Effort Estimation across Mobile App Platforms using Agile Processes: A Systematic Literature Review

Abdullah R Altaleb[1, 2*], Andrew M Gravell[1]

[1] Department of Electronics and Computer Science, University of Southampton, Southampton, UK
[2] College of Computer and Information Sciences, Al-Imam Mohammad Ibn Saud University, Riyadh, KSA

* Corresponding author. Tel.: +447533287832; email: a.altaleb@soton.ac.uk
Manuscript submitted March 15, 2018; accepted April 30, 2018.
doi: 10.17706/jsw.13.4.242-259

Abstract: It is predicted that smartphone numbers will increase to 2.8B in 2018, up from around 2B in 2016. Moreover, revenue from application stores is predicted to reach $189B by 2020, up from $88.3B in 2016[1]. Software effort and size estimation are essential when it comes to project managers being able to manage and plan a project so as to prevent it from failing. The planning and development of mobile applications differs from other traditional software applications due to the characteristics of the mobile environment, high autonomy requirements, market competition, and many other constraints. Therefore, this paper presents the results of a Systematic Literature Review (SLR) concerning effort and size estimation models in mobile application development; this is followed by a summary of estimation techniques used across mobile apps. In particular, we focus on the software estimation models that are applicable to the Agile Software Development (ASD) process. The aim of this SLR is to provide researchers and practitioners with an overview of the current state-of-the-art software estimation techniques used in mobile applications. At the end of this review, some suggestions, research gaps and possible future work will be presented.

Key words: Agile, effort estimation, software development, mobile application.

1. Introduction

Effort estimation plays an important and critical role in any software project. Both project managers and clients use the effort estimation model to measure the effort, size, cost, and duration related to the design and implementation of the software project to generate a contract. Good estimation is one of the success factors for companies, as offering a good price could mean that they win a contract. Incorrect estimation can negatively affect companies' marketing and sales, while also leading to monetary losses.

Years ago, mobile devices were used only for receiving and sending messages and calls; today, however, they are becoming essential to human life. Among the many examples of smartphone usage are making flight bookings, GPS navigation, banking transactions, gaming, and communicating with others on social media. Mobile applications represent a new trend in the software industry, and the demand for such apps is set to increase expeditiously with mobile technology development. Planning and developing a mobile application is different from the development of other traditional software systems in many aspects. Mobile devices are becoming increasingly complex, and designing their application development presents some

[1] https://www.statista.com/statistics/269025/worldwide-mobile-app-revenue-forecast/ statistics and studies from more than 22,500 sources.

special requirements and constraints that are less commonly found in other traditional software applications.

The main goal of this paper is to examine the state-of-the-art techniques used in the field of effort estimation across mobile applications in Agile Software Development (ASD) by means of a Systematic Literature Review (SLR). We follow the guidelines proposed by Kitchenham and Charters [1] to construct this SLR. We found no previous literature or studies concerning all software estimation models in mobile applications. This paper identifies, critically evaluates and integrates the findings of all studies which are relevant to the subjects addressed by the research questions. This paper also details our SLR and points out some research gaps and future directions in the field of effort estimation with mobile app developments in ASD.

The present paper is organised as follows: Section 2 puts forth a brief background of mobile application and its characteristics, as well as effort estimation models; the steps undertaken in this SLR are detailed in Section 3; following this, the results of the SLR and discussion regarding said results will be presented in Section 4. Research gaps and possible directions for future works are provided in Section 5.

2. Background

2.1. Mobile Application Development

Traditional software development methodologies are not well suited to mobile application development [2]. Mobile software development faces exceptional constraints and requirements. It is necessary to apply suitable methods that are able to tackle the challenges faced during mobile app development. One of the notable challenges is the multiple platform issue [3], [4]. Developers always plan for, and are concerned about, the compatibility of their app with all mobile operating systems, such as IOS, Android, Microsoft Windows Mobile, and Blackberry. Additional challenges include time-to-market, UI limitation, power consumption, processor efficiency, memory performance and intermittent network connection [5].

2.2. Effort, Size, and Cost Estimation

Software Estimation in General:

For each software project, an accurate software estimation is desirable and interesting for any aspect of the project. Predicting the project's timescale, scheduling the budget, and estimating the resources will help the software project to succeed and avoid overrun [6]. There are four different types of software estimation:

1) Effort estimation: amount of effort needed to complete a project. It is usually measured by person/hour or person/month.
2) Size estimation: usually measured by number of code line (LOC) or Function Point (FP) to implement the software.
3) Cost estimation: amount of money that is required to develop the project.
4) Schedule estimation: amount of time (period) required to complete the project.

There are several types of estimation models and techniques that can be used for software estimation. Among these estimation methodologies, there is no best approach; indeed, each technique has its pros and cons. The estimation techniques can be classified into two major types, namely algorithmic and non-algorithmic techniques [7]. The former is based on equations and mathematics, which are used to process the software estimation; in contrast, the latter is based on analogy and deduction.

Britto, Freitas, Mendes and Usman [8] presented a systematic literature review of effort estimation in the context of Global Software Development (GSD). Five studies out of 24 were selected in this SLR [9]–[13], and the expert judgment based approach, included planning poker, Delphi and expert review, were the most used technique in effort estimation for GSD. Furthermore, this SLR identified a wide range of cost drivers;

however, cultural, language and time zone were the most popular cost driver factors that used to estimate the effort in GSD projects.

Software Estimation in Agile SD:

A systematic literature review (Usman, Mendes, Weidt, and Britto) [14] makes it possible for us to present a detailed overview of the effort and size estimation techniques used in the ASD process. A total of 20 primary studies were selected, all of which satisfied the inclusion criteria and passed the quality assessments which were used during this literature review. Planning poker and expert judgment were the most frequently-used techniques in the ASD process [15]–[22], while story point, use-case point and function point were the most commonly-employed size metrics in these techniques. In total, 12% of the estimation techniques studied the estimation effort during the implementation phase, while this figure was around 30% in the testing phase, and 12% in the I & T phase; however, none of the primary studies had estimated the effort in the design or analysis phase of the development process.

Another SLR was presented by Bilgaiyan, Sagnika, Mishra and Das that focused on software cost estimation in ASD [23]. Varieties of estimation mechanisms were founded in this SLR such as Neural Network, Expert Judgment, Planning Poker and Use Case Point; however, Expert Judgment was one of the most common estimation techniques among conventional methods in ASD [24]–[28].

