TOWARDS DECISION-MAKING TO CHOOSE AMONG DIFFERENT COMPONENT ORIGINS

Deepika Badampudi

Blekinge Institute of Technology
Licentiate Dissertation Series No. 2016:01

Department of Software Engineering

Towards Decision-Making to Choose Among Different Component Origins

Deepika Badampudi

Blekinge Institute of Technology Licentiate Dissertation Series
No 2016:01

Towards Decision-Making to Choose Among Different Component Origins

Deepika Badampudi

Licentiate Dissertation in
Software Engineering

Department of Software Engineering
Blekinge Institute of Technology
SWEDEN

Department of Software Engineering
Publisher: Blekinge Institute of Technology,
SE-371 79 Karlskrona, Sweden
Printed by Lenanders Grafiska, Kalmar, 2016
ISBN: 978-91-7295-323-9
ISSN 1650-2140
urn:nbn:se:bth-11653

ABSTRACT

Context: The amount of software in solutions provided in various domains is continuously growing. These solutions are a mix of hardware and software solutions, often referred to as software-intensive systems. Companies seek to improve the software development process to avoid delays or cost overruns related to the software development.

Objective: The overall goal of this thesis is to improve the software development/building process to provide timely, high quality and cost efficient solutions. The objective is to select the origin of the components (in-house, outsource, components off-the-shelf (COTS) or open source software (OSS)) that facilitates the improvement. The system can be built of components from one origin or a combination of two or more (or even all) origins. Selecting a proper origin for a component is important to get the most out of a component and to optimize the development.

Method: It is necessary to investigate the component origins to make decisions to select among different origins. We conducted a case study to explore the existing challenges in software development. The next step was to identify factors that influence the choice to select among different component origins through a systematic literature review using a snowballing (SB) strategy and a database (DB) search. Furthermore, a Bayesian synthesis process is proposed to integrate the evidence from literature into practice.

Results: The results of this thesis indicate that the context of software-intensive systems such as domain regulations hinder the software development improvement. In addition to in-house development, alternative component origins (outsourcing, COTS, and OSS) are being used for software development. Several factors such as time, cost and license implications influence the selection of component origins. Solutions have been proposed to support the decision-making. However, these solutions consider only a subset of factors identified in the literature.

Conclusions: Each component origin has some advantages and disadvantages. Depending on the scenario, one component origin is more suitable than the others. It is important to investigate the different scenarios and suitability of the component origins, which is recognized as future work of this thesis. In addition, the future work is aimed at providing models to support the decision-making process.

Keywords: *Component-based software development, component origin, decision-making, snowballing, database search, Bayesian synthesis.*

Papers included in the thesis:

Chapter 2: Deepika Badampudi, Samuel A. Fricker, and Ana M. Moreno. "Perspectives on Productivity and Delays in Large-Scale Agile Projects." In: Proceedings of the 14th International Conference on Agile Processes in Software Engineering and Extreme Programming (XP 2013). Springer Berlin Heidelberg, 2013, pp. 180–194. ISBN: 978-3-642-38314-4.

Chapter 3: Deepika Badampudi, Claes Wohlin and Kai Petersen. "Software Component Decision-making: In-house, OSS, COTS or Outsourcing - A Systematic Literature Review." Under revision after review for Journal of Systems and Software (2016).

Chapter 4: Deepika Badampudi, Claes Wohlin, and Kai Petersen. "Experiences from Using Snowballing and Database Searches in Systematic Literature Studies." In: Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering (EASE 2015). ACM, 2015, 17:1–17:10. ISBN: 978-1-4503-3350-4.

Chapter 5: Deepika Badampudi and Claes Wohlin. "Bayesian Synthesis in Software Engineering: Method and Illustration." Submitted to International Conference on Evaluation and Assessment in Software Engineering (EASE 2016).

Contribution statement

Deepika Badampudi is the lead author of all the chapters. As the lead author she took the responsibilities in designing, executing and reporting of the studies. The contributions to individuals chapters are as below:

Chapter 2: Deepika Badampudi designed the case study, participated in data collection by conducting the interviews, transcribed and analyzed all the interviews and reported the findings. The co-authors reviewed the case study design and commented on the final draft of the conference paper.

Chapter 3: Claes Wohlin contributed with the idea to conduct a systematic literature review. The review consists of two search strategies, Deepika Badampudi designed the review protocol and Claes Wohlin contributed to the design of the snowballing search strategy. Kai Petersen contributed in the database search strategy. Deepika Badampudi extracted the data from 18 (of 24) studies and wrote large parts of the final draft. Claes Wohlin and Kai Petersen reviewed and commented on the final draft of the journal paper.

Chapter 4: Claes Wohlin contributed with the idea to report experiences from conducting the snowballing search strategy. Deepika Badampudi contributed to the design of the study and reported the findings. Claes Wohlin and Kai Petersen commented and reviewed the final draft of the conference paper.

Chapter 5: Deepika Badampudi mainly contributed with the idea of the study. Claes Wohlin participated and contributed in all the brainstorming sessions. Deepika Badampudi reported the study and Claes Wohlin commented and reviewed the final draft of the conference paper.

Papers related but not included in the thesis:

Paper 1: Kai Petersen, Deepika Badampudi, Syed Muhammad Ali Shah, Krzysztof Wnuk, Tony Gorschek, Efi Papatheocharous, and Axelsson Jakob. "Choosing among Architectural Asset Origins: A Case Survey." To be submitted as a journal paper (2016).

Paper 2: Claes Wohlin, Darja Smite, Ulrik Franke, Krzysztof Wnuk, Deepika Badampudi, and Cicchetti Antonio. "A Decision-making Framework for Selecting Software Assets." To be submitted as a conference paper (2016).

ACKNOWLEDGMENTS

Firstly I would like to sincerely thank my advisor Professor Claes Wohlin for his continuous guidance, support, immense knowledge and motivation throughout this thesis. I'm grateful for his timely feedback, constructive criticism and his willingness to listen and consider my ideas.

I would also like to thank my co-advisor Professor Tony Gorschek for his inputs. The support and insightful comments from my co-authors Kai Petersen, Samuel A. Fricker and Ana M. Moreno are much appreciated. I thank my colleagues for providing a supportive work environment. Also, I thank Farnaz, Indira and Siva for the fun times and most importantly their friendship.

Words cannot express my true gratitude and love towards my husband, Gurudutt Velpula, who has been supporting and helping me to remain positive even through difficult times.

I thank my parents, Kishore and Indira for their love and care, my sister, Deepthi for being my best friend and my niece Kanisha for all the happy times.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

CBSE Components-based software engineering

COTS components off-the-shelf

OSS open source software

SB snowballing

FSB forward Snowballing

BSB backward Snowballing

DB database

1 INTRODUCTION

1 OVERVIEW

The use of software is becoming more and more common in solutions provided by different domains such as automotive, automation, health and telecommunication. Systems that consists of large amounts of software that provides value to its users are referred to as software-intensive systems. Companies are constantly seeking to improve the software development/building process to gain profit and competitive advantage. The improvements are usually aimed to provide timely, high quality and cost efficient solutions.

