sustainability

MDPI

Article

Agile Beeswax: Mobile App Development Process and Empirical Study in Real Environment

Hazem Abdelkarim Alrabaiah * and Nuria Medina-Medina

Department of Languages and Computer Systems, University of Granada, 18010 Granada, Spain; nmedina@ugr.es

* Correspondence: hazem@correo.ugr.es

Abstract: Mobile application development is a highly competitive environment; agile methodologies can enable teams to provide value faster, with higher quality and predictability, and a better attitude to deal with the continuous changes that will arise in the mobile context application (App), and the positive impact of that on sustainable development through continuous progress. App development is different from other types of software. For this reason, our objective is to present a new agile-based methodology for app development that we call Agile Beeswax. Agile Beeswax is conceived after identifying the mobile development process's issues and challenges, and unique requirements. 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. 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. One of its main strengths is that it has been created with academic and business perspectives to bring these two communities closer. To achieve this purpose, our research methodology comprises four main phases: Phase 1: Extensive literature review of mobile development methodologies, Phase 2: Interviews with mobile application developers working in small to medium software companies, Phase 3: Survey to extract valuable knowledge about mobile development (which was carefully designed based on the results of the first and the second phases), and Phase 4: Proposal of a new methodology for the agile development of mobile applications. With the aim of integrating both perspectives, the survey was answered by a sample of 35 experts, including academics and developers. Interesting results have been collected and discussed in this paper (on issues such as the development process, the tools used during this process, and the general issues and challenges they encountered), laying the foundations of the methodology Agile Beeswax proposed to develop mobile apps. Our results and the proposed methodology are intended to serve as support for mobile application developers.

Keywords: agile methodology; mobile application development process; mobile application issues and challenges; process improvement; software engineering; sustainable software development; survey

Citation: Alrabaiah, H.A.; Medina-Medina, N. Agile Beeswax: Mobile App Development Process and Empirical Study in Real Environment. *Sustainability* **2021**, *13*, 1909. https://doi.org/10.3390/su13041909

Academic Editor: Alok Mishra

Received: 2 January 2021

Accepted: 7 February 2021

Published: 10 February 2021

1. Introduction

Smartphones, tablets, and apps have become words and technologies central to our lives. Mobile applications (apps) have become the center of attention for everyone, given the favorite features and opportunities smartphones offer. The availability of high-speed internet combined with the remarkably fast, easily accessible interface in a smartphone has increased the convenience apps bring. We have apps for advertising, education, communication, shopping, cooking, and much more, all at our fingertips; there is an urgent need to develop apps.

According to statista.com [1], users downloaded 28.3 billion, 8.2 million apps from Google Play and Apple App Store, respectively, in the third quarter of 2020. In contrast, in 2019, the number of mobile app downloads worldwide was 204 billion, and the global mobile app revenue was 462 billion dollars. The number of apps available in the Google Play

Store as of Q4 2019 was 2.57 million, and in the Apple App Store 1.84 million. Furthermore, statistica.com projects that the total number of app downloads will increase to 258 billion by 2022. The increase in apps has led to an increase in the growth of developers and app development companies.

Our objective in this work is to propose an agile methodology for the mobile application development process. We intend that the methodology integrates expert knowledge from an academic and industrial point of view. To do this, first, a review of the scientific literature is carried out where the methodologies proposed by other researchers are studied, at the same time that some expert developers are interviewed to analyze their mobile development processes. With everything learned, a survey is designed and conducted to extract information more specific and complete from experts both in industry and the academic community. Finally, a new methodology for mobile development is proposed, which tries to transfer the knowledge obtained from these two contexts.

Consequently, this paper describes two studies performed in the field of mobile development. On the one hand, our literature review reveals interesting mobile methodologies. However, it also shows the existence of very few works on app development methodology based on practical research in real environments—that is, involving developers. This lack of information implies an insufficient understanding of the development process that developers adopt in companies and of the tools and challenges they face during this process. Challenges include user experience and user interface, time to market, planning, requirements, development skills, complication testing, and intense competition with app competitors.

On the other hand, the paper proposes and analyzes a questionnaire designed to be administered to specialists in mobile application development and researchers and academics in the same field. The questionnaire is focused on the adopted development process, the tools used during the development process, the main issues and challenges involved in the process, and the valuable experts' opinions and recommendations. The results of the experience with experts are shown and discussed in the paper. Finally, the knowledge gathered from these two studies is integrated to form the basis of the new agile methodology to develop mobile applications, which we have called Agile Beeswax.

Therefore, the two main research contributions of this work are: (1) an in-depth analysis of the existing mobile development methodologies, which has been carried out using theoretical and empirical methods and including specialists in mobile application development and researchers in this field during that process; and (2) the proposal of the methodology Agile Beeswax for the agile development of mobile applications. An important innovation factor of this work is the intersection of the academic and business worlds in the definition of this new methodology designed from previous experience in both worlds and the combination of the three main practices: management agile practices, technical engineering practices, and operational practices.

The rest of the paper is structured as follows: Section 2 discusses the literature review and related research. Section 3 describes the study design and description of our research methodology, data collection, and the questionnaire description. Section 4 discusses the questioner information and data collected. Section 5 discusses mobile application development. Section 6 presents the proposed mobile app development methodology. Section 7 concludes with recommendations for future work.

2. Literature Review and Related Work

2.1. Background

First published in 2001, agile methodologies for software development came into existence to accommodate changing business requirements and to manage the challenges facing software developers. These methods rely on practitioners' experience and many developmental practices with a focus on early delivery of quality software. Agile software development (ASD) reinforces iteration, development, adaptability, and collaboration throughout the development process. "Agile" means continuous integration, simple design,

working software over comprehensive documentation, and customer collaboration over contract negotiation [2].

Agile methodology is not a straightforward linear waterfall model; it takes an iterative approach to software development. Agile projects consist of several smaller cycles (sprints in the Scrum model). Sprint is one timeboxed iteration in four weeks or less of a continuous development cycle. Agile development enables the development life cycle to adopt changes more quickly, minimizing risk. Its customer-centric focus helps to respond to changes during software development through its iterative approach. The agile development methodology enables companies continually to redesign their releases to improve their value throughout the development process, making their apps competitive on the market. The agile methodology's main objective is to focus on customers' needs to deliver customers' requirements on time during the development cycle. According to Flora et al. [3], the continuous interaction between the development team and customers makes the process more flexible and transparent.

In turn, mobile software development is the set of processes and procedures used in developing software programs for mobile devices such as smartphones and tablets (mobile apps). Mobile app development differs from traditional software development for personal computers. Because mobile devices are battery-powered, smartphones have poor, unstable connectivity and less energy. They also have small screens, many different screen sizes, and central processing unit (CPU) and memory limitations.

Wasserman et al. [4] summarized how mobile app development differs from traditional programs' development: interaction with other apps or embedded apps; and the combination of native and web apps. Mobile apps have very complex tests and problems of transmission through gateways and phone networks. Some apps may have restrictions from telecom providers. Mobile app development is now moving toward cloud computing, which affects development processes. Stressing the difference between a mobile device and a desktop PC, Kumar et al. [5] identify the greater challenges and concerns that face app developers, such as compatibility with various platforms, the incongruity of hardware utilities, total cost, and scheduled time. Kumar also discusses the importance of user convenience, front-end design, and restructuring for greater usability, use of the screen, design pattern, and development of worthy apps.

Agile methods are expected to overcome some of the limitations and constraints in the apps development process. Because they enable lighter and faster development, agile methodologies are a natural fit for mobile app development, as Mahmud et al. [6] note.

2.2. Related Work

Agile methods can thus be adjusted as needed to develop a mobile app. To this aim, researchers in the agile development of mobile apps have proposed a variety of agile methodologies to match these challenges and special requirements. Some of them are reviewed below.

Mobile-D: In 2004, Abrahamsson and his team [7] presented Mobile-D as a development methodology for mobile apps. This methodology was developed in the VTT Technical Research Center in Finland, with the collaboration of three companies that develop mobile software products and services as part of the AGILE-ITEA project. It is an agile development approach confined to developing practices (Programming XP), life cycle coverage (rational unified process (RUP), and Scalability (Crystal methodology). Mobile-D was adjusted for small teams working on a short development cycle and seeking to deliver a functional mobile app within 10 weeks. The methodology has five phases composed of different tasks, stages, and practices. The exploration phase plans and establishes the project. The initialization phase verifies all critical issues during the development process. The production phase implements the product requirements by applying an iterative, incremental development cycle. The stabilization phase checks the project quality. Finally, the test and fix phases provide user feedback, after which the development team corrects any defects found.

Mobile application software agile methodology (MASAM): YJ Jeong introduced MASAM [8] in 2008 for developing apps that run on a mobile platform. This method is based on agile methodology with special properties. According to YJ Jeong, MASAM's design supports small mobile development teams; its approach is agile for the rapid development process in four phases. The preparation phase summarizes the project and assigns roles, responsibilities, and set-up. The embodiment phase seeks to understand user needs and defines the software product architecture. The product development phase includes an iterative extreme programming development sequence, carried out through test-driven development with continuous integration. Finally, the commercialization phase integrates product launch and sales.

Hybrid methodology design process: in 2008, Rahimian and Ramsin et al. [9] presented this new approach. Motivated by the belief that no single process fits all situations, they merged agile methodology and information technology (IT) principles based on a combination of agile methodologies: ASD and new product development (NPD). Methodology engineering aims to construct new methods from existing ones, and this hybrid process emphasizes "the design, construct, and adapt methods, techniques, and tools for the development of information systems" [10]. Devised as a top-down, iterative, incremental process, the hybrid methodology design process contains the following tasks. At the end of each iteration, it prioritizes the requirements and selects the design approach, application of the selected design approach to defining the methodology, revision and refinement of the methodology, description of the abstraction level for the next iteration, and revision and enhancement of the requirement, in turn prioritizing these for the next iteration.

The hybrid methodology design process has four iterations. The first uses a generic software development life cycle (SDLC), adding practices commonly found in agile methodologies. The second includes activities from NPD. In the third iteration, new practices and activities are integrated into ASD. The fourth and final iteration adds prototyping to decrease the expected technology-related risks.