3. Literature Review Method

The design of the literature review in this study was inspired by the Evidence-based Software Engineering (EBSE) model; indeed, the guidelines from this model were followed Kitchenham [29] in order to synthesise the best quality scientific studies. In addition, a mixture of assessment criteria was used in this SLR, including the AMSAR (A Measurement Tool to Assess Systematic Review) standard [30] and some assessment guidelines from Kitchenham and Charters [1]. These were employed to assess and measure the reliability, validity and feasibility of measurement tools in this literature. In the first part of the present section, the scope of the research and the research questions will be laid out. Following this, the second section will detail the search strategies as well as the inclusion and exclusion criteria. This literature review will end with a presentation of the studies assessment criteria.

3.1. Scope and Research Questions

Research Scope:

Before starting the research review, it was necessary to ask a very important question: is the systematic review really needed? We had to be aware of asking general questions, otherwise it would have been difficult to determine the usefulness of the answers to the questions.

Moreover, before stating the questions, it was helpful to break down said questions into sub-questions to make them more precise and detailed. For the following questions, we used the PICOC model [31]. This model is very effective when it comes to framing and considering the components of questions.

Population: ASD process and Mobile Application platforms projects.

Intervention: Effort/Size Estimation Methodologies/Model for mobile apps.

Comparison: Different effort estimation techniques in mobile apps.

Outcome: The precision and accuracy of studies that have focused on effort estimation in mobile applications.

Context: All studies (Empirical, Hypotheses) related to Agile SD in effort estimation range.

Research Questions:

RQ1. What methods or techniques have been used to estimate effort in mobile application development using Agile methods?

RQ2. What are the effort predictors that have been used in the software estimation technique in Agile SD?

RQ3. What are the characteristics of the dataset/knowledge that have been used in studies to estimate effort in mobile applications?

- What is the type of project (educational/economical etc.)?
- What type of framework has been used on the mobile app (mobile web/native/hybrid application)?
- What type of operating system has been used on the mobile app (Android/IOS/Windows)?

RQ4. How accurate and efficient are the estimation methods?

3.2. Primary and Secondary Search Strategy

Search Criteria:

After stating the research questions, we broke said questions down into certain terms which we could use to find any relevant articles. We had to be wary of related terms that addressed the same question in a wide-ranging way. Within the search terms, it was important to include all synonymous words, singular/plural words, and words with different spellings. Identifying the major terms from the questions and obtaining their synonyms and alternatives is one advantage of using the PICOC model. AND/OR operators were used to link the terms.

It is very important to understand how to design the query and state the search criteria. Before starting to design the query, we had to consider which format we wanted to follow. Each library's database has a specific format when it comes to building their query in order to fetch and reiterate data from its DB engine. In this literature review, we were also concerned about the titles, abstracts and the papers' keywords while constructing our search query.

We designed this query based on the Google Database Search Engine and other DB sources which addressed how to use the search engine for advanced searching. We chose to focus our research on the following database libraries: IEEE Explore, ACM Digital Library, Scopus, Web of Science, Compendex and Inspec. This was due to the fact that they are specialist libraries in the computer science and engineering field. Since the Compendex and Scopus libraries have an overlap of around 100% [32], we chose to combine their results.

Search Scope:

For this literature review, we covered both published and unpublished studies relevant to our study. Based on the research topic and field [29]; [33], this study covered all databases containing high-quality and reputable primary studies related to our topic questions. These databases are listed in Table 1: Search scope database. With regard to unpublished studies, we found some different databases that specialise in technical reports. The most common of these is the Directory of Open-Access Repository; this is a database which gathers literature and studies from thousands of universities across the world. In addition, Google Scholar is also considered one of the most popular search engines with which to find all unpublished studies.

Search Query:

(effort OR efforts OR cost OR size OR performance) AND (estimation OR estimating OR estimate OR prediction OR predicting OR predict OR assess OR assessment OR calculate Or calculation OR measure OR measuring) AND (mobile OR smartphone OR screen touch OR phone OR apple OR android OR IOS) AND (app OR apps OR application OR applications OR software OR program OR programs) AND (agile OR "software

development" OR scrum OR XP OR "extreme programming" OR Crystal OR DSDM OR "dynamic system development method" OR lean)

For each of the database libraries, we followed the criteria and tips regarding how to use their database engines to obtain highly-accurate results for the articles. We produced a schema of database queries and their results. Each of the queries generated different results. We categorised the value and importance of each query into three parts:

- *Very Important Query (Query-ID 2-4,8,11,13,16): this* query contained all or most of the terms/ideas of our research topic. These kinds of research results were usually very close to our topic and were very relevant; as such, we must read all of these papers/articles in more detail. It is usually rare to have a large number of results for this query.
- *Normal Query (Query-ID 1,5,6,9,10,14,15):* this query contained some of the terms/ideas of our research topic. This query was not too close to the topic at hand, but was still relevant. We usually read the abstracts and summaries of these articles and assessed whether or not they were related to our topic. Indeed, we designed this query in order to collect some background knowledge on related topics.
- *Not Important Query (Query-ID 7,12):* We designed this query to pinpoint all topics that were being discussed in our research field/area. This kind of query helped us to expand our search scope and acquire more knowledge regarding what kinds of research had been published in this field (mobile apps software development).

Table 1. Search Scope Database

Database Name/Search Engine	Search Results
IEEE Explore	126
ACM Digital	308
Inspec and Web Science Core Collection	134
Scopus and Compendex	124
OpenDOAR and Google Scholar (for unpublished works)	9

Inclusion and Exclusion Criteria:

Defining inclusion and exclusion criteria helped to clarify the boundaries of the study.
The inclusion criteria:

- We included all studies related to the effort or size estimation models for mobile applications.
- All studies that applied the ASD process.
- All empirical studies (evidence-based studies).
- All peer-reviewed studies that had been reported or presented in workshops, conferences or journals.
- All works that had been presented in one of the highly-reputable databases listed in the above table (Table 1), and most of the trending unpublished databases which were sourced.
- In 2002, the Blackberry company released its first smartphone. The smartphone was very limited and could only handle very small applications, such as calendar and so on [34]. In 2008, touchscreen phones (e.g. iPhone, Android) became the revolution of the century; indeed, these phones are based on the application concept [34]. The Apple App Store launched in July 2008, while the Android market launched a couple of months later in October. As such, we included all studies published since 2008 and ignored all the studies which came before.
- We included all studies that had been presented and published in the English language.

The exclusion criteria:

- We excluded all studies that did not discuss the effort estimation for mobile applications.

- All studies that did not employ ASD methods to make the estimation.
- All the reported works that were not published in one of the listed databases.
- All studies not published in the English language.