Components-based software engineering (CBSE) is known to increase productivity, save costs, increase quality and reusability [105]. CBSE promotes the development for reuse and building from existing software components [105]. Different alternatives to in-house development such as outsourcing [59], adopting COTS [11] and OSS [39] components are gaining popularity. The different alternatives are referred to as component origins.

Like any other technology or process, no component origin is universally good or universally bad, some component origins are more appropriate than others based on the context of the projects and the organizations. Therefore, it is important to understand how the component origins are traded-off, i.e. what criteria are used to select the component origins in which context using what process, techniques or tools. It is also important to understand the challenges of developing software-intensive systems. Knowing the limitations or shortcomings of in-house development for software-intensive systems will help the decision-makers to evaluate if the existing challenges of in-house development can be mitigated by considering the alternative component origins.

The overall research goal is to provide a decision-support system for selecting component origins. Figure 1 depicts the focus of this

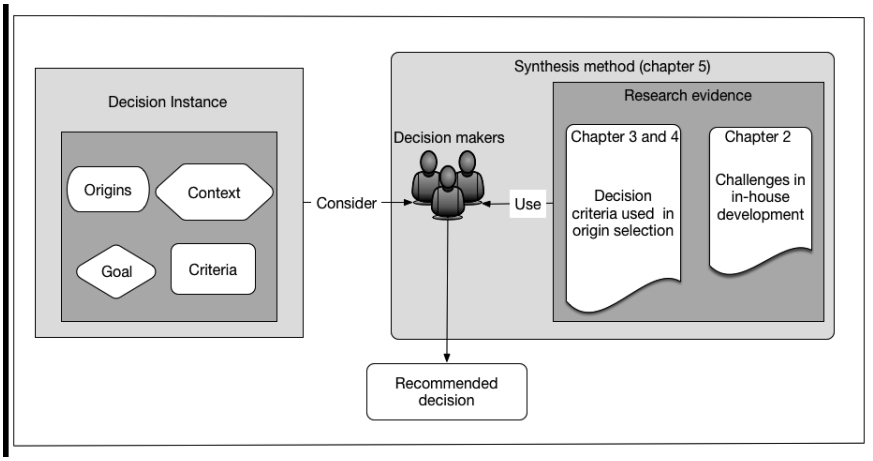

Figure 1: Overview of the thesis

thesis to explore the challenges in in-house development and the trade-off between the different component origins.

We begin by exploring the in-house development environment through a case study [115]. Five software-intensive large-scale agile projects from different domains are investigated. The investigation allowed to identify challenges that lead to productivity issues and delays in software-intensive systems (Chapter 2).

After understanding the limitations of in-house development, we investigate the alternatives to in-house development. The objective is to explore how the decision to choose the component origin is made. To understand the existing research on decision-making to choose the component origin, we conducted a systematic literature review (Chapter 3).

The main findings of the review are a list of criteria used to choose a component origin. The review also allowed us to investigate the existing solutions to address the decision-making process for component origin selection. The systematic literature review was conducted using DB search and SB, we reported our experiences on the reliability and efficiency of the SB method in comparison to the DB search (Chapter 4).

Through the systematic literature review, we identified a number of important criteria that might influence the decision to choose a component origin. The next step is to integrate the data collected

from literature with the practitioners' opinions so that they can use the information in their decisions. We wanted a systematic approach to integrate the data collected from literature and the practitioners' opinions. This lead us to design a synthesis approach that integrated the literature results and practitioners opinion (Chapter 5).

2 BACKGROUND

This section provides background information of the concepts that are mentioned in the chapters.

Component-based software development: Component-based software development is defined as follows: "The primary role of component-based software engineering is to address the development of systems as an assembly of parts (components), the development of parts as reusable entities, and the maintenance and upgrading of systems by customizing and replacing such parts" [25]. The different parts (components) can be either developed in-house, outsourced or obtained as external components (COTS or OSS).

Component Origin: The different options to get components from are called component origins. The description of each development option is as follows:

- *In-house:* This option is the most straightforward. It involves the development of the software component within the company. It also includes offshoring where the software component is developed within the same company however, in a different location.

- *Outsourcing:* The development of the software component is outsourced to another company often called supplier or vendor. The requirements for the development are provided by the company outsourcing the development.

- *OSS* [44, 65]: It refers to "open source". The OSS components are available for free and the component is provided along with the source code. These components are pre-built and the requirements for building these components are derived from several factors such as developer's interest or sometimes even the market trend.

- *COTS* [11, 74]: It refers to "components off-the-shelf". These components like the OSS components are pre-built however, they are not free and often source code is not provided along with the components. The COTS components are mainly driven by the market trend.

Decision to choose component origin [113]: In this thesis the decision-making is on the strategic level, i.e. to choose a component origin. Such decisions are often derived by some goal such as: improving the software development process or return of investment. The decisions are based on criteria such as quality, time and cost. The decision-making process requires different models to estimate the criteria values such as the cost to buy/build the component, or the time it takes to build, integrate and test the component. The decision-making process involves decision-makers from different roles, these roles might have different expertise in terms background and responsibility and different contributions such as decision initiator, supporter and decider.

3 CONTRIBUTION AND RESEARCH GAPS

The contributions of this thesis are as mentioned in the below sections.

3.1 *Chapter 2*

One possible way to produce more with high quality in less time and cost is to use agile methodologies [3]. Studies have reported positive relationship between productivity and the use of agile methods [14]. One of the agile principles promotes continuous delivery of valuable software [3]. However, when the software has hardware dependencies, it is not always feasible to delivery working software frequently.

Research gap 1: There is no study conducted to better understand the challenges in in-house development for software-intensive systems. Hence, we conducted a case study to address the identified gap.

This leads us to the objective as follows:

Objective 1: To explore context, practices, challenges and impacts of software-intensive development projects.

3.2 *Chapter 3*

The context of software-intensive systems imposes limitations in software development improvements. However, utilizing alternative component origins such as external components might contribute in the improvement. For example, the pressure to develop faster due to market competitiveness is identified as a challenge in in-house development (Chapter 2). COTS and OSS are known to be viable options when time to market is a criterion [65]. The constant requirement changes might result in a lot of wasted development effort (Chapter 2), which can be minimized when the right external components are used in particular OSS components as no additional purchasing costs are involved [18, 65, 94]. Understanding complex requirements can be challenging and time consuming (Chapter 2). The developers prefer to use pre-built components such as OSS libraries when the task complexity is high to improve productivity [104]. Using pre-built COTS and OSS components may improve productivity by reducing development effort [65]. Although, using COTS is beneficial when time to market is critical, the time to test and integrate COTS components can be greater than the in-house development effort [40]. This indicates that one option is not always best for all possible scenarios. It is important to know how decisions are made and what criteria are used to select the component origins.

Research gap 2: Primary studies on selection between the component origins have been conducted, however this evidence has not been aggregated and interpreted so far.

Hence, this leads us to our next objective, which is:

Objective 2: To identify factors that could influence the decision to choose among different component origins and solutions for decision-making in the literature.

3.3 *Chapter 4*

We used SB as the main search strategy to conduct the systematic literature review in Chapter 3.

Research gap 3: In software engineering there are fewer studies that use SB as compared to a DB search strategy for searching for primary studies. The potential of SB in terms of its efficiency and reliability in finding the primary studies is not fully understood.