Scrum for the development of mobile applications: in 2010, Scharff and Verma [11] published a study that put Scrum into practice as a development process and method for use in mobile app development. In a case study at Pace University, these authors defined a model for working with Scrum in a classroom with a group of students, a professional certified Scrum Master in the software industry, and a real product owner to provide the requirements. The aim was to develop a mobile app with which waiters could manage orders and bills more efficiently in a restaurant in Senegal.

Scrum Lean Six Sigma (SLeSS): In 2011, Cunha et al. [12] proposed SLeSS as integration of Scrum and Lean Six Sigma for mobile apps. To achieve performance and quality, LSS should be implemented as a quality framework with teams already using Scrum in their development process. SLeSS supports adaptation to change requirements in the earlier stages of the project life cycle and delivers versions more rapidly with fewer failures. SLeSS is above all an incremental approach. After executing Scrum as a development methodology, LSS should be implemented as a quality framework. The SLeSS approach has five phases: DMAIC (define, measure, analyze, improve, and control). Based on a real-world project, SLeSS was used in a practical environment with real mobile app software development in the P&D laboratory. The customer was a cell phone manufacturer with a team of 12 developers working for six months. The project achieved a functional product within the required working hours.

MADeM: In 2016, Alsabi and Dahanayake [13] proposed MADeM. Based on Mobile-D methodology [7], 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 Unified Modeling Language (UML), modelling approaches, activity diagrams, class diagrams, use case diagrams, and user view diagrams.

Mobile Ilities: Danielo Martinez et al. [14] in March 2020 presented a framework called Mobile Ilities for mobile app development based on agile and Scrum methodologies that consider the specific needs, characteristics, and challenges of mobile development. Mobile Ilities propose elements such as connectivity, security, platform, flexibility, and energy. The authors applied this agile Scrum framework as work assignments for computer science students and assessed the results as positive, potentially providing a guide for novice developers during the mobile app development process. It is difficult to generalize these results, however, especially because the framework was applied in an academic environment and the app created was limited to one university. Furthermore, the developers were unfamiliar with the iterative approach in general, had no interaction with any client, and performed no user testing.

2.3. Literature Review

The reviewed methodologies are impressive, there is an important gap in terms of 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 propose a new methodology to develop these mobile applications, these studies helped us in developing our methodology, which will be presented later in this paper. We show some of them.

Flora et al. [3] (2014) presented a survey related to improving the mobile application development process. The main participants were the mobile app development team members, Agile experts, and researchers. This survey considered the use of various agile approaches for successful mobile app development, Agile approaches such as XP, Scrum, and Lean. Furthermore, the survey was performed to determine what appropriate Agile practices are currently being used in mobile app development, practices such as development in iterations and short development cycles, customer involvement, and if the team can adapt to changes. The study's findings show that Agile naturally fits the needs of mobile application development and how these Agile methods have the chance to enhance the speed and quality of mobile application development.

Kirmani M. [15] (2017), in his investigation, indicates that agile methods are suitable for the development of mobile applications. The work's issues were: agile methods are suitable for fast-paced markets, customer's satisfaction is important, frequent delivery, the scope for changes, the delivery cycle is short, there is an appropriate collaboration between businesses and developers, and where a good design and simplicity. This study found that 86% of survey participants thought agile methods and practices are suitable for app development. This study shows adopting these agile methods are applicable to enhance speed and quality of app development.

Arshad et al. [16] (2018) investigated the main challenges during the development process of native, web or cross-platform hybrid mobile applications in an empirical study on the mobile application. They identify challenges via a systematic literature review. As a second phase, they try to identify more challenges by collecting data from mobile application developers using structured interviews to enable practitioners to specify additional challenges. This systematic literature review showed that fragmentation, testing, user experience, compatibility, and change management are the most common challenges in mobile application development. The identified challenges considered as critical by the participants in the interviews were fragmentation, testing, reuse of code, lack of tools support, lack of expertise, and change management. Further challenges were found through interviews, which were not found in the literature review, for example, lack of training, lack of teamness and enrollment, lack of communication, and lack of knowledge management. Authors conclude that to be successful and an effective application, these challenges and problems should be considered. In their opinion, identifying these challenges will assist teams in the development of mobile applications. They recommend that practitioners should give more attention to the challenges identified in academia and industry.

According to our review of related works, we can indicate that the analyzed works contribute to the problem of mobile development but are very specific and/or partial. None considers how mobile apps are being developed in the industry during the design and definition of the phases of a new methodology (the main objective of our work). Since mobile apps are becoming more complex and critical, companies and researchers should adopt development processes that address more aspects of the process covered by today's agile methodologies. Such methods are necessary to provide a fully structured understanding of the development process [12] with extended documentation. In response to this need, we have conducted a study to improve understanding of the mobile app development process in the industrial market to assess how similar they are and determine the similarities and differences between industrial processes and academic proposals. Our goal is to provide a single methodology that integrates expert knowledge from both the academic and industrial communities. We need to integrate and research other agile methods that can be developed or improved to make them compatible with the development of mobile apps. More importantly, we must determine how research can be linked to and integrated into the real development process adapted by professionals and researchers. We believe that our study will lead to an increase in scientific research on methodologies to develop mobile apps, especially on integrating agile methods available with practices learned from development companies and app experts.

3. Materials and Methods

In this study, we deepened our understanding of the mobile app development process in the industrial market and the academic community. We outline the main development process followed by some mobile app companies and researchers in the field to determine the aspects and perspectives shared by researchers and academics.

3.1. Methodology

The research methodology comprises four main phases (see Figure 1): Phase 1: Extensive literature review, Phase 2: Interviews, Phase 3: Survey, and Phase 4: Proposal of a new methodology for mobile app development.

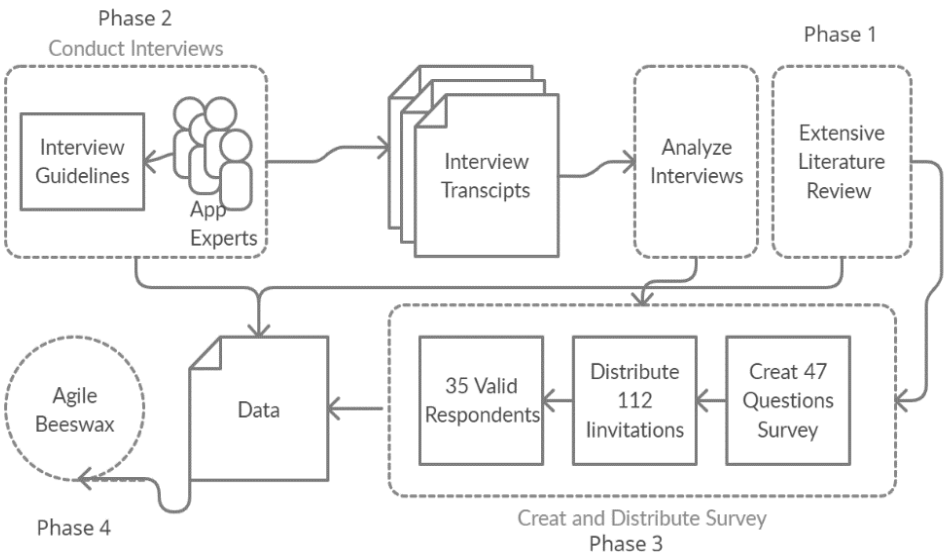

Figure 1. The applied research methodology.

The main results of phase 1 have been summarized in the section Literature Review and Related Work (Section 2). Then, the initial interviews (phase 2) were performed using a simple guideline and we obtained very useful information from five companies of mobile development (as will be described in Section 4.1). In order to design these informal interviews, based on our study goals, we began with an in-depth analysis of the mobile app

development methods recorded in previous research, such as in [15,16]. These interviews with specialists who develop mobile apps allowed us to achieve a deeper understanding of the mobile app development process, issues, and challenges [17].

After the initial interviews, we performed a more extensive literature review (phase 1 again) with the aim of maintaining a balance between the proposals made by researchers in the academic world and the everyday reality in mobile development companies. Based on the knowledge obtained in phases 1 and 2, an extensive online questionnaire was designed for the process of developing mobile applications (phase 3), (a full questionnaire with the answers is available in Appendix A). After completing the questionnaire design based on literature reviews and interviews with experts, an interview was conducted with software developers and academics researcher to test the validity of the content and the construct of our questionnaire; it seeks to cover all phases and objectives in the mobile app development process. Thus, phases 1, 2 and 3 complete a comprehensive study (theoretical and practical) about the development of mobile applications. Phase 3 will be detailed in Sections 3–5.

The main contributors to our survey were software, mobile app companies, mobile experts, researchers, and academics. The conducted survey is explanatory and exploratory, and very important conclusions have been extracted from the surveyed experts (detailed in the following sections), constituting the basis for the design of a new methodology, Agile Beeswax, presented as a result in this paper (phase 4 in Section 6).

3.2. Data Collection

We surveyed many specialists in mobile software development, researchers, and academics in the field of software development. The questionnaire respondents were also experts in the mobile application development process, academics, and researchers in this field. We chose the participants carefully; some interviewees were questionnaire respondents. We went to the respondents' workplaces, contacted them by phone, or sent an email or paid a visit to motivate them to complete the questionnaire. These individuals were invited to participate voluntarily in the survey. Finally, the survey was a combination of closed and partially open-ended questions. We began analyzing data by studying each response individually, analyzing first the academics' and company experts' responses separately and then all responses together.

3.3. Sections of the Questionnaire Described

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 (A full questionnaire with the answers is available in Appendix A).

The questionnaire contained 46 questions, divided into three main sections as follows:

- The first section requested information about the participants, such as specialty, country of origin, and age.
- The second section involved the participant's organization, including information such as whether or not the organization developed apps, size of the company or organization, number of apps developed in the organization, whether or not the organization used agile methods in app development, and whether or not it believed that agile systems are ideal for app development.
- The third and main section of the questionnaire, which asked about the app development process, consists of seven subsections, with 5–6 questions in each section, as follows:

  (1) The idea and strategy: the focus of this section was how the app idea originates, the tools used to develop the idea and manage the team, whether the organization conducted marketing research on the competitor and user ratings and reviews, and whether they were developing a roadmap and target group for the app.

