Contents lists available at Science-Gate

International Journal of Advanced and Applied Sciences

Journal homepage: http://www.science-gate.com/IJAAS.html

Motivations, challenges, and process support for the evolution of existing software to mobile computing platforms

Ibrahim Alseadoon *

College of Computer Science and Engineering, University of Ha'il, Ha'il, Saudi Arabia

ARTICLE INFO

Article history:
Received 19 September 2020
Received in revised form
13 December 2020
Accepted 5 February 2021

Keywords:
Software engineering
Software quality
Mobile computing
Software maintenance and evolution
Evolution process

ABSTRACT

Software maintenance and evolution support changes in the structure and behavior of existing software to change it as per the needs and demands of new requirements. The majority of the existing software systems lack features of mobile computing such as portability, context-awareness, connectivity, and high interactivity. The evolution of the existing software for mobile computing platforms can enable these systems to retain their core data and logic while acquiring new features that are compatible with mobile systems. The objectives of this research are to (i) systematically identify the motivations and challenges of software evolution for mobile computing, and (ii) develop and validate a process model that supports the evolution of existing software to a mobile computing platform. To conduct this research, an empirical software engineering approach has been adopted to investigate existing solutions (30 published studies from 1996 to 2019) and empirically derive a process model that supports software evolution for mobile computing. A case study-based approach is adopted to demonstrate the process-centric evolution of existing software as a mobile-enabled application. Case study-based demonstration highlights that the proposed process (i) supports an incremental evolution and (ii) allows user-decision support to guide the evolution process. Evaluation results highlight computation and energy efficiency along with enhanced usability of a mobile application when executed on resource-constrained mobile devices. The results of this research could help researchers and practitioners to rationalize motivations and challenges to utilize a process-based approach to evolve existing or aging software for mobile computing platforms. Future research is focused on providing patterns and tool support to automate and customize the evolution process.

1. Introduction

Software maintenance and evolution as a phenomenon of software engineering supports changes in structure and behavior of existing software to update software functionality as per new or emerging requirements (Mens, 2008). Lehman's law of software evolution implies that real-world systems must be continuously adapted to prolong their productive life and operational value (Jamshidi et al., 2013a). The research and development on software maintenance and evolution started in the 70s and proved successful in evolving existing software systems to enhance the functionality or extend the productive life of existing or aging software (Jamshidi et al., 2013a; Ahmad et al., 2014a). In recent years, research in software evolution has progressed to address the issues of migration or modernization of legacy software in three main areas including service oriented architectures (Khadka et al., 2013), cloud computing (Jamshidi et al., 2013b), and software product lines (Assunção et al., 2017). Software evolution refers to fundamental principles and practices of software change that can be applied to existing software to evolve them and execute them on modernized platforms such as mobile computing (Jamshidi et al., 2013b). However, in order to support a systematic evolution, systematic processes, patterns, frameworks, and tool support are required that can enable existing software to be incrementally evolved to new platforms as per new requirements (Ahmad et al., 2012).

* Corresponding Author.
Email Address: i.alsedon@uoh.edu.sa
https://doi.org/10.21833/ijaas.2021.05.011
Corresponding author's ORCID profile:
https://orcid.org/0000-0003-2289-356X
2313-626X/© 2021 The Authors. Published by IASE.

Mobile or handheld computing has seen significant growth in recent years to become a pervasive technology that powers-up enterprise computing and supports individuals with their daily life activities (Pejovic and Musolesi, 2015). For example, users can use mobile computing technologies (i.e., handheld devices, wireless networks, and mobile software applications) to perform a variety of tasks such as ride-hailing, mobile commerce, health monitoring, and social networking on the go. Specifically, mobile computing allows users to utilize their mobile computing devices to perform context-aware computing and portable communications. Despite these benefits such as context-awareness and portability, mobile computing faces some challenges that include limited device resources (battery, memory, and processor) along with issues of data security and privacy (Lane et al., 2010). Moreover, the majority of existing software systems are based on traditional computing technologies that run on traditional (web or workstation) platforms. These existing or legacy systems consist of valuable data and core logic but cannot exploit the features of mobile computing environments (Ahmad et al., 2019a). Therefore, there is a need for research and development efforts that can support the evolution of existing software (running on web or workstation) platforms so they can be executed on mobile computing platforms to benefits from features of mobile computing. For example, a web-based online shopping portal can be modernized as a mobile-based application to support context-aware (location, time, gender-based) recommendations of relevant products to its users (Ahmad et al., 2019a).

1.1. Research method

Empirical software engineering (ESE) as a research discipline and method provides experimental foundations for the research and development of software-intensive systems (Petersen et al., 2008). In recent years, ESE approaches have been used to conduct empirical studies such as comparison analysis, systematic literature reviews, and systematic mapping studies (Brereton et al., 2007). These empirical studies helped researchers to systematically investigate research areas to highlights their impacts, challenges, and existing solutions to outline the scope for futuristic research. The proposed research aims to exploit the principle and guidelines of ESE to conduct a study that investigates the challenges and motivations along with a processes model for the evolution of existing software as a mobile computing application. This research study is conducted by following the guidelines from Petersen et al. (2008) with an objective to 'identify, analyze, and synthesize the existing research and development on the challenges, factors that motivate, and processes that support evolution of existing software systems towards mobile computing platforms'. There is a lot of research on software migration or evolution for services oriented and cloud computing systems, however, there is no concrete effort to systematically study the evolution of existing software for mobile computing platforms. The proposed outcomes of this study are identification of the motivations and challenges along empirical development of a process that supports an incremental evolution of legacy software as a portable mobile computing application. Case study based approach is adopted to evaluate the evolution process. Fig. 1 shows solution overview-process view for software evolution towards mobile computing platform.

1.2. Solution overview

A high-level view of the proposed solution is presented in Fig. 1 that also highlights the scope of the research. As in Fig. 1, the first phase of this research conducts a mapping study to systematically investigate an existing solution to identify the challenges and motivations for software evolution to mobile. Based on the identified motivations and challenges, a process has been derived that comprises five different activities to support an incremental evolution of legacy software from web or workstation-based platforms to the context-aware mobile computing platform.

As in Fig. 1, the input to the process is existing software that is evolved as a mobile computing application representing the output of the process. Software development or evolution engineer represents a human intervention in the process to guide software evolution. Based on the proposed solution in Fig. 1, contributions of this research are highlighted as:

- Systematic identification of the critical factors that pose challenges and provide motivations for software evolution in the context of mobile computing.
- Development of a process-centric approach and solution that support an incremental software evolution for mobile computing platforms.
- Case study-based evaluation of the process to demonstrate its applicability and identify the areas of future research to support software engineering of mobile computing systems.

The proposed research advances state-of-the-art software migration and evolution for service and cloud computing systems with process-centric support that specifically focuses on software evolution for mobile computing. An objective evaluation of the evolved application shows the application's efficiency and usability on resource-constrained mobile devices.

The proposed evolution process and its evaluation can provide foundations for tool-supported evolution of existing software towards mobile computing. The proposed research is among the pioneering efforts to address the issues of software maintenance and evolution for mobile computing that are currently lacking in the existing research. The results of this proposed research can be beneficial for:

- Researchers who need quick identification of motivations and challenges that affect the research and development focused on the evolution of existing software towards mobile computing platforms.
- Practitioners who can exploit the proposed evolution framework to develop new processes, patterns, and tool support that can guide and execute software evolution for mobile in an industrial context.

The rest of the paper is organized as follows. Background and related research are presented in Section II. The research method is detailed in Section III. Section IV to Section VI present results of proposed research in the context of the evolution process and case study-based demonstrations of the solution. Section VII presents the results of the evaluation. Section VIII concludes the paper with a discussion about potential future research.

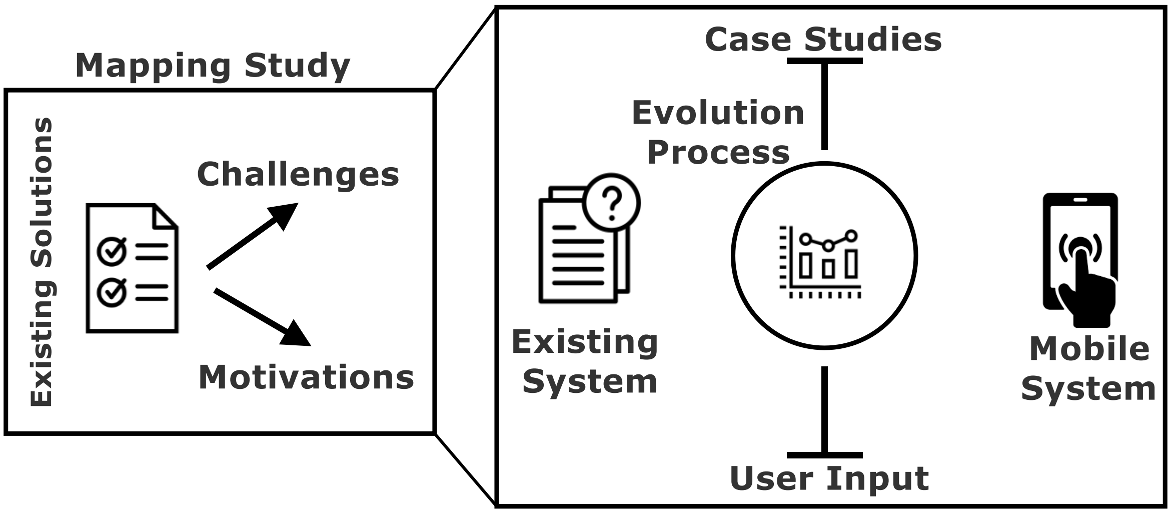


Fig. 1: Solution overview-process view for software evolution towards mobile computing platform

2. Background and related research

This section presents background details about software evolution in the context of mobile computing systems (Section 2.1). The overview and comparison of the most relevant related research are also presented (Section 2.2). Discussion about background information and related research justifies the scope of the proposed research.