Hence, this leads us to our next objective, which is:

Objective 3: To find efficient and reliable search strategies.

3.4 *Chapter 5*

After aggregating and interpreting evidence from the literature (Chapter 3), we had the results of how the decisions for component selection was made. The next step is to make the evidence available to practitioners in a way that the allows practitioners to integrate the evidence from literature into their decisions.

Research gap 4: None of the synthesis methods in software engineering research is primarily designed to transfer evidence from literature into practice by taking the experience of practitioners into account.

Therefore, our next objective is –

Objective 4: To improve the methodological support for research problems to integrate evidence from literature and practitioners knowledge.

3.5 *Research questions*

Research questions help us address the research objectives and gaps. The research questions in this thesis are formulated based on the objectives and the research gaps. Four main research questions are formulated which are further divided into sub-research questions. The research questions addressed in the thesis and their mapping to the objectives and research gaps are as shown in Figure 2. Note that the diagram shows the key research questions only, the further sub-questions are mentioned in the individual chapters.

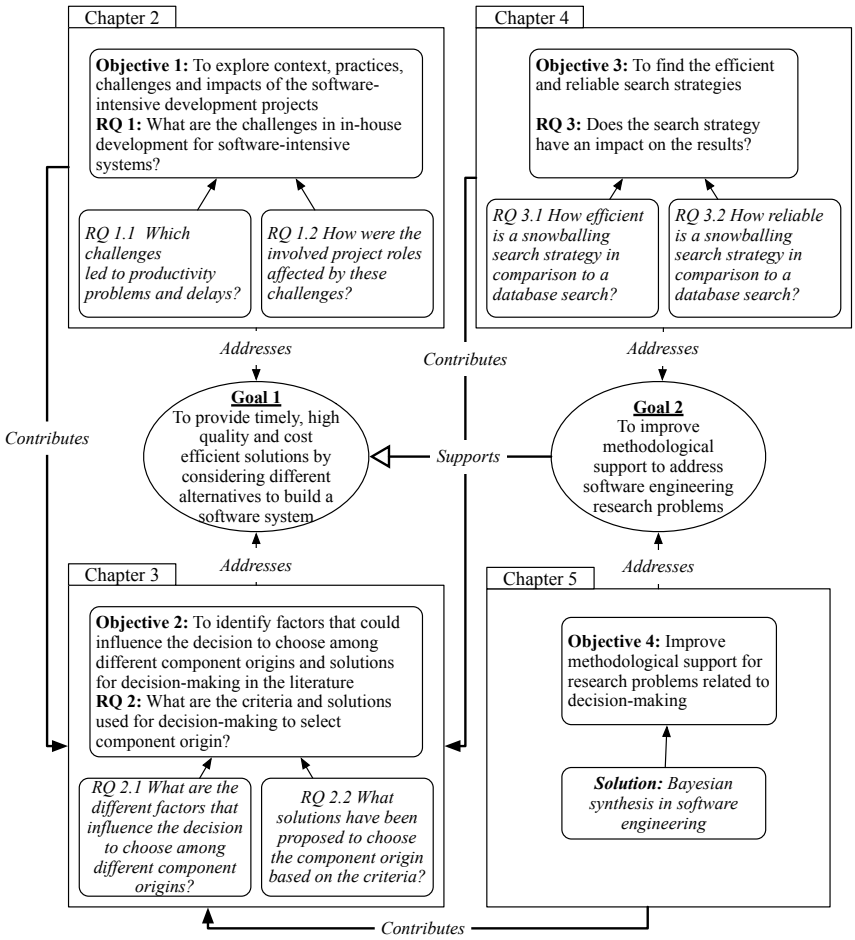

Figure 2: Mapping of research questions and objectives

The goals and chapters of the thesis are represented by the circles and blocks respectively in Figure 2. The blocks within each chapter represent the objective and research questions for each chapter, whereas the arrows within the blocks show the connection between the sub-research questions that answer the main research question and the objective. The arrows outside the blocks represent the connection between the Chapters and goals, the connections are labelled accordingly.

The results from Chapters 2 and 3 address goal 1. In addition, Chapter 2 contributes to Chapter 3. The results from Chapter 2 were discussed along with the results of Chapter 3.

Goal 2 is formulated to support goal 1 and is addressed by Chapters 4 and 5. As mentioned earlier the reflections on the methodology used in Chapter 3 are reported in Chapter 4. Chapter 5, contributes to Chapter 3 by providing methodological support to transfer the results from Chapter 3 to practitioners. Overall goal 2 provides methodological support required to address goal 1.

4 RESEARCH METHODOLOGY

Research methods help answering research questions in a systematic and repeatable way. The rigor of research methods not only allow a thorough investigation of a phenomenon but also allow the users/readers to trust and rely on the results. The research methods used in this thesis are: Systematic literature review and case study.

A brief summary of the research methods and their application to the chapters is provided in the following sections.

4.1 *Case study (Chapter 2)*

Case study is regarded as a suitable research method for software engineering research as it is hard to study a phenomenon in isolation. A case study allows to investigate the phenomenon in its real-life context. The steps involved in case study research are listed and summarized by Runeson and Höst [90] as follows:

1. Case study design: The objectives of the study are defined and the case study is planned based on the objective and research questions. The plan includes details about the case to be studied, the data collection method to be used, and the selection strategy to be used.

2. Preparation for data collection: The procedures and protocol for data collection are defined in this step. The protocol includes details such as: what questions should be asked?

3. Collecting evidence: The data collection is performed to collect evidence.

4. Analysis of collected data: The collected data is analyzed using different analysis methods that suits the collected data.

5. Reporting: The results of the analyzed data are reported. The report should include an elaborate description of the research work and details of the conclusions such as the context they affect.

Case study is used in Chapter 2 to identify the challenges in in-house development of software-intensive systems. Five software-intensive systems were investigated. Fourteen practitioners with different roles involved in the software development were interviewed. Semi-structured interviews were conducted using open-ended questionnaire to collect the data. In addition, the company's standard process documents were reviewed and the first author participated in the project meetings which allowed to triangulate the collected data. The interviews were analyzed using grounded theory. We used the steps in grounded theory to structure and organize the interview data. The data collection and analysis were conducted in parallel. Which means that after conducting one interview, the interview was transcribed and coded. Based on which the follow-up questions for the next interviews was identified.

4.2 *Systematic literature review (Chapters 3 and 4)*

Systematic literature reviews are used to aggregate and interpret the evidence through a scientific and repeatable process. It mainly includes the following steps:

1. Study identification. This step includes searching and selecting the primary studies. Mainly two search strategies are used in software engineering: DB search and SB. The primary studies are selected from the search results based on defined inclusion/exclusion criteria.

2. Data extraction. In this step the data from primary studies is extracted in an explicit and consistent way. Data extraction

forms are used to extract data. The design of the extraction forms is driven by the research questions.

3. Quality assessment. The quality of the primary studies is assessed in this step. The assessment is based on rigor and relevance of the primary studies.

4. Analysis. The data from the primary studies is analyzed in this step. Analysis methods such as thematic analysis or narrative analysis are commonly used to analyze data from qualitative studies. Whereas meta-analysis is commonly used to analyze data from quantitative studies.