(2) User experience design: this section asked about whether the organization was implementing wireframe, workflow, or a clickable user experience (UX) prototype, and whether it performed iterations to improve the UX. This section also asks about the tools used in this stage and about user feedback.
(3) User interface (UI) design: we asked how the developers move from the wireframe to the UI mockup 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 asked about choosing the host environment, how this environment affected performance and scalability, what tools were used to create the web API (Application Programming Interfaces), whether Structured Query Language (SQL) was used, what tools were used to develop a web technology platform, and which approaches were used to build the app—native or cross-platform hybrid or web technology.
(5) Development: we asked about reuse of code throughout the development and team management processes, as well as user integration into the process, and whether developers used specific tools to help them integrate users. We focused on agile principles, for example, whether the developers used the Sprint in Agile methodology. We also asked about sprint planning and the tasks to be performed during a sprint 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, regression testing, performance testing, and user acceptance testing. We also asked if the designers reviewed the testing process to make sure their vision was implemented.
(7) Deployment and monitoring: this section asked if the developers performed monitoring for their apps, which tools they used, whether they monitored app store ratings and reviews, and how these ratings could affect updates?

4. Results

More than 110 invitations to complete the questionnaire were sent, approximately 40 responses were received, and five responses were excluded. The full questionnaire is available in Appendix A. In this section, we will present the main results and founding of our research. This section contains four subsections. Section 4.1 discusses previous work that is part of our study. Section 4.2 will talk about general information about the participants. Section 4.3 will discuss the development process adopted by experts and in Section 4.4 the development tools used in the development process.

4.1. Our Previous Work

At the end of 2018, to better understand the mobile application development processes adopted in real companies and the issues and challenges related to it, we interviewed five mobile application development experts from five different companies. Our study was focused on the development process they usually follow, and the tools used in this process, as well as general issues and challenges. In [17], we published the main process adopted for each company, followed by the main conclusions learned in the mobile app development process.

Table 1 [17] summarizes the professional characteristics of the interviewees. In these interviews, we ask the candidate persons to participate in a semi-structured interview. The interviewees were experts in mobile app development. We interviewed five experts (ID1 to ID5) from five different companies. Each interview session took about 25 to 40 min. The

interviewees have 5 to 7 years an average experience in mobile apps, with the average size of the teams was 11 to 50 persons [17]. These interviews focused on the development process that had been followed, and the tools used in this process, as well as general issues and challenges during the development process. The interviewer used a list of questions and key points to be covered through the interviews: for example, (a) what is the method followed in developing mobile applications? From ideas and requirements gathering to the end of the project. (b) What are the tools used in each stage of mobile application development? (c) What are the obstacles and problems facing you during the development process?

Table 1. The first round of interviewees [18].

Interviews Summery	ID1	ID2	ID3	ID4	ID5
Role	Project Manager	Project Manager	Project Manager	Project Manager	App Developer
Software experience years	11–20	6–10	11–20	6–10	11–20
Mobile Apps Experience	6–10	5	5	4	6–10
Organization Size	21–50	21–50	11–20	11–20	6–10
Number Of Apps Developed	5–20	5–20	5–20	5–20	20–30

During this work, we noted that the main phases on mobile apps development are: prototyping, design (user experience, user interface), development, testing, and launch and deployment. In addition, our little study revealed that all companies had the development phase as a core focus and an essential stage of mobile application development. In this study, another point in common between the five software companies was that obtaining the customer's requirements is the first challenge the developers face at the beginning of the project. One step further, it was possible to notice that all companies were working to integrate the customer, making them part of the development cycle, keep them updated and take timely notes to get together to the desired mobile application. Most interviewed experts indicated that they prefer not to go to the graphic design phase until most of the customer's requirements are taken. Finally, another interesting result was that all companies that use native tools to develop mobile applications were preferred to perform testing through the same native tools' simulators. The knowledge gained from this previous work was exploited to design the survey conducted in this work.

4.2. General Information about the Participants

In this section, we will present and argue the main results and findings of our research, some of these data will be represented graphically. Although the results obtained in each question of the survey can be consulted in Appendix A. Our research participants comprised experts at companies (80%) and academics 20% (see Table 2 for more information).

Table 2. 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%) eXtreme Programing (XP) (13%) Kanban (13%)

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%). However, 90% of experts worked in companies that developed apps (see Figure 2); 40% of participants had an experience, having developed more than 10 apps; 17% had developed more than 20 apps. Fewer than 51% of participants worked at a small organization and about 40% at a medium-sized one (Figure 3). Table 2 represents the participants' information that includes the demographic characteristic, roles of participants, organization size, and methodology and tools usage.

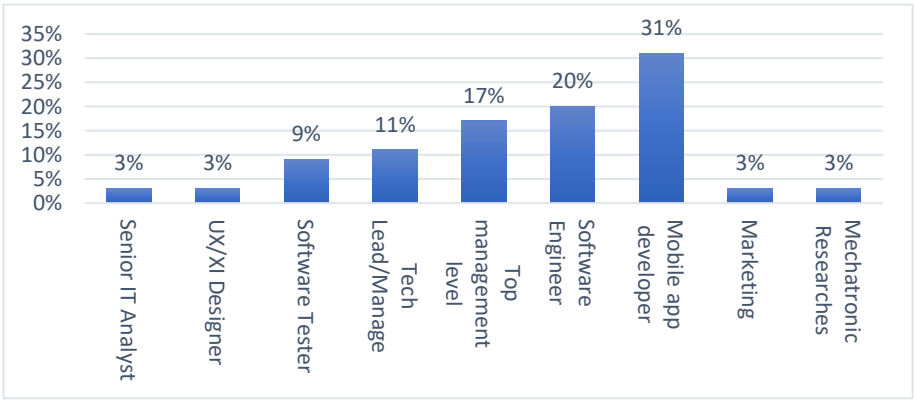

Figure 2. Participants Work.

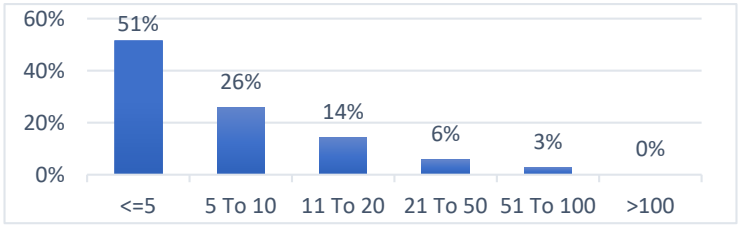

Figure 3. Organization size.

In terms of tools and methodology, nearly 43% of participants used JIRA as a team management tool, 40% used Trello, and 33% used Teamwork Project. Also, approximately 84% have used agile methodologies to develop mobile apps (Figure 4a), 56% used Scrum, about 13% eXtreme Programing (XP), and about 13% Kanban methodologies in the development process (Figure 4b). Nevertheless, 61% of participants believed that agile was the best method for app development; about 36% said this was sometimes the case.

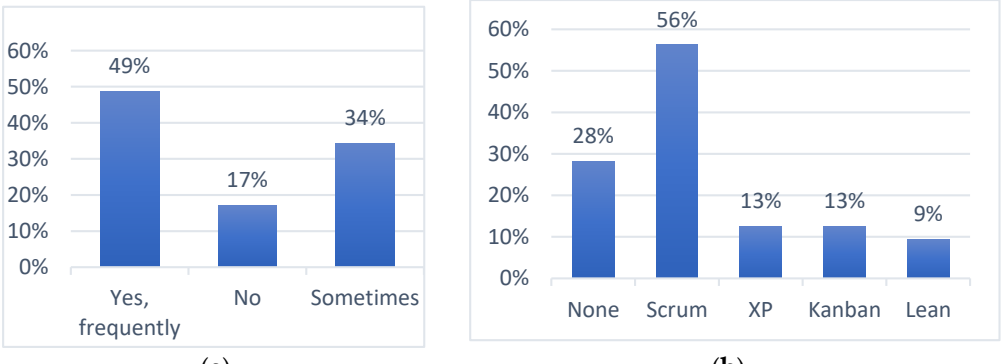

Figure 4. Adopting agile approach in the organization. (**a**): using agile in the development process; (**b**): agile approaches in the organization.

4.3. Development Process

This study identified six main phases in participants' mobile app development lifecycle. These phases were: idea and strategy, UX design, UI design, design to development, handoff—issues and decisions, development, and deployment and monitoring. This section presents all the results obtained in these six main phases, as well as their sub phases, as shown below.

(1) The Idea and Strategy

This stage contains three main sub phases: idea, strategy, and marketing campaign. The resulting phase output follows this sub-phase.

(a) Idea: once you have the idea, you must start planning for your app. The best place to start planning is market research for competitors. See the user's ratings and reviews. Nearly all participants start with market research (59% said always; 43% said sometimes).

All great and major achievements start with an idea. You ask, "How can I make a new idea reality or make an existing idea better?" Think about a problem you can fix. Talk to an expert about the problem to understand why there is no solution yet, or how they might fix the problem. Ask how an app can be part of the solution. Such questions can give you a new or improved idea. Can you solve the problem with an app with the time that you have? Try to think about different ways to find customer needs or customer adaptations. Once you have your idea, start planning your app. The best place to start planning is market research to identify competitors. Check the reviews to learn what users say about your idea.

(b) Strategy: tries to identify challenges when planning an app strategy. It is essential to construct a roadmap. About 59% of participants always define app roadmaps, and 27% sometimes do.

It is important to identify your roadmap. What is your app? What is its future? What do you want your app to become someday? Write what your app has to do, consider its core functionality, and think about what you could add in the future.

(c) Marketing campaign: start with the marketing strategy. Participants agreed on the importance of marketing to the app's success. Approximately 53% of participants made a marketing plan; 30% did sometimes, unless the app was for internal or business to business (B2B) use. Start with marketing strategy, you can see that participants agree with the importance of marketing for the app's success.

