rules has no statistically significant effect on the adoption of the Agile Beeswax methodology, as the calculated T-value amounted to 1.337, which are no significant values at a significance level of **Sig**. α = 0.193 (α ≥0.05). Accordingly, there is no statistically significant impact of the Agile Beeswax General Rules of the methodology, and the adoption of this methodology at the level of significance is α ≥ 0.05.

6. By following the T-test values (Table 6.19), the relationship represented by the Team Roles has a negative statistically significant effect on the adoption of the Agile

Beeswax methodology, as the calculated T-value amounted to -2.466, which is a significant value at a significance level of **Sig.** $\alpha = 0.021$ ($\alpha \leq 0.05$). A negative T-value indicates that the direction of the effect is reversed. Accordingly, there is a negative statistically significant impact on the Team Roles of the methodology, and the adoption of this methodology at the level of significance is $\alpha \leq 0.05$.

7. By following the T-test values (Table 6.19), the relationship represented by the Adaptation to Changes has no statistically significant effect on the adoption of the Agile Beeswax methodology, as the calculated T-value amounted to -1.577, which is no significant value at a significance level of **Sig.** $\alpha = 0.127$ ($\alpha \geq 0.05$). Accordingly, there is no statistically significant impact of the Adaptation to Changes of the methodology, and the adoption of this methodology at the level of significance is $\alpha \geq 0.05$.

8. By following the T-test values (Table 6.19), the relationships represented by the Learning have a positive statistically significant effect on adopting the Agile Beeswax methodology, as the calculated T-value amounted to 3.360, which are significant values at a significance level of **Sig.** $\alpha = 0.002$ ($\alpha \leq 0.05$). Accordingly, there is a statistically significant impact of the Learning of the methodology, and the adoption of this methodology at the level of significance is $\alpha \leq 0.05$.

6.10 Discussion and Conclusion

The main objective of this thesis is to build a methodology that helps in developing mobile applications. Both qualitative and quantitative methods have been used to achieve this goal based on the knowledge of mobile application experts and researchers. The Agile Beeswax methodology presents a framework that outlines processes, tools, activities, and resources that may support mobile app design, development, and deployment.

This proposal, like other any process, tool, or technology, needs to measure its utility. Hence, we defined Agile Beeswax, we studied the components of the methodology and its impact on the level of its adoption among experts and academics. Accordingly, a Technology Acceptance Model (TAM) was used as an essential part of evaluating Agile Beeswax adoption. Therefore, the study's most important findings can be indicated based on the results of the questionnaire and the statistical analysis:

- Most participants (80%) were under 44 years, and most of the participants (77%) work in companies; 23% were academics. This explains that participants are experts and can assess the impact of the dimensions of the independent variable on adopting the

Agile Beeswax methodology. Most of the participants are mobile application developers and in the highest level of software management (Figure 6.1). The experience of these team members is important to evaluate the proposed methodology. In addition, the study participants were anonymous which enable them to fill out the study questionnaire with complete impartiality and objectivity and assess the impact of the eight dimensions of the independent variable on adopting the Agile Beeswax methodology.

- The size of the organisation in relation to the mobile application development project ranged from 5 to 20 members (about 91%). This indicates that the team size is small to medium, which is suitable for organisations that develop mobile applications. This is in line with what Agile Beeswax adopted.
- Based on the testing results, it appears that Learning and Phases components (.002 and .007 respectively) are the most significant adoption factors for the Agile Beeswax methodology.
- The mean score for the Phases variable was relatively high, gaining a score of 4.19 from the point of view of the study participants. Furthermore, there is a statistically significant impact of the Phases of the methodology, and the adoption of this methodology at the level of significance is $\alpha \leq 0.05$. Expert agrees with us that Phases cover all the steps and tasks of mobile application development (69.4% Strongly agree). In addition, the Phases presented at Agile Beeswax are easy to understand (67% Strongly agree) (Appendix B). Furthermore, none of the mentioned application development phases can be eliminated or waived to develop a high-quality mobile application (56% Strongly agree, 42% Agree). Furthermore, the experts agree with the sequence of phases corresponding to the mobile application development methods (81% Strongly Agree) (Appendix B). Thus, we adopt the Phases as an essential part of the components of the Agile Beeswax, which is fundamental for the success to develop a high-quality mobile application. Agile Beeswax phases contain: idea and strategy, user experience design, user interface design, design to development handoff and technical decisions, development (sprint planning, development, testing, review), and deployment and monitoring.
- Most participants (93%) believe that mixing the main practices from management (Agile and Scrum), technical and software engineering practices, and operational practices makes handling the main apps development process more powerful. Thus, we adopt the usage of main practices from management (Agile and Scrum), technical and

software engineering practices, and operational practices as an essential part of the components of the Agile Beeswax, which is an key piece for the success and development of a high-quality mobile application.

- Results showed the mean score for the Management Practices variable was relatively high, gained score with 4.33 from the point of view of the study sample members. Furthermore, there is a statistically significant impact of the Management Practices of the methodology, and the adoption of this methodology at the level of significance is $\alpha \leq 0.05$. This confirms the importance of Management Practices in the development of mobile applications. Therefore, we adopt the usage of Management Practices as an essential part of the components of the Agile Beeswax, which is key piece in the success and development of a high-quality mobile application.
- The mean score for the Software Engineering Practices variable was relatively high, gained score with 4.07 from the point of view of the study sample members. Furthermore, there is a statistically significant impact of the Software Engineering Practices of the methodology, and the adoption of this methodology at the level of significance is $\alpha \leq 0.05$. Therefore, we adopt the usage of Software Engineering Practices as an essential part of the components of the Agile Beeswax, which is a critical factor in the success and development of a high-quality mobile application.
- The mean score for the Operational Practices variable was relatively high, gained score with 4.22 from the point of view of the study sample members. However, there is no statistically significant impact of the Agile Beeswax General Rules of the methodology, since the adoption of this methodology at the level of significance is $\alpha \geq 0.05$ ($\alpha = .193$). Therefore, we adopt the usage of Agile Beeswax General Rules because they have been very good valuated by the experts and provide a framework for teams to work within the methodology, but these rules are not identified as an essential part of the components of the Agile Beeswax (at least until further experiments are conducted that indicate otherwise).
- The mean score for the Agile Beeswax General Rules variable was relatively high, gaining a score of 4.46 from the point of view of the study sample members. Furthermore, there is no statistically significant impact of the Agile Beeswax General Rules of the methodology, and the adoption of this methodology at the level of significance is $\alpha \geq 0.05$ ($\alpha = .193$). Therefore, we adopt the usage of Agile Beeswax

General Rules as an essential part of the components of the Agile Beeswax, which is a critical part in the success and development of a high-quality mobile application.

- The mean score for the Team Roles variable was relatively high, gaining a score of 4.30 from the point of view of the study sample members. However, there is a negative statistically significant impact on the Team Roles of the methodology, and the adoption of this methodology at the level of significance is $\alpha \leq 0.05$ (α=.021). The negative impact means when there are more Team Roles, we have less Adoption of the Agile Beeswax by experts. Thus, we recommend that companies adopt this component with the submitted Team Roles (job title) and have the option of combining job titles into a single job title or having one person adopt more than one job title. The same person can play two roles in some teams. One person can fill the Scrum Master and Project manager as an example. Or even a Scrum Master or Product Owner can fill the role of an analyst. In short, the methodology must make the adoption of team roles more flexible in order to adapt to the human resources available to each company.
- The mean score for the Adaptation to Changes variable was relatively high, gaining a score of 4.51 from the point of view of the study sample members. However, there is no statistically significant impact of the Adaptation to Changes of the methodology, and the adoption of this methodology at the level of significance is $\alpha \geq 0.05$ (α = .127). In line with the high valuation received from the experts, we continue to emphasize the need for the methodology to be flexible in the face of changing requirements, but we do not consider this component in isolation as an essential element of the methodology (but rather as an element that is integrated in each phase of the methodology).
- The mean score for the Learning variable was relatively high, gaining a score of 4.51 from the point of view of the study sample members. Furthermore, there is a statistically significant impact of the Learning of the methodology, and the adoption of this
- methodology at the level of significance is $\alpha \leq 0.05$. Therefore, we adopt Learning as an essential part of the components of the Agile Beeswax, which is a necessary part of the success and development of a high-quality mobile application.

Summary

The Agile Beeswax methodology presents a framework that outlines processes, tools, activities, and resources that may support mobile app design, development, and deployment. Based on experts' opinions, Agile Beeswax is flexible concerning mobile app development, starting from idea to release. Also, the tasks are easy to understand, well structured, and

prioritised, keeping the path of projects transparent and understandable from its initiation till closedown.

Since the Agile Beeswax has been shown to the experts for evaluation, to help us improve our proposed model, we will start the summary of the evaluation of each component one by one. For this, we will analyse if we have to keep this component as it, or we have to remove it, or we have to modify it in future work. Table 6.20 summarises the main experts' validation, we sort the components based on their impact on the adoption of the proposed methodology, from the highest impact first (highest Sig. α). We can see too if the components are affected or not the adoption.

As we can see in Table 6.20, Learning Phases, Operational Practices, Software Engineering Practices, and Management Practices were the most acceptable of the Agile component and have a positive and high impact on the adoption of the proposed Agile Beeswax methodology. Since these components have a positive impact and highly affect adoption, we do not need to modify these five components of the proposed methodology.