Chapter 3: In chapter 3, a systematic literature review was conducted to identify factors that could influence the decision to choose among different component origins and solutions for decision-making in the literature. We used SB search strategy using the guidelines in [111] to search for primary studies and DB search using the guidelines in [56] was used to validate and ensure the completeness of the search process. A data extraction form was used to extract data. The quality of the primary studies was evaluated using the rigor and relevance criteria defined in the guidelines proposed in [47]. Thematic analysis using the guideline in [26] was used to analyze the results of the primary studies.

Chapter 4: In Chapter 4, reflections of conducting a systematic literature review using two different search strategies: SB and DB search are reported.

5 OVERVIEW OF THE CHAPTERS

In this section, an overview of the chapters is provided. The overview includes the objective of the chapter, the methods used to achieve the objective and a description of main findings.

5.1 *Chapter 2: Perspectives on productivity and delays in large-scale agile projects*

The software development process is constantly evolving; methods, process and models supporting software development are used to

improve the software development process. Agile methodologies are such examples that are used to improve the software development process. However, the agile methodologies are implemented in projects contexts that are not consistent with agile principles. The objective of this chapter is to explore context, practices, challenges and impacts of software-intensive development projects.

An exploratory multi-case study is conducted to identify the challenges and impacts of developing software-intensive systems. The employee experiences were collected to identify areas of improvement in the software development practices. The description of the research method used in this chapter is briefly described in Section 4.1 and detailed description along with the validity threats considered is provided in Chapter 2.

The main findings of this chapter are challenges that affected software development, causes of the challenges and the impact of the challenges on different roles. The identified challenges are related to requirements (creation, understandability, selection, estimation and stability), time-to-market, testing (completeness and infrastructure), collaboration (communication, decision-making, team dynamics and team stability), domain knowledge and product repository. The causes of these challenges are due to project complexity, product characteristics, distributed teams and domain knowledge limitations. The challenges had an impact on planning, shared understanding, coordination, capacity and software quality assurance.

5.2 *Chapter 3: Software component decision-making: in-house, OSS, COTS or outsourcing - A systematic literature review*

Four widely used component origins are COTS, OSS, outsource and in-house development. Decision-makers make decisions on choosing a component origin for developing a component/s. The objective of this chapter is to present results from the literature interpreting the factors that influence such decisions and the existing solutions supporting the decisions.

A systematic literature review is conducted to identify primary studies. A total of twenty four primary studies were identified. The details of the review protocol are reported briefly in Section 4.2 and in detail in Chapter 2.

Eleven factors that have an influence on the decision to choose among different component origins were identified. The factors are: Time, cost, effort, market trend, source code availability, technical support, license, integration, requirement, maintenance and quality. Most of the primary studies considered two component origins in the decision. The decision between in-house vs. COTS and COTS vs. OSS were the most researched decisions. The solution models proposed in the literature are based on optimization techniques. The criteria considered in the solutions models are time, cost and reliability.

5.3 *Chapter 4: Experiences from using snowballing and database searches in systematic literature studies*

Systematic literature reviews are ways to aggregate and interpret findings from primary studies. DB search is commonly used for searching the primary studies in software engineering. It is recommended to use SB after DB search to find additional papers. There are few studies that use SB as the main or only method. The effort required in terms of the number of papers to be reviewed to find the primary papers (efficiency) and the capability to find all relevant papers (reliability) might raise some concerns. Therefore, the objective of this chapter is to find efficient and reliable search strategies. A brief summary of the results are provided below and the detailed description is reported in Chapter 4.

The main findings of this chapter are that DB search and SB search are comparable. More papers were reviewed using SB in comparison to the DB search. However, most papers were either duplicates, gray literature or non-English papers which were easy to exclude based on the title. In DB search, such entries can be automatically removed using DB search engines. The total number of abstracts reviewed using DB search and SB were approximately the same. The SB strategy identified 83% of the papers, while DB search identified 46% of the papers and 29% of the papers were commonly found by both DB search and SB. Hence, it can be concluded that SB strategy was efficient and reliable for searching the primary studies to achieve the objective in Chapter 3.

5.4 *Chapter 5: Bayesian synthesis in software engineering: Method and illustration*

Often the aim of software engineering is to identify or produce best practices, processes, tools and methods. It is important that the best practices are made available to practitioners. Evidence-based software engineering practice encourages integrating practitioner opinions with evidence from literature. This is done to use the evidence from literature and to adapt it to the practitioners' context using the practitioners' and/or researchers' knowledge and experience. We propose the use of Bayesian synthesis to systematize the integration of evidence from literature and practitioners' subjective opinion. This contributes towards decision-making, as most decisions are based on subjective opinions.

Bayesian synthesis can be summarized in three steps:

- Prior probability: Prior probability is formulated by collecting subjective opinions from practitioners and/or researchers.
- Likelihood: The likelihood is a summary of evidence from literature.
- Posterior probability: Posterior probability is formulated when prior probability and likelihood is combined.

The detailed process to conduct Bayesian synthesis is reported in Chapter 5 along with examples from software engineering research.

6 CONCLUSIONS AND FUTURE WORK

The overall goal of this thesis is to contribute to the improvement of the software development process to provide timely, high quality and cost efficient solutions. To address this goal we consider the use of different component origins. The methods used in this thesis are case study and systematic literature review. The case study research method is used to identify the challenges of developing software-intensive systems (Chapter 2). The criteria and solutions used to choose between different component origins are identified through a systematic literature review (Chapter 3) and the experiences of conducting the systematic literature review are reported in Chapter

4. A method to integrate the results from literature into practice is proposed in Chapter 5.

The results from Chapter 2 indicate that the context of software-intensive might hinder software development improvement. Some of the hindrances identified were regulations imposed by the domain. In Chapter 3 we investigated alternative component origins (COTS, OSS and outsource) to build software-intensive systems. The results indicate that the decision is not only between internal (in-house) and external component origins (COTS, OSS and outsource), the decision between the external components (e.g. COTS vs. OSS) is also considered in the primary studies. However, it might be the case that the decision between internal and external was taken first before the decision between the external components (e.g. COTS vs. OSS) is considered.

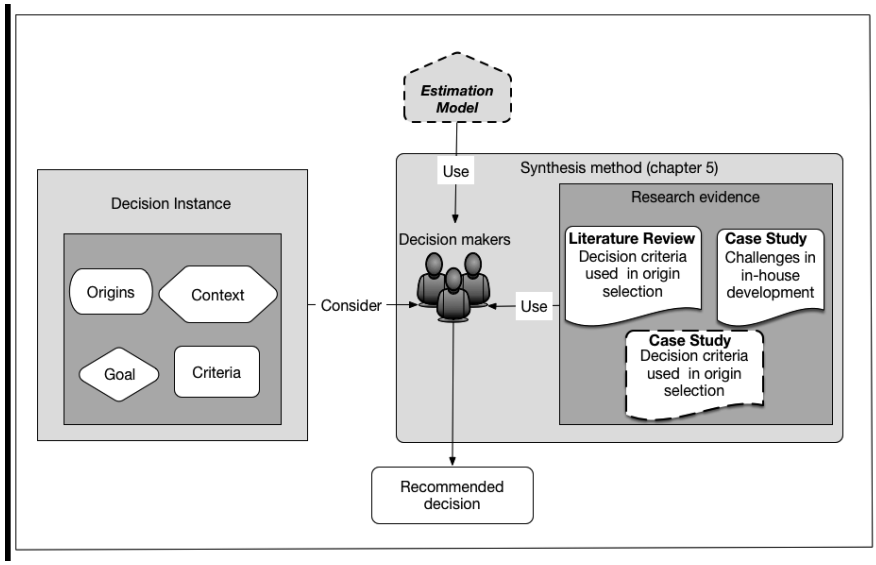

Figure 3: Thesis future work

It is important to note that the use of alternative component origins such as, outsourcing, COTS and OSS are not the complete solutions to mitigate all the challenges of in-house development, rather some component origins are more beneficial than others in certain scenarios. A deeper investigation of the context or scenarios was not possible as the primary studies did not report sufficient details of

