Software Development Methodologies and Practices in Startups - Systematic Literature Review

University of Oulu
Faculty of Information Technology and
Electrical Engineering/ M3S
Master's Thesis
Esubalew Workineh Tegegne
Date: 27.2.2018

Abstract

Startups are newly created companies with little or no operating history and with limited resources and markets. Software startup are aiming to develop cutting edge software products under highly uncertain conditions, overcoming fast-growing markets under multiple influences. The significance of startups for economic development is growing in software business due to their ability to quickly create innovative technologies and their potential to scale to a wide market. The objective of this thesis is to identify and analyze existing scientific literature regarding software development methodologies and practices used in startup companies published between Jan 2006 and Dec 2016.

A systematic literature review was conducted to explore the previous research on software development methods and practices in startups. A review protocol was developed and used as a procedure to conduct the review. The data was collected from three databases i.e. IEEE Xplore, Scopus, ISI Web of Science. This research identified 30 relevant primary studies out of a total 1249 papers to answer defined research questions. The result show that agile and Lean Startup methodologies are the most used methodologies across startup companies. Due to the flexible nature of startups, tailoring specific methodology is a common practice among startup companies. Additionally, a total of 95 software development work practices are extracted from primary studies.

This thesis concludes that software development methodologies in startups are informal, customized, and highly dependent on the experience of project managers. It is also noted that principles related to these development methodologies are not strictly followed due to limited resources and uncertainty in the market. Furthermore, the results shows that software startups choose development methodologies and practices which are flexible, light-weight and allow rapid changes.

Keywords

Startups, software startups, software development, systematic literature review

Supervisor

Dr. Muhammad Ovais Ahmad

Foreword

I would like to express my sincere gratitude to my supervisor Dr. Muhammad Ovais Ahmad who gave me professional guidance and valuable feedback throughout the process of conducting systematic literature review and thesis writing. He has always been providing me valuable suggestions and motivation whenever I was trapped with some issues.

At last but not least I want to thank my families and friends who always been with me encouraging me to complete this thesis. The thesis process has been challenging but with your encouragements and supports of various kinds, I am able to complete it.

Esubalew Workineh Tegegne
Oulu, February 27, 2018

Abbreviation

SLR	Systematic Literature Review
SMS	Systematic Mapping Study
RQ	Research Question
SE	Software Engineering
SWEBOK	Software Engineering Body of Knowledge
XP	Extreme Programming
MDD	Model-Driven Development
RUP	Rational Unified Process
ESSSDM	Early Stage Software Startup Development Model
MVP	Minimum Viable Product

Contents

1. Introduction

Startups are new phenomena in the technology market, creating new businesses and jobs (Giardino, 2015). In recent years, the number of software startups has increased globally and their role in technology and economic development have become significantly high (Seppänen, Tripathi, Oivo & Liukkunen, 2017). Technological advancements has brought an increased demand of new software and services. As a result, new software companies are established every day, and emerging technologies (such as smartphones, cloud infrastructure platforms and enhanced web development tools) have made it easier to get started. Software startups are playing an important role in addressing the high demand of software products, by developing software in short time and with a limited resources (Klotins, Unterkalmsteiner & Gorschek, 2015). Many success stories surrounding software startups, such as Facebook, Instagram and Uber, contribute to their popularity of startups. These success stories are used as a driving fuel in the software startups environment. However, despite these success stories in the media, all startups companies do not succeed. Studies (Blank, 2006; Klotins et al., 2014; Giardino et al., 2014; Blank, 2011) indicate that most of startup companies fail before getting any significant success. According to Giardino et al. (2014), 60% of the total startups in the world do not survive the first five years from their creation. This has led researchers (Giardino et al., 2014; Paternoster et al., 2014; Blank, 2011) to study and identify what factors contribute to software startups succeeding or failing. For example, high failure rate among startups is due to lack of resources, immaturity, multiple influences and dynamic technologies. On the other hand, there is a high speculation indicating that inadequate software development practices might be a significant contributing factor for the high failure rates in startups (Klotins et al., 2015).

There are several empirical studies conducted in investigating how established companies use software development methods and practices (Budgen, 2015; Giardino et al., 2014; Atkinson & Benefield; 2013, Meng & Yao, 2008). From a software engineering perspective startups differ from established companies, because startups develop software in a situation where resources are very limited and processes are not followed by a prescriptive methodology (Paternoster, 2014). Nevertheless, research on software development activities in startups is scarce. There is an identified research gap with the particular focus on software development in startups.