(d) Phase Output: after you have your idea and your strategy, you could summarize them in an internal report. Approximately 44% of participants said they did this; 35% said they sometimes wrote a final report. About 43% of participants used JIRA as a team management tool, 40% Trello, and 33% Teamwork Project. At the end of this phase, after some iterations, you will obtain a final report.

(2) User Experience Design

After reviewing the goals and requirements with the user and the team, you start writing and creating blogs containing the data and functions that should be in the application and how to organize and present them. You can start with just a pencil and paper to sketch the wireframe, representing what you need and visualizing the customer needs. Over 93% of participants start with a whiteboard or paper and pencil for wireframe design.

The user experience phase has three main sub phases: wireframe, workflow, and test workflow, followed by the phase output resulting from this phase.

(a) Wireframe: the wireframe is the way of representing design content. At this stage, you should start by drawing and creating screens with their functionality and data. Most participants used whiteboards to create their wireframe (43%); others used paper and pencil (46%); still others used other tools, such as Sketch (23%) or InVision (23%).

At this stage, it is important to have conceptualized and included most of the app's functionality. Changing something now costs only the drawing, but later you will have to rewrite the code and redesign.

(b) Workflow: workflow refers to the pathways between wireframe badges that the user can navigate within the app. Most participants agreed on doing a workflow (53%); 38% said they sometimes did a workflow. Think carefully when you draw the workflow. See how many clicks you need to perform or finish each task; it should not take many clicks. If the task takes too many clicks, go back to your wireframe and try to fix the problem there. Be sure you have made the app more usable. Check and recheck all features in each iteration. Additionally, check the functionality and usability of your design.

Most participants perform iterations between the wireframe and workflow (46% yes; 36% sometimes), as shown below in Figure 5. You can draw your workflow on the whiteboard or on a paper with a pencil. InVision can help too.

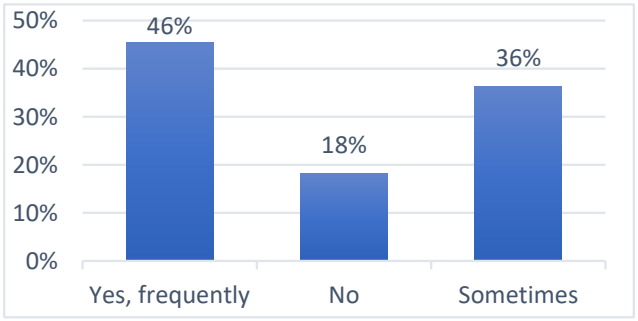

Figure 5. Using iterations to improve the design.

(c) Test workflow: building a tappable click-through model for your app can improve design. Participants were less excited about building a tappable click-through model: 24% did not use one, 45% did, and 30% sometimes did. For this model, 50% of participants used Sketch, and 22% used InVision. InVision can help to test the wireframe and workflow, after which you can import the design. The model can even be sent to users to be tested without functionality, with no code on the phone. Try to find problems, update, and continue.

(d) Phase Output: at the end of this phase, after some iterations, you will have a tappable UX prototype.

(3) User Interface Design

The user interface phase contains three main sub phases: style guidelines, replacement/mock-up, and test again, followed by the phase output resulting from this phase.

(a) Style guideline: follow the style guidelines to improve app usability and avoid mismatched parts and disjointed design. Familiarity is an important property of good UI design. Choose the right color and font. Consistent design language makes the app more comfortable to use. Try to determine who your customers or users are. Use your app day or night to minimize clutter. Teams with extensive experience help to deliver an app that works superbly for everyone. It is important to understand that there is no one-size-fits-all approach. User interface style guides are custom-made to the app's specific needs.

(b) Replacement-Mockup: a mockup is the version that most closely approximates a real product. To move from wireframe design to UI, start replacing the app wireframe elements'—colors, buttons, logo, photos, etc. Approximately 37% of participants used a Sketch for this task. You can design your style with a Sketch, import it from InVision, and construct your click-through model prototype.

(c) Test Again: most participants tested their model again; more than 57% answered that they did, whereas only 9% said they did not. After finishing the user interface design, 62% of participants always asked for user feedback and 32% sometimes did (see Figure 6).

Based on these findings, we conclude that it is important to go back to your click-through model prototype, repeat testing, take your time, and be sure of the design, usability, and user feedback. Remember that changes after this point can be costly.

(d) Phase Output: at the end of this phase, with some iterations, you will have a tappable user interface prototype.

(4) Design to development handoff- issues and decisions

One management issue/problem in developing an app is the transition from design to development. Most obstacles to a smooth transition come from the gaps in understanding and poor communication between the design and development teams. Try to have the same team handle design and development. It is good to use some tools to ensure this smooth transition, such as Sketch (27%), Photoshop (42%), or Zeplin (6%).

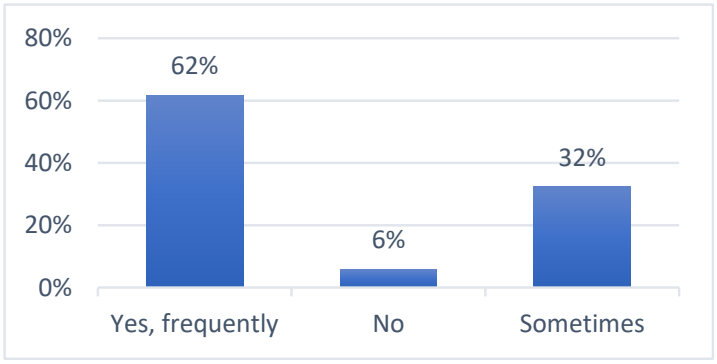

Figure 6. User feedback after user interface (UI) design.

The design to development handoff phase contains four main sub phases based on the issues and technical decisions to be considered: customer approval, technical design, front-end/back-end, and database.

(a) Customer approval: Figure 7 shows that about 90% of participants got final approval from the customer after finishing the app design. When you finish designing your app, it is important to obtain approval from customers to be sure that the design is what they need and want.

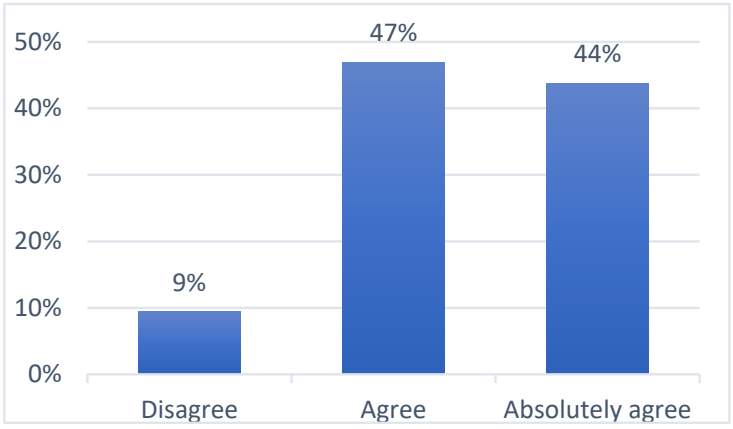

Figure 7. Starts the development phase after the customer accepts the design.

(b) Technical Design: you can use many technologies and programming languages to build a mobile app. Every technology has specific strengths. Some are less expensive, some perform less. You must choose a reliable technology.

(c) Front-End: it is the layer above the back end and it includes all software or hardware that is part of a UI, including user-entered data, buttons, programs, websites and other features. Most of these features are included in the UX. There are three approaches for a mobile app: native platform, cross-platform hybrid, and web technology. Native apps are exclusively developed for a specific operating system. They are native to a device or platform. They use the development tools and languages that the respective platform supports, like Objective C for iOS (mobile operating system created by Apple)or Java

for Android. Thus, what is built for an Android platform will not work for an iPhone iOS platform. Native apps are known for their high performance and speed with high development costs. For their part, web-based applications can be accessed through a mobile browser; they are responsive websites that adapt to the user's device. Web apps are known for accessibility from almost any device, and their performance depends on internet connections. Finally, cross-platform hybrid apps are mixtures of native and mobile web apps. They look and act exactly like native apps but are built using multiplatform technologies. Hybrid apps are known as easy to build but slower than native apps. Another way to classify mobile applications is by their purpose. There are 33 app categories on Google Play Store and 22 categories on iOS App Store, apps such as: games (21.86%), business (10.11%), education (8.68%), lifestyle (8.62%), utilities (6.12%), entertainment (5.79%), and travel (3.8%) [18].

It is good to know that most participants develop their apps on the native platform (60%), cross-platform hybrid (36%), and web technology (33%). Participants use HyperText Markup Language (HTML) (46%), JavaScript (45%), and Ionic (30%). Participants use Google-Cloud (44%) as an API database hosting, and Amazon Web Services AWS (39%).

(d) Backend (Web API/Server): it is the part of a mobile app that is responsible for security, data storage, business logic, and code that allows it to work, and a user cannot access that. When you develop a web-based app, you must choose the language. Participants used JavaScript (47%), PHP (Hypertext Preprocessor)(44%), and C# (34%).

(e) Database: it is so important to design a reliable and will organize data for your app; most of the participants use SQL, 12% do not.

As to hosting infrastructure, the location where the database and API will be hosted, 45% of participants used Google Cloud and 31% Amazon's Web Services AWS. In choosing the hosting infrastructure, it is essential to consider its effect on cost, performance, scalability, and reliability. The hosting infrastructure should be shared with your customer. Most participants share this with their customers; about 53% said they did, while 31% said they sometimes did.

(5) Development

Mobile app development is an iterative process. Most participants used iterations or sprints in the development processes (about 50% said yes; 35% said sometimes, as shown in Figure 8).