3.3. Study Selection Process

Collected data from systematic searches should be documented and organised in an appropriate format. Since we had a large number of results, we exported all the data results (701 papers) to Microsoft Excel so that they were easy to manage. The Excel sheet contained the most valuable information from the papers, such as: 1- Paper name, 2- Publishing year, 3- Author's name, 4- Abstract, 5- Paper link URL, and 6- Query-ID (1-16). After managing and grouping the information into one source, we retrieved the duplicated papers.

To start screening and filtering the papers, it was necessary to split the screening process into three phases:

Screen the Less Important Search Queries:

We first needed to filter out the less important queries, namely Query-ID 7 and 12. These queries were classified as not important, as previously mentioned in the search query section. We had 369 papers that needed to be filtered. Most of the papers were far removed from our research area due to the quick skimming of the abstracts of those papers. The subjects of said papers included: mobile network, GPS, mobile signals and radio, mobile security, mobile energy of phones and consumption, telescope, health field "heart pulses", etc. We excluded all the non-related studies (354 papers) based on the inclusion and exclusion criteria used in this research; we also moved the relevant studies to the second phase of the search category.

Screen the Normal and Highly-important Queries:

This phase included all studies from Query-ID 1-6,8-11,13-16, as well as all studies which had been passed and moved from the previous phase to the second phase. We had 111 unique papers, and 53 duplicated papers (from 236 recorded), as shown in Figure 1. The titles, abstracts and summaries of the papers were read carefully, and the inclusion and exclusion criteria were applied to those papers. As a result, we obtained 35 papers that satisfied the inclusion criteria.

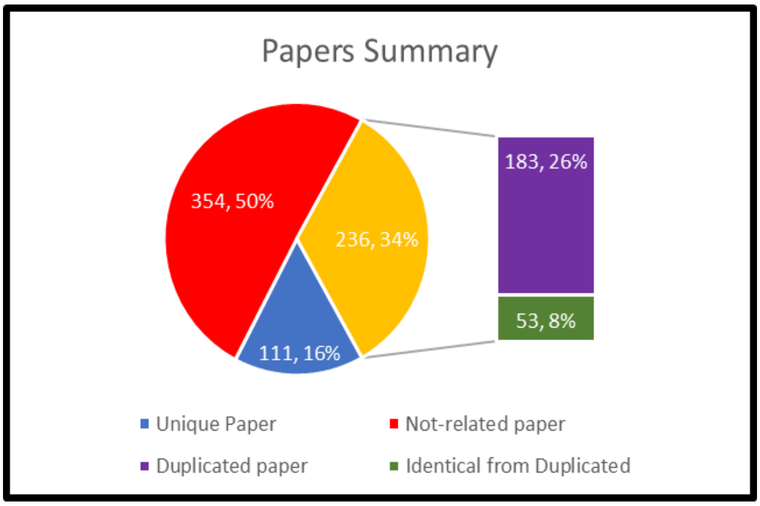

Fig. 1. Studies summary.

Full Text Screening:

During this phase, we applied the inclusion and exclusion criteria to all of the text from the 35 papers. As a result, we obtained 21 selected papers related to the literature review study. Table 2 presents a list of the selected papers. Moreover, the quality assessment criteria were applied to the selected papers to evaluate and establish the quality of the study.

Table 2. Information for Selected Papers

Study ID	Reference	Year	Study ID	Reference	Year	Study ID	Reference	Year
S1	[42]	2018	S2	[43]	2017	S3	[40]	2017
S4	[54]	2017	S5	[55]	2016	S6	[46]	2015
S7	[49]	2015	S8	[45]	2015	S9	[58]	2015
S10	[51]	2015	S11	[44]	2014	S12	[47]	2014
S13	[53]	2014	S14	[50]	2014	S15	[57]	2013
S16	[56]	2013	S17	[48]	2013	S18	[60]	2010
S19	[61]	2008	S20	[63]	2008	S21	[59]	2004

3.4. Study Quality Assessment

With regard to evaluating the quality of the selected studies, the guideline criteria put forth by [29] and [1] made it possible to assess the studies based on certain questionnaires. It was very important to assess the studies, as this helped to establish which of said studies should be focused on, and which were of less importance and in some cases could be ignored. The quality assessment criteria supported and provided more details regarding the inclusion and exclusion criteria. The quality assessment checklist was customised based on the suggestions provided by Kitchenham and Charters [1] , as well as by Usman [14]. Below are some of the questions which were used to evaluate the selected studies:

1. Are the research aims and objectives clearly specified?
2. Is the study designed to achieve these aims?
3. Are the estimation methods used for mobile applications clearly described and their selection justified?
4. Are the variables used in the study adequately measured?
5. Are the data collection methods adequately described?
6. Are the statistical approaches used to analyse data adequately described?
7. Does the researcher discuss any problems or issues with the validity or reliability of their results?
8. Are the measures in the study fully defined?
9. Are the study participants or observational units appropriately defined?
10. Are all of the research's questions answered adequately?

In terms of the answers to these questions, there were three possible options alongside said questions:

- Y: Yes, the answer is explicitly defined in the study = 1 point.
- P: Partially answered in the study = 0.5 points.
- N: No, the answer is not defined = 0 points.

Table 3. Assessment for Selected Studies

Study ID	Score	Study ID	Score	Study ID	Score
S1	8	S2	8.5	S3	7
S4	8	S5	5.5	S6	8

S7	8	S8	8.5	S9	5
S10	4.5	S11	6	S12	5
S13	5	S14	5.5	S15	5
S16	5	S17	4.5	S18	4.5
S19	5	S20	5	S21	5

Each study could obtain a score ranging from 0 to 10 points. Any study that scored below 3 was eliminated from the review. The final scores of the quality assessment for the selected papers can be seen in Table 3 上方. Some papers were excluded due to the fact that they did not satisfy the inclusion criteria (e.g. [35] and [36]), reported a study already included in another published paper (e.g. [37] and [38]), and had a low of quality assessment score (e.g. [39]).

4. Discussion and Interpretation

At this point we can return to the research questions. For example, RQ1 asked: What methods or techniques have been used to estimate effort in mobile application development using Agile methods? The results revealed various software estimation techniques and models that have been used to estimate the effort and size of mobile app projects; however, none of these techniques employed or used any of the Agile methodology processes with the estimation techniques.

We will answer all of the SLR questions in Section 4.1 below, which is entitled "Effort Estimation VS Mobile Application", regardless of which software development process has been used. Following this, all studies which have investigated the ASD with mobile app development will be discussed below in Section 4.2, entitled "ASD Process VS Mobile Applications".