For its part, Team Roles has a negative and high impact on the adoption of the Agile Beeswax methodology with a high mean (4.30 out of 5). Therefore, we will let the development teams decide their team roles based on the proposed team roles in a flexible way. A negative impact means for us, that if we have more team roles, we have less adoption. It is just a message from the experts to say that the app development team should not be a larger group because usually the companies have limitations in the numbers of persons working in each project. However, the roles offered in the methodology are appreciated by the experts since on average they have a satisfaction of 4.3, therefore we offer these roles and permits adapt them as required.

Adaptation to Changes and Agile Beeswax General Rules do not affect the adoption of the proposed methodology ($\alpha \geq 0.05$), since the mean of these two components was 4.51 and 4.46 respectively. However, since the total impact of all components is Sig = .000 (the best result for the impact), it represents a vote confirming that choosing to keep these components is a correct decision. Consequently, both components will be kept in the methodology, although they are not considered essential components, but rather transversal guidelines during the development of the phases and execution of the methodology's practices.

In conclusion, Agile Beeswax was focused on being helpful in the mobile app development lifecycle to be more effectively and efficiently. Based on experts' opinions, Agile Beeswax is important and seems to be an ideal development app methodology for small to medium businesses and industries. This methodology based on the result we had, will be beneficial for companies in terms of building mobile apps that will organise the project activities, and the input and output will be clear for the project phases. In terms of theoretical implication, this methodology will increase the knowledge among researchers of how to develop mobile apps. This methodology can be used as essential to developing any other mobile app development model and can relate to another theoretical framework from different areas, as we have done with the TAM model.

Table 6. 20: Summary of the Main Experts' Validation

Sorting	**Methodology components**	**Impact Direction**	**Sig. α**	**Impact/ Segnificant**	**Mean**
1	Learning	Positive	.002	Affect	4.51
2	Phases	Positive	.007	Affect	4.19
3	Operational Practices	Positive	.012	Affect	4.22
4	Team Roles	Negative	.021	Affect	4.30
5	Software Engineering Practices	Positive	.033	Affect	4.07
6	Management Practices	Positive	.036	Affect	4.33
7	Adaptation to Changes	Negative	.127	No affect	4.51
8	Agile Beeswax General Rules	Positive	.193	No affect	4.46
	Total Sig. α		.000		

CHAPTER 7. CONCLUSIONS AND FUTURE WORK

In this thesis, we carried out the mobile application development process based on the companies and academic perspectives. We had four main sections (mobile application and previous studies, study through interviews, study through online surveys, and the proposed methodology and its validation) each section has its conclusions (sections 3.7, 4.1.4, 4.2.4, and 6.10). In this section, we will resume the main conclusions and contributions with the study limitation and future work of our study.

7.1 General Conclusions

Since the release of Apple's first iPhone in 2007, mobile applications (apps) have increased since thin mobile apps are present everywhere. As a result, companies take advantage of this to reach the largest number of users through these applications to achieve profits. Thus, the smartphone market and mobile applications have turned into a very profitable market for companies. However, this market is quite competitive. These companies should follow software engineering practices in the context of the mobile app development process, which will give them more advanced in the competition. This study provides the importance of the development process for mobile apps.

At the same time, software development is moving from a heavyweight to a lightweight method such as the agile approach (Oyong & Ekong, 2019). This occurs in software development in general and, more specifically, mobile app development also requires being agile. The results of this thesis have supported our initial thinking that adopting agile methods is one of the main practices in mobile application project. Agile methodologies can facilitate the creation of mobile applications where requirements are often highly changeable, and they are able to restructure the software engineering to match these requirements.

As for companies, they should enhance their agility to respond quickly to customer demands and market changes while developing mobile apps. Agile methods present itself as a suitable solution for the mobile app development process (Abrahamsson et al. (2003)). Agile is a suitable solution for the mobile app development process because its main practices such as small teams, small iterations, continuous delivery, continuous integration, and frequent change of requirements are very suitable for app development. However, the development

methodologies geared to mobile app development are still unsatisfactory. Inadequate awareness of the real concern in companies and issues that arises during the creation of mobile apps in any context, stimulates more research initiatives in this field.

With this aim, in this thesis, we have performed an organized an extensive search of the pertinent scientific literature to determine the theoretical and practical elements of our study. This study focused on agile methodologies developed for mobile applications. For this purpose, the main unique characteristics of mobile applications were analyzed in detail, and a literature review of the development of mobile applications along with a knowledge extraction process from experts who work in developing apps was performed. Therefore, in order to fulfil our research goals, we performed an in-depth analysis of the methods for developing mobile apps through an extensive review of previous research and with specialists in developing mobile apps.

Only a few mobile app development models have been presented in the literature review research. Across these models, agile approaches are used. The development of mobile apps has its unique requirements. Agile methods can deal with some of these requirements, such as continuous changing requirements. This research reviewed the existing methodologies and models for developing mobile apps and existing methodologies adopted by experts in the development communities. After studying the specific agile methodologies for developing mobile applications, it was found that some of these agile methods are better-organized, more flexible, more fun, more able to make changes, easier to judge on project status, with greater owner involvement and more predictable than other methodologies; but many of them need a higher degree of administration, have bad documentation, ignore or insufficiently define some stages of the mobile development process or do not take into account the special needs of mobile applications. In addition, app idea and concept workshops, requirements gathering, deployment, maintenance, and project assessment activities receive very little attention in the presented models. At the same time, most models are inefficiently verified and rationalized for use in dealing with unique restrictions and needs in mobile apps, such as user participation in the development process.

Consequently, we conclude that the existing agile models that are presented with a mobile development approach are few and not fully suitable for the development of mobile applications. We found how mobile app development models are not yet sufficiently mature. Developing mobile apps have a unique characteristic, new practical approaches are needed. Currently, we believe that agile software development methods provide a good platform for developing mobile applications for smartphones. In our opinion, the agile methods introduced

so far are not enough to fully meet the unique requirements of mobile applications. New methodologies to develop mobile applications are required.

In line with this propose, one of our study's objectives is to understand better the mobile apps development process in the industrial market. We were satisfied in interviewing 14 experts in mobile apps development. The interview's core topic was the methodology used in the company to develop a mobile application. Further, the discussions focused on the main tools and challenges in mobile application development. We can conclude that all companies use different tools in the development process, and there is no focus on a particular type of these tools. We can see that all companies are working hard to integrate customers into the development cycle and keep them updated to get together to the desired mobile application. Understanding the client's requirements presents the main challenge for the development team. They used different methods to meet the customers' needs. Representing the requirement in a prototype model is one of the main of these collecting methods. The customer familiarity with some technical and development skills also makes it easier for the team to understand the application requirements.

In addition, most companies work on an iterative and incremental process while developing mobile applications. Working in iterations between the development teams is a centre practice for the quality of the mobile app, between designer and developers, developers and QA, and designer with the customer. Sometimes there are problems between designers and developers, these problems can be solved with cumulative experience and using some tools. Besides that, non-engineering practices like business development and user training are rarely included in app development models. Most companies claims about the extent to which these models deal with mobile applications particular specifications, such as physical limitation and operating system, and others had specific goals such as connectivity and reusability.

Additionally, most of the companies interviewed have not faithfully or integrally adopted existing development methods. They define their own methodology, in an informal way. We observed that most companies have and follow a mobile application development method and repeat this method with each new project, incorporating some improvements such as using new tools or incorporating new members to the team. These methods are valuable because they accumulate many years of experience, but they need to be better structured, formalized and documented; in addition, they are often focused on the types of development in which the company specializes and are not valid development methods for any mobile application.

After initial interviews with experts and taking into account all the knowledge acquired about mobile development in the industry, we surveyed mobile software experts, developers and academics to understand better the process of creating mobile applications, leading tools, and challenges in mobile application development. All this with the final purpose of defining our own methodology that would converge the advances of industry and academics in the systematization of mobile development. We designed a lengthy, detailed online questionnaire on the process of developing mobile apps (Appendix A). With this study obtained an advanced understanding of the mobile app development process adopted in the business world and the academic community. For example, we could confirm that most of the participants adopt the agile methods in the mobile app development process. Companies think that agile methods are ideal, a natural fit for their mobile app development practices, with Scrum as the most-used method. And User-Centred Design (UCD) is the core of the development process to achieve the greatest satisfaction and best user experience possible. In addition, the mobile app development processes of the companies are iterative and incremental, focusing on short, frequent cycles and the functional product. By other hand, we could see that Project Manager, Scrum Master, Product Owner, Design Lead / Designers UX, UI Development Lead / Developers, Analyst, and Quality Assurance Lead are the main roles in the mobile app development team. And, that all participants use different tools in the development process but focus more on some tools than others. Particularly, the companies have adopted several testing mobile application methods, between manual and automated testing, with functional and usability testing.

Continuing with an exhaustive study of the information obtained from the survey, and making an effort of abstraction, we were able to be found six main phases adopted by participants in the mobile application development lifecycle: The idea and strategy, user experience design (UX), user interface design (UI), design to development handoff- issues and decisions, development, and deployment and monitoring. In addition, we were capable to detect that one of the main challenges that developers face is the transition from design to the development phase. In this line, our findings on best handoff practices suggest using the same team for design and development services in the project and using some of the tools available to manage the handoff. In addition, mobile development companies must respond more quickly to market changes. We have discussed why agile approaches are best suited and a natural fit for the mobile app development process. Mobile app development requires numerous techniques and specific project management practices from agile and Scrum, technical and engineering practices, and some operational practices for project success.