2.1. Software evolution in the context of mobile computing

Traditionally, software evolution (a.k.a software change management), is viewed as a mechanism that comprises theories, processes, patterns, tools, and practices to change an existing software (Jamshidi et al., 2013a). More recently, software evolution practices have been exploited to support modernization or migration of existing software systems to modernized computing platforms such as service computing (Khadka et al., 2013) or cloud-based platforms (Jamshidi et al., 2013b; Ahmad and Babar, 2014). Migration of existing or legacy software to the newer generation of computing platforms also motivates the need for existing systems to be migrated to mobile computing platforms. The primary needs for such evolution (i.e., evolving existing software as a mobile application) are driven by the fact that evolved software can benefit from mobile computing technologies to offer portability, context-sensitivity, and enhanced interactivity. Despite these benefits, there are some challenges such as resource limitations of mobile devices in terms of scarcity of memory, processing, and power available on handheld devices. Also, the security and privacy of mobile devices and their data are primary challenges while engineering and developing mobile computing systems (Ahmad et al., 2019a).

Fig. 2 illustrates a high-level view for the evolution of existing software that runs on web or workstation platforms to the mobile computing platform. For example, a health and fitness monitoring software that runs on a workstation-based platform gets data in terms of users' workout plans and duration. When the software is evolved as a mobile application it can automatically track user's contextual information, i.e., movements, speed, duration of exercise to support health and fitness monitoring in a portable fashion. Fig. 2 shows that the input to the evolution process is legacy software (web portal or workstation software) that is evolved so it can be executed on mobile computing platforms. The output of the evolution process is modernized software in the form of a mobile computing application. As in Fig. 2, the existing systems contain core logic and data that is vital for ensuring the correct functionality of the existing software and such core logic and data are hard to replace or overwrite. In comparison, mobile software supports portability and context-sensitivity of computations that are lacking in the existing software. The evolution from existing software to mobile application must ensure the preservation of core logic and existing data while attaining context-sensitivity and portability as new functionality.

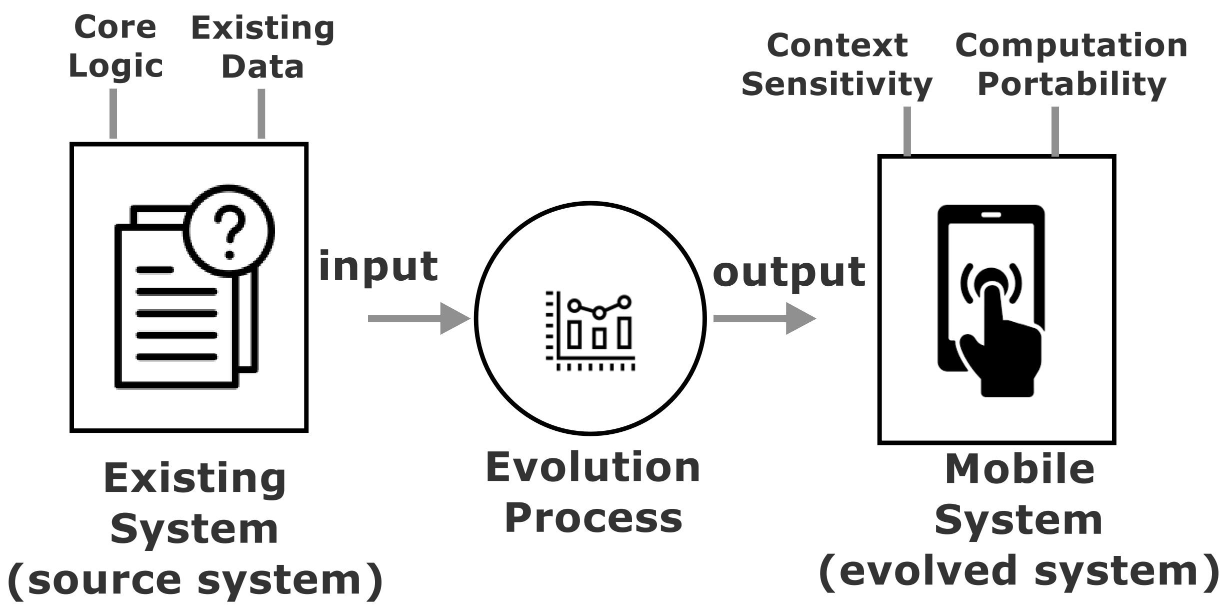


Fig. 2: Overview of the process for evolution to mobile

2.2. Related research on software evolution

This section first presents the existing research on the migration of existing or legacy software to mobile computing platforms. Afterward, the most relevant survey-based studies are presented that investigate existing research and development on maintenance and evolution of legacy software for modernized computing platforms such as service, cloud, and product line systems.

2.2.1. Software evolution for mobile computing platforms

In recent years, there has been a lot of research and development to support the migration and evolution of existing software to service-oriented and cloud computing platforms (Khadka et al., 2013; Jamshidi et al., 2013b). However, there is not much work done to support the evolution of old or aging software towards mobile computing platforms. One of the earliest research to support software evolution for mobile computing is presented in Pope (1996) to migrate software systems from resource-rich workstation platforms to portable and resource-constrained mobile platforms. The rapid proliferation of handheld devices and increased adoption of mobile technologies have resulted in growing demands for software and systems to be developed for mobile computing. Foss and Wong (2004) enabled the evolution of Microsoft Windows-based software (that executes on a workstation platform) as an interactive and touch-sensitive Palm OS-based application. The primary motivations for the adoption of mobile computing and mobile applications are to achieve portable and interactive computing offered by mobile platforms (Fan and Wong, 2016). The recent research efforts also highlight the need to develop processes, patterns, tools, and framework that support a systematic approach for legacy software evolution as a mobile application (Seffah, 2015). Existing research and development suggest that in order to keep up with the rapid adoption of mobile systems, existing software must be re-engineered so it can be evolved for and executed on mobile computing platforms (Cheng et al., 2012).

2.2.2. Survey-based studies on software evolution for modern computing platforms

In this section, comparison and analysis of existing survey-based studies on the evolution of existing or legacy software to modernized computing platforms are presented. An objective discussion of the existing research helps to justify the contributions of the proposed solution in the context of existing work. Table 1 objectively compares the existing secondary (a.k.a. survey-based) studies on software evolution towards modernized computing platforms. Table 1 acts as a structured catalog to compare the existing research-based on three-point criteria: Type of Evolution, Publication Year, and Number of Studies Reviewed. In the following, a brief summary of the most relevant secondary studies is presented that helps with highlighting the scope and contributions of the proposed research.

- Migration of Legacy Systems to SOAs: Khadka et al. (2013) conducted a systematic review of 121 published research studies to investigate the migration of legacy software systems to service-oriented architectures. The study was published in 2012 and focused on identifying the critical factors, motivations, and scenarios that modernize legacy software as per the needs for service-driven software systems. The study highlights status of existing research and also presents the needs for future research that can support the migration of outdated software that can be deployed and executed on service computing platforms.

- Evolution of Existing Software to Cloud Computing Platforms: The authors in Jamshidi et al. (2013b) conducted a systematic review to classify and compare the existing research and development that supports the evolution of legacy software systems to cloud computing platforms. The study reviews a total of 21 qualitatively selected studies to investigate the types of evolution, the types of artifacts being evolved, along past and active trends of research that support software migration to cloud computing platforms. This research study helps to understand the impacts of existing research to develop innovative solutions for the cloudification of existing software.
- Evolution of Existing Software towards Software Product Lines: The review-based study in Assunção et al. (2017), focuses on analyzing 119 research studies on the evolution of existing single instance systems to software product lines. This study analyses the factors that support and hinder software evolution to product lines. The study also highlights methods, techniques, and processes that support software evolution to software product line systems.

Table 1: Comparison of the existing survey-based studies and the focus of proposed research

No.	Type of Evolution	Publication Year	Number of Studies Surveyed	Reference
1	*Existing to SOA*	2012	121	Khadka et al. (2013)
	A systematic review of research on the migration of existing or legacy software systems to software services and service-oriented architectures.			
2	*Existing to Cloud*	2013	21	Jamshidi et al. (2013b)
	A systematic review to investigate the migration of legacy software systems to mobile computing platforms.			
3	*Existing to SPLs*	2017	119	Assunção et al. (2017)
	A survey-based study on existing research and development on the evolution of single instance to software product line systems.			
4	Existing to Mobile	NA	30	NA
	-Systematic mapping and analysis for the evolution of existing software for mobile computing platforms. -Process-centric and case study driven evolution of software to demonstrate the applicability and evaluation of the proposed solution			

Based on the data in Table 1 and a summary of the existing studies in Khadka et al. (2013), Jamshidi et al. (2013b), and Assunção et al. (2017), it is concluded that the proposed research is the first attempt towards highlighting and understanding the critical factors that influence the evolution of software systems towards mobile computing platforms. These factors include the motivations, challenges, along with process-driven frameworks to support the evolution of existing software for mobile computing. The proposed research is distinct from existing work in that it is among the pioneering efforts to understanding software evolution in the context of mobile computing platforms. The proposed research aims to support processes and patterns of evolution that reflect the need for a new generation of solutions to address evolution-specific issues for software systems in the context of mobile computing platforms.