In light of the previous two sections, Table 4 illustrates the distribution of the selected studies and their categories. 13 of 21 studies discussed the estimation techniques in mobile apps, whereas 8 studies pertained to ASD in mobile apps.

Table 4. Selected Study Categories

Related Topic	Study ID
Effort Estimation with Mobile Application	S1, S2, S3, S4, S6, S7, S8, S10, S11, S12, S13, S14, S17
Agile SD with Mobile Application	S5, S9, S15, S16, S18, S19, S20, S21

4.1. Effort Estimation VS Mobile Application

After extracting data from 13 studies concerning software estimation techniques in mobile apps, the extracted data was consolidated into 8 tables to make it readable; this also meant that it was easy to understand the relationship between the data and the research questions. The details of the 13 studies will be explained more rigorously below in parts 1 and 2.

Table 5 answers RQ1 by listing the different estimation techniques that have been used in mobile apps. The Function Size Measurement and Expert Judgment techniques were the most commonly-used across mobile apps. To answer RQ2, Table 6 presents the size metrics and cost drivers which were used by the studies. Number of screens and type of supported platform for the smartphones were the most common factors used to measure the estimation prediction.

Table 7, Table 8 and Table 9 answer RQ3 by providing a list of the Operating Systems (OS) that have supported the smartphones and the types of platforms which have been used to develop mobile applications and the domain of the studies. Table 10 investigates the development activity to which the effort estimate applies. None of the studies estimated the effort involved in the analysis, design and testing activity. For RQ4, Table 11 presents a list of various prediction accuracy metrics which were used in the studies.

Table 5. Estimation Techniques (RQ1)

Estimation Techniques	Study ID
COSMIC FSM	S6, S7, S11, S12, S14, S17
Function Point Analysis	S4a
FiSMA FSM	S13
Expert Judgment	S1, S3, S11
Analogy-based	S3, S11
Regression-based	S2, S6, S8

Table 6. Cost Drivers (RQ2)

Estimation Predictor/Metrics	Study ID
Function Point Size	S4a, S6, S7, S11, S12, S13, S14, S17
UML Diagram	S8a, S8c, S14
Supported Platform Type (IOS/Andr./Win./etc.)	S2, S3, S11
Supported Device (tablet, smartphone)	S2, S3, S11
User Interface Quality and Complexity	S2, S3, S4a, S11, S13
Back-end System Availability and Server Config. Flexibility	S2, S3, S11
Dev. Team Skills	S2, S11
App Development Flexibility and Complexity	S2, S4a, S11
Team Communication Process Complexity and Experience	S2, S11
Push Notification	S1, S2, S3
Landscape and Portrait Mode	S2, S11
Data Storage and Memory Opt. Complexity	S2, S3
Number of Screens	S2, S3, S4a, S4b, S13
Number of API Parties	S2, S3
Support Code Reusability	S2, S4a
Technology Maturity	S2, S11
Battery & Power Optimisation	S2, S13
Connection (Wireless, Bluetooth, 3G, etc.)	S2, S4, S13
Booking and Reservation, Calendar and Time, Map and Localisation, Social Sharing, Searching Contents and Messaging	S1, S3
Deadline Date	S3, S4a
Number of Functionalities/Function Size	S3, S4a, S8a, S8c
Number of Files, Classes, Method, Statement and LOC	S8b, S8d
Registration & Login, Chronological List, Hardware Access, File Upload, Comment Feature, Navigation	S1
Interrupt Handling	S2
Security Analysis Support, Budget for the Project, Compatibility with Previous Version, Multi Languages Support, Media Support, and Paying Process User Feedback	S3

Table 7. Type of Mobile OS (RQ3)

Type of Mobile App	Study ID
Android-based	S6, S7, S8
Apple-based	---
Not Described (General Smartphone)	S1, S2, S3, S4, S11, S12, S13, S14, S17

Table 8. Type of Mobile App Development (RQ3)

Development Type	Study ID
Native	S8
Web App	--
Hybrid	--
Across Platform	S14
Not Described (General)	S1, S2, S3, S4, S6, S7, S11, S12, S13, S17

Table 9. Domain of Dataset Used in the Studies (RQ3)

Domain and Type of Dataset Used	Study ID
Industrial	S1, S2, S3, S4, S6, S7, S11
Academic	S8
Not Defined	S14, S17
No Dataset Used	S12, S13

Table 10. Development Activity

Development Activities	Study ID
Analysis	---
Design	---
Implementation	S6, S7, S8
Testing	---
All	S11
Not Defined	S1, S2, S3, S4, S12, S13, S14, S17

Table 11. Accuracy of Metrics (RQ4)

Estimation Accuracy Technique	Study ID
Mean Magnitude of Relative Error (MMRE)	S2(20%), S4a (67.15%), S4b (61.99%), S6 (11.2%), S7 (33%-46%)
Median Magnitude of Relative Error (MdMRE)	S2 (12%), S6 (7.1%), S7 (21%-46%)
Magnitude of Relative Error (MRE)	S4a (19.4%-239.2%), S4b (0%-309.09%)
Pred	S2 (84%), S4a (11.77%), S4b (17.64%), S6 (87.5%), S7 (31%-77%)
Linear Regression (R2)	S2 (95%), S6 (97.5%), S7 (77%-80%), S8a (23.3%), S8b (20.2%), S8c (42.4%), S8d (70.6%)
Mean Absolute Residuals (MAR)	S8a (15.63), S8b (14.34), S8c (12.15), S8d (9.81)
Median Absolute Residuals (MdAR)	S8a (9.37), S8b (11.55), S8c (8.41), S8d (8.76)
Not Use	S3, S11, S13, S14, S17
Other	S1 (Mean)

Technique using metrics/cost drivers:

This first part contains a review of studies which have used metrics/factors/cost drivers to estimate the effort. The most recent study was conducted by Catolino, Salza, Gravino, and Ferrucci [40]. Their investigation identified a set of metrics (e.g. number of static/dynamic pages, develop new/existing app, etc.) to estimate the development effort of mobile applications in the early stage of the development process. Indeed, it would be beneficial to use these metrics in the model concerning effort estimation on mobile applications. Catolino's methodology was inspired by the work of Mendes [41] in the context of web app development. The study obtained 41 metrics from 377 discovered quotes from different companies that are available online; the metrics were then validated and analysed by four expert project managers. Since this study is new, it has not yet been evaluated; however, with regard to the present study, we plan to evaluate the validity of these metrics as a dependent variable in a predication model. Moreover, Lusky, Powilat, and Böhm [42] proposed an experience-driven approach to estimate the effort by identifying and validating 16 standard features of mobile apps and a degree of customisation level for those features. This approach was based on the Delphi method.