Finally, another interesting finding derived from our study is that testing should go through unit tests, component testing, and integration testing, and during the final iterations, the product should go through system integration testing and acceptance testing. Functional testing should start with the test plan and a list of actions to test. Designers should involve in the testing process, and the same developers should not perform testing.

The lack of general and well-defined standards and methodologies for mobile development highlighted by the experts during the evaluation confirms that work is needed to develop new methodologies that embrace the solid principles previously identified by the academic community, as well as the industry's experience. Mobile app methodology is needed to enhance flexibility for the client to have the capacity of changing the scope of the project at any time of need. It will promote flexibility in planning, customer involvement, continuous evaluation as well as management of the inherent risks associated with the mobile applications development. Usually, in mobile app the customer is not being sure or the right application to develop. Or the development team unable to select the most suitable development technology, or how to deal with the compatibility of the different devices and screen sizes. Furthermore, define the target market and security issues are common issues that the development team should learn to deal with it.

This research has enabled us to present a methodology that encompasses the specific features of mobile app development we called Agile Beeswax. Agile Beeswax is an incremental, iterative development process composed of two main iterative loops (sprints), the incremental design loop and the incremental development loop, and one bridge connecting these two sprints. In addition, two properties that are essential in Agile Beeswax are adaptive to changes and learning during the development process. Agile Beeswax is structured in six phases: idea and strategy, user experience design, user interface design, design to development, handoff and technical decisions, development, and deployment and monitoring. Agile Beeswax practices are an intersection of Agile and Scrum practices, engineering technical practices and operational practices, and a complete phase-based structure and a set of application rules that have been specified to facilitate its adaptation and use. Agile Beeswax integrates all the knowledge acquired about software development in a systematic way and proposes possible tools to assist each established task, while making an assignment of these tasks to the different roles of the team. After its definition, the methodology was presented to a panel of thirty-five experts who confirmed its validity and their predisposition to adopt this methodology.

Agile Beeswax focused on being helpful in the mobile app development lifecycle to be more effectively and efficiently. Agile Beeswax is intended as an ideal methodology for small to medium businesses and industries. In our opinion, this methodology will be beneficial for developers in terms of building mobile apps that will organize the project activities, and the input and output will be clear for the project phases. Also, it will acknowledge the management about the recent tools and technologies, and they can adjust with the external environment in terms of users' requirements and expectations. It can be effectively implemented and applied to improve the performance and the operation of the mobile app development process and knowledge domain of mobile software engineering and educational environment. Adopting the Agile Beeswax approach is expected to increase client satisfaction while increasing the mobile app development process's overall speed, performance, and quality. By following the Agile Beeswax methodology and practices, IT service providers and the academic community can ensure that they will develop and deliver strong app solutions that help companies to achieve their business objectives.

7.2 Limitation and future work

Agile Beeswax (Alrabaiah & Medina, 2021) has been focused on mobile applications and it has been conceived for small to medium-sized mobile application development since that size of projects are those committed by the experts consulted in our study. It is also necessary to specify that Agile Beeswax is not intended to be a methodology for all types of apps. For example, in the case of games, specific methodologies are used that include narrative design and tests of gameplay, among other differentiating characteristics (Garneli et. al., 2017). Therefore, Agile Beeswax can be suitable for commercial categories, but it is not suitable for gaming educational apps (for instance). In addition, Agile Beeswax is a methodology following the agile approach; consequently, it may be adapted for use. Therefore, there is no one-fits-all mobile application, although, with accumulated experience, users of this methodology will be able to choose the best practices to build their apps.

Our future research will start conducting a broader validation experience, with a larger number of experts to validate if our findings are correct and if the proposed methodology is consistent with them. Necessary specifications, modifications or extensions will be made to the methodology based on expert opinion. Then, the Agile Beeswax will be presented to some companies to adopt this methodology in developing mobile applications in order to evaluate and develop the method. The information gathered with this new evaluation will

allow for refinement and modification of the Agile Beeswax descriptions and the insertion of compiled examples of practical application to the proposed methodology.

References

(Kayes, I., Sarker, M., & Chakareski, J. (2016). Product backlog rating: a case study on measuring test quality in scrum. Innovations in Systems and Software Engineering, 12(4), 303-317.).

A. Holzinger, G. Searle, and A. Nischelwitzer. (2007). On some aspects of improving mobile applications for the elderly. *Lecture Notes in Computer Science*, vol. 4554, pp. 923-932.

Abrahamsson, P., Salo, O., Ronkainen, J., Warsta, J. (2002). Agile software development methods. Review and analysis, VTT Publications, Espoo

Abrahamsson, P., Warsta, J., Siponen, M., & Ronkainen, J. (2003). New directions on Agile methods: A comparative analysis. 25th International Conference on Software Engineering, 2003. Proceedings. https://doi.org/10.1109/icse.2003.1201204

Abrahamsson, P. (2001). Commitment development in software process improvement: Critical misconceptions. *Proceedings of the 23rd International Conference on Software Engineering. ICSE*. https://doi.org/10.1109/icse.2001.919082

Abrahamsson, P., Hanhineva, A., Hulkko, H., Ihme, T., Jlinoja, J., Korkala, M., Koskela, J., Kyllnen, P., & Salo, O. (2004). Mobile-D. *Companion to the 19th annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications - OOPSLA '04*. https://doi.org/10.1145/1028664.1028736

Adman, P. (1997). Book review: Understanding and evaluating methodologies—NIMSAD: A systemic framework. Nimal

Agile software development. (2004). Wikipedia, the free encyclopedia. Retrieved October 16, 2020, from https://en.wikipedia.org/wiki/Agile_software_development

Aguilar, M., & Zapata, C. (2016). Integrating UCD and an Agile methodology in the development of a mobile catalog of plants. Advances in Ergonomics Modeling, Usability & Special Populations, 75-87. https://doi.org/10.1007/978-3-319-41685-4_8.

Ahmad, A., Li, K., Feng, C., Asim, S. M., Yousif, A., & Ge, S. (2018). An empirical study of investigating mobile applications development challenges. IEEE Access, 6, 17711-17728. https://doi.org/10.1109/access.2018.2818724

AL-Allaf, O. N. A. (2008). A Proposed Hybrid Web Engineering Process Model for Large-Scale Web-Based Applications Development in Large Web Development Enterprises.

Unpublished Doctoral Thesis, Faculty of Information System and Technology, the Arab Academy of Banking and Financial Science, Amman, Jordan.

Albuquerque Santos, P., Porfírio, R. P., Madeira, R. N., & Correia, N. (2021). Computational framework to support development of applications running on multiple Co-located devices. Companion of the 2021 ACM SIGCHI Symposium on Engineering Interactive Computing Systems. https://doi.org/10.1145/3459926.3464758

Alexander. (2006). *Extreme projects*. The Chief Happiness Officer Blog. https://positivesharing.com/2006/01/extreme-projects

Al-Muhtadi, J., Shahzad, B., Saleem, K., Jameel, W., & Orgun, M. A. (2017). Cybersecurity and privacy issues for socially integrated mobile healthcare applications operating in a multi-cloud environment. Health Informatics Journal, 25(2), 315-329. https://doi.org/10.1177/1460458217706184

Alrabaiah, H. A., & Medina-Medina, N. (2021). Agile beeswax: Mobile app development process and empirical study in real environment. Sustainability, 13(4), 1909. https://doi.org/10.3390/su13041909

Alrabaiah, H.A., and Medina-Medina, N. (2019.) Mobile application development process in real environments. *Proceedings of the International Conference on Software Engineering Research and Practice (SERP)*. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), *Las Vegas, NV, USA* (Vol. 29).

Alsabi, E., & Dahanayake, (2016). A. Smart modeling for lightweight mobile application development methods. Communications in Computer and Information Science, 168-179. 2016. https://doi.org/10.1007/978-3-319-44066-8_18

Alsaqqa, S., Sawalha, S., & Abdel-Nabi, H. (2020). Agile software development: Methodologies and trends. International Journal of Interactive Mobile Technologies (iJIM), 14(11), 246. https://doi.org/10.3991/ijim.v14i11.13269

Altaleb, A., & Gravell, A. (2019). An empirical investigation of effort estimation in mobile apps using agile development process. *Journal of Software*, *14*(8), 356-369.

Altuwaijri, F. S., & Ferrario, M. A. (2021, May). Awareness and Perception of Agile in Saudi Software Industry. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)* (pp. 10-18). IEEE.

Ambler, S. (2005). *Quality in an agile world* (7(4), 34). Software Quality Professional. https://www.oamk.fi/~palo/English_2/AgileQuality.pdf

Andrei, B. A., Casu-Pop, A. C., Gheorghe, S. C., & Boiangiu, C. A. (2019). A study on using waterfall and agile methods in software project management. *Journal Of Information Systems & Operations Management*, 125-135.

Aoyama, M. (1998). Agile software process and its experience. *Proceedings of the 20th International Conference on Software Engineering*, 3-12. https://doi.org/10.1109/icse.1998.671097

Ashishdeep. A., Bhatia, J., and Varma, K. (2016). Software process models for mobile application development: A review. Computer Science and Electronic Journal, 7.1 (2016): 150-153.

