*Article*

# Green Requirement Engineering: Towards Sustainable Mobile Application Development and Internet of Things

**Mahrukh Tanveer [1], Huma Hayat Khan [1,*], Muhammad Noman Malik [2] and Youseef Alotaibi [3]**

[1] Department of Software Engineering, National University of Modern Languages, H-9, Islamabad 46000, Pakistan
[2] Department of Computer Science, National University of Modern Languages, H-9, Islamabad 46000, Pakistan
[3] Department of Computer Science, College of Computer and Information Systems, Umm Al Qura University, Makkah 21955, Saudi Arabia
* Correspondence: hnauman@numl.edu.pk

**Abstract:** Mobile usage statistics show the one thing that cannot be overlooked, which is the overwhelming usage of smartphones. According to the statistics, there are approximately 6.4 billion users of smartphones. Considering the world population, this rate of smart phone usage is more than 80%. Mobile development is the fastest prominent trend, although web development cannot be denied. However, the fact is that mobile platforms are considered cumbersome and complex when it comes to accomplishing requirement engineering processes, especially when mobile applications are combined with the Internet of Things (IoT). These complexities result in barriers to sustainable mobile development. The difficulty and differences occur due to various limitations, either that of mobile devices or others. Some of those from mobile devices include processor, battery, and touch screens, user experience in terms of touch screens, user context, and interactive behaviors. Other limitations include the difference in the software development lifecycle and the difference in the software development process due to inconsistency in user requirements with the aforementioned limited device capabilities. The target objective of this research is to investigate and identify all possible challenges related to mobile applications and connected mobile devices (IoT) while executing the requirement engineering process. This study can further the existing state of knowledge by contributing to the list of challenges faced in the requirement gathering process of mobile application development. Furthermore, it can also help practitioners, specifically those involved in the requirement gathering process, to carefully consider these challenges before executing the requirement engineering process.

**Keywords:** requirement engineering process; mobile application development; sustainability; Internet of Things

check for updates

**Citation:** Tanveer, M.; Khan, H.H.; Malik, M.N.; Alotaibi, Y. Green Requirement Engineering: Towards Sustainable Mobile Application Development and Internet of Things. *Sustainability* **2023**, *15*, 7569. https://doi.org/10.3390/su15097569

Academic Editor: Xiuwen Fu

Received: 26 January 2023
Revised: 30 March 2023
Accepted: 18 April 2023
Published: 5 May 2023

## 1. Introduction

Mobile phones are becoming an important part of daily life. There are about 6.4 billion people who own smartphones worldwide in the era of 2021 and this will further rise in the coming times [1]. Similarly, throughout the last few decades, there has been a significant increase in demand for mobile applications where the desire is to connect devices with mobile phones effortlessly. This concept is recognized when mobile applications are combined with the Internet of Things (IoT). The software development sector is a good example of rapid evolution. Since iPhone sales opened in July 2008, many device engineers have generated numerous outlets for other mobile devices. It has been estimated by industry experts that more than 250,000 applications are available through various stores and markets. Mobile application development is capturing much more attention from researchers as well. It is due to the different behavior of mobile development from that of web development. As people spend more time on their mobile devices, the need for mobile applications is growing [2] and as the penetration of mobile application development has increased, its percentage is also expected to be increased in the upcoming years [3,4].

Mobile applications are planned to be developed so that they can be operated on connected mobile devices and smartphones. Such apps have eccentric roadblocks as compared to other types of applications. Furthermore, it faces distinct hurdles in comparison with other application types [5]. On the other hand, the development of software takes place through the requirement engineering (RE) process. This software requirement engineering is gaining extreme importance as it carries out the development process smoothly. Every phase in requirement engineering has its rank and significance [6].

Software requirement engineering is a process that comprises various activities, from requirement elicitation (requirement gathering) to requirement specification (requirement documentation) [3]. According to IEEE, the requirement is "a condition or capability needed by a user to solve a problem or achieve an objective" [7]. Performing software requirement engineering smoothly not only leads to an efficient system but also proves to be successful in achieving a cost-effective system [3]. The main principle that the RE process follows is to meet the requirements of the customer and end-user with the minimum resources [6]. Software development becomes extremely complex as the number of new technologies is increasing rapidly. Hence with this rapid increase, it becomes problematic for mobile developers to work in such an environment as it faces several challenges during mobile development [8]. One of the major drawbacks of the mobile platform is its short development lifecycle discussed in the coming sections. The other important comparison making mobile applications different from web applications are differences in the development process and device limitations. The development process differs due to the volatility in user requirements. In addition to this, the overall lifecycle of mobile applications is much faster than any web or desktop application. The third most prominent difference is the limited device capacity or power, and others [9].

There is substantial work conducted on mobile applications but very little is conducted in the domain of sustainable requirement engineering process for mobile application development with its combination with IoT. This furthers the researcher's motivation to investigate the requirement engineering process in the domain of sustainable mobile application development. The goal of this research is to identify key challenges in the field of sustainable requirement engineering processes for sustainable mobile applications and connected devices through the internet.

The remaining paper is distributed as follows: Section 2 is about the background of the research study. Section 3 explains the research methodology. Section 4 reports the results and discussion. Section 5 specifies the threats to validity. Section 6 specifies the conclusion and future work.

## 2. Background

Network connectivity is required by a standard mobile application to communicate with other computing tools. The development of mobile applications involves constructing software packages that are installable (code, binaries, properties, and so on), integrating backend services, such as API data access, and testing the application on target devices [10]. Some of the common concerns include integration with the device hardware, and conventional problems [4]. However, mobile applications comprise a few other requirements as well which are rarely found in conventional software applications.

The typical software development method differs significantly from the mobile application development platform [3]. It includes unpredictability of user requirements, the device's capabilities, or device compatibilities, such as the CPU, battery, and touch screens, as well as the experience of the users regarding touch screens, user context, and interactive behavior, including customer or user reviews [10–13]. Apart from that, the overall lifecycle of mobile applications is shorter than that of desktop applications [14–16].

There have been numerous studies conducted on mobile application development. The researchers have gone through the existing literature regarding mobile application development in combination with IoT. Table 1 shows the summary of existing literature related to our research study.

**Table 1.** Existing studies.

| Title | Author Name/Publication Year | Domain/Publication Type | Contributions | Limitations |
|---|---|---|---|---|
| "A review on the computation offloading approaches in mobile edge computing: A game-theoretic perspective" | Ali Shakarami/2020 [17] | Mobile edge computing/Research article | GT-based computation offloading approaches in the MEC environment are reported | The perspective of game theory is discussed only |
| "Joint computation offloading and resource provisioning for edge-cloud computing environment: A machine learning-based approach" | Ali Shahidinejad/2020 [18] | Mobile edge computing/Research article | An edge server provisioning approach using LSTM prediction to estimate the future workload and reinforcement learning technique to make the appropriate scaling decision | - |
| "An automated model-based approach for unit-level performance test generation of mobile applications" | Muhammad Usman/2020 [19] | Software testing/Research article | Mobile application model-based approach is formulated specifically for performance testing | The proposed approach is specific to variants of native apps |
| "VAnDroid: A framework for vulnerability analysis of Android applications using a model-driven reverse engineering technique" | Atefeh Nirumandv/2019 [20] | Computer science/Research article | VAnDroid framework is proposed regarding Android applications. The framework can be used for eliciting the security risks and vulnerabilities | The limited sample size for tool evaluation |
| "EMaaS: Energy measurements as a service for mobile applications" | Luis Cruz/2019 [21] | Software engineering/Conference paper | A system providing reliable energy measurements for mobile applications without requiring a complex setup, named EMaaS | The proposed system is not reliable for w.r.t. security and privacy |
| "Radio and Computing Resource Allocation for Minimizing Total Processing Completion Time in Mobile Edge Computing" | Ryuji Kobaya/2019 [22] | Research article | A scheme is proposed for computing resource allocation | Time as a resource is discussed for tasks to be decomposed |
| "LeakDoctor: Toward Automatically Diagnosing Privacy Leaks in Mobile Applications" | Xiaolei/2019 [23] | Research article | The leakDoctor approach is generated | Lacking privacy declaration scenarios |
| "GUILeak: Tracing Privacy Policy Claims on User Input Data for Android Applications" | Xiaoyin Wang/2018 [24] | Software engineering/Research article | Proposed a novel approach to protect privacy policy violations due to leaks of user input data | Inadequate validation sample |
| "A systematic study on software requirements elicitation techniques and its challenges in mobile application development" | Hafsa Dar/2018 [3] | Requirement engineering/Research article | Highlighting elicitation techniques and challenges in mobile application development | No unified model or framework was proposed |
| "Why does the orientation change mess up my Android application? From GUI failures to code faults" | Domenico Amalfitano/2018 [25] | Research article/Software testing | DOC framework is formulated that categories the failures related to GUI | Limited means for the application requirements leading to miscalculation of GUI failures |

**Table 1.** *Cont.*

| Title | Author Name/Publication Year | Domain/Publication Type | Contributions | Limitations |
|---|---|---|---|---|
| "An energy-efficient algorithm for multi-site application partitioning in MCC" | Wen-li Zhang/2018 [26] | Mobile cloud computing/Research article | Proposed multi-site application partitioning algorithm based on genetic algorithm | Only considered Wi-Fi environment where bandwidth rate is relatively stable |
| "Cordovaconfig: A tool for mobile hybrid apps' configuration" | Abeer AlJarrah/2018 [27] | Conference paper/Security and privacy | Tool named as CORDOVACONFIG is developed that is used for mobile application configuration | The input was taken by the limited target audience; hence, authenticity can be a major issue |
| "Why does data deletion fail? A study on deletion flaws and data remanence in android systems" | JUNLIANG SHU/2017 [28] | Research article/Security and privacy | Framework named DataRaider is proposed that is used for recovering data from stored files. | Lack of effectiveness in disc encryption mechanism |
| "Study and refactoring of android asynchronous programming" | Yu Lin/2016 [29] | Conference Paper/Computer science | AsyncDroid tool is proposed that is focused on refactoring | Lack of mechanism for transforming communication modes |
| "A systematic mapping study of mobile application testing techniques" | Samer Zein/2016 [30] | Accepted manuscript/Software testing | Practitioners-based testing problems/challenges are reported | Validation of findings can be an issue towards result authenticity |

The latest and most relevant studies were published in 2020, from which two [17,18] studies out of three [17,19] are based on the mobile edge computing domain. The authors in their study [17] have reviewed computation offloading approaches concerning game theory while the other [18] focused on a joint merger of computation offloading with resource provisioning, giving an approach based on reinforcement learning technique, a machine learning-based approach. The authors of the study [19] researched an automated model approach for performance test evaluation of mobile devices at the unit level.

Android is the most popular platform which is widely used all around the globe by mobile developers providing communication via message passing. Another study [20] reported that the message-passing mechanism is unsafe and vulnerable in terms of its operations. To identify the security risks and vulnerabilities in the Android communication model, they described a framework that helps to solve the mentioned above issue [20]. Similarly, further research [21] has been conducted on flexible systems that are capable of switching between software and hardware-based measurements, depending on the application being tested and the operating conditions, thereby delivering measurement as a service.

In a mobile edge computing (MEC) context, finding the best solution is tricky. This ideal guide in MEC relates to the activities' competition time, and the author proposes a two-step technique for solving this optimized problem [22]. The author looked at privacy concerns in terms of privacy disclosures, and he came up with a way to automatically diagnose privacy breaches in mobile applications. The study of data protection disclosures in internet traffic is the author's main focus, as well as the first step toward automated confidentiality leak identification. The act of transferring one or more forms of private data over the internet, on the other hand, is known as private data disclosure [23]. This study [24] focused on privacy threats, also referred to as "privacy disclosure" threats, such as the aforementioned issue. Like above, it refers to network queries that convey one or more forms of private data across the internet. Android is a popular platform that allows users to access personal sensitive data with full functionality, posing major privacy risks [24].