3. Research method and questions

This section presents the method being used to plan, conduct, and document this research. An overview of the research method is presented in Fig. 3. Fig. 3 is based on the guidelines of ESE to conduct the systematic mapping studies and systematic literature reviews (Petersen et al., 2008; Brereton et al., 2007). As shown in Fig. 3, the research method has two main phases named as: (i) Specifying the Research Questions, (ii) Performing Systematic Mapping, and (ii) Enabling Process-based Evolution. As per the adopted methodology, research questions are fundamental to drive the research method and to document the results. Moreover, a case study-based approach is used to demonstrate and evaluate the evolution process. Each phase of the research method, as presented in Fig. 3 is detailed below.

3.1. Specification of the research questions

The research questions (RQs) define the scope of the research and provide an objective mechanism to document the results as an answer to each of the RQs. In the following, the outlined RQs try to investigate three main aspects and each RQ is complemented by an explanation of the primary objective(s) of the question.

3.1.1. Demographic details of published research

RQ-1: What is the frequency and growth of research publications in the area of software evolution for mobile computing environments?

Objective(s): The primary objective of this RQ is to understand the frequency and growth of the published research in the area of software evolution for mobile environments. Such demographic details of the published research provide foundations to analyze the impact and growth of research.

3.1.2. Challenges and motivation for software evolution

RQ-2: What are the challenges and motivating factors for the evolution of existing software systems towards mobile computing environments?

Objective(s): The objective of this RQ is to identify the prominent challenges that hinder software evolution for mobile computing platforms.

3.1.3. Process model for software evolution

RQ-3: Are there any processes that can support a systematic evolution of existing software towards a mobile computing environment?

Objective(s): To identify the process as a collection of activities that supports a systematic evolution of existing software towards mobile computing platforms. Case study-based demonstration and objective evaluations complement the process.

3.2. Performing systematic mapping

As in Fig. 3, the next phase of the research method is to perform systematic mapping of the existing research to identify, analyze, classify and document the progress, limitations, and impacts of the existing research (Petersen et al., 2008). The systematic mapping of the existing research helps to identify:

- Frequency and types of research publications to understand the progression and growth of research over the years (answer to RQ-1).
- Motivations and Challenges that support or hinder the process of evolution for existing software systems for mobile computing platforms (answer to RQ-2).
- Process model that can support an incremental evolution of existing software as a mobile-enabled application (RQ-3).

3.3. Enabling process-based evolution

The last phase of the research method as in Fig. 3 relates to the process-centric evolution of the software that includes:

- Empirically derived process and its underlying activities that aim to support an incremental evolution of the software.
- Case study-based demonstration and evaluation of the process model to support the evolution of existing software as a mobile-enabled application.
- Evaluation of the evolved application in terms of its efficiency and usability on resource-constrained mobile devices.

Based on the research method that is illustrated in Fig. 3 results of this research are presented in the remainder of this paper. The results provide answers to RQ-1 to RQ-3 from Section 4 to Section 7.

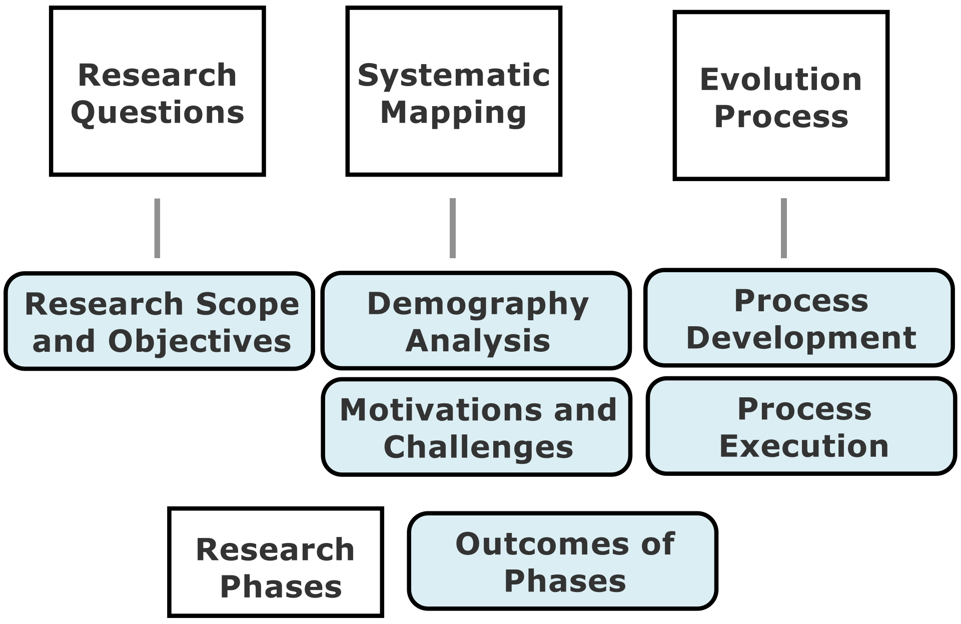


Fig. 3: Research method for the study

4. Frequency, types, and growth of published research: Demography analysis

This section answers RQ-1 that aims to analyze the frequency and growth of published research over the years in the area of software evolution for mobile computing environments. The answer to this question highlights the historical progression of research from fundamental theories to recent trends in the form of published results. Based on the analysis, the frequency of research publications is plotted in Fig. 4 that shows published research from the years 1996 to 2019 (Jun 2019). Fig. 4 shows two main types of information in terms of (a) Years of Publication, and (b) Types of Publications. In the following, both of these types are discussed to highlight the growth and progress of research over the years.

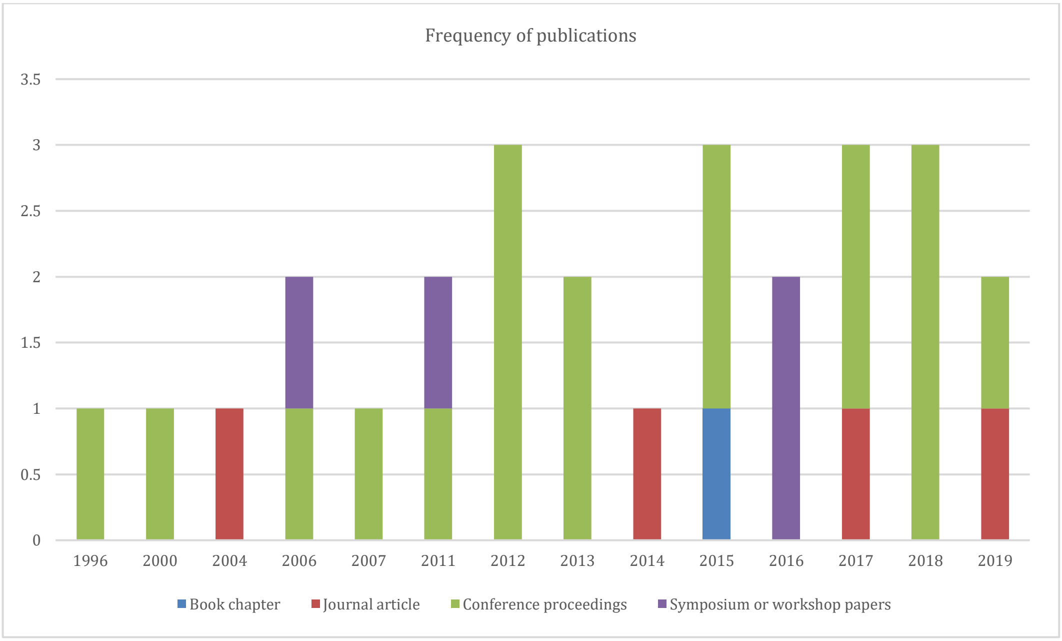


Fig. 4: Overview of the frequency and types of publications

4.1. Years of publication on software evolution for mobile

The research on software evolution for mobile computing environments started to emerge since the year 1996. From 1996 to 2003 only three studies (3/30, 10%) were published with a primary focus on the portability of source code from traditional workstation-based computers to palm devices (Emmerich et al., 2000; Pope, 1996). Since the year 2004, there is a steady growth of published research that focused on various aspects such as application state migration (Pope, 1996), runtime evolution (Hansen et al., 2017), and design time evolution (Foss and Wong, 2004) of existing software systems to mobile computing platforms. There was no published research during the years 1997, 1998, 1999, 2001, 2002, 2005, 2008, 2009, 2010. There can be two possible reasons for no published research during these years. One possibility is that during those years, there were no relevant publications or any publications from those years that could not pass the qualitative assessment and got excluded based on the quality assessment criteria. The years from 2004 to 2018 can be collectively considered as most progressive with a total of 25 (25/30, 83%) published studies. Individually, the year 2012 can be considered as most progressive with a total of 4 studies.

4.2. Types of publications on software evolution for mobile

After presenting the years of publications, new types of publications are discussed that reflect the diversity of published research. The types of publications can be classified into four main types namely: Book Chapter, Journal Article, Conference Proceedings, Symposium or Workshop Papers as in Fig. 4. Fig. 4 shows only 01 Book Chapter published in the year 2015 that focused on pattern-based reengineering of existing software to mobile computing platforms. A total of 04 Journal Articles and 04 Symposium and Workshop Papers have been published during those years. Most numbers of the published papers are in the form of Conference Proceedings amounting to 20 which represents 66% of the total published research. Based on Fig. 4, it is concluded that since 2012 there is a relative growth of published research in the area of software evolution for mobile computing environments. Also, conference publications represent the most number of published research. However, there is a need for more Journal Article based publications with rigorous validations of the results.

5. Motivations and challenges for software evolution to mobile computing environments

In this section, RQ-2 has been answered. The answer to RQ-2 (i) presents the motivations for the evolution of existing software to mobile computing platforms (Section 5.1) and (ii) highlight critical challenges that need to be considered during software evolution to mobile (Section 5.2). The discussion of motivations and challenges highlights trade-off analysis (benefits vs. limitations in terms of motivating factors and challenges) that can help research and practices to develop and execute evolution processes and activities.