Awad, M. A. (2005). A Comparison between Agile and Traditional Software Development Methodologies [Master's thesis]. http://pds10.egloos.com/pds/200808/13/85/A_comparision_between_Agile_and_Traditional_SW_development_methodologies.pdf

Balaziuk, O., Sysoieva, I., & Pilyavets, V. (2020). Control and accounting aspects of introducing agile methodology for software development projects. *Financial and credit activity: problems of theory and practice*, *3*(34), 92-100.

Barras, C., Geoffrois, E., Wu, Z., & Liberman, M. (2001). Transcriber: development and use of a tool for assisting speech corpora production. Speech Communication, 33(1-2), 5-22.

Barry, C., & Lang, M. (2001). A survey of multimedia and web development techniques and methodology usage. *IEEE Multimedia*, *8*(2), 52-60. https://doi.org/10.1109/93.917971

Bazarov, S. E., Kholodilin, I. Y., Nesterov, A. S., & Sokhina, A. V. (2017). Applying augmented reality in practical classes for engineering students. IOP Conference Series: Earth and Environmental Science, 87, 032004. https://doi.org/10.1088/1755-1315/87/3/032004

Beck, K., & Andres, C. (2004). *Extreme programming explained: Embrace change*. Addison-Wesley Professional.

Beck, K., Grenning, J., & Martin, R. (2001). Manifesto for Agile Software Development. https://agilemanifesto.org/

Bellatti, J. (2021). Novus UX/UI design. Jess Bellatti | Work. https://jessbellatti.com/novus.php

Berge, Z. L., & Muilenburg, L. (2013). *Handbook of mobile learning*. Routledge.

Boehm, B. W. (1995). A spiral model of software development and enhancement. *Readings in Human–Computer Interaction*, 281-292. https://doi.org/10.1016/b978-0-08-051574-8.50031-5

Ceci, L. (2021, September). Number of apps available in leading app stores as of 1st quarter 2021. Statista. https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores

Charland, A., & LeRoux, B. (2011). Mobile application development: Web vs. native. *Queue*, *9*(4), 20-28. https://doi.org/10.1145/1966989.1968203

Cheney, S., & Thompson, E. (2018, May). The 2017-2022 App Economy Forecast. appannie.com. https://www.appannie.com/en/insights/market-data/app-annie-2017-2022-forecast

Collado-Borrell, R., Escudero-Vilaplana, V., Villanueva-Bueno, C., Herranz-Alonso, A., & Sanjurjo-Saez, M. (2020). Features and functionalities of smartphone apps related to COVID-19: systematic search in app stores and content analysis. *Journal of medical Internet research*, *22*(8), e20334.

Cooling, J. E. (2013). Software design for real-time systems. Springer.

Corral, L., Sillitti, A., & Succi, G. (2013). Software development processes for mobile systems: Is agile really taking over the business? In *2013 1st International Workshop on the Engineering of Mobile-Enabled Systems (MOBS)* (pp. 19-24). IEEE.

Corral, L., Sillitti, A., & Succi, G. (2014). Software assurance practices for mobile applications. Computing, 97(10), 1001-1022. https://doi.org/10.1007/s00607-014-0395-8

Cunha, T. F., Dantas, V. L., & Andrade, R. M. (2011). SLeSS: A Scrum and Lean Six Sigma integration approach for the development of Sofware customization for mobile phones. 2011 25th Brazilian Symposium on Software Engineering. https://doi.org/10.1109/sbes.2011.38

D. Zhang, and B. Adipat. (2005).Challenges, methodologies, and issues in the usability testing of mobile applications. International Journal of Human-Computer Interaction, vol. 18, pp. 293-308.

Dehlinger, & Dixon. (2011). *In Workshop on mobile software engineering*. Mobile application software engineering: Challenges and research directions. (Vol. 2, pp. 29-32).

Denning, S. (2018). How major corporations are making sense of Agile. *Strategy & Leadership*.

Dianat, I., Adeli, P., Asgari Jafarabadi, M., & Karimi, M. A. (2019). User-centred web design, usability and user satisfaction: The case of online banking websites in Iran. Applied Ergonomics, 81, 102892. https://doi.org/10.1016/j.apergo.2019.102892

Dooms, G., Deville, Y., & Dupont, P. (2005). CP(Graph): Introducing a graph computation domain in constraint programming. *Principles and Practice of Constraint Programming - CP 2005*, 211-225. https://doi.org/10.1007/11564751_18

Düchting, M., Zimmermann, D., & Nebe, K. (2007). Incorporating user centered requirement engineering into Agile software development. *Human-Computer Interaction. Interaction Design and Usability*, 58-67. https://doi.org/10.1007/978-3-540-73105-4_7

E. Folmer, and J. Bosch. (2004). Architecting for usability: A survey, Journal of Systems and Software, vol. 70, pp. 61-78.

Ebert, C. (2018). 50 years of software engineering: Progress and perils. IEEE Software, 35(5), 94-101. https://doi.org/10.1109/ms.2018.3571228.

Eler, M. M., Rojas, J. M., Ge, Y., & Fraser, G. (2018). Automated accessibility testing of mobile apps. 2018 IEEE 11th International Conference on Software Testing, Verification and Validation (ICST). https://doi.org/10.1109/icst.2018.00021

F. D. Davis. (1986). Technology Acceptance Model for empirically testing new end-user information systems theory and results. Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, 1986.

Falloon, G. (2017). Mobile devices and apps as scaffolds to science learning in the primary classroom. *Journal of Science Education and Technology*, *26*(6), 613-628. https://doi.org/10.1007/s10956-017-9702-4

Faraj, S. and Sproull, L. Coordinating expertise in software development teams. (2000) *Management Science* 46,12, 1554–1568.

Finkelstein, A., & Kramer, J. (2000). Software engineering. Proceedings of the conference on The future of Software engineering - ICSE '00. https://doi.org/10.1145/336512.336519

Fishbein, J. N., Nisotel, L. E., MacDonald, J. J., Amoyal Pensak, N., Jacobs, J. M., Flanagan, C., Jethwani, K., & Greer, J. A. (2017). Mobile application to promote adherence to oral chemotherapy and symptom management: A protocol for design and development. *JMIR Research Protocols*, *6*(4), e62. https://doi.org/10.2196/resprot.6198

Flick, U. (2015). Introducing research methodology: A beginner's guide to doing a research project. SAGE.

Flora, H. K. (2018). Adopting an agile approach for the mobile applications development. https://shodhgangotri.inflibnet.ac.in/jspui/bitstream/123456789/5451/1/synopsis.pdf

Flora, H. K., & Chande, S. V. (2013). A review and analysis on mobile application development processes using Agile methodologies. *International Journal of Research in Computer Scienc*, *3*(4), 8-18. https://doi.org/10.7815/ijorcs.34.2013.068

Flora, H. K., Wang, X., & Chande, S. V. (2014). An investigation on the characteristics of mobile applications: A survey study. *International Journal of Information Technology and Computer Science*, *6*(11), 21-27. https://doi.org/10.5815/ijitcs.2014.11.03

Flowers, B. (2017, May). Chapter 1 review of literature. SILO of research documents. https://silo.tips/download/chapter-1-review-of-literature

Freire-Obregón, D., Narducci, F., Barra, S., & Castrillón-Santana, M. (2019). Deep learning for source camera identification on mobile devices. Pattern Recognition Letters, 126, 86-91. https://doi.org/10.1016/j.patrec.2018.01.005

Garcia, A. L., da Rocha Miguel, I., Eugênio, J. B., da Silva Vilela, M., & Marcondes, G. A. B. (2020). Scrum-Based Application for Agile Project Management. *JSW*, *15*(4), 106-113.

Garneli, V., Giannakos, M., & Chorianopoulos, K.(2017). Serious games as a malleable learning medium: The effects of narrative, gameplay, and making on students' performance and attitudes. British Journal of Educational Technology, 48(3), 842-859

Gavalas, D., & Economou, D. (2011). undefined. *IEEE Software*, *28*(1), 77-86. https://doi.org/10.1109/ms.2010.155

Ghandi, L., Silva, C., Martinez, D., & Gualotuna, T. (2017). Mobile application development process: A practical experience.12th Iberian Conference on Information Systems and Technologies (CISTI). https://doi.org/10.23919/cisti.2017.7975825.

Gorla, N. and Lam, Y. W. (2004). Who should work with whom? Building effective software project teams. Comm. ACM47, 79–82.

Gulliksen, J., Göransson, B., Boivie, I., Blomkvist, S., Persson, J. and Cajander, Å. (2003). 'Key principles for user-centred systems design'. Behaviour & Information Technology 22 (6), pp. 397–409.

Hamm, M. J. (2014). Wireframing essentials. Packt Publishing Ltd.

Hamsini, R., & Smitha, H. (2016). *Agile Development Methodology and Testing for Mobile Applications - A Survey* [Paper presentation]. International Journal of New Technology and Research (IJNTR) ISSN:2454-4116, Volume-2, Issue-9.

Haynes, R., & Friedenberg, M. (2006). *Best Practices in Agile Software Development*. College of Information Sciences & Technology, The Pennsylvania State University, Pennsylvania. https://www.researchgate.net/profile/Marc_Friedenberg/publication/228555529_Best_Practices_in_Agile_Software_Development/links/0deec520e3ad00f763000000.pdf

Heitkötter, H., Hanschke, S., & Majchrzak, T. A. (2013). undefined. *Lecture Notes in Business Information Processing*, 120-138. https://doi.org/10.1007/978-3-642-36608-6_8

Herrero, I. and Salmeron, J. L. (2005). Using the DEA methodology to rank software technical efficiency. Comm.ACM, 48, 101–105.