One study is based on requirement elicitation challenges, particularly on mobile platforms. The author contributed to the requirement gathering guideline as well as categorized the challenges [3] while a similar study discussed the limitations of mobile platform testing approaches and presented testing topics for practitioners. The paper's author presented a system for classifying such GUI failures, as well as all the defects that cause them [25]. Similarly, the study [26] is based on a multi-site partitioning problem in the mobile cloud computing domain, proposing an algorithm to overcome the problem.

According to the author of study [27], smartphones are not secure in terms of establishing mobile hybrid applications. Because of the nature of open-source programming, privacy concerns are dominant in Android development. It tends to delete sensitive data, posing numerous hazards. When information is accessed, changed, or transferred, there is a fundamental problem with data handling. To address this problem, a system for recovering files from disc fragments is proposed, as well as various data remanence mitigation measures [28].

Asynchronous structures from asynchronous programming, which is a basic aspect of mobile development, are used by mobile developers. They do so to minimize unresponsiveness; however, the problem highlighted in the research is that mobile developers are unable to employ these constructs consistently, resulting in memory leaks, lost results, and wasted energy [29].

## 3. Research Methodology

This section briefly describes the research design. A systematic review of the existing literature is conducted. We have adopted the standard practice of Kitchenham [31], the widely used decorum for conducting systematic reviews. It provides a broad review and critical examination of all related existing research in and around the area of software engineering. It helps in conducting a structured and even-handed literature review, having a predefined search strategy. As mentioned above, the primary objective of selecting SLR is to come up with the challenges that are faced during the implementation of the requirement engineering process, particularly for the mobile domain [3]. The qualitative analysis is performed by using data coding techniques of the grounded theory [32,33]. In the end, a discovered list of challenges is reviewed, evaluated, and validated via Expert Review [34].

### *3.1. Resources, Keywords, and Search Strings*

Search terms and resources to be searched are part of the approach for finding a research study. The majority of journal papers accepted manuscripts, and mature conference proceedings are being retrieved in electronic databases that are accessible online. Researchers did not collect information from books or printed sources. The list of electronic databases from which publications are reviewed for SLR is shown in Table 2.

**Table 2.** Digital libraries.

| | URL |
|---|---|
| IEEE Xplorer | https://ieeexplore.ieee.org (accessed on 15 January 2023) |
| ScienceDirect | https://www.sciencedirect.com/ (accessed on 17 January 2023) |
| WileyOnline | https://onlinelibrary.wiley.com/ (accessed on 23 January 2023) |
| ACM | https://dl.acm.org/ (accessed on 25 January 2023) |
| Springer | https://link.springer.com (accessed on 25 January 2023) |
| MDPI | https://www.mdpi.com (accessed on 25 January 2023) |

Researchers performed searches on the chosen digital libraries using the search queries composed of the keywords selected based on the following strategies: researchers have derived significant terms and keywords from the goal of our investigation. Additionally, the related terms and alternative synonyms or keywords are identified. While most papers are found with the privacy and security perspective. Researchers have developed a total of 176 search strings based on our keywords with related synonyms. Table 3 shows the possible keywords along with their related synonyms.

**Table 3.** Keywords.

| Keywords | Synonyms |
|---|---|
| Green/Sustainable Requirement Engineering Process | Sustainable Requirement Engineering, Sustainable Requirements, Sustainable Requirement Gathering, Sustainable Requirement Inception, Sustainable Requirement Elicitation, Sustainable Requirement Prioritization, Sustainable Requirement Validation, Sustainable Requirement Specification |
| Sustainable Mobile Application Development | Sustainable Mobile Platform, Sustainable Mobile Applications, Sustainable Mobile Development |
| Challenges | Issues, Problems, Barriers |

### *3.2. Inclusion/Exclusion Criteria*

Manuscripts published between 2010 and 2022 are included. The inclusion and exclusion criteria for selecting the papers in this review are as follows. It is ensured that mobile application development-related published work from 2010–2022 is included. Furthermore, journal articles, conference proceedings, research, review articles, and (accepted manuscript) are included.

The first stage of the search was undertaken based on title and time range, which was 2010–2022. Then, in stage two, a filter was used to identify the most relevant and thorough documents. Finally, selected papers were categorized based on search strings and inclusion criteria for selection from the selected content.

Using the search query mentioned earlier, we conducted systematic searches in six digital libraries: IEEE, ScienceDirect, WileyOnlineLibrary, Springer, MDPI, and ACM Digital Library. Researchers found 526 papers based on the time frame, which is from 2010 to 2022. In the first stage, researchers had a total of 230 papers based on titles and keywords. Then, after a thorough reading of the abstract of selected primary studies, researchers included 85 papers, from which 70 were repeated and were included only once. The papers were then vetted based on the criteria for inclusion and exclusion, and 20 papers were chosen. As a result, researchers have narrowed it down to about 14 papers. This was solely for a single source: IEEE. The same procedure is followed for other databases. The search and study selection method for the research, comprising all selected databases, is shown in Table 4.

**Table 4.** Study selection and distribution statistics.

| Electronic Database | Papers Found | Stage 1: Title and Keyword | Stage 2: Abstract | Stage 3: Repeated | Stage 4: Full Paper Read | Stage 5: Quality Assessment |
|---|---|---|---|---|---|---|
| IEEE | 526 | 230 | 85 | 70 | 20 | 14 |
| ScienceDirect | 31,772 | 21,929 | 2198 | 1361 | 33 | 26 |
| WileyOnline | 27,531 | 2903 | 1473 | 94 | 47 | 23 |
| ACM | 14,685 | 788 | 518 | 50 | 45 | 37 |
| Springer | 326 | 63 | 34 | 8 | 2 | 0 |
| MDPI | 457 | 116 | 26 | 6 | 4 | 0 |

Figure 1 shows the pictorial representation of the selection process of research studies. The pattern shows that the maximum studies were found in the ACM Digital Library. It correlates with the essence of maximum acknowledgment of software engineering-related publications.

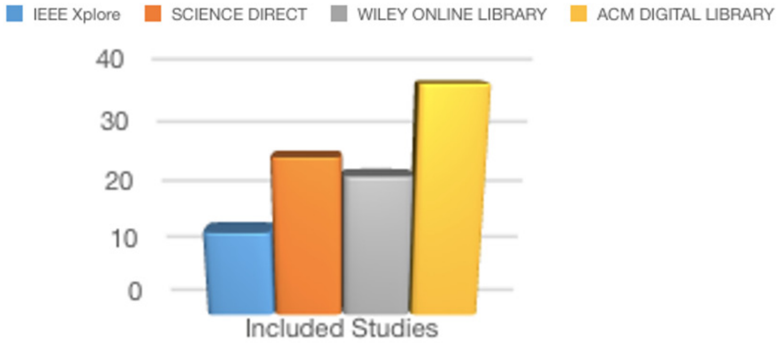

**Figure 1.** Distribution of studies on the basis of databases.

Similarly, Figure 2 shows the research paper distribution year-wise. The pattern shows that throughout these years the state of knowledge regarding sustainable requirement engineering for sustainable mobile development has grown.

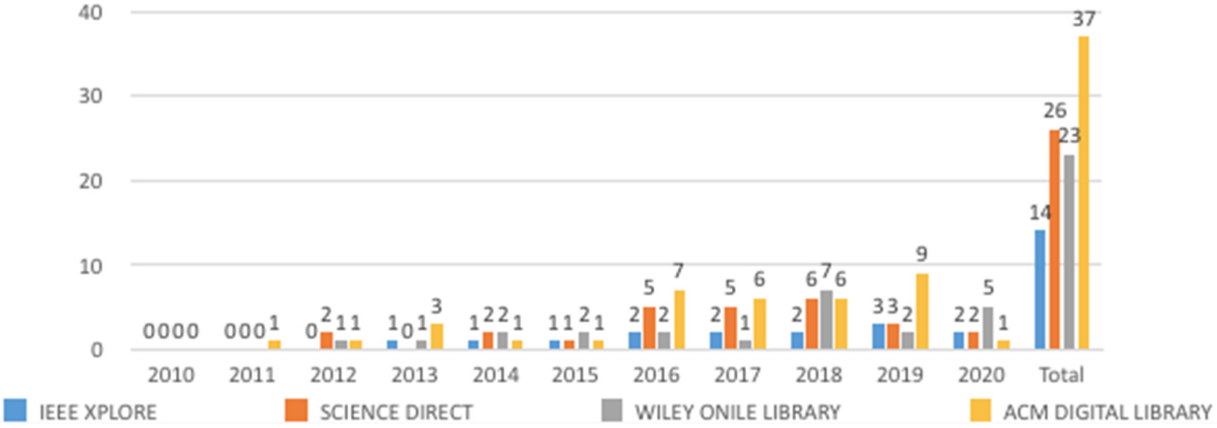

**Figure 2.** Distribution of studies on the basis of year of publication.

The study included the most relevant papers from journals and conferences. Figure 3 shows the distribution of the journal and conference papers.

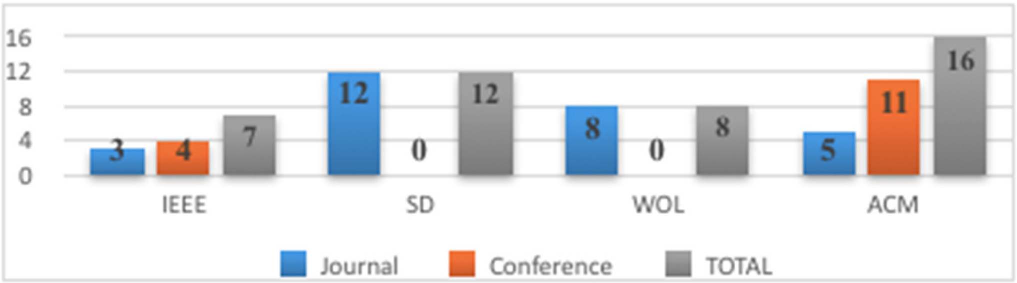

**Figure 3.** Distribution of studies on the basis of type of paper.

As shown in Figure 3, out of 43 selected papers, 29 papers were journal papers and 15 were conference papers. All these studies are peer-reviewed publications, published in high-quality and mature journals/conferences. Likewise, Figure 4 shows the type of included papers. The included papers were either categorized as 'research articles' or 'review articles', or as 'accepted manuscripts'.

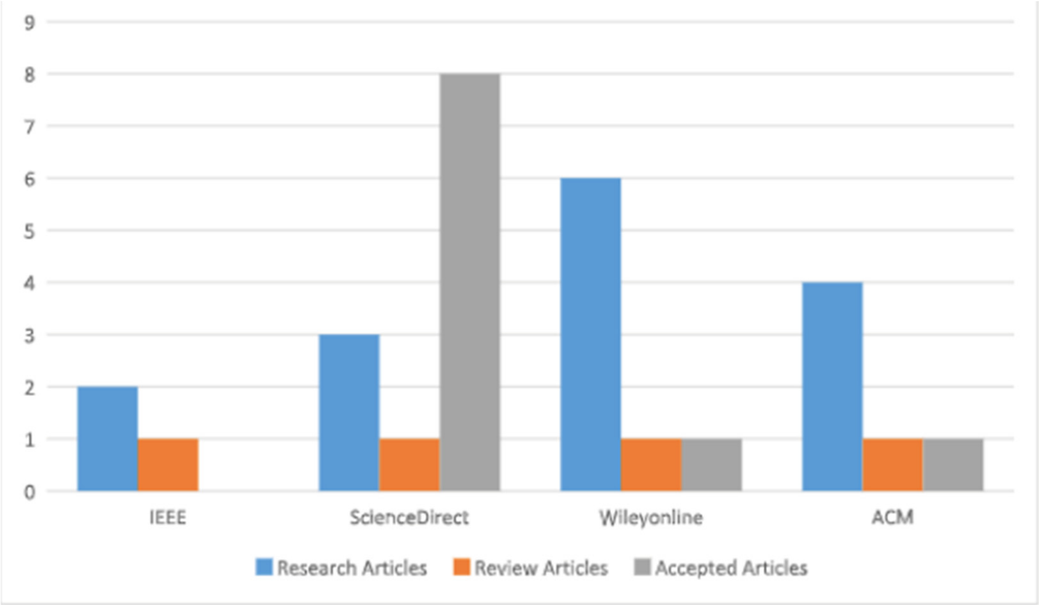

**Figure 4.** Distribution of studies on the basis of article type.

As shown in Figure 4, the maximum number of papers were 'Research articles', that are published in high-impact quality peer-reviewed journals/conferences. Eight included papers were under the 'accepted manuscript' type and only one included paper was of the type 'review article'. The overall representation of the searching and filtration process is shown in Figure 5.