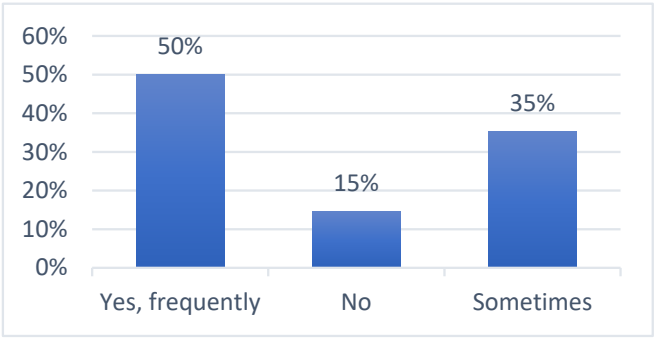

Figure 8. Working in iterations in the development phase.

The development phase contains four main sub phases: sprint planning, development, testing, and review, followed by the phase output resulting from this phase.

(a) Sprint Planning: the Sprint planning meeting's team members must decide what elements need to be achieved in the next Sprint. 60% of participants started with sprint planning; 27% said they sometimes did. Sprint starts from sprint planning, the list of tasks to be implemented during one iteration. Each task must clearly define the time needed for each task and its requirements. Developers must be involved in the process to guarantee

a full understanding of each task. Each sprint starts with sprint planning, development, testing, review, and customer or user feedback.

When developers start planning for the sprint, they must begin to explore how they can reuse code. About 65% of participants reuse code through the development process; only 8% do not (Figure 9).

At this point, you can begin to review your design and change it if necessary. Updating design is much better than updating code, and good design apps cause fewer problems, with fewer changes during the development phase.

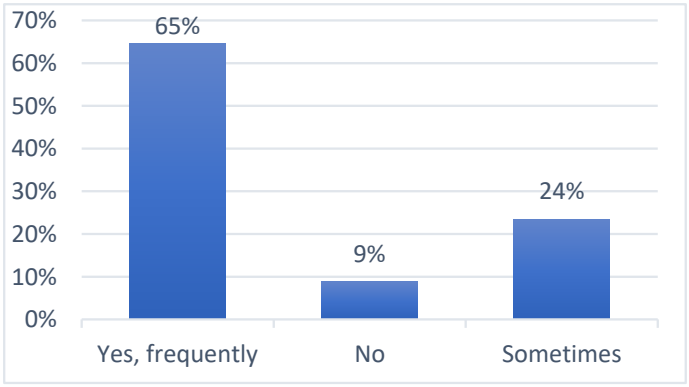

Figure 9. Reuse code.

(b) Development: you start to develop the functionality of your app when you complete the sprint. It is important to return the results to the project manager or to quality assurance for review. Figure 10 shows that about 70% of participants do this; 15% said they sometimes do. The team is crucial to success in this phase—a full understanding of the specific backlog requirements was worked on. If success is not achieved, the team must let the project manager know what is missing. Agile principles in developing apps are a means to break all requirements for product backlog milestones and starting development by building your app backlog by backlog in iterations and cycle/sprint backlog. Use an iterative, incremental development cycle.

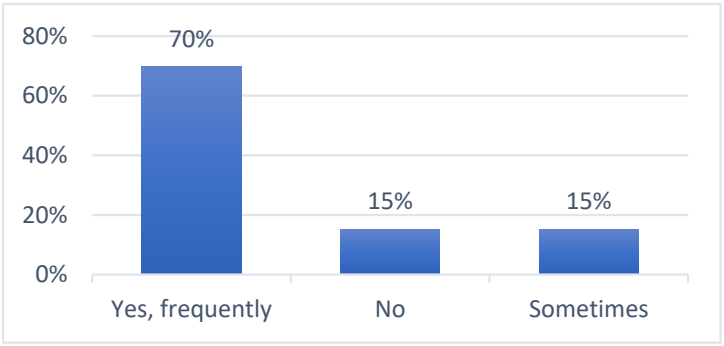

Figure 10. PM and QA review after each sprint.

(c) Testing: For a more genuine experience, developers should not test the apps. If the QA team is doing this job, start with functional testing, including a test plan and a list of actions to check. About 67% of participants performed functional testing with a list of actions to check; 27% sometimes performed functional testing (Figure 11). Functional testing usually focuses on user-friendly usability, performance testing, and responsiveness. Performance testing and app responsiveness are important in late sprints. 77% of participants use and focus on user-friendly usability testing, 67% on user acceptance testing, and 55% on performance testing. Try to involve designers in the testing process; let them check whether the vision they described was developed. Most participants agreed (47% said yes;

50% said sometimes), as shown in Figure 12. Remember to do regression testing. After you finish testing a sprint, you must go back and test the previous sprints.

On-device testing: be sure to test your app on numerous screen sizes and in different versions. You can use native tools or automated testing tools such as Google Firebase (28% of participants) and native tools (28%).

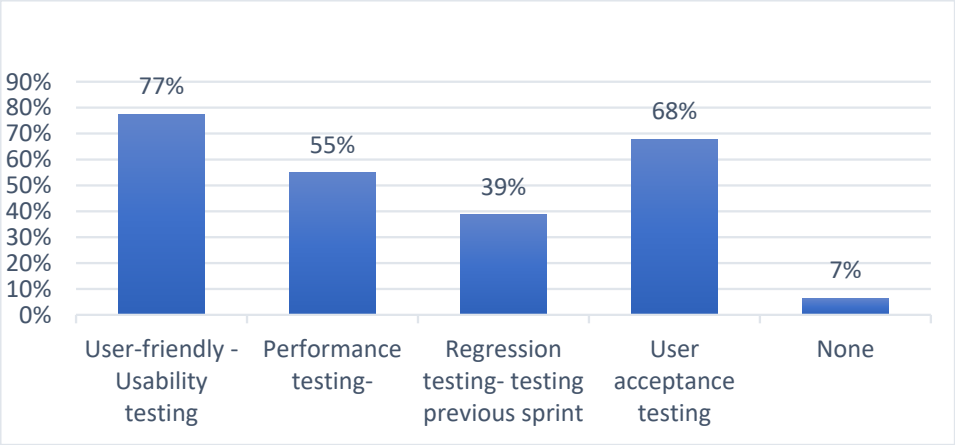

Figure 11. Application testing.

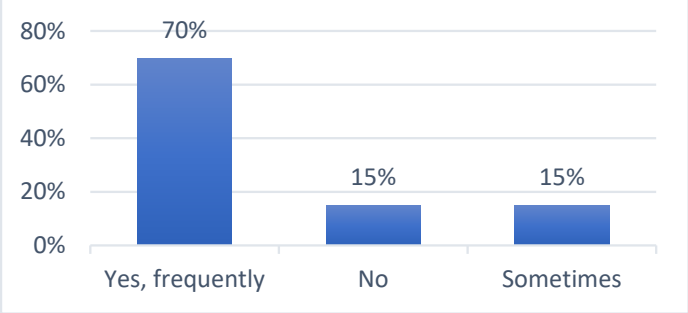

Figure 12. Designers review in the development and testing phases.

We believe it is important to test the app on physical devices. The most important testing is user acceptance testing and security testing. Owners or candidate users are the reason we build the app.

(d) Review: at the end of each sprint, talk with stakeholders to learn from everything you did in the sprint. Try to learn; the goal is to improve the process in each iteration. Try to find testers such as potential users, listen to their feedback, and then make decisions. You can launch a beta test or give your app to a real user to test. After you finish your reviews, it is good to perform one final development sprint to fix any remaining problems.

(e) Phase Output: after each iteration, you will have a beta app. At the end of this phase, after some iterations, you will have a fully functional app.

(6) Deployment and Monitoring

The deployment and monitoring phase contains two main sub phases, deployment and monitoring.

(a) Deployment: this is the last stage in the mobile app development process. It involves the selection of a day and finally conducting a formal launch of the application. After the deployment process is complete, the application becomes accessible. The web app requires a server backend to host your data. You must choose which server to use. Participants recommend a server with a scalable environment, such as Amazon Web Services. With such a server, your app will not fail if it becomes popular. To be deployed in the Apple store or Google play store, your app must meet these companies' requirements.

(b) 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 you to monitor your apps, such as Google Analytics for app analytics (67% of participants use it).

4.4. Development Tools

Note that all participants use different tools in the development process. While they do not focus on a particular type of tool [18], they focus more on some tools than on others. Figure 13 summarizes the tools that the participants used in each phase and sub phases. Most participants use the native platform for developing the app (61%), cross-platform hybrid (36%), web technology apps (33%). We also note that most participants use the native platform in the testing phase.

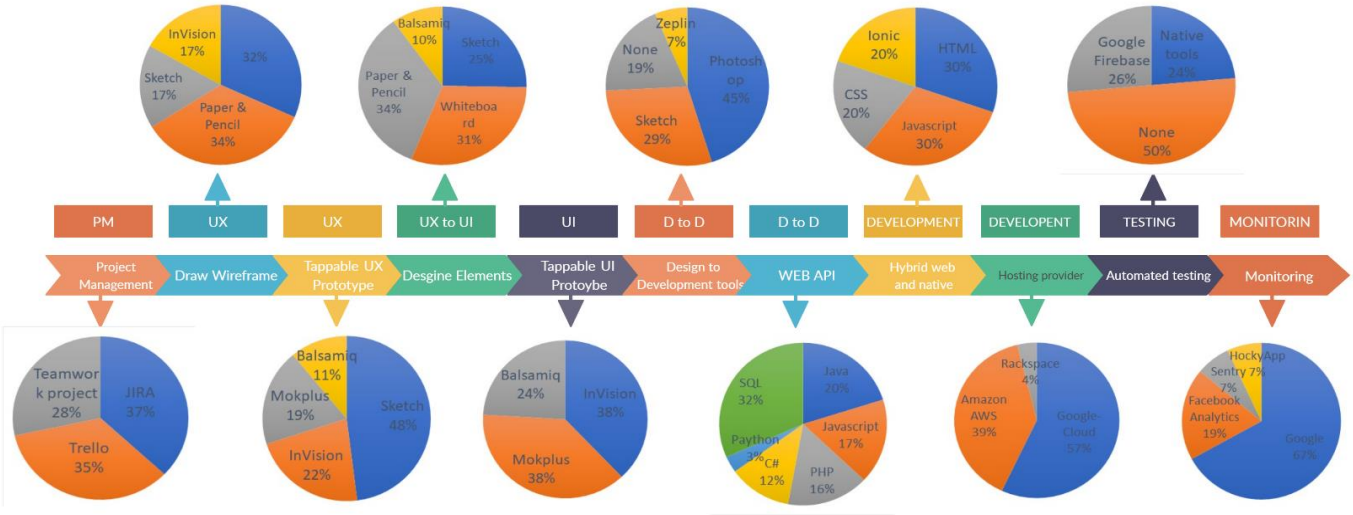

Figure 13. Tools used during the application development process.