5.1. Motivating factors for software migration to mobile

This section answers RQ-2 to highlight the motivating factors for the evolution of existing software systems to mobile computing platforms. These motivating factors are also considered as desired quality attributes or non-functional properties that need to be attained by software that executes on mobile computing platforms. To discuss the motivating factors, illustration in Fig. 5 and data in Table 2 are used to answer RQ-2. Based on the thematic analysis process, a total of five major motivating factors have been identified, as in Table 2. In the following, each factor is discussed individually with the help of Table 2 that provides a structured catalog to present a summary of the motivating factors and evidence for each of the factors as a reference to the published research. Moreover, Fig. 5a highlights the relative distribution of studies that support a given motivating factor. The identified motivating factors include:

- Context-Sensitivity: One of the primary motivating factors for the evolution of existing software to mobile computing is to achieve context-sensitivity of computations. Context-sensitive computations refer to computing devices performing their computations based on consideration of the contextual information such as location information, day/time, mood, temperature. For example, context about user location can be exploited to offer users the ability to locate nearby services (restaurants, events, social gatherings, etc.). Therefore, context-sensitivity is a major driving factor for software evolution to mobile that is identified in a total of 7/30 studies (Table 3, Reference Evidence) that represents a total of 23% of available evidence (Ahmad et al., 2019a).
- Computation Portability: It refers to portable and mobility-driven computations and communications that can be provided by mobile applications. Based on the review, portability is the most important factor for evolution as it empowers a mobile user to perform complex computations such as health monitoring, crowdsensing, mobile commerce, and social networking on the go. The review suggests a total of 14/30, i.e., 47% studies that support computation portability as the primary factor for motivating software evolution to mobile (Emmerich et al., 2000; Pope, 1996; Canfora et al., 2004; Fan and Wong, 2016; Foss and Wong, 2004; Seffah, 2015; Hansen et al., 2017).
- Enhanced Interactivity: It refers to better user experience and better interactivity that is offered by mobile devices. Enhanced interactivity allows users to perform tasks in an intuitive and efficient manner that minimizes laborious and manual efforts to complete the computation tasks. As per the review, a total of 4/30, i.e., 13% of studies support enhanced interactivity as one of the motivating factors for software evolution (Bruschi et al., 2018).
- Software Scalability: It refers to scaling up or down a software system as per the requirements of the computing platform. Scalability in the context of evolution is driven by the need to have software systems that are continuously connected and available on the go as a self-sufficient platform. A total of 2 studies have been identified, i.e., 07% of studies supporting software scalability as a motivating factor for software evolution to mobile computing (Fan and Ma, 2017; Businge et al., 2018).
- Service Availability: It refers to the continuous availability of computing and software services to its users. Service availability is related to context-sensitivity that supports computation portability, enabling users to have better and continuous access to computing services to perform their tasks. The review suggests a total of 3/30 studies, i.e., 10% of studies supporting service availability as the motivating factor for software evolution as a mobile computing application (Cheng et al., 2012).

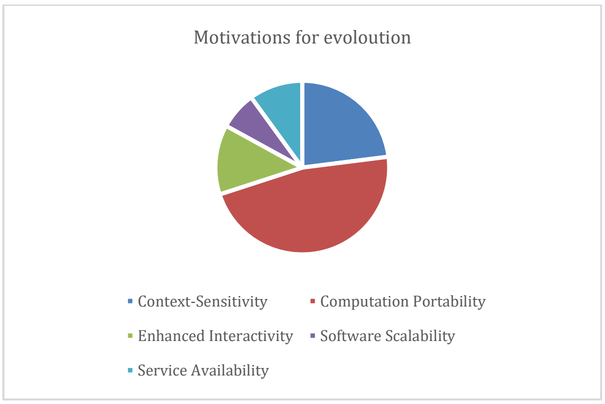
5.2. Critical challenges for software migration to mobile

After presenting the motivations, now answer to RQ-3 is presented that aims to investigate the critical challenges that are associated with the evolution process or evolved software when it is executed on a mobile platform. Table 3 and Fig. 5b are used to discuss the critical challenges for software evolution. Based on the review, a total of six critical challenges for software evolution to mobile have been identified. In the remainder of this section, each of the challenges is discussed individually to highlight how such challenges could possibly be tackled.

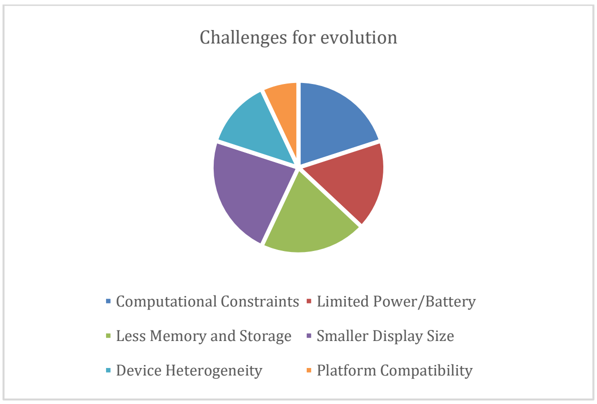
- Computational Constraints: It refers to the constraints or limitations that are inherited by mobile devices that have smaller processors. It is vital to mention that while achieving the portability, computational capabilities of a device are compromised in terms of a smaller processor. This can lead to a lack of performance for some computation-intensive tasks. For example, an image processing, multimedia, or gaming app can suffer from performance issues while being executed on mobile devices. The review suggests a total of 6/30 studies, i.e., 20% of studies highlighting computation constraints as the critical challenge for the evolution of existing software to mobile platforms (Businge et al., 2018).
- Limited Power/Battery: Another challenge that relates to resource constraints of mobile devices is limited power or battery. It refers to a limitation of mobile devices that can also compromise the availability, performance, and efficiency of tasks. Any computation-intensive task impacts the power consumption of a mobile device that leads to a short life cycle for available battery, unlike traditional workstation-based systems. The review suggests that a total of 5/30, i.e., 17% of studies have highlighted limited power/battery as a critical challenge for mobile applications as indicated in (Bruschi et al., 2018).

Table 2: Summary of Motivations and Challenges for Evolution along with Available Evidence

Motivating Factors for the Evolution of Existing Software to Mobile Computing Platform		
Motivating Factor	Description for Motivations	Reference Evidence
Context-Sensitivity	Evolved software can exploit context-aware computing offered by the mobile computing platforms.	(Ahmad et al., 2019a)
Computation Portability	The user is empowered with computations and communications on the go.	(Emmerich et al., 2000; Pope, 1996; Canfora et al., 2004; Fan and Wong, 2016; Foss and Wong, 2004; Seffah, 2015; Hansen et al., 2017)
Enhanced Interactivity	Mobile displays offer enhanced interaction and user experience.	(Bruschi et al., 2018)
Software Scalability	Software can be scaled to a mobile device rather than bigger processing terminals.	(Fan and Ma, 2017; Businge et al., 2018)
Service Availability	Software services are available to the device and users on the go.	(Cheng et al., 2012)
Critical Challenges for the Evolution of Existing Software to Mobile Computing Platform		
Critical Challenges	Description for Challenge	Reference Evidence
Computational Constraints	Mobile devices have a limited processor and computation capabilities for the evolved software.	(Businge et al., 2018)
Limited Power/Battery	Battery or device power hinders the capabilities of mobile computing and software	(Bruschi et al., 2018)
Less Memory and Storage	Smaller memory and storage size can be an issue for data-intensive applications	(Pope, 1996; Cheng et al., 2012; Hansen et al., 2017)
Smaller Display Size	The size of the display could be limiting compared to the traditional bigger displays	(Fan and Ma, 2017; Canfora et al., 2004; Fan and Wong, 2016; Seffah, 2015; Ahmad et al., 2019a)
Device Heterogeneity	Multiple devices can affect the performance of the evolved software	(Emmerich et al., 2000)
Platform Compatibility	Compatibility issues related to data size, computation, efficiency, and screen size may affect the evolved software.	(Foss and Wong, 2004)



a



b

Fig. 5: (a) Motivations and (b) Challenges for software evolution to the mobile computing platform

- Less Memory and Storage: It refers to limited data storage space that can be available on resource-constrained mobile devices. Similar to smaller processors and batteries, portable mobile computing devices suffer from a smaller amount of storage space that can be available on these devices. One of the solutions to overcome this issue is mobile cloud computing that can offload memory-intensive tasks and data on remote cloud servers (AccuBattery (version-1.1.7). The review suggests a total of 6/30, i.e., 20% of studies highlighting limited storage as a challenge for mobile applications (Pope, 1996; Cheng et al., 2012; Hansen et al., 2017).
- Smaller Display Size: Mobile devices and their displays support high interactivity and touch-sensitive computing. However, one inherent limitation of enhanced interactivity is the limited size of the display that is available on mobile devices. This can be particularly limiting for multimedia and graphics-based applications when scaled down from larger displays to smaller ones. The review suggests that a total of 7/30, i.e., % studies have highlighted smaller display sizes as one of the challenges for mobile applications (Fan and Ma, 2017; Canfora et al., 2004; Fan and Wong, 2016; Seffah, 2015; Ahmad et al., 2019a).
- Device Heterogeneity: It refers to a mix and diverse set of mobile computing devices that can vary significantly in terms of device specifications, display size, and the platforms that are available on those devices (e.g., Android, iOS, and Windows-powered devices). This can be a challenge for the evolved software application to function and execute consistently across all such diverse devices. The review suggests that a total of 4/30, i.e., 13% of total studies highlighting device heterogeneity as a critical challenge.
- Platform Compatibility: It refers to the compatibility of the evolved software when it is transitioned from a traditional (web or workstation-based) platform to a mobile device. A

typical example of such transition of the migration of software applications from Microsoft Windows-based platform to Google's Android platforms (Foss and Wong, 2004; Fan and Wong, 2016). Such migration or transition could result in consistency, scalability, and data compatibility issues. The review suggests that only a total of 2/30, i.e., 7% of studies have highlighted platform compatibility as a critical challenge for the evolved software (Foss and Wong, 2004).