Moreover, Shahwaiz, Malik, and Sabahat [43] proposed a parametric model to estimate the effort needed to create mobile application software. Shahwaiz identified 20 effort predictors (e.g. number of screens, application complexity, memory opt. complexity, etc.), and used an online questionnaire format that resulted in 169 data points; these points were used in the regression-based model. Moreover, Nitze, Schmietendorf, and Dumke [44] proposed a conceptual approach using analogy-based and function point-based techniques to estimate the effort needed for mobile app development. The method proposed certain influential factors that may affect the estimation of cost for a project.

Francese, Gravino, Risi, Scanniello, and Tortora [45] built a formal, regression-based estimation model which addressed 23 Android mobile apps; the purpose of this was to estimate the effort needed for mobile app development as well as the graphical component numbers based on certain information from the mobile project's requirements; this information included number of use cases, number of classes, and number of actors etc. The present study investigated and compared the accuracy of the prediction level – which was obtained from the requirements – between the effort needed for mobile app development and the number of graphical components in a mobile app. As such, we obtained four different types of Stepwise Linear Regression (SWLR) Model that must be studied in this paper:
S8a: SWLR Model built around the Requirement Analysis Document (RAD) variables to predict the effort.
S8b: SWLR Model built around the Source Code (SC) variables to predict the effort.
S8c: SWLR Model built around RAD to predict the number of XMI files and graphical components.
S8d: SWLR Model built around SC to predict the number of XMI files and graphical components.

Techniques using Functional Size Measurement:

This second part focuses on the functional size measurement used to estimate the effort needed for mobile apps. D'Avanzo, Ferrucci, Gravino, and Salza [46] applied the COSMIC functional size measurement method to mobile applications, with a particular focus on Android OS. The present study examined how COSMIC FP can be applied successfully in mobile apps and verifies the utilisation of the functional size to predict the code size (memory size kb) of mobile applications. Our study was empirically validated by extracting the functional user requirement (FUR) from eight mobile applications from the Google Play Store; following this, the data movement (Entry-E, Exit-E, Write-W and Read-R) was obtained from the FUR. As a result, the study revealed that the prediction model is highly accurate. Other studies, such as that conducted by Heeringen and Gorp [47], described guidelines regarding the application of an approximate method using COSMIC to measure the functional size of mobile apps in a rapid manner. In a similar vein, Nitze [48] examined an adapted version of the COSMIC FP approach and assessed its suitability to measure the size of mobile apps from the function user requirements.

In addition, Ferrucci, Gravino, Salza, and Sarro [49] investigated the use of the COSMIC functional size measure to estimate the size (line of code and memory size-byte) of mobile applications. This study included 13 mobile apps from the Google Play Store and applied the same guidance as that applied by D'Avanzo [46] . In relation to the previous invitation, it is fitting to refer to a study by Abdullah, Rusli, and Ibrahim [50], which addressed the complexity of game parameters (requirements and characteristics) in terms of accurately estimating the degree of size and effort needed for mobile game applications. This study adapted COSMIC FSM to estimate the cost and effort incurred when using the Unity3D game engine; the latter is a multi-platform development tool that allows developers to create and design 3D games and applications for mobiles. Moreover, the same authors [51] conducted a review of the functional size measurement (FSM) on a mobile application development using UML modelling methods in terms of the measuring process and rules. The same authors also had a case study that simulated the game Angry Bird mobile app to demonstrate the advantages of uses UML representations in COSMIC FSM calculations [52]. Along similar lines, Souza and Aquino [53] proposed a new method that is adopted in the FiSMA method; however, their study suffered from a lack of empirical evidence and no experts were included when conducting said method. This approach has not been evaluated, nor has the accuracy of the predication method been measured. Souza constructed a systematic review that covered all characteristics of mobile applications and presented the differences between mobile applications and other traditional software.

Moreover, an empirical validation study proposed by Arnuphaptrairong and Suksawasd [54] validated and compared the effort estimation accuracy between a traditional effort estimation model, function point method, (S4a) and a proposed method based on the number of screens of the mobile app (S4b).

4.2. ASD Process VS Mobile Applications

This section includes all studies which have explored the use of the ASD process in mobile applications.

Santos, Kroll, Sales, Fernandes, and Wildt [55] investigated the adoption of Agile methods during the mobile application development process. This study determined certain challenges and reported the experiences of 20 undergraduate students when adopting mobile practice in order to develop mobile applications. The study noted five major challenges faced by the participants; in contrast, the authors identified eight advantages which the participants benefitted from when using the Agile method to develop mobile apps.

Corral, Sillitti, and Succi [56] examined the suitability of the Agile method to meet the needs of the mobile business environment; the authors also discussed the real use of the proposed frameworks that adopted the Agile method during the mobile app development process. The study revealed that there is a need for evidence-based research to link the proposed frameworks with the real project in the industrial field. In other words, their study revealed that there is a need for actual evidence-based research that declares which software development process is suitable for use during mobile app development. Similar to the above-mentioned paper, Corral, Sillitti, and Succi [57] presented a more detailed review of the current Agile-based framework used in mobile app development, and put forth some discussions on, and limitations of, the current methodologies.

Mahmud and Abdullah [58] published a review study that examined and discussed all existing studies which have adopted ASD methods during the mobile application development process. Below, we allude to most of the important studies that have discussed the adoption of Agile methods during the mobile app process.

Mobile-D Approach and an improved Mobile-D approach:

Abrahamsson *et al.* (2004) [59] applied a new development process approach that is suitable for mobile application development. There are certain challenges and constraints which arise when applying the Agile SD process during mobile app development due to the physical and technical characteristics of the mobile environment [59], [60]. The Mobile-D approach is suitable for a small team of developers (10 developers or less) and for the short-term development cycle (10 weeks or less). The new approach has adopted the XP methodology, and is based on the Crystal methodology and Rational Unified Process (RUP). Spataru [60] provided an evaluation of using the Agile methods during the mobile app development process and then proposed a set of improvements for the Mobile-D approach.