the context. In addition, it was not possible to explore the perspectives of all the stakeholders involved in the decision. None of the primary studies focused on the perspective of different stakeholders involved in the decision. To address these limitations a case study is planned to investigate the decision-making process in its real-life context (software-intensive systems). The results from the case study, along with the evidence from the literature will be integrated using the Bayesian synthesis model proposed in Chapter 5. The future work based on this thesis is depicted in Figure 3. The boxes with dotted lines represent future work.

Proposing a solution to support decision-making is also part of future work. The solution will focus on criteria and estimation models to estimate outcomes of the criteria.

REFERENCES

[1] Pekka Abrahamsson, Outi Salo, Jussi Ronkainen, and Juhani Warsta. "Agile software development methods: Review and analysis." In: *VTT Publications* 478 (2002).

[2] Silvia Teresita Acuña, John W Castro, Oscar Dieste, and Natalia Juristo. "A systematic mapping study on the open source software development process." In: *Evaluation & Assessment in Software Engineering (EASE 2012), 16th International Conference on*. IET. 2012, pp. 42–46.

[3] Agile Alliance. "Agile manifesto." In: *Online at http://www.agilemanifesto. org* 6.6.1 (2001).

[4] Nguyen Duc Anh, Daniela S Cruzes, Reidar Conradi, Martin Höst, Xavier Franch, and Claudia Ayala. "Collaborative resolution of requirements mismatches when adopting open source components." In: *Requirements Engineering: Foundation for Software Quality*. Springer, 2012, pp. 77–93.

[5] Claudia Ayala and Xavier Franch. "A goal-oriented strategy for supporting commercial off-the-shelf components selection." In: *Reuse of Off-the-Shelf Components*. Springer, 2006, pp. 1–15.

[6] Victor R Basili, Lionel C Briand, and Walcélio L Melo. "How reuse influences productivity in object-oriented systems." In: *Communications of the ACM* 39.10 (1996), pp. 104–116.

[7] Joseph D Blackburn, Gary D Scudder, and Luk N Van Wassenhove. "Improving speed and productivity of software development: a global survey of software developers." In: *Software Engineering, IEEE Transactions on* 22.12 (1996), pp. 875–885.

[8] Barry Boehm and Richard Turner. "Management challenges to implementing agile processes in traditional development organizations." In: *Software, ieee* 22.5 (2005), pp. 30–39.

[9] Hongyu Pei Breivold, Muhammad Aufeef Chauhan, and Muhammad Ali Babar. "A systematic review of studies of open source software evolution." In: *Software Engineering Conference (APSEC), 2010 17th Asia Pacific*. IEEE. 2010, pp. 356–365.

[10] Lionel Briand, Khaled El Eman, and Walcélio L Melo. "An Inductive Method for Software Process Improvement: Concrete Steps and Guidelines." In: (1995).

[11] Lisa Brownsword, Tricia Oberndorf, and Carol A Sledge. "Developing new processes for COTS-based systems." In: *IEEE Software* 17.4 (2000), pp. 48–55.

[12] Tilmann Bruckhaus, Nazim H Madhavji, Ingrid Janssen, and John Henshaw. "The impact of tools on software productivity." In: *IEEE Software* 13.5 (1996), p. 29.

[13] James Westland Cain and Rachel Jane McCrindle. "An investigation into the effects of code coupling on team dynamics and productivity." In: *Computer Software and Applications Conference, 2002. COMPSAC 2002. Proceedings. 26th Annual International*. IEEE. 2002, pp. 907–913.

[14] E Cardozo, JBFA Neto, Alexandre Barza, A França, and F da Silva. "SCRUM and productivity in software projects: a systematic literature review." In: *14th International Conference on Evaluation and Assessment in Software Engineering (EASE)*. 2010.

[15] E Cardozo, JBFA Neto, Alexandre Barza, A França, and F da Silva. "SCRUM and productivity in software projects: a systematic literature review." In: *14th International Conference on Evaluation and Assessment in Software Engineering (EASE)*. 2010.

[16] David J. Carney and Fred Long. "What Do You Mean by COTS? Finally, a Useful Answer." In: *IEEE Software* 17.2 (2000), pp. 83–86.

[17] Lianping Chen, Muhammad Ali Babar, and Nour Ali. "Variability management in software product lines: a systematic review." In: *Proceedings of the 13th International Software Prod-*

uct Line Conference. Carnegie Mellon University. 2009, pp. 81–90.

[18] Weibing Chen, Jingyue Li, Jianqiang Ma, Reidar Conradi, Junzhong Ji, and Chunnian Liu. "An empirical study on software development with open source components in the chinese software industry." In: *Software Process: Improvement and Practice* 13.1 (2008), pp. 89–100.

[19] Tsun Chow and Dac-Buu Cao. "A survey study of critical success factors in agile software projects." In: *Journal of Systems and Software* 81.6 (2008), pp. 961–971.

[20] Mike Cohn and Doris Ford. "Introducing an agile process to an organization." In: *Computer* 6 (2003), pp. 74–78.

[21] Juliet Corbin and Anselm Strauss. *Basics of qualitative research: Techniques and procedures for developing grounded theory*. Sage publications, 2014.

[22] Vittorio Cortellessa, Fabrizio Marinelli, and Pasqualina Potena. "Automated selection of software components based on cost/reliability tradeoff." In: *Software Architecture*. Springer, 2006, pp. 66–81.

[23] Vittorio Cortellessa, Fabrizio Marinelli, and Pasqualina Potena. "An optimization framework for "build-or-buy" decisions in software architecture." In: *Computers & Operations Research* 35.10 (2008), pp. 3090–3106.

[24] Jamie L Crandell, Corrine I Voils, and Margarete Sandelowski. "Bayesian approaches to the synthesis of qualitative and quantitative research findings." In: *Synthesizing Qualitative Research: Choosing the Right Approach* (2012), pp. 137–159.

[25] Ivica Crnkovic. "Component-based software engineering—new challenges in software development." In: *Software Focus* 2.4 (2001), pp. 127–133.

[26] Daniela S Cruzes and Tore Dybå. "Recommended steps for thematic synthesis in software engineering." In: *Empirical Software Engineering and Measurement (ESEM), 2011 International Symposium on*. IEEE. 2011, pp. 275–284.

[27] Daniela S Cruzes and Tore Dybå. "Research synthesis in software engineering: A tertiary study." In: *Information and Software Technology* 53.5 (2011), pp. 440–455.