6. Process and case study for software evolution as mobile computing application

In this section, RQ-3 aims to present an empirically derived process for the evolution of existing software to mobile computing platforms. To answer RQ-3., first, we present the process for software evolution in Section 6.1. We then present a case study that demonstrates the process-driven evolution of an existing web-based portal to a mobile-enabled application in Section 6.2.

6.1. Process model for software evolution to mobile computing

This section presents details of the process and its underlying activities for the evolution of existing software to mobile computing platforms as illustrated in Fig. 6. Fig. 6 highlights that the process has four main activities namely, Requirement Analysis, Design Modeling, Change Management, Software Validation. The activities decompose the process into small steps for incremental evolution of existing software as a mobile application. Incremental evolution refers to the decomposition of coarse-grained steps of change into smaller fine-grained actions that can be executed and managed in an incremental manner. The process demonstrates what needs to be done? and activities show how it is to be done? In the following, we detail each of the four process activities as presented in Fig. 6. Each of the activities is discussed in terms of the income, outcome, and human input for the individual activity. The theoretical details about the process activities provide the foundations to discuss the case study in the next subsection.

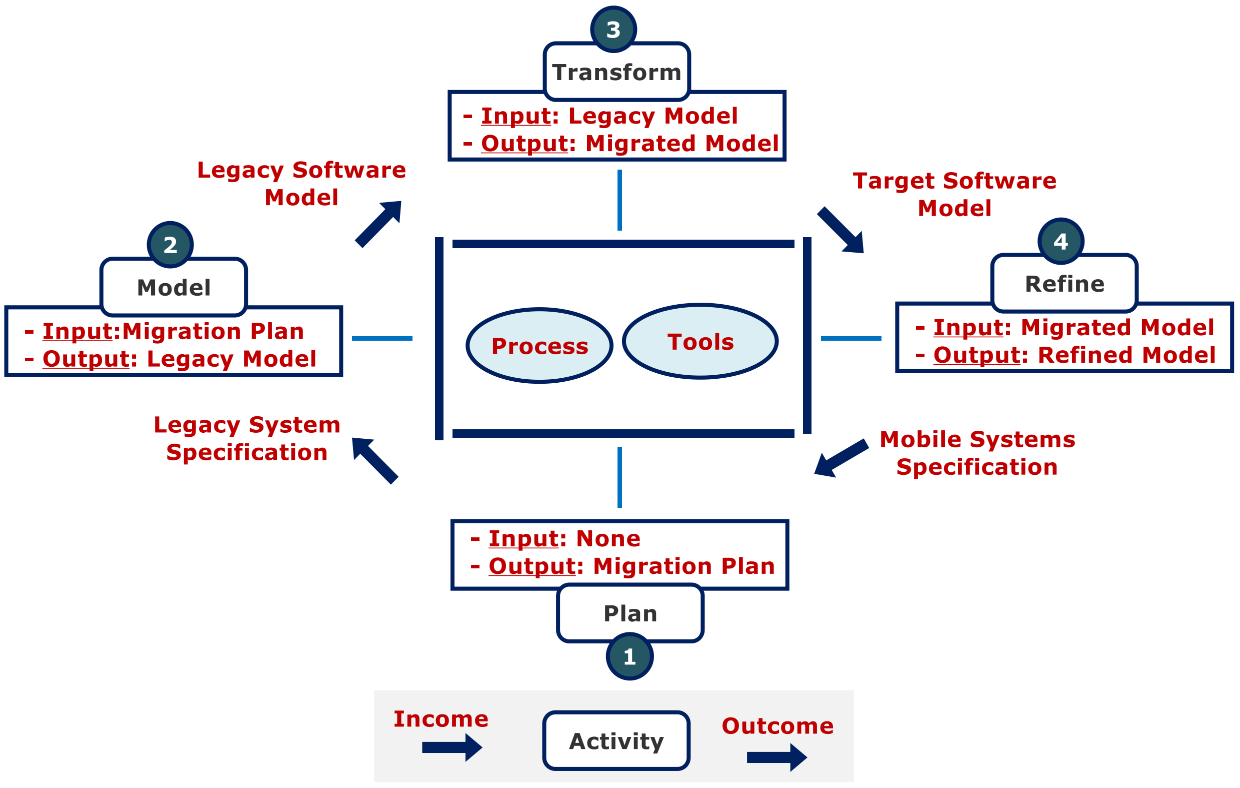


Fig. 6: Overview of the process for the evolution of legacy software to the mobile computing platform

6.1.1. Activity I–Requirement analysis and planning

The first activity in legacy to mobile evolution, as presented in Fig. 6 relates to requirement analysis and planning for the evolution of existing software. Requirement analysis helps to identify, prioritize and manage the requirements for software evolution. Based on the requirement analysis, an evolution plan is derived that guides the further activities of the process. The activity I aims to investigate how to develop an evolution plan that accommodates the needs for software evolution. It also aims to assess the feasibility of executing such an evolution to support an incremental change of existing software as per the needs for mobile computing platforms.

- Activity Income and Outcome: As the initial activity of the process, it does not require any input. Instead, the needs and motivations for software evolution can be considered as the pre-requisites for evolution planning. The outcome of the activity

is an evolution plan that acts as a blueprint and a documented reference for the stakeholders before proceeding further with the evolution.
- Human Input: This activity requires input from the requirements engineer to identify and specify the requirements for evolution. Based on the identified requirements, an evolution plan is also created based on human intervention.

6.1.2. Activity II–Design modeling for existing software

After requirement analysis and planning, the next activity is design modeling (i.e., structural representation) of existing software that needs to be transformed. Design modeling provides an overall structural view of the existing software as a blueprint of the system. The model can abstract implementation-specific complexities of the software with a design or architectural view that presents a graphical (high-level) view of software. Therefore, the model as a high-level graphical representation of existing software supports analysis and transformation of the system at a higher level of abstractions.

- Activity Income and Outcome: The income to this activity is the specification (source code or design) of the existing software system that is the candidate for the evolution. The outcome is the model (high-level view of the system) in terms of the design or architecture of the existing system that needs to be evolved.
- Human Input: Human input is required in terms of the software designer providing the existing source code for its design model. The designer also needs to check the structural consistency of the legacy architecture.

6.1.3. Activity III–Change management to support evolution

After modeling the design or architecture of the existing software, this activity supports change management in the existing architecture so it can be evolved. Change management refers to the addition, removal, or modifications of the components and connectors of the existing architecture. These changes update the existing architecture as per the requirements of the architecture for the mobile application. For example, the addition of the Location Sensing component can be added during change management activity to enable location sensing as part of context-awareness of the mobile application.

- Activity Income and Outcome: The income for this activity is the architecture model of the existing software that has been created in the design modeling activity. This architecture can be changed to support evolution. The outcome is the changed or evolved architecture that has new functionality for mobile applications.
- Human Input: It is needed in terms of guiding and supporting change management. The human role in terms of designer or architect needs to ensure that the structural consistency of the architecture is preserved as a consequence of the change management.

6.1.4. Activity IV–Evaluation of the evolved software

The last activity in the process is about the evaluation of the evolved software to analyze if the evolved software supports the required functionality and desired quality. During the evaluation activity, specific quality attributes such as usability of the evolved software and computation efficiency are met. Despite the benefits such as portability and context-sensitivity, mobile devices lack computation (process), storage (memory), and energy (battery) resources. Therefore, there is a need to objectively evaluate the performance of the evolved software on resource-constrained mobile devices.

- Activity Income and Outcome: The income to this activity is the evolved system after change management that needs to be evaluated. The outcome is evaluation results that help with an objective interpretation of the usability and efficiency of the evolved software when it is executed on resource-constrained mobile devices.
- Human Input: The input from the human is required in terms of the testing engineer to evaluate the application and analyze the results in the context of benchmarks for usability and performance.

6.2. Case study for the evolution of a web portal to mobile application

A case study-based approach has been used to demonstrate the applicability of the framework and the role of the framework's processes and activities as in Table 3. The case study helps to present the results of the preliminary evaluation for computational efficiency of evolved software when it is transitioned from a traditional platform (web/desktop system) to a resource-constrained (handheld/mobile) platform.

Evolution of Web Portal to Mobile Recommender Application: The existing web-based portal enables its user to search and identify the products of interest (such as cellular devices, watches, and accessories) driven by user's preferences and search histories. In the presence of handheld, context-sensitive computing devices, the web portal offers limited functionality by lacking context-aware recommendations that can be provided to the users. Therefore, the basic requirement of the web portal is to evolve it so that it acquires new functionality by offering relevant products based on user's context information. The evolution of the web portal as a mobile application can exploit dynamic context information based on the user's location, time, weather, and other information to generate context-driven recommendations. In addition, the mobile user can also share the selected recommendations with his/her contact list. The evolution of the web portal system as a mobile application is supported by the evolution process in Fig. 6, where each of the processes is detailed as below in the context of the above-detailed case study.

Table 3: Summary of activities and elements of the evolution process

Process Activity	Process Sub Activities	Process Income	Process Outcome	Process Automation (Tool Support)	Process Supervision (Human Decision)
Activity I Evolution Analysis and Planning	- Perform Trade-off Analysis - Identify Level of Evolution	None	Evolution Plan - Motivations - Challenges - Level of Evolution	No	Yes
This activity takes into consideration the source and target platforms, benefits, and limitations along with the required efforts to enable the evolution.					
Activity II Design Modeling for Evolution	- Code Analysis - Architectural Representation	Legacy Software Specifications	Legacy Software Model	Yes	Yes
This activity aims to develop an overall understanding of the overall structure of the legacy software and identify areas that require structural changes during evolution.					
Activity III Software Change Management	- Architecture Transformation - Code Representation	Legacy Software Model	Target/Evolved Software Model	Yes	Yes
This activity supports the changes in the structure of existing software by means of change implementation to evolve it as per the requirements of the new platform/software.					
Activity III Evaluation of the Evolved Software	- Software Efficiency - Software Usability	Target/Evolved Software Model	Refined Software Model	No	Yes
This activity supports the evaluation of the evolved software in terms of desired quality requirements.					