Holler. R., (2006) Mobile application development: a natural fit with agile methodologies. VerisonOne LLC, Alpharetta.
https://csce.ucmss.com/cr/books/2019/LFS/CSREA2019/SER4050.pdf

Humphrey, W. S., & Kellner, M. I. (1989). Software process modeling. *Proceedings of the 11th international conference on Software engineering - ICSE '89*. https://doi.org/10.1145/74587.74631

Hussain, A., & Ferneley, E. (2008). Usability metric for mobile application. *Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services - iiWAS '08*. https://doi.org/10.1145/1497308.1497412

International Organization for Standardization (2010). ISO 9241-210. Ergonomics of human-system interaction — part 210: human-centred design for interactive systems. Geneva, Switzerland: ISO.

InVision. Wireframe. (2021). Transformative collaboration for all the work you do | InVision. https://www.invisionapp.com/defined/wireframes

J. Heo, D-H. Hamb, S. Park, C. Song, and W.C. Yoon. (2009). A framework for evaluating the usability of mobile phones based on multi-level, hierarchical model of usability factors. Interacting with Computers, vol. 21, pp. 263-275.

Jabangwe, R., Edison, H., & Duc, A. N. (2018). Software engineering process models for mobile app development: A systematic literature review. Journal of Systems and Software, 145, 98-111. https://doi.org/10.1016/j.jss.2018.08.028

Jain, P., Sharma, A., & Ahuja, L. (2019). A customized quality model for software quality assurance in agile environment. *International Journal of Information Technology and Web Engineering (IJITWE)*, *14*(3), 64-77.

Jayaratna. McGraw-hill book company, information systems, management and strategy series, London, 1994. 259 pp. ISBN 0 07 707882 9 (paperback). Systems Research and Behavioral Science, 14(5), 352-354. https://doi.org/10.1002/(sici)1099-1743(199709/10)14:5<352::aid-sres186>3.0.co;2-g

Jayaswal, B. K., & Patton, P. C. (2006). Design for trustworthy software: Tools, techniques, and methodology of developing robust software. Pearson Education.

Jeffries, R. (2003). What is extreme programming? RonJeffries.com. https://ronjeffries.com/xprog/what-is-extreme-programming/

Jeffries, R. (n.d.). *XP Practices*. xProgramming.com

Jeffries, R., Anderson, A., & Hendrickson, C. (2001). *Extreme programming installed*. Addison-Wesley Professional.

Joorabchi, M. E., Mesbah, A., & Kruchten, P. (2013). Real challenges in mobile app development. *2013 ACM / IEEE International Symposium on Empirical Software Engineering and Measurement*. https://doi.org/10.1109/esem.2013.9

Jumaat, N. F., & Tasir, Z. (2013). Integrating project based learning environment into the design and development of mobile apps for learning 2D-Animation. *Procedia - Social and Behavioral Sciences*, *103*, 526-533. https://doi.org/10.1016/j.sbspro.2013.10.369

K.Flora, H., V. Chande, S., & Wang, X. (2014). Adopting an Agile approach for the mobile applications development. *International Journal of Computer Applications*, *94*(17), 43-50. https://doi.org/10.5120/16454-6199

Kangas, E., & Kinnunen, T. (2005). Applying user-centered design to mobile application development. *Communications of the ACM*, *48*(7), 55-59. https://doi.org/10.1145/1070838.1070866

Katranas, G., Riel, A., Corchado-Rodríguez, J. M., & Plaza-Hernández, M. (2020). The SMARTSEA education approach to leveraging the Internet of Things in the maritime industry. In *European Conference on Software Process Improvement* (pp. 247-258). Springer, Cham.

Kayes, I., Sarker, M., & Chakareski, J. (2016). Product backlog rating: A case study on measuring test quality in Scrum. *Innovations in Systems and Software Engineering*, *12*(4), 303-317. https://doi.org/10.1007/s11334-016-0271-0

Khaddage, F., & Lattemann, C. (2013). The future of mobile apps for teaching and learning. *Handbook of Mobile Learning*, 119-128. https://doi.org/10.4324/9780203118764.ch11

Khalid, M., Asif, M., & Shehzaib, U. (2015). Towards improving the quality of mobile app reviews. International Journal of Information Technology and Computer Science (IJITCS), 7(10), 35.

Khan, M. U., Abbas, S., Lee, S. U., & Abbas, A. (2021). Measuring power consumption in mobile devices for energy sustainable app development: A comparative study and challenges. Sustainable Computing: Informatics and Systems, 31, 100589. https://doi.org/10.1016/j.suscom.2021.100589

Kirmani, M. M. (2017). Agile Development Method for Mobile applications: A Study. *International Journal of Advanced Research in Computer Science*, *8*(5).

Krehbiel, T. C., Salzarulo, P. A., Cosmah, M. L., Forren, J., Gannod, G., Havelka, D., & Merhout, J. (2017). Agile Manifesto for Teaching and Learning. *Journal of Effective Teaching*, *17*(2), 90-111.

Kumar, K., Gupta, P. K., & Upadhyay, D. (2011). Change-oriented adaptive software engineering by using Agile methodology: CFDD. 2011 3rd International Conference on Electronics Computer Technology. https://doi.org/10.1109/icectech.2011.5941947

Lantz, K. (1997). *Manageknowledge*. Manageknowledge. https://www.manageknowledge.com/prototyp.html

Lee, S., & Geum, Y. (2020). How to determine a minimum viable product in app-based lean start-ups: Kano-based approach. Total Quality Management & Business Excellence, 1-17.

Leffingwell, D. (2007). *Scaling software agility: Best practices for large enterprises*. Pearson Education.

Li, L., Bissyandé, T. F., & Klein, J. (2019). Rebooting research on detecting repackaged android apps: Literature review and benchmark. *IEEE Transactions on Software Engineering*.

Lima, I. R., de Castro Freire, T., & Costa, H. A. X. (2012). Adapting and using Scrum in a software research and development laboratory. Revista de Sistemas de Informação da FSMA, 9, 16-23.

Losada, B. (2018). Flexible requirement development through user objectives in an Agile-UCD hybrid approach. Proceedings of the XIX International Conference on Human Computer Interaction. https://doi.org/10.1145/3233824.3233865

Mahajan, R., & Kaur, P. (2010). Extreme programming: newly acclaimed agile system development process. *International Journal of Information Technology and Knowledge Management* , *3*(2), 699-705. http://csjournals.com/IJITKM/PDF%203-1/101.pdf

Mahalakshmi, M., & Sundararajan, M. (2013). Traditional SDLC vs scrum methodology–a comparative study. International Journal of Emerging Technology and Advanced Engineering, 3(6), 192-196.

Martinez, D., Ferre, X., Guerrero, G., & Juristo, N (2020). An Agile-based integrated framework for mobile application development considering Ilities. IEEE Access, 8, 72461-72470. https://doi.org/10.1109/access.2020.2987882.

Martinez, D., Ferre, X., Guerrero, G., & Juristo, N. (2020). An Agile-based integrated framework for mobile application development considering Ilities. *IEEE Access*, *8*, 72461-72470. https://doi.org/10.1109/access.2020.2987882

Martins, P. V., & Zacarias, M. (2017). An agile business process improvement methodology. *Procedia Computer Science*, *121*, 129-136.

Mascheroni, M. A., and Irrazábal, E Mascheroni, Maximiliano Agustín, and Emanuel Irrazábal. (2018) Problemas que afectan a la Calidad de Software en Entrega Continua y Pruebas Continuas. XXIV Congreso Argentino de Ciencias de la Computación.

Masi, E., Cantone, G., Mastrofini, M., Calavaro, G., & Subiaco, P. (2013). Mobile apps development: A framework for technology decision making. *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, 64-79. https://doi.org/10.1007/978-3-642-36632-1_4

Mathew, J. (2019). Scrum master certification: Questions with answers and explanations, part: 1.