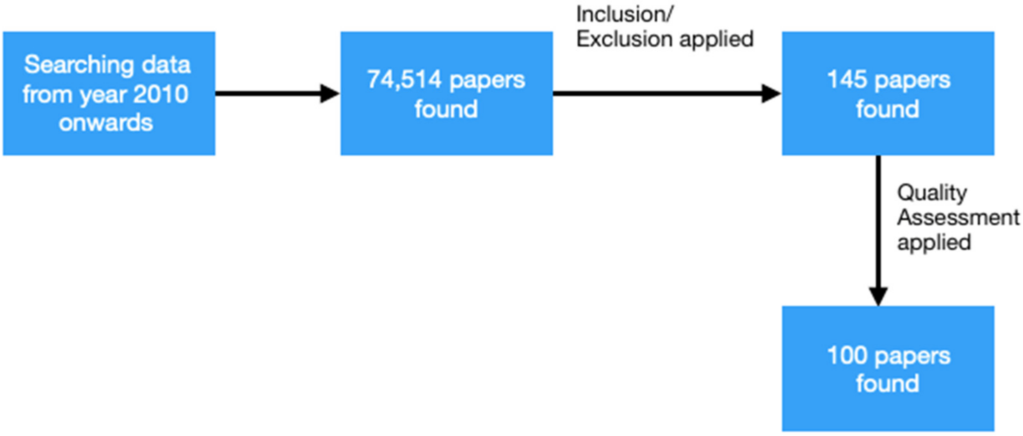
Searching data from year 2010 onwards
74,514 papers found
Inclusion/ Exclusion applied
145 papers found
Quality Assessment applied
100 papers found

**Figure 5.** Process of searching and filtering research studies.

As shown in Figure 5, firstly, the research papers related to sustainable requirement engineering for sustainable mobile application development were searched from various databases. In total, 74,514 papers were found. After applying the inclusion/exclusion strategy, 145 papers were left. The quality assessment process was conducted to further validate the selected papers. Quality assessment is aimed to evaluate the worth of your selected studies to avoid bias. Researchers have selected questions in the form of questionnaires adapted from [35], based on which, the respondents have to give answers to each question for the selected study. The answers are given based on a certain scale of Yes (1), No (0), and Partial (0.5). The studies were distributed among various researchers who belong to the software engineering domain and have knowledge of sustainable requirement engineering besides mobile application development. As a result, 100 papers were left, which were thoroughly studied for the investigation.

## 4. Results and Discussions

Based on the research objective, researchers have a list of challenges tabulated in the next section. Based on the findings, researchers have categorized the challenges into five sub-categories, under nature requirement-related, resource-related, communication-related, security and privacy-related, and stakeholder-related domains. This section of the study aimed to describe the findings achieved. At the first stage of review, researchers have identified a total of 113 data units with the help of data coding techniques of grounded theory. After applying data encoding and implicit/explicit duplication removal among the data units, 92 challenges along with the 5 sub-categories were identified.

The first research objective was to discover the mobile developer-related challenges faced during R.E.'s accomplishment while developing a sustainable mobile application. The identified challenges are tabularized in Table 5. The table comprises four columns namely 'Paper Id', 'Ref #', 'Category', and 'Challenge'.

**Table 5.** Challenges for sustainable green requirement engineering.

| Paper Id | Ref Number | Category | Challenge |
|---|---|---|---|
| X1 | [36] | Communication | Anonymous communication |
| X4 | [3] | | Lack of effective requirement articulation |
| X4 | [3] | | Lack of verbal and presentation skills |
| X4 | [3] | | Lack of communication participation |
| W3 | [37] | | Low bandwidth |
| W3 | [37] | | Unavailability of services for network connection |
| W3 | [37] | | Heterogeneity in terms of the involvement of different networks |
| W13 | [38] | | Frequent server disconnections |
| W13 | [38] | | Problems in accessing server data |
| A9 | [39] | | Data communication overheads |
| X4 | [3] | Requirements | Incomplete requirements gathering |
| X4 | [3] | | Lack of accurate requirement prioritization |
| X4, X11 | [3,40] | | Unstable requirements |
| X4 | [3] | | Change of user needs and understanding |
| X4 | [3] | | Requirement over-scoping |
| X11 | [40] | | Guideline compliance |
| X12 | [41] | | The problem of minimization of service latency |
| X12 | [41] | | The optimal revenue maximization problem |
| X12 | [41] | | Absence of high quality-of-service requirement to offload the computational service offloading |
| X13 | [42] | | Inefficient provisioning and delivery of mobile application |
| S1, S15, S14 | [17,43,44] | | Lack of consideration of user and application requirements for offloading decision making |
| S2, S4 | [26,45] | | Incorrect requirement partitioning |
| S4 | [45] | | Lack of useful information extraction |
| S8, S12, S21, S22 | [46–49] | | Inefficiency in constructing power models |
| W3 | [37] | | CO issues in terms of energy savings (static environment) |
| X6, S23, W7 | [50–52] | | Complexity in GUI testing |
| W8 | [53] | | Lack of appropriate battery life |
| W9 | [54] | | Complexity in evaluating applications' performances |

**Table 5.** *Cont.*

| Paper Id | Ref Number | Category | Challenge |
|---|---|---|---|
| W11 | [55] | Requirements | Time consumption of mobile devices |
| W11, W18 | [56,57] | | Energy consumption of mobile devices |
| A1 | [58] | | Diversity of mobile surroundings |
| W15 | [20] | | Changing the orientation of the app |
| W17 | [56] | | Inefficient web browsing on mobile systems |
| W21 | [59] | | Problem with touch gesture and embedded sensor-based interfaces |
| W22, W23 | [60,61] | | Incorrect requirement identification |
| A7 | [62] | | Complexity in bundling of HTTP requests |
| A9 | [39] | | Depletion of battery power |
| A11 | [63] | | Event-driven structure |
| A11 | [63] | | Complex contextual features |
| A14 | [64] | | High power consumption of smartphones |
| A14 | [64] | | Surface over-heating of smartphones |
| A18 | [65] | | Impact of delay on UX |
| A20 | [38] | | Impact of network on the performance of mobile devices |
| A21, A27 | [66,67] | | Environmental factors' impacts on the biometric authentication method |
| A25, A26, A32 | [68–70] | | Lack of power requirement for battery of mobile devices |
| A33 | [71] | | Lack of quality assurance |
| A26 | [69] | | Frequent changing requirements |
| A36 | [72] | | Inadequate app performance |
| W10, X3, A5 | [73–75] | | Inefficient requirement completion time |
| X2, X9, X11, W20, W13 | [18,38,40,73,76,77] | Resource | Lack of appropriate resource allocation |
| X3, S11, S13, S18, S22, S23, S25, S26 | [74,78–84] | | Platform incompatibility |
| W4 | [85] | | Inaccurate task scheduling |
| X5, W6 | [22,86] | | Service selection problem |
| X7 | [29] | | DVFS-control problem |
| X8, A5 | [75] | | Lack of resource optimization |
| X8, A5 | [75] | | Energy inefficiency |
| W4, W13 | [38,85] | | Limited resources/resources lacking |
| W12 | [87] | | VM-migration problem |
| S18 | [80] | | Higher resource utilization by mobile cross-platform development approaches |
| S23, A1 | [58,83] | | Lacking/limited computational resources |
| A3, A4 | [88,89] | | Incorrect estimation of battery life |
| A12, A13 | [23,24] | | Lack of accurate quantification about the consumption of energy by the app |
| A24, A27, A29 | [69,90,91] | | Compatibility across various OS versions |

**Table 5.** *Cont.*

| Paper Id | Ref Number | Category | Challenge |
|---|---|---|---|
| X10 | [29] | Security and Privacy | Lack of asynchrony retrofitting |
| S3, S6, | [92,93] | | Malware detection problem |
| S6 | [93] | | Timing attacks |
| S7, W14 | [94,95] | | Tampering during offloading data |
| S8 | [46] | | Limitation of profilers |
| W3 | [37] | | CO in terms of network connection status and bandwidth |
| W3 | [37] | | Security w.r.t. mobile users |
| W3, S24, A16 | [37,96,97] | | Data security on the cloud |
| W14, S19, W19 | [95,98,99] | | Side-glance attack |
| W15, A2 | [20,100] | | Lack of identification of risky actions and vulnerabilities |
| A4, A6 | [101,102] | | OTP-vulnerability |
| A8, A15 | [103,104] | | Inconsistent and inefficient testing |
| A10 | [27] | | Lack of configuration of mobile hybrid apps |
| S20, A16, A17 | [23,24,96] | | Unclear requirements for app functionality toward privacy threats |
| A19, A22, A30 | [105–107] | | Patching for updating, correction, or improvement |
| A34, A35, | [108,109] | | Data reconstruction attack |
| A28 | [28] | | Data erasure |
| X4 | [3] | Stakeholder | Unawareness of needs |
| X4 | [3] | | Cultural and language barrier |
| X4 | [3] | | Lack of domain knowledge |
| X4 | [3] | | Ambiguities |
| X4 | [3] | | Conflicts |
| X8, A5 | [65,75] | | Response time inefficiency |
| W23 | [110] | | Ignorance of usability |
| S5 | [111] | | Lack of Requirement Task Efficiency and Responsiveness |
| S9 | [112] | | Lack of knowledge regarding development standards and practices |
| S10 | [30] | | Testing issues |
| A31 | [113] | | Impact of screen size on usability |
| A36 | [72] | | Poor user experience |

## *4.1. Communication*

### 4.1.1. Anonymous Communication

An anonymous communications strategy conceals the IP address from the host server the user is accessing [36]. Mobile developers face a challenge when it comes to sustainable mobile development or the mobile domain, especially when building mobile applications, such as the Watts app, Telegram, and others. These apps use end-to-end encryption to ensure that users' data are kept safe. Those programs that do not use encryption techniques, on the other hand, have a challenge.

### 4.1.2. Lack of Requirement Effective Articulation

This issue is centered on the rivalry between stakeholders. Because of the diversity of the mobile environment, both consumers and developers evaluate needs from different perspectives, resulting in disagreements [3].

#### 4.1.3. Lack of Verbal and Presentation Skills

The goal of this challenge is to explore oneself and perceptions in a way that is difficult for others to comprehend. One may be unable to share ideas or offer his or her opinions and thoughts to other stakeholders due to the diversity and complexity of the mobile development domain, posing a significant difficulty in sustainable mobile development [3].

#### 4.1.4. Lack of Communication Participation

This problem is distinct from the preceding one in that the person is skilled but unwilling to communicate his or her opinions [3]. As the domain of mobile applications develops in a new dimension of sustainability, novel expressions and technologies emerge that can be difficult to explain or communicate because of crowd phobia, which leads to hypertension.