6.2.1. Process I–Evolution analysis and planning

Evolution analysis and planning as the initial process streamlines the activities and steps that analyze the existing software requirements for the evolution and analysis of the target platform (Seffah, 2015). The initial process involves two main activities that include (i) performing trade-off analysis and (ii) selection of the level of software abstraction for change implementation as in Table 3, Fig. 6.

- Process Activities: The initial activity of the analysis and planning phase relates to trade-off analysis in terms of motivations and challenges for software evolution (Ahmad and Babar, 2014). Details about the motivations and challenges have already been detailed in Section 5 (Table 2, Fig. 5). Specifically, the prime motivation for the evolution of existing software to the mobile application is context-sensitivity and mobility of product recommendations that also enhance user interaction. To attain context-sensitivity and portability mobile devices lack computation, power, and storage resources that make them resource-poor. For example, the evolution of the web portal as a mobile application empowers its users to exploit location information for product recommendations. Along with resource poverty and smaller display size, data security and privacy are also primary challenges for mobile applications. After analysis and planning, the second activity relates to the selection of software abstraction level in terms of software architecture or source code for change management and software evolution. Architecture-centric evolution can support change management at higher abstraction levels while hiding the complexities of source code refactoring (Fan and Wong, 2016).
- Process Input and Output: As the initial phase, evolution analysis and planning have no formal income. The outcome of the process is an evolution plan that guides software evolution.
- Process Support: Analysis and planning is a manual process that requires human intervention to derive the plan.

6.2.2. Process II–Design modeling

The design modeling process aims to model a high-level design of the existing software that needs to be changed. By modeling design, an overall structure of the existing software is identified with points in software structures that can be evolved as in Table 3, Fig. 6.

- Process Activities: In the case study, the architectural model of the web portal is presented as in Fig. 7. Please note that for demonstrative purposes only a partial architectural model is presented in Fig. 7. An architectural model is presented in terms of architectural components and connectors as in Fig. 7 (Process II–Design Modeling). Architecture modeling supports components and connector level representation that abstract complex implementation-specific details of the software. In the case study, the architecture represents the Product List component that depends on the User Preferences component to generate the list of recommended products. The mapping between source code and architectural representation is presented in Fig. 7.

Architecture-based evolution is supported by exploiting existing research and practices that have proven useful to support software evolution (Jamshidi et al., 2013b; Fan and Wong, 2016).

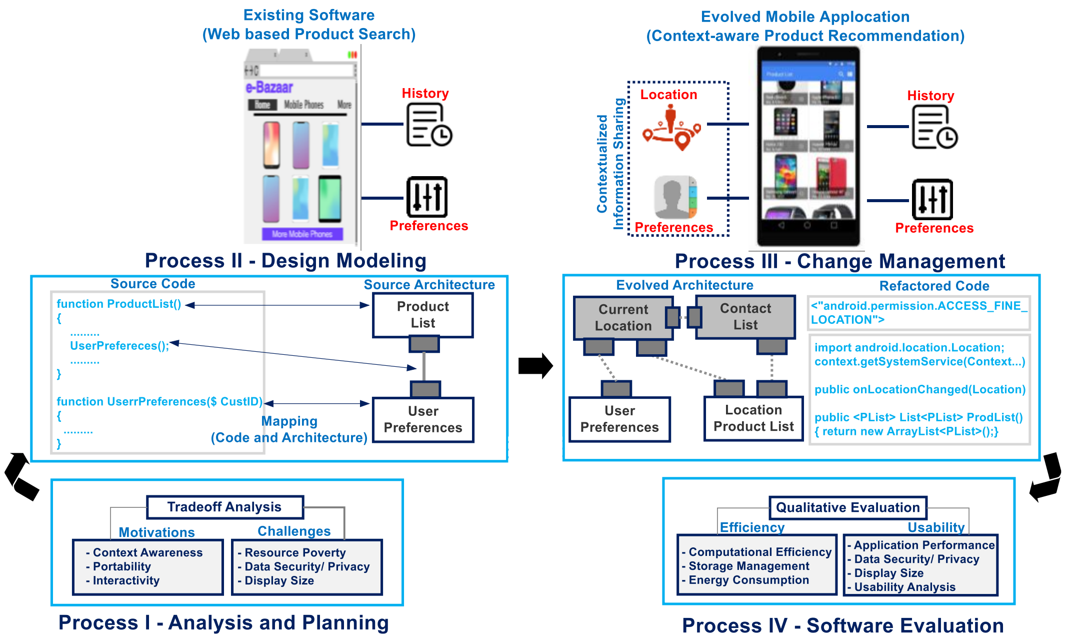
- Process Input and Output: The input to this process is the source code of the software that needs evolution. The output is the architecture of existing software that needs to be evolved as in Table 3.
- Process Support: The process is supported with the relevant tool and user intervention that automates and supervises the design modeling of existing software.

6.2.3. Process III–Change management

After the design modeling, the next process focuses on implementing the desired changes in the architectural model to support software evolution. Specifically, by adding or removing architectural components and connectors, the architecture of the existing web portal is evolved as a mobile-enabled application.

- Process Activities: Architecture base change management adds or removes architectural components and connectors in the existing architecture as in Fig. 6 (Process III–Change Management). As per Fig. 7, new components Current Location and Contact List with desires connectors are added to the existing architecture, whereas the connection between Product List and User Preferences has been removed. The newly introduced components in the existing architecture namely Current Location and Contact List enable the mobile application to first allow the user to get product recommendations based on geo-proximity of the user and then allow them to share those recommendations with the context list. The evolved architecture model supports context-aware recommendations that can also be shared with device contacts. The evolved architecture is used to generate the source code that can be executed on android powered mobile devices. The changes in the architecture are implemented based on architecture-centric software evolution (Ahmad et al., 2018).
- Process Input and Output: The input to the process is the design model (i.e., architecture) of the existing software. The output is the evolved architecture that represents the blueprint for context-sensitive mobile-enabled applications.
- Process Support: Change management is supported by means of architectural transformation. Architectural transformation is a semi-automated process that involves an appropriate human intervention to guide and execute change management.

After presenting the first three phases of the solution from Fig. 7, the final process that involves software evaluation is presented as a dedicated Section 7.



```
function ProductList()
{
   .........
   UserPrefereces();
   .........
}

function UserrPreferences($ CustID)
{
  .........
}
```

```
<"android.permission.ACCESS_FINE_
LOCATION">

import android.location.Location;
context.getSystemService(Context...)

public onLocationChanged(Location)

public <PList> List<PList> ProdList()
{ return new ArrayList<PList>();}
```

Fig. 7: Case study overview–The evolution of a web portal as a mobile-enabled application

7. Results for evaluation of the existing software

After presenting the process and its demonstration with a case study, we now present the results of the evaluation. Evaluation results are obtained by validating the evolved software based on some predefined criteria. Specifically, the ISO-IEC (2006) model for software quality has been used to select the evaluation criteria in terms of efficiency and usability of the evolved software in Section 7.1. and Section 7.2., respectively. Moreover, we also present some threats to the validity of the research in Section 7.3. The discussion about the results of the evaluation is guided by Fig. 8.

7.1. Evaluating the efficiency of evolved software

As highlighted earlier, we need to evaluate the efficiency of evolved software. Efficiency evaluation refers to validating the utilization of computation, memory, energy resources of resource-constrained mobile devices. By evaluating the efficiency of the evolved software we aim to find an answer to: How efficient is the evolved software in terms of its utilization of the computation, memory, and energy resources of resource-constrained mobile devices?

7.1.1. Evaluating the computation efficiency of the evolved software

In order to evaluate the computation efficiency, i.e., utilization of the device's CPU we measured it by using the CPU Monitor software. The software runs as a background application and profiles the usage of the device CPU. We executed the application and conducted approximately 100 trials of CPU utilization when the evolved application is running. We have plotted the CPU utilization as an average of 100 trials in Fig. 8. Fig. 8 shows a graph based on the following two cases:

- Case 1– Maximum CPU utilization in Fig. 8a) (red bound area) that profiles the maximum use case.
- Case 2– Minimum CPU utilization in Fig. 8a) (green bound area) that profiles the minimum use case.

The CPU utilization graph reflects that maximum usage of CPU is approximately 10% as in Fig. 8a), while minimum utilization reaches up to 3% of the total CPU in Fig. 8b). Maximum utilization happens when the application is running as a foreground application (active state), while the minimum utilization occurs when the application is running as a background application (idle state). We can conclude that based on our trials we observed that CPU utilization is quite acceptable that remains between 03% to 10%, as illustrated in Fig. 8.

7.1.2. Evaluating the energy efficiency of the evolved software

After evaluating the computation efficiency, we now also need to evaluate the energy efficiency of the evolved application. Evaluating energy efficiency refers to analyzing the utilization of battery resources of mobile devices by evolved software.

In order to profile and evaluate the battery utilization, we have used the AccuBattery software that logs the usage of batteries by any specific app. Like the evaluation of CPU utilization, we also conducted approximately 100 trials for the usage of the evolved software and its impacts on the device's battery. We present two cases:

- Case 1–Maximum use of the device's battery in Fig. 8b) (blue bound area) that profiles the maximum use case for battery.
- Case 2–Minimum use of the device's battery in Fig. 8b (yellow bound area) that profiles the minimum use case for battery.