Miro. (n.d). (2021). Introducing App Development Canvas and Business Model Canvas, Miro. https://miro.com/templates/app-development-canvas/

Mobile apps for language learning. Language Learning & Technology, 15(2), 2-11 [Paper presentation]. (2011). Language Learning & Technology, hawai.edu. https://scholarspace.manoa.hawaii.edu/bitstream/10125/44244/15_02_emerging.pdf

Mojica, I. J., Adams, B., Nagappan, M., Dienst, S., Berger, T., & Hassan, A. E. (2014). A large-scale empirical study on software reuse in mobile apps. IEEE Software, 31(2), 78-86. https://doi.org/10.1109/ms.2013.142

Mollet, L. S. (2021). Emergent practices in agile people management: a multiple-case study of SMEs in New Zealand and Switzerland: [Doctoral dissertation]. https://researcharchive.lincoln.ac.nz/handle/10182/14088

Morris, J., & Maynard, V. (2010). Pilot study to test the use of a mobile device in the clinical setting to access evidence-based practice resources. *Worldviews on Evidence-Based Nursing*, *7*(4), 205-213. https://doi.org/10.1111/j.1741-6787.2009.00171.x

Mota, J. M., Ruiz-Rube, I., Dodero, J. M., & Arnedillo-Sánchez, I. (2018). Augmented reality mobile app development for all. *Computers & Electrical Engineering*, *65*, 250-260. https://doi.org/10.1016/j.compeleceng.2017.08.025

Motta, R. C., De Oliveira, K. M., & Travassos, G. H. (2018). On challenges in engineering IoT software systems. Proceedings of the XXXII Brazilian Symposium on Software Engineering - SBES '18. https://doi.org/10.1145/3266237.3266263

Muccini, H., Di Francesco, A., & Esposito, P. (2012). Software testing of mobile applications: Challenges and future research directions. *2012 7th International Workshop on Automation of Software Test (AST)*. https://doi.org/10.1109/iwast.2012.6228987

Nandyal, A. B., & Rafi, M. (2020). Determining feature gaps of open source cloud platforms for mobile backend as service (MBaaS) in enterprise mobile applications. 2020 IEEE

International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER). https://doi.org/10.1109/discover50404.2020.9278113

Newkirk, J. (2002). Introduction to Agile processes and extreme programming. *Proceedings of the 24th international conference on Software engineering - ICSE '02*, 19-25. https://doi.org/10.1145/581441.581450

Norgren, A. (2004). *Requirements Engineering and prototyping in a Legacy Software Setting* [Doctoral dissertation]. https://www.diva-portal.org/smash/get/diva2:215122/FULLTEXT01.pdf

Nur, T., & Joviando, J. (2021). Determination of E-wallet usage intention : Extending the TAM model with self efficacy. *2021 3rd International Conference on Cybernetics and Intelligent System (ICORIS)*. https://doi.org/10.1109/icoris52787.2021.9649568

O'Dea, S. (2021, August). Smartphone users worldwide 2016-2021. Statista. https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide

Oracle. (2022, January). What is ERP? https://www.oracle.com/erp/what-is-erp/

Oyong, S., & Ekong, V. (2019). An explorative survey of formal and Agile software development methods. Global Journal of Pure and Applied Sciences, 25(1), 71. https://doi.org/10.4314/gjpas.v25i1.10

Pardo-Calvache, C. J., Chilito-Gómez, P. R., Viveros-Meneses, D. E., & Pino, F. J. (2019). Scrum+: A scaled Scrum for the agile global software development project management with multiple models. *Revista Facultad de Ingeniería Universidad de Antioquia*, (93), 105-116.

Pathak, S., & Saxena, P. (2012). Hybrid Methodology Involving Scrum and Waterfall Model towards the Software Project Development in Academic Knowledge Centers. *International Journal of Evaluation and Research in Education (IJERE)*, *1*(1), 25-32.

Pecorelli, F., Catolino, G., Ferrucci, F., De Lucia, A., & Palomba, F. (2020). Testing of mobile applications in the wild. Proceedings of the 28th International Conference on Program Comprehension. https://doi.org/10.1145/3387904.3389256

Pendharkar, P. C., & Rodger, J. A. (2009). The relationship between software development team size and software development cost. *Communications of the ACM,* 52(1), 141-144.

Pendharkar, P.C. and Rodger, J.A. (2007) An empirical study of the impact of team size on software development effort, Information Technology and Management 8, 253262.

Peters. (2007). *Software engineering: An engineering approach*. John Wiley & Sons.

Pham, Y. D., Fucci, D., & Maalej, W. (2018). A first implementation of a design thinking workshop during a mobile app development course project. In *Proceedings of the 2nd International Workshop on Software Engineering Education for Millennials* (pp. 56-63).

Pigrum, D. (2021). 'The Board Behind My Desk': The Wall-Beside-the-Work in Architectural Practice. In The Wall Beside the Work (pp. 185-194). Springer, Cham

Preece, J., Rogers, Y. and Sharp, H. (2001). Interaction design: beyond human-computer interaction. New York, U.S.A.: John Wiley & Sons.

Pressman, R. S. (2005). Software engineering: A practitioner's approach. Palgrave Macmillan.

Pressman, R., & Maxim, B. (2014). Software engineering: A practitioner's approach. McGraw-Hill Education.

process: A practical experience. 2017 12th Iberian Conference on Information

Prototyping software life cycle model. (n.d.). Visual Basic Tutorial - Learn VB Programming with source code. https://www.freetutes.com/systemanalysis/sa2-prototyping-model.html FIX

R. Baharuddin, D. Singh, and R. Razali. (2013). Usability dimensions for mobile applications—A review, *Res. J. Appl. Sci. Eng. Technol*, vol. 5, pp. 2225-2231.

Rahimian, V., & Habibi, J. (2008). Performance evaluation of mobile software systems: Challenges for a software engineer. *2008 5th International Conference on Electrical Engineering, Computing Science and Automatic Control*. https://doi.org/10.1109/iceee.2008.4723426

Ramanathan, M. (2007). *A new software process model designed from the basics of evolutionary biology and software evolution* [Master's thesis]. https://shareok.org/bitstream/handle/11244/8226/Ramanathan_okstate_0664M_2582.pdf?sequence=1&isAllowed=y

Rasnacis, A., & Berzisa, S. (2017). Method for adaptation and implementation of agile project management methodology. *Procedia Computer Science*, *104*, 43-50.

Ribeiro, A., & Domingues, L. (2018). Acceptance of an agile methodology in the public sector. *Procedia computer science*, *138*, 621-629.

Rivera, M. (2020, November 17). 2x2 matrix. https://www.fool.com. https://www.fool.com/the-blueprint/prioritization-matrix/

Rosales-Morales, V. Y., Sánchez-Morales, L. N., Alor-Hernández, G., Garcia-Alcaraz, J. L., Sánchez-Cervantes, J. L., & Rodriguez-Mazahua, L. (2019). ImagIngDev: A new approach for developing automatic cross-platform mobile applications using image processing techniques. The Computer Journal, 63(5), 732-757. https://doi.org/10.1093/comjnl/bxz029

Rubin, J. and Chisnell, D. (2008) Handbook of usability testing: how to plan, design and conduct effective tests. John Wiley and Sons.

Ruiz, I. J., Nagappan, M., Adams, B., & Hassan, A. E.(2012). Understanding reuse in the Android market. 20th IEEE International Conference on Program Comprehension (ICPC). https://doi.org/10.1109/icpc.2012.6240477.

Saleh, S. M., Huq, S. M., & Rahman, M. A. (2019). Comparative study within Scrum, Kanban, XP focused on their practices. In *2019 International Conference on* Electrical, Computer and Communication Engineering (ECCE) (pp. 1-6). IEEE.

Satpathy, T. (2013). *Una guía para el conocimiento de Scrum (Guía SBOK™). Phoenix, Arizona:*. SCRUMstudy.

Schwaber, K. (2004). *Agile project management with Scrum*. O'Reilly Media.

Schwaber, K., & Beedle, M. (2002). *Agile software development with Scrum*. Pearson.

Schwalbe, K. (2012, September). Managing a project using an agile approach and the PMBOK® guide. In Proceedings of the Information Systems Educators Conference ISSN (Vol. 2167, p. 1435).

Serrano, N., Hernantes, J., & Gallardo, G. (2013). Mobile web apps. *IEEE Software*, *30*(5), 22-27. https://doi.org/10.1109/ms.2013.111

Sharif, A. M. (2010). It's written in the cloud: The hype and promise of cloud computing. *Journal of Enterprise Information Management*, *23*(2), 131-134. https://doi.org/10.1108/17410391011019732

Shivageeta, S. Choodi. (2020). Testing in iterative product development environment." [Online]. Available: http://www.qaielearning.com/KnowledgePapers/Testing_In_Iterative.pdf.

Shmatko, N. A., & Volkova, G. L. (2019). Willingness of Russian researchers to digital transformation: Basic digital literacy and advanced skills. Proceedings of the Middle-Term Conference RC04 Sociology of Education International Sociological

Association (ISA), Institute of Foreign Languages RUDN University, Moscow, 24-26 July 2019. https://doi.org/10.22363/09669-2019-616-625

Sin, D., Lawson, E., & Kannoorpatti, K. (2012). Mobile web apps - The non-programmer's alternative to native applications. *2012 5th International Conference on Human System Interactions*. https://doi.org/10.1109/hsi.2012.11

Sketch. The digital design toolkit. (2021, September). Sketch. https://www.sketch.com/

Sommerville, I. (2004). *Software engineering*. Addison-Wesley.

Statista. (2021). *Smartphone subscriptions worldwide 2016–2026*. https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/

Systems and Technologies (CISTI).

Tao, C., Guo, H., & Huang, Z. (2020). Identifying security issues for mobile applications based on user review summarization. Information and Software Technology, 122, 106290. https://doi.org/10.1016/j.infsof.2020.106290

Vaupel, S., Taentzer, G., Gerlach, R., & Guckert, M. (2016). undefined. *Software & Systems Modeling*, *17*(1), 35-63. https://doi.org/10.1007/s10270-016-0559-4

Vijayasarathy, L. R., & Butler, C. W. (2015). Choice of software development methodologies: Do organisational, project, and team characteristics matter?. IEEE software, 33(5), 86-94. Chicago

Vithani, T., & Kumar, A. (2014). Modeling the mobile application development lifecycle. In *Proceedings of the International MultiConference of Engineers and Computer Scientists* (Vol. 1, pp. 596-600).