Hybrid Method Approach:

Rahimian and Ramsin [61] stated that, while Agile methods are the most appropriate means of mobile SD through the existing methods of software development processes, some particular attributes of mobile devices require specific adjustments so that they match the existing software development methodologies. This study [61] revealed some differences regarding the development of mobile apps when compared with traditional software development in various aspects. Mobile apps should satisfy special requirements and constraints. Their study constructed a new methodology known as the Hybrid Methodology design approach. This framework has been structured as a top-down iterative and incremental process. The framework consists of two major tasks: Prioritisation of the Requirements and Iterative Design Engine. The framework combines ideas from the Adaptive Software Development (ASD) to ensure more quality insurance processes, and New Product Development (NPD).

Other Studies:

There also exist other proposed software development methods for mobile applications, such as the Spiral process put forth by Nosseir, Flood, Harrison, and Ibrahim (Nosseir *et al.*, 2012) [62], which is based on a usability-driven model. This methodology focuses on risk reduction and support iterations to ensure that the requirements have been addressed and validated. Mobile Application Software development based on the Agile Methodology (MASAM) was adopted in another study conducted by Jeong, Lee, and Shin [63], and is closely tied to the Mobile-D approach. MASAM comprises four phases, namely development preparation, embodiment (representing the customer's intention), product development, and commercialisation; SLeSS, MASEF and DSDM are other methodologies that have adapted the Scrum, XP and Iteration concepts respectively. Indeed, all of these used the Mobile-D approach as a reference [58].

The research questions RQ1-RQ4 have been answered in this section, and the details of all studies related to SLR have been described adequately.

5. Research Gap and Future Work

From the previous discussion and results, we have seen various software estimation techniques that have been applied in mobile applications; however, there exist many research gaps and possible further avenues for future works. As can be seen in Fig. 2, there are three intersections, in the middle plot, between Agile SD, effort estimation, and mobile application. These intersections contain many gaps and ideas for possible future works, which will be discussed in the following:

5.1. The Relationship between "Effort Estimation Techniques in Agile SD" AND "Effort Estimation in Mobile App"

- A possible future study could examine the need to employ and validate the existing effort estimation methods using Agile processes in mobile app development.
- Another possible future study could employ the prediction factors and cost drivers that have been used to estimate the effort involved in Agile methods; information on this can be found in Section 2 under background, Software Estimation in Agile SD, during mobile application development.
- One particular research gap calls for the need to study the relationship between the cost drivers and effort predictors from effort estimation in Agile SD models, and the effort estimation in mobile app models.

5.2. Special Attention Must also is Paid to the Relationship Between the "Agile Software Process in Mobile Apps" and "Effort Estimation in Mobile App Development". This could be Achieved by Conducting:

- A study that investigates the newly-adapted Agile development process in mobile software

development, such as Mobile-D, alongside existing estimation models for mobile apps.
Moreover, there exists only one study which has applied a cross-platform (Unity3D) approach to estimate the size of the software application [50]; however, this tool is primarily designed for game development and is not suitable for industrial applications [64]. There is a need to conduct more studies that examine and investigate the effort estimation for other cross-platform mobile applications.

For the native application development, a regression-based technique has been applied to estimate the effort needed to develop a mobile application; however, we must put in place other estimation models to measure the effort estimation using additional estimation techniques, such as COSMIC FSM, or based on expert judgment.

As we can observe from previous studies, there are no estimation techniques for analysing, designing and testing mobile application development.

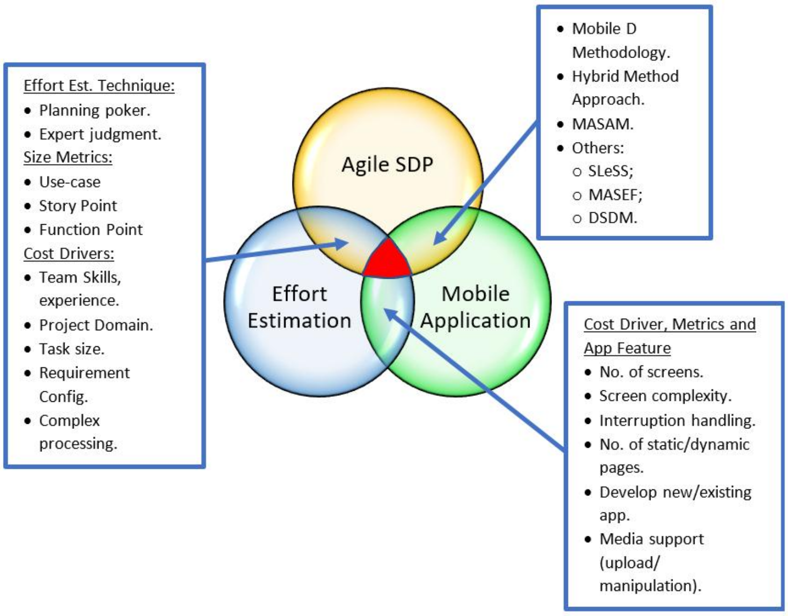

Fig. 2. Research gap.

6. Threat to Validity

In terms of the threat to the SLR validity, the major issue is that we failed to cover and find all the relevant studies. However, we implemented a very deep search strategy to mitigate the threat. We built a good query that contained most of the keywords of relevant works concerning Agile processes, effort estimation and mobile application. Some aspects of the SLR are subjective. To mitigate this effect, the two authors have discussed and agreed on the findings presented here.

7. Conclusion

The study presents a Systematic Literature Review (SLR) regarding effort estimation in mobile applications that use an Agile software development method. The initial search phase returned 701 papers, of which only 21 were selected for the primary study. Various forms of information have been extracted from these selected studies, such as estimation techniques, cost drivers, domain and type of dataset, and estimation accuracy techniques. A couple of possible future directions and works have been proposed at the end of this SLR.

8. Reference

[1] Kitchenham, B., & Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering version 2.3. *Softw. Eng. Group, Sch. Comput. Sci. Math. Keele Univ.*, *45(4)*, p. 1051.

[2] Kumar, N. A., Krishna, K. T. H., & Manjula, R. (2016). Challenges and best practices in mobile application development. *Imp. J. Interdiscip. Res.*, *2(12)*, 1607–1611.

[3] Xanthopoulos,S., & Xinogalos, S. (2013). A comparative analysis of cross-platform development approaches for mobile applications. *Proceedings of the 6th Balk. Conf. Informatics - BCI '13*.

[4] Hammershoj, A., A. Sapuppo, & Tadayoni, R. (2010). Challenges for mobile application development. *Proceedings of the 14th Int. Conf. Intell. Next Gener. Networks (ICIN 2010) Second Int. Work. Bus. Model. Mob. Platforms (BMMP 2010)*, 1–8, 2010.