In the past, very few studies have identified, characterized and mapped software development methods and practices in software startups (Sutton, 2000; Paternoster et al., 2014; Klotins et al., 2015; Coleman, 2005). According to Klotins et al. (2015), existing knowledge about software engineering practices are limited among the startups and existing body of knowledge are insufficient to support engineering aspects in startups. As a result of growing interest in software startups, studying various aspects of software development in startup companies become an important issue. The emerging of new startup approaches in recent years, like Lean Startups, has resulted an increasing interest in studying the existing approaches and methods in startups (Giardino et al., 2014). Consequently, studies are essential to fill the identified gap and support startups in minimizing the chance of failure due to improper use of development methods and practices. The following Table 1, shows the comparison between existing mapping studies (Klotins et al., 2015; Paternoster, 2014) and work conducted in this thesis.

Table 1. Comparison of previous studies on software startup.

Comparison Context	**Paternoster et al. (2014)**	**Klotins et al. (2015)**	**This study (2017)**
Purpose	Extract and analyze software development work practices used in startups	Identify and categorize software engineering practices utilized in startups and mapped to Software Engineering Book of Knowledge (SWEBOK)	Identify and analyze software development methodologies and work practices used in startups
Years included	1994-2013	1994-2014	2006-2016
Number of Primary studies	43 primary studies • Experience Report • Empirical studies	14 primary studies • Experience Report • Empirical studies	30 primary studies • Experience Report • Empirical studies
Key Results and Conclusions	• Extraction of 213 software development work practices in startups. • The existing body of knowledge is limited to a few high quality studies. • Software development work practices are chosen opportunistically.	• Identified 54 software engineering practices & mapped to 11 main software engineering knowledge areas from SWEBOK. • Existing studies addressing software engineering in startups are insufficient to support all engineering aspects and do not create a solid body of knowledge.	

The study conducted by Paternoster et al. (2014) has identified, categorized and analyzed 213 software development work practices used in software startup companies. Klotins et al. (2015) identified 54 software engineering practices in startup companies which are mapped to the SWEBOK knowledge areas (IEEE Computer Society, 2004). However, both studies (Paternoster et al., 2014; Klotins et al., 2015) didn’t extract software development methodologies used by software startups. Therefore, this thesis will be conducted as a complementary study to the previous SLRs, with focus on both software development methodologies and practices in startups. The identified gap will be fulfilled by investigating software development methodologies and practices used in startups by conducting a Systematic Literature Review (SLR).

This SLR consists of 30 primary studies that were identified from an initial set of 1249 papers. A total of 12 software development methodologies and 95 software development work practices that are used by startups are extracted from relevant primary studies. Software startups can take advantage of these lists development methodologies and practices in improving their decision making process regarding software development. Furthermore, this SLR on software startups provides researchers with directions for future work.

This thesis is structured as follows: Section 2 describes the background of this thesis; Section 3 presents the SLR and its procedure that has been followed to perform the review; The results are presented in Section 4. Section 5 discuss answer of the research questions; Section 6 provide conclusion to the thesis.

2. Background

This chapter explains the background of the startups, describing the key concepts of software development in startups. Section 2.1 explains the definition and characteristics of a startup company. Section 2.2 discusses the characteristics of software startups and section 2.3 explains the general overview of software development methodologies and practices used in software startup.

2.1 Startups

Startups are described as a human institutions designed to create a new product under conditions of extreme uncertainty (Ries, 2011). Blank (2006) defines a startup as temporary organization that builds high-tech innovative products and has no previous operating history. Startups are new organization who seek scalable, repeatable, and profitable business model in order to survive and grow (Blank, 2006). Sutton (2000) also has defined a startup as an organization that is challenged by youth and immaturity, extremely limited resources, multiple influences and dynamic technologies and markets. Different researchers and practitioners have defined startups differently, but they agreed with the common elements and on the definition that "startup is a small company exploring new business opportunities, working to solve a problem where the solution is not well-known and the market is highly volatile" (Giardino et al., 2014). Nevertheless, being newly founded does not in itself make a company a startup, the two key natures of startups, high-uncertainty and rapidly-evolvement, characterize them as startups and better differentiate them from more established companies.

In startup companies, economical, human and physical resources are extremely limited. Startups are characterized as high uncertainty towards market, aim to grow and scale up rapidly, and with a small numbers of individuals in the team. These characteristics mainly indicate the differences between startups and well-established companies which have more resources, already established market and a scalable business model. Startups are also unique in the sense that they are designed to search for a repeatable and scalable business model, in order to transform to a large company. Due to dynamic, unpredictable and even chaotic environment, they are forced to act quickly, fail fast and learn faster to find a suitable market to acquire a sustainable income (Unterkalmsteiner et al., 2016).