#### 4.1.5. Low Bandwidth

Bandwidth refers to the amount of data that can be sent between two points on a network. The bandwidth of your network or internet connection is the maximum Mbps transmission rate. It also has an impact on how fast your internet is. Any connection that is faster than 25 Mbps is considered decent, thus larger families with 3–5 people should search for rates between 200 and 300 Mbps. Low bandwidth might be caused by the Mbps rate of your existing broadband connection, or it can be slowed down because there are too many users connected to the network, and in some cases, to your device [37].

#### 4.1.6. Unavailability of Services for Network Connection

This issue mostly concerns essential services or services that are not available at the required time for network connection, posing a significant barrier to sustainable mobile applications. In other words, due to network congestion, mobile users are unable to connect to the cloud [37].

#### 4.1.7. Heterogeneity in Terms of Involvement of Different Networks

Diversification is a term used to describe heterogeneity. It has everything to do with MCC's participation in numerous networks. Managing multiple wireless connections while adhering to MCC standards is difficult, as it poses problems in such a context, making sustainable mobile development difficult [37].

#### 4.1.8. Frequent Server Disconnections

The frequent server disconnections that mobile client users suffer from are one of the major impediments to the development of mobile information systems as a top priority. As a result, data access from the server in the wireless network becomes expensive, posing a barrier to the sustainable mobile environment [38].

#### 4.1.9. Problems in Accessing Server Data

The server will inevitably become disconnected as a result of high data traffic, making it impossible for mobile users to access data from the server and resulting in expensive wireless network expenses in a sustainable mobile environment [38].

#### 4.1.10. Data Communication Overheads

On mobile devices, the major usability challenge is the rapid consumption of battery power. Data connectivity is the foremost waste of mobile device energy after the display. End-user requirements and sustainable mobile application requirements have not kept pace with advances. The device will consume scarce energy due to minimum data transfer via the interface, but this does not happen, making data communication difficult [39].

### *4.2. Requirement*

#### 4.2.1. Incomplete Requirement Gathering

This challenge focuses on the requirements gathering phase. Incomplete requirement gathering refers to the lack of vital and mandatory requirements [3]. Due to the vastness of the mobile domain, it is expected that obligatory restrictions may lag or be ignored at times.

#### 4.2.2. Lack of Accurate Requirement Prioritization

One of the processes in the requirement engineering process is requirement prioritization, often known as software requirement prioritization. This problem emerges as a result of a lack of requirements prioritization [3]. The sustainable mobile domain is made up of a range of events that might be tough to manage since it is difficult to grasp. To comprehend such event management, a knowledgeable specialist is required. As a result, it is not always clear which criterion should take precedence or be given top priority in this situation.

#### 4.2.3. Unstable Requirements

Unstable needs do not follow a predictable pattern. Alternatively, the requirements might not have met their expectations [3,40]. As previously said, handling sensitive information in a mobile context is necessary. The intricacy of sustainable mobile requirements prevents the requirement analyst or mobile developer from understanding them.

#### 4.2.4. Change of User Needs and Understanding

The issue is that users' needs evolve. In other words, user needs change over time, leaving them confused about what they desire. Changes in user needs can cause misunderstandings about users' genuine needs [3]. As a result of the innovation and introduction of new complicated technologies, user needs, combined with changes in mobile development, have created a barrier to sustainable mobile platforms.

#### 4.2.5. Requirement Over-Scoping

This challenge is focused on the scope's inaccuracy. The requirements, or project scope, are either under or over the set limit. Alternatively, the scope of requirements is not clearly defined, resulting in stakeholder conflicts and dissatisfaction [3]. Due to the complexity of innovative technologies in mobile development, developing a scope for sustainable mobile requirements is tough in this situation.

#### 4.2.6. Lack of Guideline Compliance

There have been many security proposals for sustainable mobile development, but none of them are technically possible or meet security standards. The failure to follow recommendations is a concern [40].

#### 4.2.7. Inefficient Requirement Completion Time

When a need is not fulfilled in the time provided, it is referred to as "inefficient requirement completion time". Alternatively, we can say that we failed to produce an optimal solution when the need could not be met within the estimated timeframe [73–75]. Sustainable mobile development varies from traditional development in that it requires more computational resources, such as memory and storage. Completing mobile needs within the required time range is difficult due to resource restrictions.

#### 4.2.8. Minimization of a Service Latency Problem

Mobile edge computing (MEC) is a revolutionary architecture that gives mobile base stations at the network's edge cloud computing capabilities. MEC's main feature is that it distributes mobile computing, network control, and storage to network edges (e.g., base stations and access points), allowing resource-constrained mobile devices to run computation-intensive and latency-critical applications and various purposes, such as traffic monitoring and healthcare. These systems are closer to users, allowing for seamless and low-latency access to edge services. In general, latency refers to a period. Service latency, on the other hand, refers to the time it takes for a service to be delivered. The delay in services from the server site is the subject of this challenge. As a result of the lack of resources compared to traditional development, the sustainable mobile platform suffers from service latency [41].

#### 4.2.9. The Optimal Revenue Maximization Problem

Similar to the previous problem, in the MEC environment, edge devices make computational offloading requests to edge service providers that include source and destination addresses as well as offloading time intervals. Edge service providers create a revenue maximization dilemma at the edge. This problem stems from the fact that the edge service provider is unable to give the maximum number of services to gain the most income. Sustainable mobile platforms, in comparison to traditional web development, are constrained by restricted resources [41].

#### 4.2.10. Absence of High Quality-Of-Service Requirement for Computational Service Offloading

In comparison to traditional web development, sustainable mobile development lacks high-quality services for offloading computational services because the mobile environment is hampered by processing resources, rendering offloading computational services difficult [41].

#### 4.2.11. Inefficient Provisioning and Delivery of Mobile Application

Mobile consumers have sporadic and diverse computing needs. As a result, this issue concerns the efficient transmission of data and the fulfillment of user requests using cloud IT resources. Because mobile devices have limited resources, it is difficult to meet user requests while maintaining QoS in the face of intermittent network access at sustainable platforms [42].

#### 4.2.12. Lack of Consideration of User and Applications Requirements for Offloading Decision Making

The failure to address either user or application requirements, or both types of requirements, while offloading decision-making, is the source of this problem. Because of the limited resources available in a sustainable mobile environment, user and application needs were often overlooked or ignored while making decisions on the migration of computation-intensive tasks [17,43,44].

#### 4.2.13. Incorrect Requirement Partitioning

The requirement is inappropriate. Partitioning means that the demands are not successfully partitioned or given the necessary partition due to a lack of resources in the mobile domain, which can be problematic for sustainable mobile development. Alternatively, inadequate partitioning of jobs to be run in the cloud versus those to be run on mobile devices for computation offloading could be the cause of this problem [26,45].

#### 4.2.14. Lack of Extraction of Useful Information

This issue is specifically pertinent to crowd-sensing applications on mobile devices. Crowdsensing (MCS) is a method in which people disseminate data and elicit information using mobile devices to process for the common interest. Because these types of applications execute in a dynamic platform, it is critical to schedule sensing processes in an efficient manner [45].

#### 4.2.15. Inefficiency in Constructing Power Models

To begin with, it is related to the precision of several factors, including changing power co-efficient, model design for one or more specific smartphones, and system accessibility limitations. Second, while models are designed to be as realistic as possible, they are not capable of meeting all potential operational situations. In addition, recent technological advancements have rendered traditional power models and associated devices obsolete [46–49].

#### 4.2.16. Issues in Terms of Energy Savings (Static Environment)

The technique of offloading compute-intensive workloads to other platforms, such as the cloud, is known as computation offloading. Energy savings are not as successful in the cloud environment, particularly when it comes to sustainable mobile applications, as they are in the case of code compilation, because in fewer data scenarios, high energy is utilized than in local processing [37].

#### 4.2.17. Complexity in Graphical User Interface (GUI) Testing

GUI testing is a sort of mobile application testing that verifies the functionality of the user interface components. Manual or automated GUI testing for sustainable development might take significant time and effort [50–52].

#### 4.2.18. Lack of Appropriate Battery Life

MCC (mobile cloud computing) is a modern technology used to execute intricate programs on mobile devices. To boost their suppleness and mobility, users using mobiles are encouraged to use their devices for a variety of tasks. Mobile devices, on the other hand, have a short battery life, which puts such advantages to the test [53].

#### 4.2.19. Complexity in Evaluating Applications' Performances

Due to the scarce resources obtainable, a rigorous evaluation of the performance of the mobile application is critical. The difficulty of the operation is increased by the broad variety of sustainable mobile devices accessible [54].

#### 4.2.20. Time Consumption of Mobile Devices

Mobile platforms have fewer computing capabilities when compared to traditional or online development. So, when we talk about multiple computing environments at sustainable platforms, such as MCC or MEC, ewe are talking about environments that cannot supply on-demand resources and ill cannot fulfil requests from user/client sites. As a result, time consumption becomes a challenge in such contexts [55].

#### 4.2.21. Energy Consumption of Mobile Devices

Similar to the previously mentioned issue, the unavailability of on-demand resources due to the limited processing capabilities of cloudlets in MCC or MEC makes it difficult to accomplish large tasks. As a result, mobile devices consume more energy, posing difficulty in the mobile environment [56,57].

#### 4.2.22. Changing the Orientation of the Application

Converting between both the running app's portrait orientation layout options is a distinct phenomenon in mobile application development platforms, commonly mentioned as the difficulty of shifting orientation. Due to this reason, it is recommended that the new layouts should be adopted. It is to avoid issues like memory leakage. On the other hand, it is difficult to execute this in a practical environment, and as a result, the programming challenges can be seen by Android programmers [20].

#### 4.2.23. Inefficient Web Browsing on Mobile Systems

The slowness of sustainable mobile web browsing is caused by the limited processing capabilities of mobile devices, wireless latency, and incremental page or resource loading. The browser renders resources [56] in between two successive resource downloads.

#### 4.2.24. Problem with Touch Gesture and Embedded Sensor-Based Interfaces

This issue is addressed in particular in mobile computer-aided design. Mobile CAD refers to software that is typically used in mobile devices. It is characterized as computer-based software that aids in the creation of 2D and 3D model generation procedures. A touch gesture is a feature that allows the user to interact with an application by using their hands. Incorporated sensor-based interfaces, on the other hand, are the interfaces used to communicate with sensors that are embedded in mobile devices. The problem is that these interfaces obstruct the user's view of the display and are therefore unsuitable for the task in a sustainable environment [59].

#### 4.2.25. Incorrect Requirements Identification

Because mobile development is such a big and diverse field, the software requirements for mobile devices were not appropriately identified or omitted due to their heterogeneous nature, resulting in software failures [60,61].

#### 4.2.26. Diversity of Mobile Surroundings

The rise of a diverse variety of mobile platforms, each designed in its programming language, has resulted in a diversified sustainable mobile environment, creating a challenge for sustainable mobile platforms [58].

#### 4.2.27. Complexity in Bundling of HTTP Requests

HTTP requests are an energy-intensive task. It is known through research that the combination of the HTTP requests to generate a request that is a bigger HTTP request can reduce consumption of the network communication energy consumption. As a consequence, bundling is a complex technique to carry out [62].

#### 4.2.28. Depletion of Battery Power

Because computing resources are limited in the mobile environment, sustainable mobile applications must rely on device energy to execute. As a result, the depletion of mobile device battery power poses a challenge [39].

#### 4.2.29. Event-Driven Structure

The main focus of this challenge is on numerous events occurring at the same time in a sustainable mobile context. The mobile environment is constrained by a complex structure that contains several events that cannot be handled appropriately at the same time. As a result, it poses a new challenge to the mobile environment [63].

#### 4.2.30. Complex Contextual Features

Contextual elements in mobile development, such as sensor handling, are challenging to incorporate due to dependency in platform dependence, resulting in roadblocks in the sustainable mobile environment [63].