[5] Wasserman, A. I. (2010). Software engineering issues for mobile application development. *Proceedings of the FSE/SDP Work. Futur. Softw. Eng. Res. ACM, 2010.*, 397–400.

[6] Nasir, M. (2006). A survey of software estimation techniques and project planning practices. *Proceedings of the Seventh ACIS Int.* Conf. *Softw. Eng., Artific. Intell. Netw., Parallel/Distributed Comput. S*, 305–310.

[7] Shekhar, S., & Kumar, U. (2016). Review of various software cost estimation techniques. *Int. J. Comput. Appl.*, *141(11).*

[8] Britto, R., Freitas, V., Mendes, E., & Usman, M. (2014). Effort estimation in global software development: a systematic literature review. *Proceedings of the 2014 IEEE 9th Int. Conf. Glob. Softw. Eng.*

[9] Björndal, P., Smiley, K., & Mohapatra, P. (2010). Global software project management: A case study. *Lect. Notes* Bus*. Inf. Process.*, *54*.

[10] Muhairat, M., Al-Daajeh, S., & Al-Qutaish, R. (2010). The impact of global software development factors on effort estimation methods. *Eur. J. Sci. Res.*, *46(2)*, 221–232, 2010.

[11] Narendra, N. C., Ponnalagu, K., Zhou, N., & Gifford, W. M. (2012). Towards a formal model for optimal task-site allocation and effort estimation in global software development. *Proceedings of the 2012 Annu. SRII Glob. Conf. IEEE*, 470–477.

[12] Peixoto,C. E. L., Audy, J. L. N., & Prikladnicki, R. (2010). Effort estimation in global software development Projects: Preliminary results from a survey. *Proceedings of the 2010 5th IEEE Int. Conf. Glob. Softw. Eng.*, 123–127.

[13] Ramasubbu, N., & Balan, R. K. (2012). Overcoming the challenges in cost estimation for distributed software projects. *Proceedings of the Int. Conf. Softw. Eng.*, 91–101.

[14] Usman, M., Mendes, E., Weidt, F., & Britto, R. (2014). Effort estimation in agile software development: A systematic literature review. *ACM Int. Conf. Proceeding*, 82–91.

[15] Logue, K., McDaid, K., & Greer, D. (2007). Allowing for task uncertainties and dependencies in agile release planning. *Proceedings of the 4th Softw. Meas. Eur. Forum*.

[16] Santana, C., Celio, L. F., Vasconcelos, & Alexandre, G. (2011). Using function points in agile projects. *Lect. Notes Bus. Inf. Process.*, 176–191, 2011.

[17] Mahnič, V., & Hovelja, T. (2012). On using planning poker for estimating user stories. *J. Syst. Softw.*, *85(9)*, 2086–2095.

[18] Haugen, N. C. (2006). An empirical study of using planning poker for user story estimation. *Proceedings of the Agil. 2006 Conf. IEEE*, 23–34.

[19] Parvez, A. W. M. M. (2013). Efficiency factor and risk factor based user case point test effort estimation model compatible with agile software development. *Proceedings of the 2013 Int. Conf. Inf. Technol. Electr. Eng, Intelligent Green Technol. Sustain. D*ev.

[20] Abrahamsson, P., Fronza, I., Moser, R., Vlasenko, J., & Pedrycz, W. (2011). Predicting development effort

from user stories. *Proceedings of the 2011 Int. Symp. Empir. Softw. Eng. Meas.*, 400–403.

[21] Tamrakar, R., & Jørgensen, M. (2012). Does the use of fibonacci numbers in planning poker affect effort estimates? *Proceedings of the 16th Int. Conf. Eval. Assess. Softw. Eng*.

[22] Power, K. (2011). Using silent grouping to size user stories. *Proceedings of the Int. Conf. Agil. Softw. Dev*.

[23] Bilgaiyan, S., Sagnika, S., Mishra, S., & Das, M. (2017). A systematic review on software cost estimation in agile software development. *J. Eng. Sci. Technol. Rev.*, *10(4)*, 51–64.

[24] Jørgensen, M., & Grimstad, S. (2009). Software development effort estimation — Demystifying and improving expert estimation. *Simula Res. Lab.*, 381-403.

[25] Jørgensen, M. (2004). Top-down and bottom-up expert estimation of software development effort. *Inf. Softw. Technol.*, *46(1)*, 3–16.

[26] Grimstad, S., & Jørgensen, M. (2007). Inconsistency of expert judgment-based estimates of software development effort. *J. Syst. Softw.*, *80(11)*, 1770–1777.

[27] Gandomani, T. J., Wei, K. T., & Binhamid, A. K. (2014). A case study research on software cost estimation using experts' estimates, wideband delphi, and planning poker technique. *Int. J. Softw. Eng. its Appl.*, *8(11)*, 173–182.

[28] Jørgensen, M. (2004). A review of studies on expert estimation of software development effort. *J. Syst. Softw.*, *70(1–2)*, 37–60.

[29] Kitchenham, B., Brereton, O. P., Budgen, D., Turner, M., Bailey, J., & Linkman, S. (2009). Systematic literature reviews in software engineering - A systematic literature review. *Inf. Softw. Technol.*, *51(1)*, 7–15.

[30] Shea, B. J. *et al.* (2009). AMSTAR is a reliable and valid measurement tool to assess the methodological quality of systematic reviews," *J. Clin. Epidemiol.*, vol. 62, no. 10, pp. 1013–1020, 2009.

[31] Petticrew, M., & Roberts, H. (2006). *Systematic Reviews in the Social Sciences*, Blackwell publishing.

[32] Elsevier.com. How do Scopus and Compendex differ?

[33] A. (University of S. Siddaway. What is a systematic literature review and how do I do one?, *What is a Syst. Lit. Rev. How do I do one?*, 1–13.

[34] Tracy, K. (2012). Mobile application development experiences on apple's iOS and android OS. *IEEE Potentials*, *31*, 30–34.

[35] Ponnalagu, K., & Narendra, N. (2012). Automated trendline generation for accurate software effort estimation.

[36] Preuss, T. (2013). Mobile applications, function points and cost estimating. *Int. Cost Estim. Anal. Assoc. Conf.*

[37] Francese, R., Gravino, C., Risi, M., Tortora, G., & Scanniello, G. (2016). Estimate method calls in Android apps. *Proceedings of the Int. Work. Mob. Softw. Eng. Syst.*

[38] Souza, L. S. D., & Jr, G. S. D. A. (2014). "Estimating the Effort of Mobile Application Development," *CS IT*-CSCP.