Authors should discuss the results and how they can be interpreted from the perspective of previous studies and of the working hypotheses. The findings and their implications should be discussed in the broadest context possible. Future research directions may also be highlighted.

5. Discussion

Developing mobile apps is still a challenge. During the conducted three-phases study, we have shown 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 identified agile methods for mobile app development. We also explained how agile methods are naturally suited to app development and how apps are mapped onto agile themes. This study sought to understand a mix of agile techniques and methods based on what we have learned and experts and researchers' recommendations in app development to construct a general model for app development. We found no studies that analyzed a mixture of all the above.

In this study, we found six main phases adopted by participants in the mobile application development lifecycle, which are: The idea and strategy, UX design, UI design, design to development handoff- issues and decisions, development, and deployment and monitoring. The agile approach was preferred, that is, adopted by most participants; 84% used agile methods to develop mobile apps, while 61% believed agile methods were the best way to develop apps and 36% believed they sometimes were. Most participants (56%) used Scrum methodology in mobile app development, and about 13% each used XP and Kanban.

Agile methods have a history in mobile app development. We included some of these methods adopted in previous studies. Furthermore, some studies concluded that agile methods are an ideal, natural fit for their mobile app development practices, with Scrum as the most used method. Martinez et al. [14], Flora et al. [3], 20. Ghandi et al. [19], Holler et al. [20], and Ashishdeep et al. [21] 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.

When you adopt agile approaches in your organization, you know that one point in the Agile Manifesto is customer collaboration over contract negotiation. User-centered design (UCD) is the core of your 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 [22]. This study shows that 62% of participants request user feedback during and after finishing UI; 32% said they sometimes do. 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; 32% said they sometimes perform these tasks.

Aguilar and Zapata [23] argue that it is useful to integrate UCD and agile methodologies for software development to achieve software products with a higher degree of usability. Pratt et al. [24] recommended reaching the highest user satisfaction rate. We must place the user in the center of the design process. According to Rubin et al. [22], UCD is valuable for achieving greater usability in software.

One of the main challenges facing developers was moving from the design to the 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 80% of participants tried to use some tools to handle the transition from design to development.

When you start 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 match many studies and reports. One interesting finding in the report by Mojica et al. [25] is the practice of high software reuse among mobile app developers. Additionally, Ruiz et al. [26] argued that 61% of Android app classes appeared across two or even more different apps.

Iteration and incremental model/Sprints: most participants (48%) answered that they used an iterative design between the wireframe and workflow; 36% reported that they sometimes do. Mobile app development is an iterative process; most participants used iterations in the development processes. Most participants (50%) answered that they performed sprints and iterations in the development process; 35% reported that they sometimes do.

For Rubin et al. [22], the design, modification, and testing of the product should be an iterative process. The Manifesto for Agile Software Development [2] states that agile methodologies are iterative and incremental, focusing on short, frequent cycles and the functional product.

When you complete the sprint, it is essential to send the results back to the project manager or quality assurance for review. About 70% of the participants do this, and 15% do it sometimes. Team collaboration is the key to success.

The iterative model demands a new process for product development and testing. In Shivageeta et al. [27], testing in the iterative model 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. Approximately 67% 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. According to Mascheroni et al. [28], various authors affirm that continuous tests are the key to solving quality problems in continuous delivery based on continuous testing. Try to involve designers in the testing process. 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 et al. [22] recommend that users and the design team are in direct contact throughout product development, including the testing phase. To ensure a more genuine experience, testing should not be performed by the developers.

It is evident that none of the companies included in our previous study [17] has faithfully or integrally adopted any existing development methodology, and this corresponds to the majority of questionnaire participants, much less reviewing existing academic proposals to improve these processes. We believe work is needed to develop new methodologies that embrace the solid principles previously identified by the academic community, as well as 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.

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, these are: idea and strategy, UX design, UI design, design to development handoff- issues and decisions, development, and deployment and monitoring. In addition, the new methodology should be based on agile and Scrum management practices and project management practices. This methodology should support: small release, customer involvement and feedback, self-organization, fixed meetings, and short sprint iterations. In addition, an really useful methodology for mobile development should add more operational practices, manage the mobile app as a product, and control the process in order to effectively move from the design to the development. On the other hand, an iteration and incremental model 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. In the next section, a new methodology is proposed following these considerations (reflecting in what way the results obtained are incorporated within our proposal).

6. Proposed Mobile App Development Methodology

This section presents our proposed methodology for developing mobile apps: Agile Beeswax. After the extensive research performed in our in-depth study of mobile app development methods used by mobile application experts and academic communities, as well as an extensive literature review, we concluded that many practices and principles from both communities should be combined for the successful development of mobile apps. We had to classify and scrutinize many of these practices to determine which were the best, most widely used, and easiest to implement for all communities. After classifying these practices, we found that they could be divided into three main categories: Agile and Scrum management practices, engineering practices, and operational practices (see Figure 14). Agile Beeswax practices are based on the intersection of all these practices.

Figure 14. Agile Beeswax Practices.

6.1. Main Practices

Agile and Scrum practices: in the first place, some Agile and Scrum practices are adopted in Agile Beeswax because they have been demonstrated to be effective methods for developing mobile apps [3]. As Figure 4b shows, most participants use Scrum in developing mobile apps. 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 task. Scrum's approach is to direct and manage iterative cycles at project level. Relatively lightweight and timeboxed, Scrum provides fixed meetings and an all-time project overview, with details of delays, to-do lists, and a completed task list. Scrum is also well known, widely used, and easy to understand. Schwaber [29] 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, small release, customer involvement and feedback, self-organization, fixed meetings, short sprint iterations, sprint retrospective meetings, high control process, sprint and sprint planning, backlog creation, scrum master, retrospective and review meeting, product backlog, sprint backlog, the product owner, the scrum team, and fitting small-to-medium project size. Agile Beeswax uses some of these practices, not all of them. 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, which are essential to mobile app development.

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 behavior-driven development, continuous integration, continuous delivery, small release, refactoring, accept changes in iteration at any time, test-driven development, requirement prioritization, unit testing, coding reviews, and automated acceptance tests. Agile Beeswax includes some technical engineering practices, as shown in Figure 14.

Operational practices: in the 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 practice, eliminating waste, continual learning and improvement, continuous delivery, planning for unplanned work, flexible processes, accepting

changes in iteration at any time, ready to adapt to changing priorities, whole team master, customer involvement, issues that require immediate response and cannot wait for the next sprint planning session. Since the participants used Kanban and Lean, we decided to incorporate some of their operational practices to Agile Beeswax.

Agile Beeswax practices: as a conclusion, the development of mobile apps requires the intersecting of 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 14 and 15). Agile Beeswax has major adaptation, little adaptation, and no adaptation practices in all above practices.

Agile Beeswax practices include techniques and practices used by experts in app development, major adaptation practices such as: 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 center 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.

Intersecting parts such as operational engineering practices, or Agile–Scrum engineering practices that are shared with the same practices, such as operational engineering practices, we can clearly find that they are in common with, for example accepting changes in iteration at any time, customer involvement, or ongoing delivery. These intersections are outside the scope of our proposal. Our interest is in using practices from among these three types.

6.2. Agile Beeswax Main Phases

Figure 15 presents the proposed methodology for developing mobile apps: Agile Beeswax. Main rules and phases are explained in the following sections. (This section repeats some parts of Section 4.2. See Figure 13 for reference to the best tools).

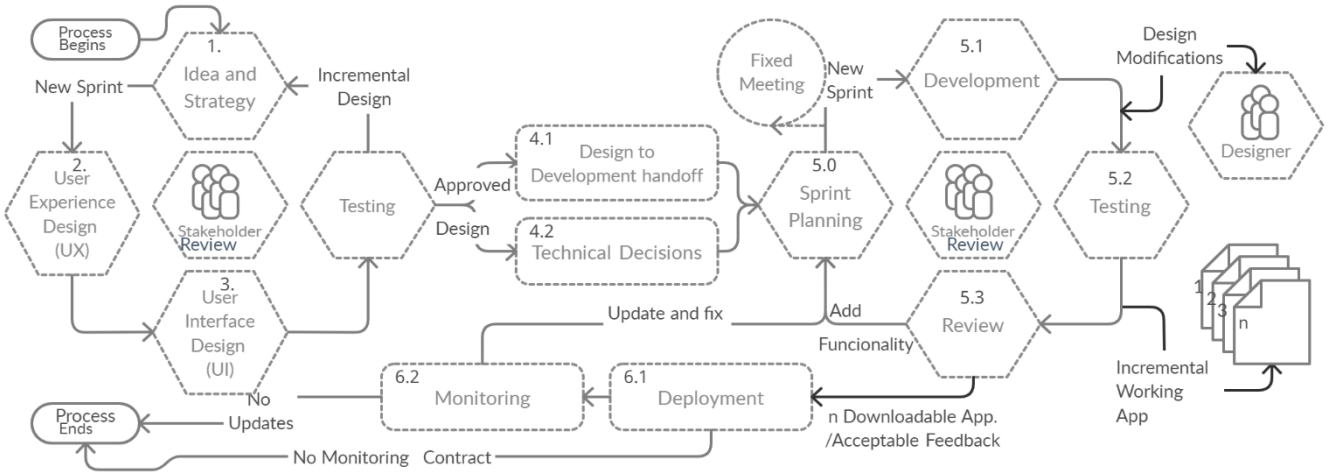

Figure 15. Agile Beeswax Methodology.