#### 4.2.31. High Power Consumption of Smartphones

In comparison to traditional development, mobile environments require more resources for high-performance and resource-intensive jobs, which is troublesome in such a setting. This causes a large battery drain, which is a problem in a mobile context [64].

#### 4.2.32. Surface Over-Heating of Smartphones

Similar to the previously mentioned difficulty, increased power consumption results in surface rising temperatures of the smartphone, posing a barrier for sustainable mobile systems [64].

#### 4.2.33. Impact of Delay on UX

Frequently using mobile applications should be efficient, like with health and situational awareness-related applications. Delaying the execution will utilize less energy and is an attractive method for energy savings. Incorporating delays to save power can harm the user experience [65].

#### 4.2.34. Impact of Network on the Performance of Mobile Devices

Mobile applications use internet services to access multimedia content, allowing users to share and receive photographs, audio, and video from the internet using their mobile devices. Because multimedia content exists, application developers, users, and manufacturers place a premium on energy conservation. Mobile devices, as well as the wireless networks and communications that support them, are faced with a variety of restraints, such as conditions related to battery or storage capacity, and others. This ultimately influences the overall network performance [38].

#### 4.2.35. Environmental Factors' Impact on the Biometric Authentication Method

Recent biometric techniques are majorly reliant on various external factors. Lighting, camera motion, and picture framing, for example, affect face authentication, whereas finger location affects fingerprint scanning. All of these traits make handling such systems complex for users, as well as time-consuming [66,67].

#### 4.2.36. Lack of Power Requirement for Battery of Mobile Devices

There are a lot of advancements (such as augmented reality, and others) in mobile technologies which are putting an exponentially higher demand on the battery. Furthermore, to improve usability, manufacturers of mobile devices are trying to create lighter and thinner devices, which require a significant amount of power while operating. This leads to battery life being one of the most popular complaints about current mobile devices [68–70].

#### 4.2.37. Frequent Changing Requirements

To deal with new needs, adjust to changing conditions, resolve errors, and improve software architecture, software programmers must reflect on changes. The quality of the software may worsen due to aging, independent of the type of improvements made to mobile applications or mobile software. This leads to poor design and implementation choices, and the quality of mobile software, as well as the performance of software applications, deteriorates with time. Wrong selections are indicated by anti-patterns and a code smell in the code [69].

#### 4.2.38. Lack of Quality Assurance

Whether it is a web application or a mobile application, software quality is paramount. However, testing and producing a high-quality application is tough in the case of mobile development. The expense of testing a mobile application is significant, as are the levels of complexity. In addition to its high cost, it takes a long time to develop a high-quality application for mobile devices [71].

#### 4.2.39. Inadequate Applications Performance

The introduction of mobile applications as new software systems has resulted in the emergence of new performance development needs. App location performance can be severely hampered by development techniques that do not adhere to these needs [72].

### *4.3. Resources*

#### 4.3.1. Lack of Appropriate Resource Allocation

Due to mobile application development-based resource constraints, tasks that require greater resource allotment may not be allocated, or the work may require fewer resources if given more allotment, resulting in resource wastage and difficulty in sustainable mobile application development [18,38,40,76,77].

#### 4.3.2. Platform Incompatibility

This is a problem with platforms that are incompatible with one another. As a result of insufficient resources, a few platforms and devices can be supported by the developers. This leads to platform incompatibility, which is a problem for sustainable mobile developers [74,78–84].

#### 4.3.3. Service Selection Problem

MEC (multi access edge computing) gives high quality services to customers through tasks execution at the network, but as there are scarce resources and the difficulty of service needs, selecting the appropriate services to reduce user interaction latency, and mobile device energy consumption is a major challenge [22,86].

#### 4.3.4. DVFS-Control Problem

DVFS alters the power and speed settings on a computer's different CPUs, controller chips, and peripheral devices to optimize resource allocation for activities while also maximizing power savings when those resources are not in use. DVFS allows devices to perform needed functions while using the least amount of power possible. In addition, the recent practice of combining many cores in mobile processors exacerbates the problem. In this situation, we must synchronize the power modes for effectiveness and efficiency, which raises the DVFS control complexity enormously, making power management policy implementation difficult [29].

#### 4.3.5. Lack of Resource Optimization

The mobile environment, in comparison to the web environment, lacks optimal resources or is unable to supply the needed resources for job execution, posing difficulty [75].

#### 4.3.6. Energy Inefficiency

Due to the energy efficiency challenge, computation-intensive occupations in a mobile context require more energy due to limited resources [75].

#### 4.3.7. Inaccurate Task Scheduling

This problem stems from the ineffective scheduling of tasks in a mobile domain. Although mobile cloud computing (MCC) is a data processing system with a higher performance rate, it is critical to forecast the precise work scheduling. As a consequence, estimating and predicting the exact task scheduling of mobile applications on mobile platforms has become difficult [85].

#### 4.3.8. Limited Resources/Resources Lacking

As opposed to traditional development, the sustainable mobile environment is hampered by a lack of resources. We place great importance on performance due to the restricted resources available. Native applications are designed and maintained individually for each device as there are so many different mobile platforms, each with its operating system and hardware, resulting in a wide range of performance [38,85].

#### 4.3.9. VM-Migration Problem

In a MEC system, virtual machine (VM) migration is the process of relocating a virtual machine (VM) from one edge node to another. User mobility, on the other hand, is a problem when it comes to moving around freely within a running program's environment. To achieve the best performance in the MEC environment, it is necessary to choose how to transfer VMs between nodes [87].

#### 4.3.10. Higher Resource Utilization by Mobile Cross-Platform Development Approaches

The construction of software applications that are compliant with several mobile operating systems is known as cross-platform mobile development. Mobile cross-platform development (MCPD) methodologies are being used to create an increasing number of applications. MCPD techniques provide developers with several benefits when it comes to porting a mobile application across several platforms. These approaches, however, come at the expense of increased resource utilization on a mobile device [80].

#### 4.3.11. Lacking/Limited Computational Resources

Because mobile platforms have less memory and processing capability than typical PCs, they pose a challenge [58,83].

#### 4.3.12. Incorrect Estimation of Battery Life

Users of mobile devices are often worried about energy notifications, and they take methods to extend the battery life of their devices regularly. On the other hand, commercial smartphone platforms, such as Android and iOS, lack the ability that shows how much battery is remaining. As a result, itis difficult to predict the battery life available to running programmers [88,89].

#### 4.3.13. Inaccurate Quantification Regarding Energy Consumption by the Application

Sustainable mobile application developers struggle to measure energy consumption and its fluctuation due to uncontrollable factors, such as network congestion, and others [23,24].

#### 4.3.14. Compatibility across Various Operating Systems

Sustainable mobile application developers cannot test their applications because there are so many distinct OS versions. As a result, many problems occur, leading to a bad user experience [69,90,91].

### *4.4. Security and Privacy*

#### 4.4.1. Lack of Asynchrony Retrofitting

Programmers can use many asynchronous techniques in Android. However, developers can carry on employing ineffective asynchronous techniques, which can lead to problems, such as memory discharge, lost data, and wasted resources. Asynchronous retrofitting, on the other hand, is the proper application of asynchronous constructs to avoid the drawbacks outlined. The Android OS includes many async structures that developers can use. Data corruption, delayed information, and resource loss occur when mobile application developers are unable to apply async constructs or techniques [29].

#### 4.4.2. Malware Detection Problem

Mobile application security is becoming increasingly vital for sustainable mobile development. It is an appealing mark for unethical hackers to make use of flaws and breed malware [92,93].

#### 4.4.3. Timing Attacks

A timing attack is a form of side-channel attack in which an attacker examines how long a system takes to respond to various queries to corrupt it. Because offloading sometimes needs various sending or receiving times, it is sensitive to timing attacks [93].

#### 4.4.4. Data Tampering While Offloading

It relates to the intrusion of unknown dangers when the data object is transmitted to the public cloud. Consequently, security concerns occur, making the MCC environment's ability to obtain a high performance harder to achieve and offering a substantial obstacle to sustainable mobile development [94,95].

### 4.4.5. Limitation of Profilers

Profilers are software codes that monitor the settings of the operating system (surrogate and network). They also monitor the mobile device's accessible resources. Profilers must be lightweight in addition to having high-quality profiling data for cost effectiveness. The profiler's issue is lack of assessment towards offloading data-related benefits. This is due to the unpredictability of mobile device behavior during runtime [46].

### 4.4.6. CO in Terms of Network Connection Status and Bandwidth

This problem typically emerges in a dynamic network setting when data are broadcast but not received by the destination node or lost on the server side, causing a problem in a sustainable mobile environment [37].

### 4.4.7. Security w.r.t. Mobile Users

Due to limitations in processing and power in mobile devices, it is difficult to defend mobile devices from threats, and when using location-based services, such as GPS, there is a greater risk of private data leaking, such as current location [37].

### 4.4.8. Data Security on the Cloud

Data security also includes the integrity of user data on the cloud, which ignores mobile users' energy consumption; authentication, which considers the complexity of using difficult-to-remember passwords, which protects piracy of digital content, such as audio, video, and images from illegal access for content providers of MCC [37,96,97].

### 4.4.9. Side-Glance Attack

These attacks are possible while surfing the mobile page and using image code on a mobile device. A side-glance assault occurs when an attacker stares directly at the screen of a victim's mobile device in an attempt to steal the victim's identification and relevant security information [95,98,99].

### 4.4.10. Lack of Identification of Risky Actions and Vulnerabilities

Android has a built-in messaging system that helps them interact with each other. Because of the risks this system poses, it is vital to detect any potentially destructive activities or faults. Malicious applications can take advantage of Android's communication architecture to force other applications to perform unwanted actions [20,100].

### 4.4.11. OTP-Vulnerability

User passwords are leaked due to security breaches in user accounts. When such a situation occurs, Android application developers frequently employ the short messaging service to provide additional One-Time Password (OTP) authentication to strengthen the Password Authentication Protocol (PAP) (SMS). However, SMS is not designed to be a secure service [101,102].

### 4.4.12. Inconsistent and Inefficient Testing

Interfaces based on touch or gestures are widespread in mobile applications. Specialist software engineering methodologies have become more important as smartphones and tablets become more widespread. Software development demands frequent and effective testing to produce high-quality solutions. At the same time, testing these applications is still unmanageable, time-taking, and open to error. Touch-based interactions in smart phones, and gestures, are critical to incorporate [103,104].

### 4.4.13. Lack of Configuration of Mobile Hybrid Applications

These are web applications that connect to the mobile platform through a browser embedded in the application. The users of the smartphone are usually worried about their privacy, and they are reluctant to perform any transactions. Similarly, it is found that smartphones have some concerns regarding configuration. These concerns should not be considered as non-functional parts [27].

#### 4.4.14. Unclear Requirements for Application Functionality toward Privacy Threats

Android is a well-known platform that gives users a lot of access to personal and sensitive data, which raises severe privacy concerns. Several detection algorithms have been presented, however none of them apply privacy policies or define what is required for application performance. Google Maps, for example, needs the user's location to offer driving instructions, whereas an application related to weather may require the user's location to provide weather information [23,24,96].

#### 4.4.15. Patching for Updating, Correction, or Improvement

It is a set of changes to a computer program or its associated data to rectify or improve the program. It entails the correction of security weaknesses and other concerns, known as bug fixes. The security hole is, when discovered, repaired quickly. On the other side, this technique of patching is restricted to Google-connected or supported applications solving the patching problem. This method is not used to update software from other manufacturers or third parties [105–107].