[28] Daniela Damian and James Chisan. "An empirical study of the complex relationships between requirements engineering processes and other processes that lead to payoffs in productivity, quality, and risk management." In: *Software Engineering, IEEE Transactions on* 32.7 (2006), pp. 433–453.

[29] Piergiorgio Di Giacomo. "COTS and open source software components: are they really different on the battlefield?" In: *COTS-Based Software Systems*. Springer, 2005, pp. 301–310.

[30] Mary Dixon-Woods, Shona Agarwal, David Jones, Bridget Young, and Alex Sutton. "Synthesising qualitative and quantitative evidence: a review of possible methods." In: *Journal of health services research & policy* 10.1 (2005), 45–53B.

[31] Tore Dybå, Torgeir Dingsøyr, and Geir K Hanssen. "Applying systematic reviews to diverse study types: An experience report." In: *Proceedings of the 1st International Symposium on Empirical Software Engineering and Measurement*. IEEE Computer Society. 2007, pp. 225–234.

[32] Tore Dybå, Barbara Kitchenham, and Magne Jørgensen. "Evidence-based software engineering for practitioners." In: *Software, IEEE* 22.1 (2005), pp. 58–65.

[33] Samuel Fricker, Tony Gorschek, Carl Byman, and Armin Schmidle. "Handshaking with implementation proposals: Negotiating requirements understanding." In: *IEEE software* 27.2 (2010), p. 72.

[34] Samuel Fricker and Susanne Schumacher. "Release Planning with Feature Trees: Industrial Case." In: *REFSQ*. Springer. 2012, pp. 288–305.

[35] Cristina Gacek and Budi Arief. "The many meanings of open source." In: *Software, IEEE* 21.1 (2004), pp. 34–40.

[36] Rosanna Garcia and Roger Calantone. "A critical look at technological innovation typology and innovativeness terminology: a literature review." In: *Journal of product innovation management* 19.2 (2002), pp. 110–132.

[37] David A Garvin. "Building a learning organization." In: *Org Dev & Trng, 6E (Iae)* 274 (1985).

[38] Trisha Greenhalgh and Richard Peacock. "Effectiveness and efficiency of search methods in systematic reviews of complex evidence: audit of primary sources." In: *Bmj* 331.7524 (2005), pp. 1064–1065.

[39] Øyvind Hauge, Claudia Ayala, and Reidar Conradi. "Adoption of open source software in software-intensive organizations–A systematic literature review." In: *Information and Software Technology* 52.11 (2010), pp. 1133–1154.

[40] Tuija Helokunnas. "The dimensions of embedded COTS and OSS software component integration." In: *Product Focused Software Process Improvement*. Springer, 2002, pp. 509–518.

[41] Tuija Helokunnas and Marko Nyby. "Collaboration between a COTS Integrator and Vendors." In: *Software Quality—ECSQ 2002*. Springer, 2002, pp. 267–273.

[42] James D Herbsleb and Deependra Moitra. "Global software development." In: *Software, IEEE* 18.2 (2001), pp. 16–20.

[43] Scott A. Hissam, Daniel Plakosh, and C Weinstock. "Trust and vulnerability in open source software." In: *IEE Proceedings-Software* 149.1 (2002), pp. 47–51.

[44] Scott A Hissam and Charles B Weinstock. "Open source software: The other commercial software." In: *1st Workshop on Open Source Software at ICSE*. 2001.

[45] Rashina Hoda, Philippe Kruchten, James Noble, and Stuart Marshall. "Agility in context." In: *ACM Sigplan Notices*. Vol. 45. 10. ACM. 2010, pp. 74–88.

[46] Martin Höst and Alma Oručević-Alagić. "A systematic review of research on open source software in commercial software product development." In: *Information and Software Technology* 53.6 (2011), pp. 616–624.

[47] Martin Ivarsson and Tony Gorschek. "A method for evaluating rigor and industrial relevance of technology evaluations." In: *Empirical Software Engineering* 16.3 (2011), pp. 365–395.

[48] Samireh Jalali and Claes Wohlin. "Systematic literature studies: database searches vs. backward snowballing." In: *Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement*. ACM. 2012, pp. 29–38.

[49] PC Jha, Shivani Bali, U Dinesh Kumar, and Hoang Pham. "Fuzzy optimization approach to component selection of fault-tolerant software system." In: *Memetic Computing* 6.1 (2014), pp. 49–59.

[50] PC Jha, Vikram Bali, Sonam Narula, and Mala Kalra. "Optimal component selection based on cohesion & coupling for component based software system under build-or-buy scheme." In: *Journal of Computational Science* 5.2 (2014), pp. 233–242.

[51] Joseph B Kadane and Lara J Wolfson. "Experiences in elicitation." In: *The Statistician* (1998), pp. 3–19.

[52] Lena Karlsson, Åsa G Dahlstedt, Björn Regnell, Johan Natt och Dag, and Anne Persson. "Requirements engineering challenges in market-driven software development–An interview study with practitioners." In: *Information and Software technology* 49.6 (2007), pp. 588–604.

[53] Siffat Ullah Khan, Mahmood Niazi, and Rashid Ahmad. "Factors influencing clients in the selection of offshore software outsourcing vendors: An exploratory study using a systematic literature review." In: *Journal of systems and software* 84.4 (2011), pp. 686–699.

[54] Barbara A Kitchenham and S Charters. *Guidelines for performing systematic literature reviews in software engineering*. Tech. rep. Technical report, EBSE Technical Report EBSE-2007-01, 2007.

[55] Barbara A Kitchenham, Tore Dyba, and Magne Jorgensen. "Evidence-based software engineering." In: *Software Engineering, 2004. ICSE 2004. Proceedings. 26th International Conference on*. IEEE. 2004, pp. 273–281.

[56] Barbara Kitchenham, O Pearl Brereton, David Budgen, Mark Turner, John Bailey, and Stephen Linkman. "Systematic literature reviews in software engineering–a systematic literature review." In: *Information and software technology* 51.1 (2009), pp. 7–15.

[57] Stefan Koch. "Open source as a sourcing strategy for corporations." In: *International Journal of Business Innovation and Research* 5.1 (2010), pp. 1–16.

[58] Tommi Kramer and Michael Eschweiler. "Outsourcing location selection with SODA: a requirements based decision support methodology and tool." In: *Advanced Information Systems Engineering*. Springer. 2013, pp. 530–545.

[59] Tommi Kramer, Armin Heinzl, and Kai Spohrer. "Should this software component be developed inside or outside our firm?-A design science perspective on the sourcing of application systems." In: *New Studies in Global IT and Business Service Outsourcing- 5th Global Sourcing Workshop 2011, Courchevel, France, March 14-17, 2011*. Springer, 2011, pp. 115–132.

[60] Philippe Kruchten. "Scaling down large projects to meet the agile "sweet spot"." In: *IBM developerWorks* 13 (2004).

[61] Rikard Land, Laurens Blankers, Michel Chaudron, and Ivica Crnković. "COTS selection best practices in literature and in industry." In: *High Confidence Software Reuse in Large Systems*. Springer, 2008, pp. 100–111.

[62] Dean Leffingwell. *Scaling software agility: best practices for large enterprises*. Pearson Education, 2007.