(1) Idea and strategy:

This phase contains three main sub phases: idea, strategy, and marketing campaign. Idea: apps start from an idea. This idea may be a solution to an existing problem, a way to cover some of users' needs, etc. Once you have the idea, start planning for your app. You need a strategy. Strategy: in defining the strategy, start by evolving the idea into a successful app. Identify challenges, a roadmap, and your app's futures. Write out what your app must do and its core functionality. App strategy involves identifying what your app is, what is it for, what you want from your app, and what you hope it will become someday.

Marketing campaign: start with a marketing strategy. Develop a marketing plan. Phase Output: Once you have your idea and strategy, write an internal report to record all of these elements. With some iterations, you will have a final report at the end of this phase.

(2) User experience design:

After reviewing the goals and requirements with the user and the team, start writing and creating blogs that contain the data and functions to appear in the mobile app, and ways to organize and present them. With just a pencil and paper, you can start to make the wireframe, which presents what you or customers need visually. Start with a medium-sized mobile screen when wireframing. The UX phase contains three main sub phases: wireframe, workflow, and test workflow. Wireframe: the way to show design content; this phase should begin by drawing and creating screens, and the app's functionality and data. Workflow: the pathways between wireframe badges that the user can navigate within the app. Check functionality and usability. Perform iterations between the wireframe and workflow. Test workflow: Build a tappable click-through model to improve app design. The goal of such testing is to see whether the screens work together and to review your design decisions. Using tools like InVision can help to test the wireframe and workflow, find problems, update, and continue. Phase output: with some iterations, you will have a tappable UX prototype at the end of this phase. Ask for user feedback while developing and after completing UX design.

(3) User interface design:

The UI phase contains three main sub phases: style guidelines, replacement/mockup, and test again. Style guidelines: Follow style guidelines to improve app usability and to avoid mismatched parts and disjointed design. Replacement/mockup: start replacing the wireframe with app elements, colors, buttons, logo, photos, etc. A sketch can help in this phase. Test again: test your model again, and ask for user feedback. Phase output: with some iterations, you will have a tappable user interface prototype at the end of this phase.

(4) Design to development handoff- issues and decisions:

The goal of this phase is a smooth transition from design to development. Most problems come from the gaps between teams of designers and developers. Tools such as Sketch, Photoshop, and Zeplin can help to ensure a smooth transition (see Figure 15). The design to development handoff phase contains four main sub phases based on the issues and technical decisions to be considered: customer approval, technical design, front-end/back-end, and database. Try to obtain final approval from the customer. Then choose the technologies and programming languages to be used to build the mobile app, including approaches using native platform, cross platform hybrid, and Web technologies. You must choose the right tools—HTML, JavaScript, ionic, etc.; the API and database hosting; and so on. It is extremely important to design a reliable API that will organize data for your app. In choosing where your database and API will be hosted, we recommend sharing this with your customers. Section 4 discusses this issue in more detail.

(5) Development:

Mobile app development is an iterative process. After identifying the project manager or project master he will be mainly responsible for leading the sustainability work. That means identifying and prioritizing different work items. The development phase contains four main sub phases; sprint iteration: developers must be involved in the process to guarantee full understanding of each task. Development: once you complete the sprint, start to develop the app's functionality. Agile principles in developing apps are a means to break down all requirements to product backlog milestones. They enable you to start development by building the app backlog by backlog in iterations and cycles–sprint backlog. Apply an iterative and incremental development cycle. Testing: start with functional testing using a test plan and checklist of actions. Try to include designers in the testing process. You must perform regression testing. On-device testing: make sure to test your app on numerous screen sizes and in different versions, using native tools or automated testing

tools (see Figure 15 for tools you can use in this phase). The most important testing is user acceptance testing: We built the app for the users. Review: At the end of each sprint, talk with stakeholders to learn from everything done in the sprint; try to learn for continuous improvement in the next sprint. Your goal is to improve the process in each iteration. Try to find testers who resemble potential users. Try to accept their feedback and then make decisions. After you finish your reviews, it is good to perform one last development sprint to fix any previously discovered problems. Phase output: with some iterations, each iteration will produce a beta app, and you will have a fully functional app at the end of this phase. Section 4 discusses this issue in more detail. List of tasks to be implemented during one sprint: sprint planning, development, testing, and review.

In the development phase, the technical debt and issues become part of the sustainability work. Common types of technical issues are: refactoring bad code, removing dead code, updating old libraries, bad log messages, introducing better techniques and tools, cleaning up any code, not following agreed upon standards, fixing technical defects and many more. Fixing these is an important part in the sustainability in mobile application development process. Writing some automated tests should be standard in most cases. It is one thing to write automated tests, the other is to make your application more testable. Improving APIs and refactoring your code can result in a much easier application to test. This is an excellent example of sustainability, as you are doing work that will benefit the project in the future by making it easier and faster to write strong tests. Having a stable development experience is important for sustainability as you have a good codebase. With that in mind, making sure you are using the right tools becomes an important part of project development.

(6) Deployment and monitoring:

The deployment and monitoring phase has two main sub phases, deployment and monitoring. Deployment: The hybrid-web app requires a backend server to host your data. You must choose which server to use. If you deploy in the Apple Store or Google Play Store, you must make sure that your app meets their requirements. Monitoring: monitoring the app for updates includes bug fixes, changes, new features, performance updates, etc. Some tools can help you monitor your apps, such as Google Analytics for app analytics. Section 4 discusses this issue in more detail. Developers will also need tools to measure energy consumption and emerging tools that identify expected energy use. Reducing energy consumption results in better sustainable systems impact. Making the application easy to support is highly important feature of sustainable mobile app development. Adding the ability to alert and creating monitoring dashboards are common ideas to improve monitoring and support.

6.3. Main Rules of Agile Beeswax

There are two main iterative loops, the iterative incremental design loop and the iterative incremental development loop. One bridge connects these loops (sprints). We call it the design to development handoff and technical decisions.

(1) First loop: design sprint.
- Incremental UX, incremental UI.
- The idea of Incremental UX and UI is to design iterations of a component that can be developed one by one.
- The first design iteration must satisfy the basic user needs.
- We apply the first phase of requirements engineering.
- Start by creating the wireframes; then add user stories for each screen.
- Start the design sprint by defining the new requirement and blocks.

(2) When you complete one sprint in the design phase and have approved the front end, send it to the second loop, crossing the bridge (design to development handoff and technical decisions).

(3) Second loop: development.

- In the second loop development phase; we apply some Scrum practices to the development phase.
- Before starting the first sprint, we apply the second phase of requirements engineering.
- 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.
- Product owner writes user stories as a Scrum product backlog and initiates a prioritization session with the architect and some developers.
- Sprint typically lasts 2–4 weeks.
- In Sprint 1, Day 1, start with the sprint planning and sprint planning meeting. The product owner presents the Scrum backlog items, prioritized from highest to lowest. 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. The deeper the development cycle, the higher the costs incurred in fixing bugs. It is often important to fix the planning and original design documents when creating different test cases.
- Complete the sprint by reviewing its results and adding functionality.

(4) General rules.

- Focus on continuous feedback.
- Have a fixed meeting. Meetings need not be daily, but it is important that they are fixed schedule meetings.
- Build your app incrementally, block by block, until you reach the final product with acceptable feedback.
- Some operational practices, such as continuous learning and improvement, are needed at the end of each sprint.
- As an operational practice, sprints must be flexible so that they can incorporate some modifications during the sprint's life.
- After reaching the last sprint and receiving feedback, move to the deployment phase.
- Some companies do not reach and interact with customers through a maintenance contract. If this is the case, you have reached the end of the process.
- If you have a maintenance and upgrade contract, move to the monitoring phase and continue updates and upgrades.
- Designers are always in the development phase, working on updates, and in the sprint review.

The same occurs in test-driven development and continuous integration, which increase value delivery. Using XP practices provides quick frequent delivery, teamwork, and nominally documents upfront. It is a high-discipline method that requires committed personnel who are not necessarily IT department personnel. It also requires each team member to have an intimate knowledge of XP for successful development of mobile projects.

6.4. Applicability and Limitations

Agile Beeswax has been focused on native 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 [30]. 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. This section is not mandatory but may be added if there are patents resulting from the work reported in this manuscript.

6.5. Comparison with the Reviewed Methodologies

Table 3 compares our methodology Agile Beeswax with others methodologies proposed by other researchers who have made significant advances in their understanding of relevant factors that influence the mobile development processes (see Section 2.2). The table summarizes the main methods proposed for developing mobile applications to perform the comparison we have selected the most common characteristics that have been used to evaluate each contribution. Consequently, the proposals are collated in terms of: used agile methods (Such as Agile Software Development ASD, used non-agile methods (such as New Product Development NPD and Software and systems Process Engineering Metamodel SPEM), 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, if it implements a management approach (or not), if it applies a technical engineering approach (or not), if it applies an operational approach (or not), and the structure of the methodology (depending on its weight).

Table 3. Comparing the existing methodologies and our proposal (Agile Beeswax).

Mobile Process	Mobile D	Hybrid	MASAM	Scrum	SleSS	MADeM	Mobile Ilities	Agile Beeswax
Year	2004	2008	2008	2010	2011	2016	2020	2021
Agile	XP, Crystal	ASD	XP	Scrum	Scrum, Lean	XP, Crystal	Scrum	Scrum
Non-Agile	RUP	NPD	RUP, SPEM	_	Lean Six Segma	RUP	_	Lean, Kanban
Experimental	Yes	No	No	Yes	Yes	No	Yes	Yes
Based on Experts	Yes	No	Yes	No	Yes	No	No	Yes
Based on Academic	No	No	No	No	No	No	No	Yes
Management Approach	No	Yes	No	Yes	Yes	No	Yes	Yes
Technical Engineering Approach	Yes	No	Yes	No	No	Yes	No	Yes
Operational Approach	No	Yes	Yes	Yes	Yes	No	Yes	Yes
Methodology Structure	Heavy Weight	Light-weight	Heavy Weight	Light-weight	Light-weight	Heavy Weight	Lightweight	Light-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 3 into one methodology. The proposed method, Agile Beeswax, is a set of mobile development processes that would be used in different contexts and environments (with the limitations previously indicated), which was born from an experimental approach (interviews, surveys, etc.) involving academic and industry.