Wang, R. J. (2020). Branded mobile application adoption and customer engagement behavior. Computers in Human Behavior, 106, 106245. https://doi.org/10.1016/j.chb.2020.106245

Wang, W., Li, Y., Wang, X., Liu, J., & Zhang, X. (2018). Detecting Android malicious apps and categorizing benign apps with ensemble of classifiers. *Future Generation Computer Systems*, *78*, 987-994. https://doi.org/10.1016/j.future.2017.01.019

Wasserman, A. I. (2010). Software engineering issues for mobile application development. *Proceedings of the FSE/SDP workshop on Future of software engineering research - FoSER '10*. https://doi.org/10.1145/1882362.1882443

Wong, C. Y., Khong, C. W., & Chu, K. (2012). Interface design practice and education towards mobile apps development. *Procedia - Social and Behavioral Sciences*, *51*, 698-702. https://doi.org/10.1016/j.sbspro.2012.08.227.

Wynn, D. C., & Clarkson, P. J. (2017). Process models in design and development. Research in Engineering Design, 29(2), 161-202. https://doi.org/10.1007/s00163-017-0262-7

Yu, L. (2011). Coevolution of information ecosystems. ACM SIGSOFT Software Engineering Notes, 36(6), 1-5. https://doi.org/10.1145/2047414.2047435

APPENDIX A

In this section we will present our survey that we used in our research for mobile application development, that was covered in chapter 4.

QN	CATEGORY/QUESTIONS	POSSIBLE ANSWERS	% EACH ANSWER
ABOUT THE PARTICIPANTS			
QN 1	Gender	Mail	22.9%
		Femail	71.1%
QN 2	Age	20-34	42.9%
		35-44	42.9%
		45-54	5.7%
		55-64	8.6%
		> 64	---
QN 3	Country		
QN 4	I identify my work in	Company	80%
		Academic	20%
QN 5	I'm a specialist in	Designer UX/XI Designer	2.9%
		Mobile app developer	34.3%
		Tech Lead/Manage	20%
		Top management level	25.7%
		Marketing	2.9%
		Software Engineer	37.1%
		Software Tester	8.6%
		Other…	2.9%
ABOUT YOUR ORGANIZATION			
QN 6	Does your organisation develop mobile apps?	Yes, frequently	48.6%
		No, never	11.4%

		Sometimes	40%
QN 7	The number of main apps that have been developed by your organisation in the last five years are	Never	5.7%
		Less than 5 apps	25.7%
		5 to 10 apps	37.1%
		11 to 20 apps	14.3%
		More than 21 apps	17.1%
QN 8	What is the staff size of the mobile development team in your organisation?	Less than 5 employees	51.4%
		5 to 10 employees	25.7%
		11 to 20 employees	14.3%
		21 to 50 employees	2.9%
		51 to 100 employees	-
		More than 100 employees	5.7%
QN 9	Does your organisation usually use agile methodologies in mobile apps development process?	Yes, frequently	48.6%
		No	17.1%
		Sometimes	34.3%
QN 10	Agile approaches used in mobile application development in your organisation are...	None	28.1%
		Scrum	56.3%
		XP	12.5%
		Kanban	12.5%
		Lean	9.4%
		Other…	--
QN 11	Do you believe that agile methods and its practices and rules are appropriate for the development of mobile applications?	Yes, always	61.8%
		No	2.9%
		Sometimes	35.3%
THE IDEA AND STRATEGY			
QN 12	When you start thinking of developing an app, do you usually implement some market	Yes, frequently	54.3%
		No	-

	research -competitive audit- focusing on other apps that are carrying out the same idea and its user rating and reviews?	Sometimes	42.9%
QN 13	Do you usually write a final report after finishing the planning-strategy stage?	Yes, frequently	38.3%
		No	20.6%
		Sometimes	35.3%
QN 14	Do you usually create a marketing campaign for your app?	Yes, frequently	29.4%
		No	17.6%
		Sometimes	52.9%
QN 15	Do you usually define the roadmap for your app to be successful from day one?	Yes, frequently	58.8%
		No	14.7%
		Sometimes	26.5%
QN 16	Which tools are you using in a project and team management?	Kanbanchi	0%
		JIRA	43.3%
		Wrike	0%
		Trello	40%
		Axosoft	0%
		Planbox	10%
		Zoho Projects	10%
		Teamwork Projects	33.3%
		Other…	3.3%
USER-EXPERIENCE DESIGN			
QN 17	Do you usually start to collect your requirement and functionality using a Whiteboard and pencil and paper?	Yes, frequently	63.7%
		No	6.1%
		Sometimes	30.3%
QN 18	If you usually draw a wireframe presenting the functionality and data, which tools are you using to draw your wireframe?	Whiteboard	43.3%
		Pencil and paper	46.7%
		Balsamiq	10%
		Sketch	23.3%

		UXPin Mockplus	0% 10%
		1. Invision	23.3%
		Other	3.3%
QN 19	Do you usually create workflows that present the pathways users can travel within the app?	Yes, frequently	52.9%
		No	8.8%
		Sometimes	38.2%
QN 20	Do you usually do iterations between wireframe and workflows to improve the design?	Yes, frequently	45.5%
		No	18.2%
		Sometimes	36.4%
QN 21	Do you usually test your wireframe and workflow on a tappable click-throw UX prototype? (We are not talking about UI Prototype- this will come later)	Yes, frequently	24.2%
		No	30.3%
		Sometimes	45.5%
QN 22	If yes; which tools do you use to make a tappable UX prototype?	Whiteboard	-
		Pencil and paper	3.7%
		Balsamiq	11.1%
		Sketch	48.1%
		UXPin	3.7%
		Mockplus	18.5%
		Invision	22.2%
		Other…	3.7%
USER-INTERFACE DESIGN			
QN 23	Which tools usually do you use to move from wireframe to UI Design elements?	Whiteboard	28.1%
		Pencil and paper	31.3%
		Balsamiq	9.4%
		Sketch	37.5%
		None	12.5%
		Other…	3.1%

QN 24	Which tools usually do you use to make a tappable UI prototype?	Balsamiq	29.6%
		Mockplus	18.5%
		2. Invision	3. 29.6%
		Other…	3.7%
QN 25	When you finish all UI screens, do you usually test again click-through "workflow- model to be sure that it still works and have correct data-flow?	Yes, frequently	57.5%
		No	9.1%
		Sometimes	33.3%
QN 26	When you finish UI design, do you usually ask for user feedback?	Yes, frequently	61.8%
		No	5.9%
		Sometimes	32.4%
QN 27	Do you usually use any tools to ensure a smooth transition from design to the development process? Which?	Sometimes	27.3%
		Zeplin	6.1%
		Sketch	27.4%
		Photoshop	42.4%
		None	18.2%
		Other…	3%
QN 28	Do you agree not to move to the development phase until you get mostly full approval from the customer that this is what he wants and needs?	Absolutely agree	43.7%
		Agree	18.8%
		Usually agree	28.1%
		Disagree	9.4%
DESIGN TECHNICAL DECISIONS			
QN 29	Which approaches do you usually use to build your app?	Native platform	60.6%
		Cross-platform hybrid	36.4%
		Web technology	33.3%
QN 30	Which languages do you usually use to build your Web API?	Java	56.3%
		C	3.1%
		C++	6.3%
		C#	34.4%

		Go-lang	0%
		Javascript	46.9%
		PHP	43.8%
		Python	9.4%
		Ruby	0%
		Haskell	0%
		Go	3.1%
		Other…	3.1%
QN 31	Do you usually use SQL for mobile apps databases?	Yes, frequently	35.3%
		No	11.8%
		Sometimes	52.9%
QN 32	Which language or tool do you normally use when you are developing a web platform?	HTML	45.5%
		Cordova	15.2%
		Javascript	45.5%
		Phonegap	3%
		Ionic	30.3%
		JQuery	24.2%
		Intel XDK	0%
		I have never developed a web-based platform	6.1%
		CSS	30.3%
		Other…	3%
QN 33	Do you share the importance of choosing the hosting environment with the customer, and how is it essential in performance and scalability?	Yes, frequently	51.5%
		No	15.2%
		Sometimes	30.3%
QN 34	Which hosting providers do you usually use for your APIs and Databases?	Amazon AWS	31%
		Rackspace	3.4%
		Google-Cloud	44.8%
		Other…	3.4%
DEVELOPMENT			

QN 35	When you start the development process, do you use an iteration model "Sprints- in agile methodology?	Yes, frequently	50%
		No	14.7%
		Sometimes	35.3%
QN 36	When you start planning for the Sprint, do you focus on the tasks to be implemented during this iteration and estimate the time needed to finish this task?	Yes, frequently	58.9%
		No	11.8%
		Sometimes	26.5%
QN 37	Do you try to reuse code throw the development process?	Yes, frequently	64.7%
		No	8.8%
		Sometimes	23.5%
QN 38	When you complete a Sprint, do you send back the results to your project manager or quality assurance for review?	Yes, frequently	67.7%
		No	14.7%
		Sometimes	14.7%
QN 39	During development, do you usually use tools or platforms (Like Hockey app) to share the reviews or testing with other developers or with the client? What are these tools?	Yes, frequently	6.1%
		No	48.5%
		Sometimes	45.5%
TESTING			
QN 40	In functional testing, have the testing team have a test plan and list of actions to check?	Yes, frequently	66.7%
		No	6.1%
		Sometimes	27.3%
QN 41	Which one of these testing do you use to test app features?	User-friendly – Usability testing	77.4%
		Responsiveness and its performance – performance testing-	54.8%
		Regression testing- testing previous Sprints-	38.7%
		User acceptance testing	67.7%
		None	6.5%