[39] Balasupramanian, N., Lakshminarayanan, R., & Balaji, R. D. (2013). Software engineering framework using agile dynamic system development method for efficient mobile application development. *Int. J. Comput. Sci. Inf. Secur*.

[40] Catolino, G., P. Salza, Gravino, C., & Ferrucci, F. (2017). A set of metrics for the effort estimation of Mobile apps. *Proceedings of the 2017 IEEE/ACM 4th Int. Conf. Mob. Softw. Eng. Syst. MOBILESoft*.

[41] Mendes, E., & Counsell, S. (2004). Investigating early web size measures for web cost estimation. *J. Syst. Softw. 77*, 1–28.

[42] Lusky, M., Powilat, C., & Böhm, S. (2018). Software cost estimation for user-centered mobile app development in large enterprises,.

[43] Shahwaiz, S. A., Malik, A. A., & Sabahat, N. (2016). A parametric effort estimation model for mobile apps. *Proceedings of the 2016 19th Int. Conf. (INMIC)*.

[44] Nitze, A., Schmietendorf, A., & Dumke, R. (2014). An analogy-based effort estimation approach for mobile application development projects. *Proceedings of the 2014 Jt. Conf. Int. Work. Softw. Meas.*.

[45] Francese, R., Gravino, C., Risi, M., Scanniello, G., & Tortora, G. (2015). On the use of requirements measures to predict software project and product measures in the context of Android mobile apps: A preliminary study. *Proceedings of the 41st Euromicro Conf. Softw. Eng. Adv. Appl. SEAA 2015*.

[46] D'Avanzo, L., Ferrucci, F., Gravino, C., & Salza, P. (2015). COSMIC functional measurement of mobile applications and code size estimation. *Proceedings of the 30th Annu. ACM Symp. Appl. Comput*.

[47] Heeringen, H. V., & Gorp, E. V. (2014). Measure the functional size of a mobile app: Using the cosmic functional size measurement method. *Proceedings of the 2014 Jt. Conf. Int. Work. Softw. Meas*.

[48] André, N. (2013). Measuring mobile application size using COSMIC FP. *Softw. Metr. Kongress*.

[49] Ferrucci, F., Gravino, C., Salza, P., & Sarro, F. (2015). Investigating functional and code size measures for mobile applications. *Proceedings of the 41st Euromicro Conf. Softw. Eng. Adv. Appl. SEAA 2015*.

[50]Abdullah, N. A. S., Rusli, N. I. A., & Ibrahim, M. F. (2014). Mobile game size estimation: COSMIC FSM rules, UML mapping model and unity3d game engine. *Proceedings of the 2014 IEEE Conf. Open Syst*.

[51] Abdullah, N. A. S., & Rusli, N. I. A. (2015). Reviews on functional size measurement in mobile application and uml model. *Proceedings of the 5th Int. Conf. Comput. Informatics*.

[52] Abdullah, N. A. S., Rusli, N. I. A., & Ibrahim, M. F. (2013). A case study in COSMIC functional size measurement: Angry bird mobile application. *Proceedings of the 2013 IEEE Conf. Open Syst. ICOS 2013*.

[53] Souza, L. S. D., & Aquino, G. S. D. (2014). Mobile application development : How to estimate the effort? Springer *Int. Publ. Switz.*, 63–72.

[54] Arnuphaptrairong, T., & Suksawasd, W. (2017). An Empirical Validation of Mobile Application Effort Estimation Models," *Proceedings of the Int. MultiConference Eng. Comput. Sci. IMECS 2017*, vol. II, 2017.

[55] Santos, A., Kroll, J., Sales, A., Fernandes, P., & Wildt, D. (2016). Investigating the adoption of agile practices in mobile application development. *Proceedings of the 18th Int. Conf. Enterp. Inf. Syst.*.

[56] Corral, L., Sillitti, A., & Succi, G. (2013). Software development processes for mobile systems: Is agile really taking over the business? *Proceedings of the 1st Int. Work. Eng. Mobile-Enabled Syst*.

[57] Corral, L., Sillitti, A., & Succi, G. (2013). Agile software development processes for mobile systems: Accomplishment, evidence and evolution. *Proceedings of the Mob. Web Inf. Syst*.

[58] Mahmud, D. M., & Abdullah, N. A. S. (2015). Reviews on agile methods in mobile application development process. *Proceedings of the 9th* Malaysian *Softw. Eng. Conf. Dec.*

[59] Abrahamsson, P. *et al.* (2004). Mobile-D: An agile approach for mobile application development. *Proceedings of the Conf. Object-Oriented Program. Syst. Lang. Appl. OOPSLA*.

[60] Spataru, A. C. (2010). Agile development methods for mobile applications. *Univ. Edinburgh, Edinburgh*, 1–68.

[61] Rahimian, V., & Ramsin, R. (2008). Designing an agile methodology for mobile software development: A hybrid method engineering approach. *Second Int. Conf. Res. Challenges Inf. Sci.*, 337–342.

[62] Nosseir, A., Flood, D., Harrison, R., & Ibrahim, O. (2012). Mobile development process spiral. 281–286.

[63] Jeong, Y. J., Lee, J. H., & Shin, G. S. (2008). Development process of mobile application SW based on agile methodology. *Int. Conf. Adv. Commun. Technol*.

[64] Ottka, S. (2015). Comparison of mobile application de- velopment tools for multi-platform in- dustrial applications. *Aalto Univ. Sch. Sci*.

Abdullah Altaleb is a PhD student specialising in Software Engineering at the University of Southampton, UK. The author accomplished his master's degree in software engineering at Southern Methodist University in Dallas, Texas in the United States. The Author worked at the GOSI and NIC organisation as a software and application developer for four years, with a focus on the Java EE platform. He also completed an internship at MRN Inc. as a software architect in San Jose, California. Abdullah is certified in CABA, which is associated with the business analysis profession.

Andrew Gravell was born 1956 in Exeter, UK. He studied mathematics and computer science at Cambridge, and was awarded a PhD by the University of Southampton in 1995. He held the position of software developer for 7 years, most of which was spent at IBM's Hursley laboratory working on a range of projects and writing software for microprocessor-based systems. In 1987 he joined ECS, Southampton, as a lecturer in computer science. He was promoted to senior lecturer/associate professor in 2002. Andrew also held the position of Associate Dean for Education and the Student Experience, Faculty of Physical and Applied Sciences from 2010 to 2015. He has supervised a number of PhD students and published on a range of approaches to software development, empirical approaches to software development, agile methods, and lean information technology.