7. Conclusions and Future Work

To fulfill our research goals, we performed in-depth analysis of the methods for developing mobile apps through an extensive review of previous research and interviews with specialists in developing mobile apps.

Based on the information and results obtained, we designed a lengthy, detailed online questionnaire on the process of developing mobile apps (Appendix A). Our study has advanced understanding of the mobile app development process adopted in the business world and the academic community. The synthesis of these processes provides a greater understanding of issues and challenges, as well as the tools currently used.

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. To succeed, mobile app development requires numerous techniques and specific project management practices from agile and Scrum, technical and engineering practices, and some operational practices.

Synthesizing our research has enabled us to present a methodology that encompasses the specific features of mobile app development, Agile Beeswax. 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. 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.

Our future research will start asking experts to validate if our findings are correct and if the proposed methodology is consistent with them. We are developing a full Agile Beeswax methodology with all explanations needed to be presented to experts developing mobile apps for validation, evaluation, and development. Some of these experts will be those previously interviewed (semi-structured interviews have already been performed with this aim). This methodology will be developed and improved based on expert advice and guidance and will be compared to present the differentiation with what has been presented in this paper. We will also present the methodology to some companies to adopt this methodology when developing mobile apps to evaluate and improve their methods. The information gathered will allow for refinement and revision of the Agile Beeswax descriptions and introduce examples compiled into the proposed methodology's practical application.

Author Contributions: Conceptualization, H.A.A. and N.M.-M.; methodology, H.A.A.; software, H.A.A.; validation, H.A.A. and N.M.-M. formal analysis, H.A.A.; investigation, H.A.A.; resources, H.A.A.; data curation, H.A.A.; writing—original draft preparation, H.A.A.; writing—review and editing, H.A.A.; visualization, H.A.A.; supervision, N.M.-M.; project administration, H.A.A. and N.M.-M. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been partly financed by the Spanish Government through the contract RTI2018-096986-B-C32 (PERGAMEX-ACTIVE).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Table A1. Questionnaire.

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		

Table A1. *Cont.*

QN	Category/Questions	Possible Answers	% Each Answer
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 organization 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 organization 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 organization?	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 organization 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 organization 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%

Table A1. *Cont.*

QN	Category/Questions	Possible Answers	% Each Answer
The Idea and Strategy			
QN 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?	Yes, frequently	54.3%
		No	-
		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%
		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%

Table A1. *Cont.*

QN	Category/Questions	Possible Answers	% Each Answer
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%
		Invision	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 dataflow?	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%

Table A1. *Cont.*

QN	Category/Questions	Possible Answers	% Each Answer
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%

Table A1. *Cont.*

QN	Category/Questions	Possible Answers	% Each Answer
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%

Table A1. *Cont.*

QN	Category/Questions	Possible Answers	% Each Answer
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?	Sentryfor 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 specify 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%, ...		

References

1. Clement, J. Mobile App Usage. Available online: https://www.statista.com/topics/1002/mobile-app-usage/ (accessed on 13 July 2020).
2. Beck, K.; Grenning, J.; Martin, R. Manifesto for Agile Software Development. 2001. Available online: https://agilemanifesto.org (accessed on 26 July 2020).
3. Flora, H.K.; Chande, S.V.; Wang, X. Adopting an agile approach for the development of mobile applications. *Int. J. Comput. Appl.* **2014**, *94*, 43–50.
4. Wasserman, A.I. Software engineering issues for mobile application development. In Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research—FoSER, Santa Fe, NM, USA, 7–8 November 2010. [CrossRef]
5. Kumar, N.; Ajit, K.; Krishna, H.; Manjula, R. Challenges and best practices in mobile application development. *Imp. J. Interdiscip. Res.* **2016**, *2*, 1607–1611.
6. Mahmud, D.; Abdullah, A. Agile: A natural fit in mobile application development process. In Proceedings of the 7th International Conference on Internet (ICONI), Abu Dhabi, United Arab Emirates, 28–31 March 2015.
7. Abrahamsson, P.; Hanhineva, A.; Hulkko, H.; Ihme, T.; Jlinoja, J.; Korkala, M.; Koskela, J.; Kyllnen, P.; Salo, O. Mobile-D. In Proceedings of the Companion to the 19th Annual ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages, and Applications—OOPSLA, Vancouver, BC, Canada, 4–28 October 2004. [CrossRef]
8. Jeong, Y.; Lee, J.; Shin, G. Development process of mobile application SW based on Agile methodology. In Proceedings of the 10th International Conference on Advanced Communication Technology, Gangwon-Do, Korea, 17–20 February 2008. [CrossRef]

9. Rahimian, V.; Ramsin, R. Designing an Agile methodology for mobile software development: A hybrid method engineering approach. In Proceedings of the Second International Conference on Research Challenges in Information Science, Marrakech, Morocco, 3–6 June 2008. [CrossRef]
10. Ågerfalk, P.J.; Brinkkemper, S.; Gonzalez-Perez, C.; Henderson-Sellers, B.; Karlsson, F.; Kelly, S.; Ralyté, J. Modularization constructs in method engineering: Towards common ground? In *Situational Method Engineering: Fundamentals and Experiences*; IFIP—The International Federation for Information Processing Seiries; Springer: Berlin/Heidelberg, Germany, 2007; pp. 359–368. [CrossRef]
11. Scharff, C.; Verma, R. Scrum to support mobile application development projects in a just-in-time learning context. In Proceedings of the 2010 ICSE Workshop on Cooperative and Human Aspects of Software Engineering—CHASE, Cape Town, South Africa, 1–8 May 2010. [CrossRef]
12. Cunha, T.F.; Dantas, V.L.; Andrade, R.M. SLeSS: A Scrum and Lean Six Sigma integration approach for the development of Sofware customization for mobile phones. In Proceedings of the 25th Brazilian Symposium on Software Engineering, Sao Paulo, Brazil, 28–30 September 2011. [CrossRef]
13. Alsabi, E.; Dahanayake, A. Smart modeling for lightweight mobile application development methods. In *New Trends in Databases and Information Systems*; Communications in Computer and Information Science; Springer: Berlin/Heidelberg, Germany, 2016; pp. 168–179. [CrossRef]
14. Martinez, D.; Ferre, X.; Guerrero, G.; Juristo, N. An Agile-based integrated framework for mobile application development considering Ilities. *IEEE Access* **2020**, *8*, 72461–72470. [CrossRef]
15. Kirmani, M. Agile methods for mobile application development: A comparative analysis. *Int. J. Adv. Res. Comput. Sci.* **2017**, *8*, 1200–1205.
16. Ahmad, A.; Li, K.; Feng, C.; Asim, S.M.; Yousif, A.; Ge, S. An empirical study of investigating mobile applications development challenges. *IEEE Access* **2018**, *6*, 17711–17728. [CrossRef]
17. Al-Rabaiah, H.A.; Medina-Medina, N. Mobile application development process in real environments. In Proceedings of the International Conference on Software Engineering Research and Practice (SERP), Las Vegas, NV, USA, 29 July–1 August 2019. Available online: https://csce.ucmss.com/cr/books/2019/LFS/CSREA2019/SER4050.pdf (accessed on 26 July 2020).
18. Clement, J. Most Popular Apple App Store Categories. 2020. Available online: https://www.statista.com/statistics/270291/popular-categories-in-the-app-store/ (accessed on 13 June 2020).
19. Ghandi, L.; Silva, C.; Martinez, D.; Gualotuna, T. Mobile application development process: A practical experience. In Proceedings of the 12th Iberian Conference on Information Systems and Technologies (CISTI), Lisbon, Portugal, 14–17 June 2017. [CrossRef]
20. Holler, R. *Mobile Application Development: A Natural Fit with Agile Methodologies*; VerisonOne LLC: Alpharetta, GA, USA, 2006.
21. Ashishdeep, A.; Bhatia, J.; Varma, K. Software process models for mobile application development: A review. *Comput. Sci. Electron. J.* **2016**, *7*, 150–153.
22. Rubin, J.; Chisnell, D. *Handbook of Usability Testing: How to Plan, Design and Conduct Effective Tests*; John Wiley and Sons: Hoboken, NJ, USA, 2008.
23. Aguilar, M.; Zapata, C. Integrating UCD and an Agile methodology in the development of a mobile catalog of plants. In *Advances in Ergonomics Modeling, Usability & Special Populations*; Springer: Cham, Switzerland, 2016; pp. 75–87. [CrossRef]
24. Pratt, A.; Nunes, J. Interactive design: An introduction to the theory and application of user-centered design. *Choice Rev. Online* **2013**, *50*. [CrossRef]
25. Mojica, I.J.; Adams, B.; Nagappan, M.; Dienst, S.; Berger, T.; Hassan, A.E. A large-scale empirical study on software reuse in mobile apps. *IEEE Softw.* **2014**, *31*, 78–86. [CrossRef]
26. Ruiz, I.J.; Nagappan, M.; Adams, B.; Hassan, A.E. Understanding reuse in the Android market. In Proceedings of the 20th IEEE International Conference on Program Comprehension (ICPC), Passau, Germany, 11–13 June 2012. [CrossRef]
27. Shivageeta, S.C. Testing in Iterative Product Development Environment. Available online: http://www.qaielearning.com/KnowledgePapers/Testing_In_Iterative.pdf (accessed on 26 July 2020).
28. Mascheroni, M.A.; Irrazábal, E. Problemas que afectan a la Calidad de Software en Entrega Continua y Pruebas Continuas. In Proceedings of the XXIV Congreso Argentino de Ciencias de la Computación, La Plata, Argentina, 8–12 October 2018.
29. Schwaber, K. *Agile Project Management with Scrum*; Microsoft Press: Washington, DC, USA, 2004.
30. Garneli, V.; Giannakos, M.; Chorianopoulos, K. Serious games as a malleable learning medium: The effects of narrative, gameplay, and making on students' performance and attitudes. *Br. J. Educ. Technol.* **2016**, *48*, 842–859. [CrossRef]