#### 4.4.16. Data Reconstruction Attack

Considering the MCS environment, which comprises a cloud server architecture that is commonly used to integrate data from participants. Uploading sensor data to the cloud and allowing third parties to expose the participants leads to privacy issues because local sensor data contains or can be used to derive users' private information. Data reconstruction assaults develop as a result of this privacy breach, which is a serious barrier in the MCS sector [108,109].

#### 4.4.17. Data Erasure

Because Android is the most extensively used mobile operating system, it has been a source of privacy concerns. In Android OS, data erasure is a common concern. The way this operating system and its applications manage data is the most fundamental fault in it. Data reappearance after erroneous erasure could be problematic due to Android's lack of transparency regarding how third-party applications process user data saved on a mobile device [28].

### *4.5. Stakeholders*

#### 4.5.1. Unawareness of Needs

A mobile application's needs are frequently different from those of traditional development. Web development has its own set of requirements and demands, whereas sustainable mobile development has its own set of requirements and needs. Stakeholders' lack of awareness of demands could be problematic in this situation [3].

#### 4.5.2. Cultural and Language Barrier

Every organization must adhere to its own culture and language. Employees are expected to follow the company's established policies and procedures. The variance in practices and growing norms among distinct workplaces [3] is one of the most serious concerns in sustainable mobile development. When it comes to mobile platforms, this difficulty emerges when a client wants to construct an iOS native application, which necessitates the hiring of a SWIFT (programming language) developer, as well as a KOTLIN/Java developer. In a mobile context, however, if the software company lacks any of the aforementioned developers, a language barrier may exist.

#### 4.5.3. Lack of Domain Knowledge

Mobile development is a big and expansive domain that involves a wide range of themes as compared to online or traditional development. lWe need domain specialists in each area. Each programmer must be an authority on a specific subject. As a result, a lack of domain knowledge is one of the most significant hurdles for sustainable mobile developers [3].

#### 4.5.4. Ambiguities among Stakeholders

This issue stems from a disagreement among stakeholders. Divergent opinions come from the stakeholders' inability to agree on a particular topic [3]. We should assume the stakeholders are pursuing complicated business requirements that the development team may not be able to meet in this situation. As a result, there may be a problem or disagreement in this case.

#### 4.5.5. Intragroup Conflicts

This issue emerges when people commonly work in groups. As a result, it should go without saying that a point of contention or disagreement must be developed, which is a major hurdle in the mobile environment [3]. Code conflicts may happen during the merging process if a group of developers is working on the same project.

#### 4.5.6. Inefficient Response Time

The mobile environment, as previously stated, is an event-driven framework in which multiple events or tasks occur at the same time. As a result, in a sustainable mobile environment, completing several tasks or events is a time-consuming process that results in delayed response times [65,75].

#### 4.5.7. Lack of Requirement Task Efficiency and Responsiveness

For mobile applications to provide a wide range of benefits to consumers, the Android mobile operating architecture is critical. Inefficient execution approaches, on the other hand, cause severe performance and reaction time issues in existing Android applications. It is vital to maximize efficiency while still providing users in this case [111].

#### 4.5.8. Lack of Development Standards and Practices Knowledge

Applications are becoming increasingly popular, creating a security risk. Such applications are typically offered for free or for a low cost on all platforms, developed by novice programmers, and are open to attacks [112].

#### 4.5.9. Testing Issues for Practitioners

The demand to elicit testing requirements at the beginning of the development cycle, conduct research in a real-world development environment, testing of mobile services, and test the usability and security of the applications, are a few of the challenges for practitioners to handle [30].

#### 4.5.10. Ignorance of Usability

The quality of a user's experience when dealing with items or systems, such as websites, software, devices, or applications, is referred to as usability. Usability is concerned with the user's efficacy, efficiency, and overall satisfaction. Usability is a highly desired feature in web and sustainable mobile applications, although it is frequently overlooked. Learnability, effectiveness, and user pleasure are all enhanced by good user interfaces. However, usability is usually overlooked in the early stages of mobile application development and hence is rarely addressed in the architectural design of a system. Furthermore, because usability is frequently overlooked or postponed, estimating the time and effort spent especially on adding or improving usability mechanisms in software architecture and code is difficult. As a result, this led to a mobile development challenge [110].

#### 4.5.11. Impact of Screen Size on Usability

One of the most important aspects that affects user experience is screen size, claiming that internet searches are slower on mobile phones with small screens. Similarly, the efficacy of pupils' learning is constantly deteriorating when they use small-screen smartphones [113].

#### 4.5.12. Poor User Experience

New requirements have arisen as a result of the dramatic expansion in software applications, particularly for sustainable mobile platforms. In mobile app development, failing to achieve these standards results in poor user experience [72].

## 5. Threats to Validity

While undertaking this research study, researchers must keep several factors in mind before generalizing the findings. During the process of locating relevant content, we looked at both published and unpublished publications. However, we only examined approved manuscripts for previously unpublished work. Studies that appeared in journals or mature conference proceedings that were not published, on the other hand, may have gone unnoticed. Second, the majority of the research discovered was Wiley Online Library special issue papers, which were excluded. Because adding these publications requires us to think outside the box in terms of our research, researchers simply looked at generic applications and left out specific ones. For instance, the field of health and social applications and others. Primary studies that lacked any form of validation or testing on a large scale were simply eliminated. The identified literature was volumetric since researchers created 176 search strings in total of relevant keywords and related substitutes.

## 6. Conclusions

Mobile application development is becoming more popular day by day but the requirement elicitation process for such applications is still an area to investigate further. Researchers have merged mobile development with R.E. and identified challenges that the mobile developers faced during the execution of the entire R.E. process. For this purpose, a review has been conducted, and chose the guide of Kitchenham, the most popular and widely used SLR protocol. In total, 100 studies have been reviewed by exploring 4 electronic databases. Finally, a list of 92 unique challenges is contributed, which is validated by domain experts through expert review conduction. These challenges are related to 'Communication', 'Requirements', 'Resources', 'Security and Privacy', and 'stakeholders'. The list of challenges can become significant as it can help the mobile developers to recognize and consider them during requirement gathering for mobile applications. We intend to extend this research study by including other application domains (other than mobile applications) in the future. This study can further the existing state of knowledge for contributing to the list of challenges towards the requirement gathering process of mobile application development. Furthermore, it can also help practitioners, specifically those involved in the requirement gathering process, to carefully consider these challenges before executing the requirement engineering process.

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

**Funding:** This research is funded by Deanship of Scientific Research at Umm Al-Qura University, Grant Code: 22UQU4281768DSR10.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Acknowledgments:** The author would like to thank the Deanship of Scientific Research at Umm Al-Qura University for supporting this work by Grant Code: (22UQU4281768DSR10).

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

## References

1. Lin, Y. 10 Twitter Statistics Every Marketer Should Know in 2021. 2021. Available online: https://www.oberlo.com/blog/twitter-statistics (accessed on 13 January 2020).
2. Costi. No Best Solutions to Top Mobile App Development Challenges. 2017. Available online: https://blog.intelligentbee.com/2017/09/04/best-solutions-top-mobile-app-development-challenges/ (accessed on 15 January 2020).
3. Dar, H.; Lali, M.I.; Ashraf, H.; Ramzan, M.; Amjad, T.; Shahzad, B. A systematic study on software requirements elicitation techniques and its challenges in mobile application development. *IEEE Access* **2018**, *6*, 63859–63867. [CrossRef]
4. Zanutto, D.; Lorenzini, E.C.; Mantellato, R.; Colombatti, G.; Sanchez-Torres, A. Software Engineering Issues for Mobile Application Development. In Proceedings of the International Astronautical Congress IAC, Naples, Italy, 1–5 October 2012; Volume 4, pp. 2577–2585.
5. Liu, Z.; Gao, X.; Long, X. Adaptive random testing of mobile application. In Proceedings of the ICCET 2010 International Conference on Computers Engineering and Technology, Chengdu, China, 16–19 April 2010; Volume 2. [CrossRef]
6. UsmanMalik, M.; Chaudhry, N.M.; Shahzad Malik, K. Evaluation of Efficient Requirement Engineering Techniques in Agile Software Development. *Int. J. Comput. Appl.* **2013**, *83*, 24–29. [CrossRef]
7. Mandler, M. Coarse, Efficient Decision Making. *J. Eur. Econ. Assoc.* **2020**, *18*, 3006–3044. [CrossRef]
8. Souza, V.E.S. Requirements-Based Software System Adaptation. Ph.D. Thesis, University of Trento, Trento, Italy, 2012.
9. Soujanya, R.; Rupa, B. Test Cases and Testing Strategies for Mobile Apps—A Survey. *Int. Res. J. Eng. Technol.* **2017**, *4*, 3024–3027.
10. Codemotion. Mobile or Web Development: Which Is the Best Career Path. 2022. Available online: https://www.codemotion.com/magazine/soft-skills/it-careers/mobile-or-web-development-which-is-the-best-career-path-to-choose-in-2022/ (accessed on 11 September 2022).
11. Employees, G. How Hard Is Mobile App Development Compared to Web Development? 2021. Available online: https://medium.com/@globalemployees116/how-hard-is-mobile-app-development-compared-to-web-development-eae8def75247 (accessed on 1 January 2022).
12. Nyakundi, H. What Is the Difference between a Website and a Web Application? 2021. Available online: https://www.freecodecamp.org/news/difference-between-a-website-and-a-web-application/ (accessed on 20 February 2022).
13. Top 4 Mobile App Development Challenges and Solutions. 2020. Available online: https://www.nixsolutions.com/blog/top-4-mobile-app-development-challenges-solutions/ (accessed on 11 January 2022).
14. Tuban, A. 7 Proven Steps of the Mobile App Development Lifecycle. 2020. Available online: https://technologyrivers.com/blog/mobile-app-development-lifecycle/ (accessed on 20 February 2022).
15. Busch, Z. 6 Stages of the Mobile Development Lifecycle. 2019. Available online: https://learn.g2.com/mobile-development-lifecycle (accessed on 15 January 2021).
16. Olson, S. 8 Steps to Understanding the Mobile App Development Lifecycle—BuildFire. Build Fire, 2016. Available online: https://buildfire.com/understanding-mobile-app-development-lifecycle/ (accessed on 24 September 2021).
17. Shakarami, A.; Shahidinejad, A.; Ghobaei-Arani, M. A review on the computation offloading approaches in mobile edge computing: A game-theoretic perspective. *Softw. Pract. Exp.* **2020**, *50*, 1719–1759. [CrossRef]
18. Shahidinejad, A.; Ghobaei-Arani, M. Joint computation offloading and resource provisioning for edge-cloud computing environment: A machine learning-based approach. *Softw. Pract. Exp.* **2020**, *50*, 2212–2230. [CrossRef]
19. Usman, M.; Iqbal, M.Z.; Khan, M.U. An automated model-based approach for unit-level performance test generation of mobile applications. *J. Softw. Evol. Process* **2020**, *32*, e2215. [CrossRef]
20. Nirumand, A.; Zamani, B.; Ladani, B.T. VAnDroid: A framework for vulnerability analysis of Android applications using a model-driven reverse engineering technique. *Softw. Pract. Exp.* **2019**, *49*, 70–99. [CrossRef]
21. Cruz, L.; Abreu, R. EMaaS: Energy measurements as a service for mobile applications. In Proceedings of the 2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emergind Results, ICSE-NIER, Montreal, QC, Canada, 29–31 May 2019; Volume 2019, pp. 101–104. [CrossRef]
22. Kobayashi, R.; Adachi, K. Radio and Computing Resource Allocation for Minimizing Total Processing Completion Time in Mobile Edge Computing. *IEEE Access* **2019**, *7*, 141119–141132. [CrossRef]
23. Wang, X.; Continella, A.; Yang, Y.; He, Y.; Zhu, S. LeakDoctor: Toward Automatically Diagnosing Privacy Leaks in Mobile Applications. In *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*; Association for Computing Machinery: New York, NY, USA, 2019; Volume 3, pp. 1–25.