		Other…	-
QN 42	Do app designers review each feature to be sure that their vision was implemented as described in the design phase?	Yes, frequently	47%
		No	2.9%
		Sometimes	50%
QN 43	If you are using automated specific device testing, which tools you are using?	Amazon AWS device farm	12.5%
		Native tools	25%
		Google-Firebase	28.1%
		I don't use	53.1%
		Other…	3.1%
DEPLOYMENT AND MONITORING			
QN 44	After deploying your app, do you monitor app store ratings and reviews?	Yes, frequently	58.8%
		No	8.8%
		Sometimes	29.4%
QN 45	Which one of these tools do you use to help you in monitoring your app?	Sentry - for app crashing	6.5%
		HockeyApp for app crashing	6.5%
		Facebook Analytics for app Analytics	19.4%
		Apptentive for app Analytics	3.2%
		Google Analytics for app Analytics	6.2%
		Appsee for app Performance	6.5%
		Prometheus for app Performance	6.5%
		None	-
		Other…	-

QN 46	Finally, can you write the time needed in every stage in the mobile development process in percentage? 1-Planning and strategy. 2-User experience and user interface design. 3-Development. 4-Testing. 5-Deployment and monitoring. % Example of answer: " 1: 20%, 2: 30%, 3: 30%, …		

APPENDIX B

In this section e will present our survey that we used in our research to validate our proposed methodology, Agile Beeswax, that was covered in chapter 6.

QN	`	POSSIBLE ANSWERS	% EACH ANSWER
Section 1: About the participants/organisation			
QN 1	Gender	Mail	
		Female	
QN 2	Age	20-34	
		35-44	
		45-54	
		55-64	
		> 64	
QN 3	I identify my work in	Company	
		Academic	
QN 4	I'm a specialist in	Designer UX/XI Designer	
		Mobile app developer	
		Tech Lead/Manage	
		Top management level	
		Marketing	
		Software Engineer	
		Software Tester	
		Other…	
About your organisation			
QN 5	Does your organisation develop mobile apps?	Yes, frequently	
		No, never	
		Sometimes	
QN 6	The number of main apps that have been developed by your organisation in the last	Never	
		Less than 5 apps	

	five years are	5 to 10 apps	
		11 to 20 apps	
		More than 21 apps	
QN 7	What is the staff size of the mobile development team in your organisation?	Less than 5 employees	
		5 to 10 employees	
		More than 10 employees	
QN 8	Does your organisation usually use agile methodologies in mobile apps development process?	Yes, frequently	
		No	
		Sometimes	
Section 2: Agile Beeswax Process (Phases)			
According to the Agile Beeswax phases that we explained before, you should indicate his/her grade of agree with each sentence.			
QN 9	I believe that Agile Beeswax phases cover all the steps and phases of mobile application development.	Strongly disagree	0%
		Disagree	3%
		Neutral	23%
		Agree	34%
		Strongly agree	40%
QN 10	I believe that mobile application development phases presented at Agile Beeswax are easy to understand.	Strongly disagree	0%
		Disagree	3%
		Neutral	17%
		Agree	31%
		Strongly agree	49%
QN 11	I believe that none of the mentioned application development phases can be eliminated or waived if we want a high-quality mobile application.	Strongly disagree	0%
		Disagree	3%
		Neutral	9%
		Agree	49%
		Strongly agree	40%
QN 12	I believe that the order and sequence of phases correspond to the mobile application development methods	Strongly disagree	0%
		Disagree	9%
		Neutral	9%
		Agree	43%

		Strongly agree	40%
Section 3: Agile Beeswax Practices			
According to the Agile Beeswax practices that we explained before, you should indicate his/her grade of agree with each sentence.			
QN 13	I believe that by mixing the main practices from management (Agile and Scrum), technical and software engineering practices, and operational practices, it is more possible to handle the main apps development process.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
I believe that to develop a mobile application, Agile and Scrum Management practices are required. These are some practices.			
Agile and Scrum Management practices			
QN 14	Working in short iterations in the design and development phases,	Strongly disagree	0%
		Disagree	3%
		Neutral	17%
		Agree	20%
		Strongly agree	60%
QN 15	Fixed meetings	Strongly disagree	0%
		Disagree	0%
		Neutral	11%
		Agree	40%
		Strongly agree	49%
QN 16	Sprint review	Strongly disagree	%
		Disagree	%
		Neutral	%
		Agree	%
		Strongly agree	%
QN 17	Product Backlog	Strongly disagree	%
		Disagree	%
		Neutral	%
		Agree	%
		Strongly agree	%
QN 18	Small releases,	Strongly disagree	%
		Disagree	%

		Neutral	%
		Agree	%
		Strongly agree	%
QN 19	Incremental design and development,	Strongly disagree	%
		Disagree	%
		Neutral	%
		Agree	%
		Strongly agree	%
QN 20	User centre design	Strongly disagree	%
		Disagree	%
		Neutral	%
		Agree	%
		Strongly agree	%
QN 21	Delivery as-fast-as-possible	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 22	Team empowerment	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 23	Continuous integration	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
Software Engineering practices			
I believe that to develop a mobile application, Technical and Software Engineering practices are required. These are some practices.			
QN 24	Test-driven development	Strongly disagree	
		Disagree	
		Neutral	
		Agree	

		Strongly agree	
QN 25	Design improvement	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 26	Coding standards	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 27	Control of design to development handoff	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 28	Continuous testing,	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 29	Regression testing,	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
DESIGN TECHNICAL DECISIONS			
I believe that to develop a mobile application Operational Practices are required. These are some practices.			
QN 30	Simple design	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 31	Eliminate waste	Strongly disagree	
		Disagree	
		Neutral	
		Agree	

		Strongly agree	
QN 32	Amplification of learning	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 33	Continuous learning and improvement	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 34	Accept changes in iteration at any time	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 35	Planning for unplanned work	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 36	Does mobile application development need other practices? May indicate which are?		
Section 4: Agile Beeswax Rules			
In the next questions, the main rules of the Agile Beeswax are established. Please, you should indicate the grade to which you agree with each rule.			
QN 37	I believe that the first design iteration must satisfy basic user needs.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 38	I believe that once the design Sprint has been completed and confirmed, it should be sent to the development phase (crossing the bridge, design to development handoff and technical	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	

	decisions)		
QN 39	I believe that the Product Owner must split the requirements into smaller user stories for each application screen in discussion with customers and other stakeholders.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 40	I believe that after writing the user stories (by the Product Owner) as a Scrum Product Backlog, a prioritization session with the architect and some developers must be initiated.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 41	I believe that testing should be performed regularly, as it will significantly reduce the financial costs incurred at each stage.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 42	I believe that Agile Beeswax certainly favors the rapid development of mobile applications.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
Section 5: Team Dynamics			
According to the Agile Beeswax team structure, you should indicate his/her grade of agree with each sentence.			
QN 43	I believe that with the using of Agile Beeswax, you will have good communication between project members.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 44	I believe that with the using of Agile Beeswax will improve team decisions	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 45	I believe that with the using of Agile Beeswax will enhance the team organizing	Strongly disagree	
		Disagree	
		Neutral	

		Agree	
		Strongly agree	
Section 6: Adaptation to changing requirements			
QN 46	I believe that using wireframing design in the early stages will help in understanding the application's requirements.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 47	I believe that designing the application workflow in the early stages will help in understanding the application's requirements.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 48	I believe that with Agile Beeswax, it will be easier to deal with changing requirements.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 49	I believe that dividing the app project to Sprint backlog (divide a task into smaller ones) and prioritizing it, will help manage core app requirements	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 50	I believe that with Agile Beeswax, most changes in requirements occur in the early stages of the development process.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 51	I believe that with Agile Beeswax, developers are not very concerned about changing customer needs.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
Section 7: Learning and Motivation			
QN 52	I believe that it is important to learn from each Sprint in the development process for the next Sprints or another project.	Strongly disagree	
		Disagree	
		Neutral	

	We can discuss that in the Sprint Review	Agree	
		Strongly agree	
QN 53	Agile Beeswax believes that keeping up with the latest trends is an integral part of any software specialist's job,	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 54	Using Agile Beeswax will motivate me during the development process.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
Section 8: Agile Beeswax Applicability and Limitations			
QN 55	I believe that Agile Beeswax is not suitable for all types of mobile apps.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 56	I believe that Agile Beeswax can be suitable for commercial categories, but it is not suitable for gaming educational apps (for instance). you can add what do you think it is suitable for.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 57	I believe that since the Agile Beeswax methodology is flexible, anyone who adopts it can choose what is the best practices for them to develop the mobile application.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 58	Adopting just the Agile management method is not enough to develop a mobile application, we need more developments practices	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
Section 9: Agile Beeswax Adoption			
QN 59	I believe that Agile Beeswax will be helpful for your job	Strongly disagree	
		Disagree	

		Neutral	
		Agree	
		Strongly agree	
QN 60	I believe that I would like my company to carry on adopting the Agile Beeswax methodology.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 61	I believe that the adoption of Agile Beeswax is not needed many efforts.	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	
QN 62	I believe that Agile beeswax is easy to adopt	Strongly disagree	
		Disagree	
		Neutral	
		Agree	
		Strongly agree	