The evaluation results suggest that maximum utilization of the battery happens when the application is active, it uses 08% (i.e., 22 mAh) of memory at max. Alternatively, when the application is idle it only consumes 3% (i.e., 9 mAh of memory). Based on the illustrations in Fig. 7, we can conclude that based on our trials, we have found the evolved mobile application is efficient in terms of battery utilization of the device.

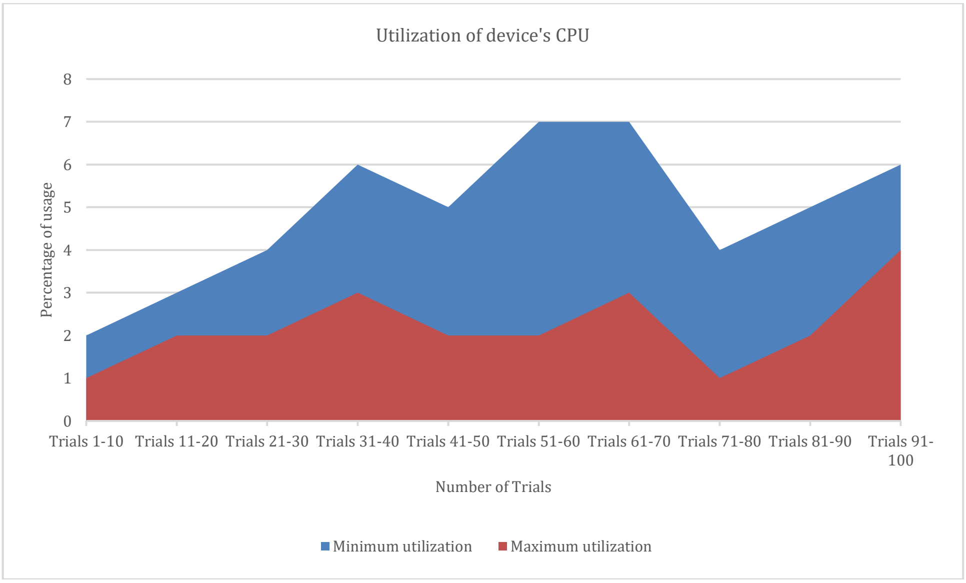
7.1.3. Evaluating the memory efficiency of the evolved software

We also discuss the memory efficiency of the evolved application in the context of mobile cloud computing. Specifically, mobile cloud computing helps front-end mobile devices to offload memory-intensive data to the cloud servers (Ahmad et al., 2014a). We haven't explicitly evaluated the memory efficiency as it represents part of the future work. The early assessment suggested that by integrating a cloud server with the evolved application memory-intensive data can be offloaded to the cloud and it can be accessed when required to address memory limitation issues of the mobile device.

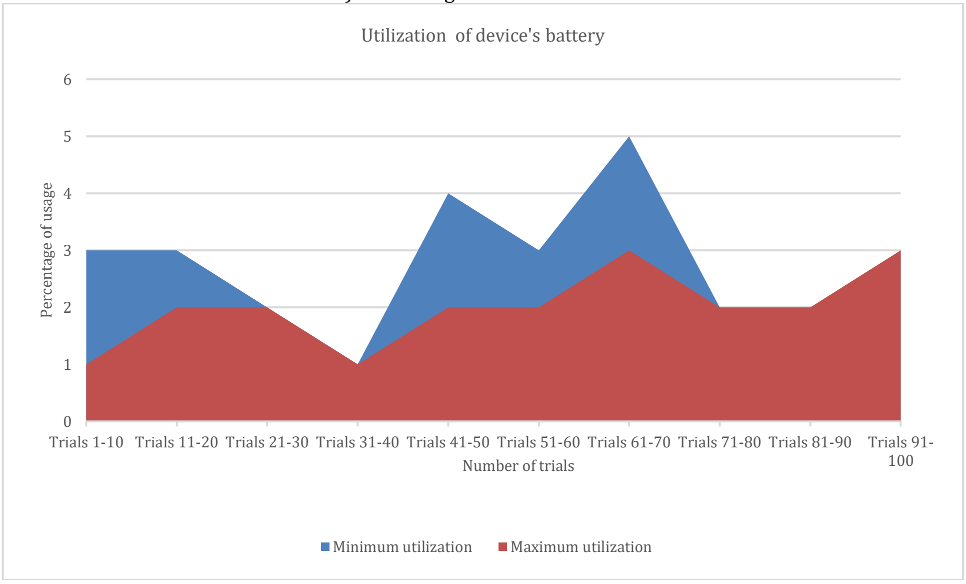
7.2. Evaluating the usability of evolved software

After evaluating the efficiency, we now also evaluate the usability of the evolved application. Usability evaluation refers to analyzing and assessing the ease of use by potential users of the software. One of the mechanisms of usability evaluation is to conduct a usability survey of the potential users of the evolved software as a mobile application. In order to do so, we selected a sample of 15 different users (age between 18 to 57, with gender diversity). We engaged these potential users to first use the application and then present a survey questionnaire to record their responses based on an 8-point criterion as illustrated in Fig. 8c). Fig. 8c) shows a bar graph based on the following data from 15 respondents:

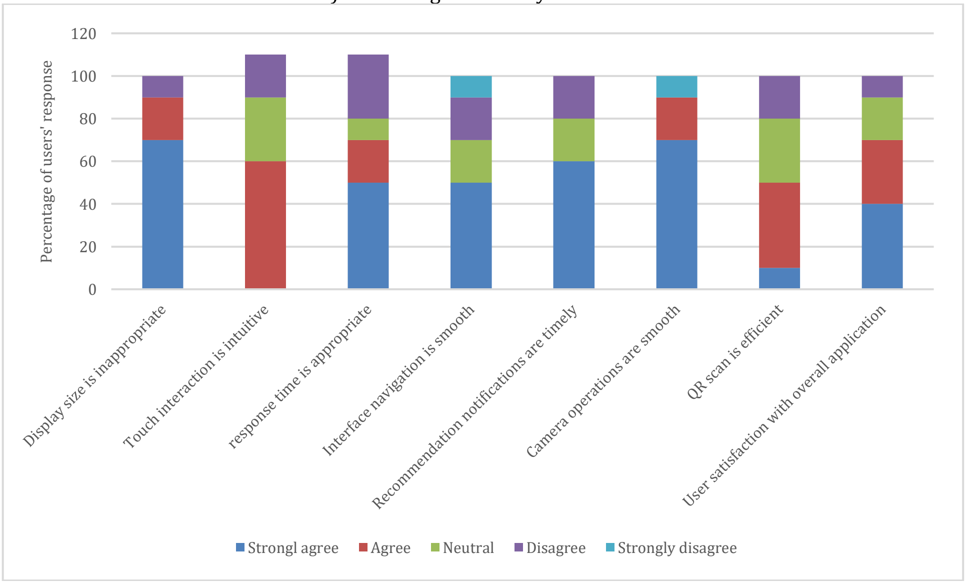
- Level of Usability: We use five standard levels as Strongly Agree, Agree, Neutral, Disagree, and Strongly Disagree each given a weightage of 20% on the y-axis in Fig. 8c).
- Criterion for Evaluation: There are a total of 8 distinct criteria to evaluate the usability on the x-axis in Fig. 8.



a) Evaluating the CPU Utilization



b) Evaluating the Battery Utilization



c) Evaluating the Usability

Fig. 8: Overview of the evaluation results

Based on the survey-based usability analysis, we can conclude that majority of the users were satisfied and suggested the evolved software as usable. The survey analysis highlights that some aspects like navigation between interfaces and display size are the factors that need to be addressed to further enhance the usability of the software. Specifically, the response time of the evolved software and smooth navigation between user interfaces needs to be optimized to further enhance usability.

7.3. Threats to validity

After presenting the results of the evaluation, we must also discuss some threats to the validity of the research and its results. Validity threats represent the conditions and scenarios that need to be addressed to eliminate any potential bias and/or threats from the evaluation. We briefly discuss each of the threats as below. Validity threats need to be addressed as part of future research to further strengthen the proposed solution.

7.3.1. Threat I-Evaluating the process in the real context

One of the prominent threats relates to the lack of evaluation of the evolution process in a real context. Currently, the process is evaluated with a small case study that needs extended evaluations with a more comprehensive system to the mobile computing platform. Currently, the case study that has been used to demonstrate and evaluate the proposed process represents a small-scale system that may not be a true reflection of a real-world system. In order for more rigorous evaluation, there is a need to identify the real-world system that needs to be evolved to mobile platforms. As part of the future work, the process needs to be customized and scaled up so it can address large-scale application evolution as a mobile-enabled application.

7.3.2. Threat II-Availably of diverse scenarios and case studies

Currently, the usability and efficiency of the evolved application are evaluated based on controlled experiments that need to be extended. A limited number of scenarios and case study only evaluate the evolved software and process at a limited scale. In order to minimize these threats, future research efforts are required to identify more diverse scenarios for evaluating the quality attributes like usability and efficiency of the software. Evolution patterns as artifacts of reuse can support a reuse-driven and knowledge-oriented approach to support software evolution for mobile computing.

7.3.3. Threat III-Evaluating the non-functional aspects

Finally, as part of the evaluation, we could not manage to analyze and evaluate other non-functional properties like data security and privacy that represent two critical challenges for mobile computing. Therefore, in order to minimize this threat, as part of future research, a security threat matrix needs to be established that can be objectively evaluated to the security and privacy of the evolved application. Similarly, other non-functional aspects or quality attributes such as computation efficiency or usability analysis can also be rigorously evaluated based on comprehensive scenarios and detailed case studies as identified earlier.