24. Wang, X.; Qin, X.; Hosseini, M.B.; Slavin, R.; Breaux, T.D.; Niu, J. GUILeak: Tracing privacy policy claims on user input data for Android applications. In Proceedings of the 40th International Conference on Software Engineering (ICSE '18), Gothenburg, Sweden, 27 May–3 June 2018; Association for Computing Machinery: New York, NY, USA, 2018; pp. 37–47. [CrossRef]
25. Amalfitano, D.; Riccio, V.; Paiva, A.C.R.; Fasolino, A.R. Why does the orientation change mess up my Android application? From GUI failures to code faults. *Softw. Test. Verif. Reliab.* **2018**, *28*, e1654. [CrossRef]
26. Zhang, W.L.; Guo, B.; Shen, Y.; Li, D.G.; Li, J.K. An energy-efficient algorithm for multi-site application partitioning in MCC. *Sustain. Comput. Inform. Syst.* **2018**, *18*, 45–53. [CrossRef]
27. AlJarrah, A.; Shehab, M. Cordovaconfig: A tool for mobile hybrid apps' configuration. In *ACM International Conference Proceeding Series*; Association for Computing Machinery: New York, NY, USA, 2018; pp. 161–170. [CrossRef]
28. Shu, J.; Zhang, Y.; Li, J.; Li, B.; Gu, D. Why data deletion fails? A study on deletion flaws and data remanence in android systems. *ACM Trans. Embed. Comput. Syst.* **2017**, *16*, 1–22. [CrossRef]
29. Lin, Y.; Okur, S.; Dig, D. Study and refactoring of android asynchronous programming. In Proceedings of the 2015 30th IEEE/ACM International Conference on Automated Software Engineering ASE, Lincoln, NE, USA, 9–13 November 2015; Volume 2016, pp. 224–235. [CrossRef]
30. Zein, S.; Salleh, N.; Grundy, J. A systematic mapping study of mobile application testing techniques. *J. Syst. Softw.* **2016**, *117*, 334–356. [CrossRef]
31. Keele, S. Guidelines for Performing Systematic Literature Reviews in Software Engineering. Technical Report, EBSE Technical Report EBSE-2007-01. Available online: https://www.elsevier.com/__data/promis_misc/525444systematicreviewsguide.pdf (accessed on 25 January 2023).
32. Charmaz, K. Grounded theory: Methodology and theory. In *International Encyclopedia of the Social & Behavioral Sciences*; SAGE Publications, Inc.: Thousand Oaks, CA, USA, 2001; pp. 6396–6399.
33. Vollsteedt, M.; Rezat, S. *Compendium for Early Career Researchers in Mathematics Education*; Springer: Berlin/Heidelberg, Germany, 2019. [CrossRef]
34. Ayyub, B. *A Practical Guide on Conducting Expert-Opinion Elicitation of Probabilities and Consequences for Corps Facilities*; Institute for Water Resources: Alexandria, VA, USA, 2001.
35. Zhou, Y.; Zhang, H.; Huang, X.; Yang, S.; Babar, M.A.; Tang, H. Quality assessment of systematic reviews in software engineering: A tertiary study. In *Proceedings of the EASE '15: Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering, Nanjing, China, 27–29 April 2015*; pp. 1–14. [CrossRef]
36. Sakai, K.; Sun, M.T.; Ku, W.S.; Wu, J. A framework for anonymous routing in delay tolerant networks. In Proceedings of the International Conference on Network Protocols ICNP, Toronto, ON, Canada, 10–13 October 2017; Volume 2017. [CrossRef]
37. Dinh, H.T.; Lee, C.; Niyato, D.; Wang, P. A survey of mobile cloud computing: Architecture, applications, and approaches. *Wirel. Commun. Mob. Comput.* **2013**, *13*, 1587–1611. [CrossRef]
38. Darwish, S.M.; El-Zoghabi, A.; El-Shnawy, A.G. Proactive cache replacement technique for mobile networks based on genetic programming. *IET Netw.* **2018**, *7*, 376–383. [CrossRef]
39. Dutta, K.; Vandermeer, D. Caching to reduce mobile app energy consumption. *ACM Trans. Web* **2017**, *12*, 1–30. [CrossRef]
40. Tien, C.W.; Huang, T.Y.; Huang, T.C.; Chung, W.H.; Kuo, S.Y. MAS: Mobile-Apps Assessment and Analysis System. In Proceedings of the 47th Annual IEEE/IFP International Conference Dependable Systems and Networks Workshop DSN-W, Denver, CO, USA, 26–29 June 2017; Volume 2017, pp. 145–148. [CrossRef]
41. Samanta, A.; Chang, Z. Adaptive service offloading for revenue maximization in mobile edge computing with delay-constraint. *IEEE Internet Things J.* **2019**, *6*, 3864–3872. [CrossRef]
42. Kumar, J.; Malik, A.; Dhurandher, S.K.; Nicopolitidis, P. Demand-Based Computation Offloading Framework for Mobile Devices. *IEEE Syst. J.* **2018**, *12*, 3693–3702. [CrossRef]
43. Junior, W.; Oliveira, E.; Santos, A.; Dias, K. A context-sensitive offloading system using machine-learning classification algorithms for mobile cloud environment. *Futur. Gener. Comput. Syst.* **2019**, *90*, 503–520. [CrossRef]
44. Goudarzi, M.; Zamani, M.; Haghighat, A.T. A fast hybrid multi-site computation offloading for mobile cloud computing. *J. Netw. Comput. Appl.* **2017**, *80*, 219–231. [CrossRef]
45. Antonić, A.; Marjanović, M.; Pripužić, K.; Podnar Žarko, I. A mobile crowd sensing ecosystem enabled by CUPUS: Cloud-based publish/subscribe middleware for the Internet of Things. *Futur. Gener. Comput. Syst.* **2016**, *56*, 607–622. [CrossRef]
46. Nguyen, Q.H.; Dressler, F. A smartphone perspective on computation offloading—A survey. *Comput. Commun.* **2020**, *159*, 133–154. [CrossRef]
47. Yoon, C.; Lee, S.; Choi, Y.; Ha, R.; Cha, H. Accurate power modeling of modern mobile application processors. *J. Syst. Archit.* **2017**, *81*, 17–31. [CrossRef]
48. Folino, G.; Pisani, F.S. Automatic offloading of mobile applications into the cloud by means of genetic programming. *Appl. Soft Comput. J.* **2014**, *25*, 253–265. [CrossRef]
49. Ahmed, E.; Gani, A.; Sookhak, M.; Hamid, S.H.A.; Xia, F. Application optimization in mobile cloud computing: Motivation, taxonomies; open challenges. *J. Netw. Comput. Appl.* **2015**, *52*, 52–68. [CrossRef]
50. Liu, Y.; Lee, M.J. Security-aware resource allocation for mobile cloud computing systems. In Proceedings of the International Conference on Computing and Communication Networks, ICCCN, Las Vegas, NV, USA, 3–6 August 2015; Volume 2015. [CrossRef]

51. Goudarzi, M.; Zamani, M.; Toroghi Haghighat, A. A genetic-based decision algorithm for multisite computation off loading in mobile cloud computing. *Int. J. Commun. Syst.* **2017**, *30*, e3241. [CrossRef]
52. Ali, A.; Maghawry, H.A.; Badr, N. Automated parallel GUI testing as a service for mobile applications. *J. Softw. Evol. Process* **2018**, *30*, e1963. [CrossRef]
53. Xu, W.; Daneshmand, L. Cuckoo: Flexible compute-intensive task offloading in mobile cloud computing. *Wirel. Commun. Mob. Comput.* **2016**, *2016*, 421–433. [CrossRef]
54. Fernandes, T.S.; Moreira, Á.F.; Cota, É. EPE-Mobile—A framework for early performance estimation of mobile applications. *Softw. Pract. Exp.* **2018**, *48*, 85–104. [CrossRef]
55. Peng, K.; Huang, H.; Pan, W.; Wang, J. Joint optimisation for time consumption and energy consumption of multi-application and load balancing of cloudlets in mobile edge computing. *IET Cyber-Phys. Syst. Theory Appl.* **2020**, *5*, 196–206. [CrossRef]
56. Mittal, S.; Mattela, V. A survey of techniques for improving efficiency of mobile web browsing. *Concurr. Comput. Pract. Exp.* **2019**, *31*, e5126. [CrossRef]
57. Ahmad, R.W.; Kjeldskov, J.; Skov, M.B. A survey on energy estimation and power modeling schemes for smartphone applications. *Int. J. Commun. Syst.* **2017**, *30*, e3234. [CrossRef]
58. Baride, S.; Dutta, K. A cloud based software testing paradigm for mobile applications. *ACM SIGSOFT Softw. Eng. Notes* **2011**, *36*, 1–4. [CrossRef]
59. Lee, S.; Ha, T.; Lim, S.J. Finger gesture input utilizing the rear camera of a mobile phone: A perspective of mobile CAD. *Hum. Factors Ergon. Manuf.* **2018**, *28*, 69–80. [CrossRef]
60. García-López, D.; Segura-Morales, M.; Loza-Aguirre, E. Improving the quality and quantity of functional and non-funtional requirements obtained during requirements elicitation stage for the development of e-commerce mobile applications: An alternative reference process model. *IET Softw.* **2020**, *14*, 148–158. [CrossRef]
61. Zhou, B.; Dastjerdi, A.V.; Calheiros, R.N.; Buyya, R. An online algorithm for task offloading in heterogeneous mobile clouds. *ACM Trans. Internet Technol.* **2018**, *18*, 2981. [CrossRef]
62. Li, D.; Lyu, Y.; Gui, J.; Halfond, W.G.J. Automated energy optimization of HTTP requests for mobile applications. In Proceedings of the International Conference of Software Engineering, Austin, TX, USA, 14–22 May 2016; Volume 14, pp. 249–260. [CrossRef]
63. Moran, K.; Linares-Vasquez, M.; Bernal-Cardenas, C.; Vendome, C.; Poshyvanyk, D. CrashScope: A practical tool for automated testing of android applications. In Proceedings of the 2017 IEEE/ACM 39th International Conference on Software Engineering Companion ICSE-C, Buenos Aires, Argentina, 20–28 May 2017; Volume 2017, pp. 15–18. [CrossRef]
64. Dai, Y.; Li, T.; Liu, B.; Song, M.; Chen, H. Exploiting Dynamic Thermal Energy Harvesting for Reusing in smartphone with mobile applications. In Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems ASPLOS, Williamsburg, VA, USA, 24–28 March 2018; pp. 243–256. [CrossRef]
65. Nikzad, N.; Radi, M.; Chipara, O.; Griswold, W.G. Managing the energy-delay tradeoff in mobile applications with tepus. In Proceedings of the 16th Annual Middleware Conference, Vancouver, BC, Canada, 7–11 December 2015; pp. 259–270. [CrossRef]
66. Silva, B.M.C.; Rodrigues, J.J.P.C.; Kumar, N.; Proença, M.L.; Han, G. MobiCoop: An incentive-based cooperation solution for mobile applications. *ACM Trans. Multimed. Comput. Commun. Appl.* **2016**, *12*, 1–23. [CrossRef]
67. Aras, S.; Gniady, C.; Venugopalan, H. MultiLock: Biometric-based graded authentication for mobile devices. In *Proceedings of the ACM International Conference Proceeding Series*; Association for Computing Machinery: New York, NY, USA, 2019; pp. 100–109. [CrossRef]
68. Zhang, X.; Brankovic, S.R.; Chen, J.; Fu, X.I.N. Toward Customized Hybrid Fuel-Cell and Battery-powered. *ACM Trans. Embed. Comput. Syst.* **2019**, *18*, 1–20. [CrossRef]
69. Hecht, G.; Benomar, O.; Rouvoy, R.; Moha, N.; Duchien, L. Tracking the software quality of android applications along their evolution. In Proceedings of the 2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE), Lincoln, NE, USA, 9–13 November 2015; pp. 236–247. [CrossRef]
70. Hoque, M.A.; Siekkinen, M.; Khan, K.N.; Xiao, Y.; Tarkoma, S. Modeling, profiling, debugging the energy consumption of mobile devices. *ACM Comput. Surv.* **2015**, *48*, 723. [CrossRef]
71. Holl, K.; Elberzhager, F. Quality assurance of mobile applications: A systematic mapping study. In *Proceedings of the MUM '16: Proceedings of the 15th International Conference on Mobile and Ubiquitous Multimedia, Rovaniemi, Finland, 12–15 December 2016*; pp. 101–113. [CrossRef]
72. Habchi, S.; Moha, N.; Rouvoy, R. The rise of android code smells: Who is to blame? In Proceedings of the 2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR), Montreal, QC, Canada, 25–31 May 2019; pp. 445–456. [CrossRef]
73. Kaewpuang, R.; Niyato, D.; Wang, P.; Hossain, E. A framework for cooperative resource management in mobile cloud computing. *IEEE J. Sel. Areas Commun.* **2013**, *31*, 2685–2700. [CrossRef]
74. Usman, M.; Iqbal, M.Z.; Khan, M.U. A model-driven approach to generate mobile applications for multiple platforms. In Proceedings of the 2014 21st Asia-Pacific Software Engineering Conference, Jeju, Republic of Korea, 1–4 December 2014; pp. 111–118. [CrossRef]
75. Gao, Y.; Liu, B. Energy Efficient and Delay Aware Service Selection in Mobile Edge Computing. In Proceedings of the 2020 IEEE 20th International Conference on Communication Technology (ICCT), Nanning, China, 28–31 October 2020; pp. 1546–1550. [CrossRef]