[63] Jingyue Li, Finn Olav Bjørnson, Reidar Conradi, and Vigdis B Kampenes. "An empirical study of variations in COTS-based software development processes in the Norwegian IT industry." In: *Empirical Software Engineering* 11.3 (2006), pp. 433–461.

[64] Jingyue Li, Reidar Conradi, Odd Petter N Slyngstad, Christian Bunse, Umair Khan, Marco Torchiano, and Maurizio Morisio. "Validation of new theses on off-the-shelf component based development." In: *Software Metrics, 2005. 11th IEEE International Symposium*. IEEE. 2005, pp. 26–26.

[65] Jingyue Li, Reidar Conradi, Odd Petter N Slyngstad, Christian Bunse, Marco Torchiano, and Maurizio Morisio. "An empirical study on decision making in off-the-shelf component-based development." In: *Proceedings of the 28th international conference on Software engineering*. ACM. 2006, pp. 897–900.

[66] Mikael Lindvall, Dirk Muthig, Aldo Dagnino, Christina Wallin, Michael Stupperich, David Kiefer, John May, and Tuomo Kähkönen. "Agile software development in large organizations." In: *Computer* 37.12 (2004), pp. 26–34.

[67] Gary S Lynn, Joseph G Morone, and Albert S Paulson. "Marketing and discontinuous innovation: the probe and learn process." In: *California management review* 38.3 (1996), pp. 8–37.

[68] Stephen MacDonell, Martin Shepperd, Barbara Kitchenham, and Emilia Mendes. "How reliable are systematic reviews in empirical software engineering?" In: *Software Engineering, IEEE Transactions on* 36.5 (2010), pp. 676–687.

[69] Josip Maras, Luka Lednicki, and Ivica Crnkovic. "15 years of CBSE symposium: impact on the research community." In: *Proceedings of the 15th ACM SIGSOFT symposium on Component Based Software Engineering*. ACM. 2012, pp. 61–70.

[70] Valerie Maxville, Jocelyn Armarego, and Chiou Peng Lam. "Applying a reusable framework for software selection." In: *Software, IET* 3.5 (2009), pp. 369–380.

[71] Birendra Mishra, Ashutosh Prasad, and Srinivasan Raghunathan. "Quality and profits under open source versus closed source." In: *ICIS 2002 Proceedings* (2002), p. 32.

[72] Ayse Tosun Misirli and Ayse Basar Bener. "Bayesian Networks For Evidence-Based Decision-Making in Software Engineering." In: *Software Engineering, IEEE Transactions on* 40.6 (2014), pp. 533–554.

[73] Mirko Morandini, Alberto Siena, and Angelo Susi. "Risk Awareness in Open Source Component Selection." In: *Business Information Systems*. Springer. 2014, pp. 241–252.

[74] Maurizio Morisio and Marco Torchiano. "Definition and classification of COTS: a proposal." In: *COTS-Based Software Systems*. Springer, 2002, pp. 165–175.

[75] Sridhar Nerur, RadhaKanta Mahapatra, and George Mangalaraj. "Challenges of migrating to agile methodologies." In: *Communications of the ACM* 48.5 (2005), pp. 72–78.

[76] Jeffrey S Norris. "Mission-critical development with open source software: Lessons learned." In: *Software, IEEE* 21.1 (2004), pp. 42–49.

[77] Kai Petersen and Nauman Bin Ali. "Identifying Strategies for Study Selection in Systematic Reviews and Maps." In: *Proceedings of the 5th International Symposium on Empirical Software Engineering and Measurement, ESEM 2011, Banff, AB, Canada, September 22-23, 2011*. 2011, pp. 351–354.

[78] Kai Petersen and Cigdem Gencel. "Worldviews, research methods, and their relationship to validity in empirical software engineering research." In: *Software Measurement and the 2013 Eighth International Conference on Software Process and Product Measurement (IWSM-MENSURA), 2013 Joint Conference of the 23rd International Workshop on*. IEEE. 2013, pp. 81–89.

[79] Kai Petersen and Claes Wohlin. "Context in industrial software engineering research." In: *Proceedings of the 3rd International Symposium on Empirical Software Engineering and Measurement*. IEEE Computer Society. 2009, pp. 401–404.

[80] Kai Petersen and Claes Wohlin. "Measuring the flow in lean software development." In: *Software: Practice and experience* 41.9 (2011), pp. 975–996.

[81] Fredrik Pettersson, Martin Ivarsson, Tony Gorschek, and Peter Öhman. "A practitioner's guide to light weight software process assessment and improvement planning." In: *Journal of Systems and Software* 81.6 (2008), pp. 972–995.

[82] Catherine Pope, Nick Mays, and Jennie Popay. *Synthesising Qualitative and Quantitative Health Evidence: A Guide to Methods: A Guide to Methods*. 2007.

[83] Pasqualina Potena. "Composition and tradeoff of non-functional attributes in software systems: research directions." In: *The 6th Joint Meeting on European software engineering conference and the ACM SIGSOFT symposium on the foundations of software engineering: companion papers*. ACM. 2007, pp. 583–586.

[84] Balasubramaniam Ramesh, Lan Cao, Kannan Mohan, and Peng Xu. "Can distributed software development be agile?" In: *Communications of the ACM* 49.10 (2006), pp. 41–46.

[85] Donald J Reifer, Frank Maurer, and Hakan Erdogmus. "Scaling agile methods." In: *Software, IEEE* 20.4 (2003), pp. 12–14.

[86] Mehwish Riaz, Emilia Mendes, and Ewan Tempero. "A systematic review of software maintainability prediction and metrics." In: *Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*. IEEE. 2009, pp. 367–377.

[87] Karen A Roberts, Mary Dixon-Woods, Ray Fitzpatrick, Keith R Abrams, and David R Jones. "Factors affecting uptake of childhood immunisation: a Bayesian synthesis of qualitative and quantitative evidence." In: *The Lancet* 360.9345 (2002), pp. 1596–1599.

[88] C Robson. *Real world research: A resource for social scientists and practitioner-researchers. 1993*.

[89] Jakub Rudzki, Kimmo Kiviluoma, Tero Poikonen, and Imed Hammouda. "Evaluating quality of open source components for reuse-intensive commercial solutions." In: *Software Engineering and Advanced Applications, 2009. SEAA'09. 35th Euromicro Conference on*. IEEE. 2009, pp. 11–19.

[90] Per Runeson, Martin Host, Austen Rainer, and Bjorn Regnell. *Case study research in software engineering: Guidelines and examples*. John Wiley & Sons, 2012.

[91] Mats Skoglund and Per Runeson. "Reference-based search strategies in systematic reviews." In: *the Proceedings of the 13th International Conference on Evaluation and Assessment in Software Engineering, Durham, England*. 2009.

[92] David J Spiegelhalter, Keith R Abrams, and Jonathan P Myles. *Bayesian approaches to clinical trials and health-care evaluation*. Vol. 13. John Wiley & Sons, 2004.

[93] Adil AA Ssaed, Wan Kadir, and Siti Zaiton Mohd Hashim. "Metaheuristic Search Approach Based on In-house/Outsourced Strategy to Solve Redundancy Allocation Problem in Component-Based Software Systems." In: *International Journal of Software Engineering and Its Applications* 6.4 (2012), pp. 143–154.