8. Conclusions and discussion of future work

This research highlights the need for the evolution of legacy software systems to mobile computing platforms. The principle of software maintenance and evolution can be exploited in terms of processes, patterns, tools, and methods to support an incremental evolution of legacy software to mobile computing systems. The primary objectives of legacy evolution to mobile are driven by the fact that evolved applications can exploit features of mobile computing such as portability and context-sensitivity of computing. However, there are certain challenges such as maintaining the privacy of device data and resource poverty of mobile platforms that needs to be addressed by the evolution process (Alreshidi and Ahmad, 2019). Before conclusive remarks, some dimensions of future research are discussed along with implications of research and its results as detailed below.

8.1. Dimensions of future research

The results highlight the following two areas as dimensions of future research. This means that future research and development can focus on these findings and alike areas to support a new generation of the solution to address the emerging challenges of legacy evolution to mobile computing

- Tools and Framework Support for Legacy Evolution: As highlighted in the legacy evolution process (Fig. 6), there is a need for tools and technologies that can support the automation of the evolution process (Ahmad et al., 2019b). Legacy evolution is a complex and multifaceted process that requires lots of effort to implement changes in existing software systems. Due to an inherent complexity and efforts of change implementation, manual efforts for evolution can be complex, time-consuming, and labor-intensive. In this context, tool support can provide the foundations for automation to plan, manage, and execute the evolution in an efficient and cost-effective manner(Ahmad et al., 2019a).
- Process Patterns for Legacy Evolution: Patterns as best practices and reusable knowledge support the reusable rationale and knowledge to support evolution. This means that the most frequent and routine evolution tasks can be represented as patterns that can be reused later. Future research needs to empirically discover, document, and

exploit patterns that support the reuse-driven and quality-oriented evolution of existing software. Moreover, patterns as elements of reuse can be integrated into the evolution process and utilized by the tool(s) to support reusable automated evolution. Some of the fundamental challenges for the discovery and application of patterns can be related to empirical research and practices for patterns-based legacy evolution (Ahmad et al., 2018).

8.2. Implications of research findings

Finally, the implications of the proposed study and its results are highlighted. The results of the mapping study can be beneficial for:

- Researchers who are interested in identifying the relevant literature and need to know about the motivations and challenges for legacy evolution to mobile computing environments. The findings of the mapping study can also help researchers to derive new hypotheses to be tested.
- Practitioners who like to know the existing research on motivations, challenges, and available processes can help to develop tools, patterns, processes that support legacy evolution. Academic research and industrial practices can be unified to develop innovative solutions for legacy evolution to mobile.

Compliance with ethical standards

Conflict of interest

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

References

Ahmad A and Babar MA (2014). A framework for architecture-driven migration of legacy systems to cloud-enabled software. In the WICSA 2014 Companion Volume, Association for Computing Machinery, Sydney, Australia: 1-8. https://doi.org/10.1145/2578128.2578232

Ahmad A, Alseadoon I, Alkhalil A, and Sultan K (2019a). A framework for the evolution of legacy software towards context-aware and portable mobile computing applications. In the International Conference on Software Engineering Research and Practice, CSREA Press, Las Vegas, USA: 3-9.

Ahmad A, Jamshidi P, and Pahl C (2012). Pattern-driven reuse in architecture-centric evolution for service software. In the 7th International Conference on Software Paradigm Trends, Rome, Italy.

Ahmad A, Jamshidi P, and Pahl C (2014a). Classification and comparison of architecture evolution reuse knowledge- A systematic review. Journal of Software: Evolution and Process, 26(7): 654-691. https://doi.org/10.1002/smr.1643

Ahmad A, Malik AW, Alreshidi A, Khan W, and Sajjad M (2019b). Adaptive security for self-protection of mobile computing devices. Mobile Networks and Applications, 1-20. https://doi.org/10.1007/s11036-019-01355-y

Ahmad A, Pahl C, Altamimi AB, and Alreshidi A (2018). Mining patterns from change logs to support reuse-driven evolution of software architectures. Journal of Computer Science and Technology, 33(6): 1278-1306. https://doi.org/10.1007/s11390-018-1887-3

Alreshidi A and Ahmad A (2019). Architecting software for the internet of thing based systems. Future Internet, 11(7): 153. https://doi.org/10.3390/fi11070153

Assunção WK, Lopez-Herrejon RE, Linsbauer L, Vergilio SR, and Egyed A (2017). Reengineering legacy applications into software product lines: A systematic mapping. Empirical Software Engineering, 22(6): 2972-3016. https://doi.org/10.1007/s10664-017-9499-z

Brereton P, Kitchenham BA, Budgen D, Turner M, and Khalil M (2007). Lessons from applying the systematic literature review process within the software engineering domain. Journal of Systems and Software, 80(4): 571-583. https://doi.org/10.1016/j.jss.2006.07.009

Bruschi R, Davoli F, Lago P, Lombardo C, and Pajo JF (2018). Personal services placement and low-latency migration in edge computing environments. In the IEEE Conference on Network Function Virtualization and Software Defined Networks, IEEE, Verona, Italy: 1-6. https://doi.org/10.1109/NFV-SDN.2018.8725635 **PMid:28347192**

Businge J, Openja M, Nadi S, Bainomugisha E, and Berger T (2018). Clone-based variability management in the android ecosystem. In the IEEE International Conference on Software Maintenance and Evolution, IEEE, Madrid, Spain: 625-634. https://doi.org/10.1109/ICSME.2018.00072

Canfora G, Di Santo G, and Zimeo E (2004). Toward seamless migration of Java AWT-based applications to personal wireless devices. In the 11th Working Conference on Reverse Engineering, IEEE, Delft, Netherlands: 38-47. https://doi.org/10.1109/WCRE.2004.38

Cheng L, Cai H, and Jiang L (2012). Research on code migration framework for mobile computing. In the 2nd International Conference on Cloud and Green Computing, IEEE, Xiangtan, China: 230-236. https://doi.org/10.1109/CGC.2012.55

Emmerich W, Mascolo C, and Finkelstein A (2000). Implementing incremental code migration with XML. In the International Conference on Software Engineering. ICSE 2000 the New Millennium, IEEE, Limerick, Ireland: 397-406. https://doi.org/10.1145/337180.337227

Fan CY and Ma SP (2017). Migrating monolithic mobile application to microservice architecture: An experiment report. In the IEEE International Conference on AI and Mobile Services, IEEE, Honolulu, USA: 109-112. https://doi.org/10.1109/AIMS.2017.23

Fan X and Wong K (2016). Migrating user interfaces in native mobile applications: Android to iOS. In the IEEE/ACM International Conference on Mobile Software Engineering and Systems, IEEE, Austin, USA: 210-213. https://doi.org/10.1145/2897073.2897101 **PMCid:PMC4778641**

Foss A and Wong K (2004). On migrating a legacy application to the palm platform. In the 12th IEEE International Workshop on Program Comprehension, 2004, IEEE, Bari, Italy: 231-235. https://doi.org/10.1109/WPC.2004.1311065

Hansen HV, Goebel V, and Plagemann T (2017). TRAMP real-time application mobility platform. IEEE Transactions on Mobile Computing, 16(11): 3236-3249. https://doi.org/10.1109/TMC.2017.2688421

ISO-IEC (2006). ISO/IEC 9126-1: 200 Software engineering-Product quality- Part 1: Quality model. Available online at: https://www.iso.org/standard/22749.html

Jamshidi P, Ahmad A, and Pahl C (2013b). Cloud migration research: A systematic review. IEEE Transactions on Cloud Computing, 1(2): 142-157. https://doi.org/10.1109/TCC.2013.10

Jamshidi P, Ghafari M, Ahmad A, and Pahl C (2013a). A framework for classifying and comparing architecture-centric software evolution research. In the 17th European Conference on Software Maintenance and Reengineering, IEEE, Genova, Italy: 305-314. https://doi.org/10.1109/CSMR.2013.39

Khadka R, Saeidi A, Idu A, Hage J, and Jansen S (2013). Legacy to SOA evolution: A systematic literature review. In: Ionita, AD, Litoiu M, and Lewis G (Eds.), Migrating Legacy Applications: Challenges in Service Oriented Architecture and Cloud Computing Environments: 40-70. IGI Global, Pennsylvania, USA. https://doi.org/10.4018/978-1-4666-2488-7.ch003

Lane ND, Miluzzo E, Lu H, Peebles D, Choudhury T, and Campbell AT (2010). A survey of mobile phone sensing. IEEE Communications Magazine, 48(9): 140-150. https://doi.org/10.1109/MCOM.2010.5560598

Mens T (2008). Introduction and roadmap: History and challenges of software evolution. In: Mens T and Demeyer S (Eds.), Software evolution: 1-11. Springer, Berlin, Germany. https://doi.org/10.1007/978-3-540-76440-3_1

Pejovic V and Musolesi M (2015). Anticipatory mobile computing: A survey of the state of the art and research challenges. ACM Computing Surveys, 47(3): 1-29. https://doi.org/10.1145/2693843

Petersen K, Feldt R, Mujtaba S, and Mattsson M (2008). Systematic mapping studies in software engineering. In the 12th International Conference on Evaluation and Assessment in Software Engineering, Bari, Italy, 12: 1-0. https://doi.org/10.14236/ewic/EASE2008.8

Pope S (1996). Application migration for mobile computers. In the 3rd International Workshop on Services in Distributed and Networked Environments, IEEE, Macau, Macau: 20-26. https://doi.org/10.1109/SDNE.1996.502443

Seffah A (2015). Patterns of HCI design and HCI design of patterns: Bridging HCI design and model-driven software engineering. Springer, Berlin, Germany. https://doi.org/10.1007/978-3-319-15687-3 **PMid:27738618 PMCid:PMC5020890**