76. Tian, Z.; Xu, J.; Li, H.; Maeda, R.K.V. Multidevice Collaborative Power Management through Decentralized Knowledge Sharing. *IEEE Trans. Very Large Scale Integr. Syst.* **2020**, *28*, 1128–1140. [CrossRef]
77. Liu, Y.; Lee, M.J.; Zheng, Y. Adaptive Multi-Resource Allocation for Cloudlet-Based Mobile Cloud Computing System. *IEEE Trans. Mob. Comput.* **2016**, *15*, 2398–2410. [CrossRef]
78. Kao, Y.W.; Lin, C.; Yang, K.A.; Yuan, S.-M. A web-based, offline-able, personalized runtime environment for executing applications on mobile devices. *Comput. Stand. Interfaces* **2012**, *34*, 212–224. [CrossRef]
79. Ciman, M.; Gaggi, O. An empirical analysis of energy consumption of cross-platform frameworks for mobile development. *Pervasive Mob. Comput.* **2017**, *39*, 214–230. [CrossRef]
80. Dorfer, T.; Demetz, L.; Huber, S. Impact of mobile cross-platform development on CPU, memory and battery of mobile devices when using common mobile app features. *Procedia Comput. Sci.* **2020**, *175*, 189–196. [CrossRef]
81. Corral, L.; Janes, A.; Remencius, T. Potential advantages and disadvantages of multiplatform development frameworks—A vision on mobile environments. *Procedia Comput. Sci.* **2012**, *10*, 1202–1207. [CrossRef]
82. Elgendy, I.A.; Zhang, W.; Tian, Y.C.; Li, K. Resource allocation and computation offloading with data security for mobile edge computing. *Futur. Gener. Comput. Syst.* **2019**, *100*, 531–541. [CrossRef]
83. Rieger, C.; Majchrzak, T.A. Towards the definitive evaluation framework for cross-platform app development approaches. *J. Syst. Softw.* **2019**, *153*, 175–199. [CrossRef]
84. Rieger, C.; Tian, A.M.B. Weighted Evaluation Framework for Cross-Platform App Development Approaches. In *Information Systems: Development, Research, Applications, Education: 9th SIGSAND/PLAIS EuroSymposium 2016, Proceedings 9, Gdansk, Poland, 29 September 2016*; Springer: Berlin/Heidelberg, Germany, 2016; Volume 2, pp. 18–39. [CrossRef]
85. Zhang, G.A.; Gu, J.Y.; Bao, Z.H.; Xu, C.; Zhang, S.B. AGILE: A terminal energy efficient scheduling method in mobile cloud computing. *Trans. Emerg. Telecommun. Technol.* **2015**, *26*, 1323–1336. [CrossRef]
86. Wang, Y.H.; Wu, I.C. An evaluation framework for cross-platform mobile application development tools. *Softw. Pract. Exp.* **2014**, *45*, 1331–1357. [CrossRef]
87. Kim, S. One-on-one contract game–based dynamic virtual machine migration scheme for Mobile Edge Computing. *Trans. Emerg. Telecommun. Technol.* **2018**, *29*, e3204. [CrossRef]
88. Kim, D.; Chon, Y.; Jung, W.; Kim, Y.; Cha, H. Accurate prediction of available battery time for mobile applications. *ACM Trans. Embed. Comput. Syst.* **2016**, *15*, 5423. [CrossRef]
89. Mittal, R.; Kansal, A.; Chandra, R. Empowering developers to estimate app energy consumption. In *Proceedings of the Mobicom '12: Proceedings of the 18th Annual International Conference on Mobile Computing and Networking, Istanbul, Turkey, 2–26 August 2012*; pp. 317–327. [CrossRef]
90. Wei, L.; Liu, Y.; Cheung, S.C. Taming android fragmentation: Characterizing and detecting compatibility issues for android apps. In *Proceedings of the ASE '16: Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering, Singapore, 3–7 September 2016*; pp. 226–237. [CrossRef]
91. Biørn-Hansen, A.; Grønli, T.M.; Ghinea, G. A survey and taxonomy of core concepts and research challenges in cross-platform mobile development. *ACM Comput. Surv.* **2019**, *51*, 1739. [CrossRef]
92. Tong, F.; Yan, Z. A hybrid approach of mobile malware detection in Android. *J. Parallel Distrib. Comput.* **2017**, *103*, 22–31. [CrossRef]
93. Meng, T.; Wolter, K.; Wu, H.; Wang, Q. A secure and cost-efficient offloading policy for Mobile Cloud Computing against timing attacks. *Pervasive Mob. Comput.* **2018**, *45*, 4–18. [CrossRef]
94. Angin, P.; Bhargava, B.; Ranchal, R. A self-protecting agents based model for high-performance mobile-cloud computing. *Comput. Secur.* **2018**, *77*, 380–396. [CrossRef]
95. Goel, D.; Jain, A.K. Mobile phishing attacks and defence mechanisms: State of art and open research challenges. *Comput. Secur.* **2018**, *73*, 519–544. [CrossRef]
96. Mai, P.X.; Goknil, A.; Shar, L.K.; Pastore, F.; Briand, L.C.; Shaame, S. Modeling Security and Privacy Requirements: A Use Case-Driven Approach. *Inf. Softw. Technol.* **2018**, *100*, 165–182. [CrossRef]
97. Mollah, M.B.; Azad, M.A.K.; Vasilakos, A. Security and privacy challenges in mobile cloud computing: Survey and way ahead. *J. Netw. Comput. Appl.* **2017**, *84*, 38–54. [CrossRef]
98. Kamesh; Sakthi Priya, N. Securing mobile access with interactive image code and image key encryption. *Secur. Commun. Netw.* **2016**, *5*, 422–438. [CrossRef]
99. Yerima, S.Y.; Sezer, S.; McWilliams, G. Analysis of Bayesian classification-based approaches for Android malware detection. *IET Inf. Secur.* **2014**, *8*, 25–36. [CrossRef]
100. Lee, Y.K.; Bang, J.Y.; Safi, G.; Shahbazian, A.; Zhao, Y.; Medvidovic, N. A sealant for inter-app security holes in android. In Proceedings of the 2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE), Buenos Aires, Argentina, 20–28 May 2017; pp. 312–323. [CrossRef]
101. Ma, S.; Feng, R.; Li, J.; Liu, Y.; Nepal, S.; Bertino, E.; Deng, R.H.; Ma, Z.; Jha, S. An empirical study of SMS one-time password authentication in android apps. In *Proceedings of the ACSAC '19: Proceedings of the 35th Annual Computer Security Applications Conference, San Juan, Puerto Rico, 9–13 December 2019*; pp. 339–354. [CrossRef]

102. Namrud, Z.; Kpodjedo, S.; Talhi, C. AndroVul: A repository for android security vulnerabilities. In *Proceedings of the CASCON '19: Proceedings of the 29th Annual International Conference on Computer Science and Software Engineering, Toronto, ON, Canada, 4–6 November 2019*; pp. 64–71.
103. Hesenius, M.; Griebe, T.; Gries, S.; Gruhn, V. Automating UI tests for mobile applications with formal gesture descriptions. In *Proceedings of the MobileHCI '14: Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices & Services, Toronto, ON, Canada, 23–26 September 2014*; pp. 213–222. [CrossRef]
104. Russello, G.; Jimenez, A.B.; Naderi, H.; Van Der Mark, W. FireDroid: Hardening security in almost-stock android. In *Proceedings of the ACSAC '13: Proceedings of the 29th Annual Computer Security Applications Conference, New Orleans, LA, USA, 9–13 December 2013*; pp. 319–328. [CrossRef]
105. Arora, A.; Peddoju, S.K. Minimizing network traffic features for Android mobile malware detection. In *Proceedings of the ICDCN '17: Proceedings of the 18th International Conference on Distributed Computing and Networking, Hyderabad, India, 5–7 January 2017*. [CrossRef]
106. Mulliner, C.; Oberheide, J.; Robertson, W.; Kirda, E. PatchDroid: Scalable third-party security patches for android devices. In *Proceedings of the ACSAC '13: Proceedings of the 29th Annual Computer Security Applications Conference, New Orleans, LA, USA, 9–13 December 2013*; pp. 259–268. [CrossRef]
107. Bhat, P.; Dutta, K. A survey on various threats and current state of security in android platform. *ACM Comput. Surv.* **2019**, *52*, 1–35. [CrossRef]
108. Wei, N.; AmiriSani, A. SchrodinText: Strong Protection of Sensitive Textual Content of Mobile Applications. *IEEE Trans. Mob. Comput.* **2020**, *1233*, 1. [CrossRef]
109. Wang, P.; Tamura, S. Software Protection on the Go: A Large-Scale Empirical Study on Mobile App Obfuscation ACM Reference Format. In Proceedings of the 40th International Conference on Software Engineering, Gothenburg, Sweden, 27 May–3 June 2018; pp. 26–36.
110. Capilla, R.; Kazman, R.; Romera, C.; Carrillo, C. Usability implications in software architecture: The case study of a mobile app. *Softw. Pract. Exp.* **2020**, *50*, 2145–2168. [CrossRef]
111. Zhao, H.; Chen, M.; Qiu, M.; Gai, K.; Liu, M. A novel pre-cache schema for high performance Android system. *Futur. Gener. Comput. Syst.* **2016**, *56*, 766–772. [CrossRef]
112. Dye, S.M.; Scarfone, K. A standard for developing secure mobile applications. *Comput. Stand. Interfaces* **2014**, *36*, 524–530. [CrossRef]
113. Raptis, D.; Tselios, N.; Kjeldskov, J.; Skov, M.B. Does size matter? Investigating the impact of mobile phone screen size on users' perceived usability, effectiveness and efficiency. In *Proceedings of the MobileHCI '13: Proceedings of the 15th International Conference on Human-Computer Interaction with Mobile Devices and Services, Munich, Germany, 27–30 August 2013*; pp. 127–136. [CrossRef]

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.