[94] Katherine J Stewart, Anthony P Ammeter, and Likoebe M Maruping. "A preliminary analysis of the influences of licensing and organizational sponsorship on success in open source projects." In: *System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on*. IEEE. 2005, pp. 197c–197c.

[95] Klaas-Jan Stol, Muhammad Ali Babar, Paris Avgeriou, and Brian Fitzgerald. "A comparative study of challenges in integrating Open Source Software and Inner Source Software." In: *Information and Software Technology* 53.12 (2011), pp. 1319–1336.

[96] Klaas-Jan Stol, Muhammad Ali Babar, Barbara Russo, and Brian Fitzgerald. "The use of empirical methods in open source software research: Facts, trends and future directions." In: *Proceedings of the 2009 ICSE Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development*. IEEE Computer Society. 2009, pp. 19–24.

[97] Abu Bakar Md Sultan, Abubakar Diwani Bakar, Hazura Zulzalil, and Jamilah Din. "Systematic Literature Review in Open Source Software Maintainability." In: *International Review on Computers & Software* 7.5 (2012).

[98] Md Sultan, Abu Bakar, Abubakar Diwani Bakar, Hazura Zulzalil, and Jamilah Din. "Systematic literature review: the important factors in assessing the trustworthiness of OSS." In: *International Journal on Communication Antenna & Propagation (IRECAP)* 3.1 (2013), pp. 56–60.

[99] Jeff Sutherland, Guido Schoonheim, Narinder Kumar, V Pandey, and S Vishal. "Fully distributed scrum: Linear scalability of production between San Francisco and India." In: *Agile Conference, 2009. AGILE'09*. IEEE. 2009, pp. 277–282.

[100] Jeff Sutherland, Anton Viktorov, Jack Blount, and Nikolai Puntikov. "Distributed scrum: Agile project management with outsourced development teams." In: *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on*. IEEE. 2007, 274a–274a.

[101] Sharifah Mashita Syed-Mohamad and Tom McBride. "A Comparison of the Reliability Growth of Open Source and In-House Software." In: *15th Asia-Pacific Software Engineering Conference (APSEC 2008)*. 2008, pp. 229–236.

[102] Mahbubul Syeed, Imed Hammouda, and Tarja Syatä. "Evolution of open source software projects: A systematic literature review." In: *Journal of Software* 8.11 (2013), pp. 2815–2829.

[103] CMMI Product Team. "CMMI for Development, Version 1.3 (CMU/SEI-2010-TR-033)." In: *Software Engineering Institute, Carnegie Mellon University* (2010).

[104] Russell Torres. "Developer-led adoption of open source software libraries: A conceptual model." In: *18th Americas Conference on Information Systems*. 2012, pp. 583–586.

[105] Tassio Vale, Ivica Crnkovic, Eduardo Santana de Almeida, Paulo Anselmo da Mota Silveira Neto, Yguaratã Cerqueira Cavalcanti, and Silvio Romero de Lemos Meira. "Twenty-eight years of component-based software engineering." In: *Journal of Systems and Software* 111 (2016), pp. 128–148.

[106] Michiel Van Genuchten. "Why is software late? An empirical study of reasons for delay in software development." In: *Software Engineering, IEEE Transactions on* 17.6 (1991), pp. 582–590.

[107] Corrine Voils, Vic Hasselblad, Jamie Crandell, YunKyung Chang, EunJeong Lee, and Margarete Sandelowski. "A Bayesian method for the synthesis of evidence from qualitative and quantitative reports: the example of antiretroviral

medication adherence." In: *Journal of health services research & policy* 14.4 (2009), pp. 226–233.

[108] Tom Wanyama and B Far. "An empirical study to compare three methods for selecting COTS software components." In: *International Journal of Computing and ICT Research* 2.1 (2008), pp. 34–46.

[109] Jane Webster and Richard T Watson. "Analyzing the past to prepare for the future: Writing a literature review." In: *Management Information Systems Quarterly* 26.2 (2002), p. 3.

[110] Roel Wieringa, Neil A. M. Maiden, Nancy R. Mead, and Colette Rolland. "Requirements engineering paper classification and evaluation criteria: a proposal and a discussion." In: *Requir. Eng.* 11.1 (2006), pp. 102–107.

[111] Claes Wohlin. "Guidelines for Snowballing in Systematic Literature Studies and a Replication in Software Engineering." In: *8th International Conference on Evaluation and Assessment in Software Engineering (EASE 2014)*. ACM. 2014, pp. 321–330.

[112] Claes Wohlin, Per Runeson, Paulo Anselmo da Mota Silveira Neto, Emelie Engström, Ivan do Carmo Machado, and Eduardo Santana de Almeida. "On the reliability of mapping studies in software engineering." In: *Journal of Systems and Software* 86.10 (2013), pp. 2594–2610.

[113] Claes Wohlin, Darja Smite, Franke Ulrik, Krzysztof Wnuk, Deepika Badampudi, and Cicchetti Antonio. "A Decision-making Framework for Selecting Software Assets." In: *Submission*. 2016.

[114] Jin Xu, Yongqin Gao, Scott Christley, and Gregory Madey. "A topological analysis of the open souce software development community." In: *System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on*. IEEE. 2005, 198a–198a.

[115] Robert K Yin. *Case study research: Design and methods*. Sage publications, 2013.

[116] He Zhang, Muhammad Ali Babar, and Paolo Tell. "Identifying relevant studies in software engineering." In: *Information & Software Technology* 53.6 (2011), pp. 625–637.

ABSTRACT

Context: The amount of software in solutions provided in various domains is continuously growing. These solutions are a mix of hardware and software solutions, often referred to as software-intensive systems. Companies seek to improve the software development process to avoid delays or cost overruns related to the software development.

Objective: The overall goal of this thesis is to improve the software development/building process to provide timely, high quality and cost efficient solutions. The objective is to select the origin of the components (in-house, outsource, components off-the-shelf (COTS) or open source software (OSS)) that facilitates the improvement. The system can be built of components from one origin or a combination of two or more (or even all) origins. Selecting a proper origin for a component is important to get the most out of a component and to optimize the development.

Method: It is necessary to investigate the component origins to make decisions to select among different origins. We conducted a case study to explore the existing challenges in software development. The next step was to identify factors that influence the choice to select among different component origins through a systematic literature review using a snowballing (SB) strategy and a database (DB) search. Furthermore, a Bayesian synthesis process is proposed to integrate the evidence from literature into practice.

Results: The results of this thesis indicate that the context of software-intensive systems such as domain regulations hinder the software development improvement. In addition to in-house development, alternative component origins (outsourcing, COTS, and OSS) are being used for software development. Several factors such as time, cost and license implications influence the selection of component origins. Solutions have been proposed to support the decision-making. However, these solutions consider only a subset of factors identified in the literature.

Conclusions: Each component origin has some advantages and disadvantages. Depending on the scenario, one component origin is more suitable than the others. It is important to investigate the different scenarios and suitability of the component origins, which is recognized as future work of this thesis. In addition, the future work is aimed at providing models to support the decision-making process.

Keywords: Component-based software development, component origin, decision-making, snowballing, database search, Bayesian synthesis.

ISSN 1650-2140

2016:01

ISBN: 978-91-7295-323-9