Startups have different phases of lifecycle, from the early stage of their inception until they become an established company. Crowne (2002) has explained different phases of startups, which includes startup, stabilization, growth and maturity phases. Through these phase, a startup company experience different challenges to become an established company. A startup is considered as evolved into a mature or established company when the market size, share and growth rate have been established, all processes necessary to support product development and sales are in place. Besides that established companies have a diversified team of multi-skilled professionals in the product development, and the company has a successful product in the market (Crowne, 2002). Therefore, startups are newly formed companies with less or no operating history, bounded by resource limitation and uncertain market But aiming to grow fast and deliver products in short period of time.

2.2 Software Startups

The term "Software Startup" has been used in software engineering literature since 1994, when Karmel (1994) first introduced the term. Since then, different researchers has given their own definitions for the term "Software Startups" (Unterkalmsteiner et al., 2016; Coleman and Connor, 2008; Sutton, 2000). Coleman and Connor (2008) describe software startups as unique companies that develop software through various processes and without a prescribed methodology. Sutton (2000) characterized software startups by the challenge they face as:

- Little or no operating history – Software startups have little accumulated experience in development processes and organization management.
- Young or immature compared to more established and mature companies
- Limited resources: Scarcity in finance, time, human resources and market; typically focus on delivering the product, promoting the product and building up customers.
- Multiple influences: Pressure from investors, customers, partners and competitors impact the decision-making in a company. Although individually important, overall they might be inconsistent.
- Dynamic technologies and markets: newness of software companies often require to develop or operate with disruptive technologies to enter into a high-potential target market.

Therefore, many researchers shared the above characterization of software startups (Paternoster et al., 2014). Software startups are also known for the absence of a business model (Coleman, 2008). From business model perspective, there are two kinds of software startups: product-oriented and project-oriented startups. Product-oriented software startups develop, market and sell their own product whereas project-oriented software startups do not develop their own product, instead they serve their customers by building products for them (Eloranta, 2014).

2.3 Software Development in Startups

Software has been part of modern society for more than 50 years. Software development has started as a chaotic activity often referred as "code and fix". Software was written with a little planning, and the design of the product was set by many short term decisions (Awad, 2005). This approach has been used to develop simple products but as the product grows bigger it became more difficult to add new features and to fix bugs. This style of development was used for several years until an alternative, called 'Methodology', was introduced (Awad, 2005). "A development methodology imposes a disciplined process upon software development with the aim of making software development more predictable and more efficient." (Awad, 2005).

A software development methodology has been defined as a set of activities, practices, or processes that an organization use to develop and maintain software and the associated products (e.g., project plans, design documents, code, test cases and user manuals) (Zahran, 1998; Sanchez & Connor, 2015). Chapman (2004) also defines software development methodology as "the documented collection of policies, processes and procedures used by a development team or organization to practice software engineering". Software development practices are concrete activities and actions that a methodology defines to be used in the process (Abrahamsson et al., 2017).

It is important to note that the software development methodology does not simply imply the development of software code; it covers the entire process involved in developing and maintaining a software product (Chapman, 2004). However, the process is not a rigid instruction how to develop a software product. Rather, it is a flexible approach that enables the software team or organization to do the work by choosing the appropriate set of work actions and tasks (Sanchez & Connor, 2015). As a result, software companies are able to deliver a software product in a timely manner and with sufficient product quality to satisfy their customers. In the past, there has been a number of different methodologies designed to help companies manage their software development activities. However, no single approach has achieved a generalized acceptance, as there are number of other contextual and situational factors that influence the choice of methodologies and processes (Clarke & O’Connor, 2012).

Startups are ultimately concerned with survival rather than establishing working procedures. Most of the work is done by energetic and committed people without defined development approach (Coleman & O’Connor, 2008). In general, the methods and processes of software development in startups has been neglected in the studies (Sutton (2000) or little known (Paternoster et al., 2014; Giardino et al., 2014). Due to this, there is limited knowledge concerning development methodologies among project managers.

3. Systematic Literature Review

This chapter discusses the procedures of conducting the SLR. First, the overview of the research method is presented and its key stages are discussed. Then, each stage of the SLR is further elaborated accordingly.

3.1 Overview of Systematic Literature Review

According to Kitchenham & Charters (2007), "A systematic literature review (often referred to as a systematic review) is a means of identifying, evaluating and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest". This thesis followings SLR guidelines provided by Kitchenham & Charters (2007). It can be conducted with the following set of reasons as discussed by Kitchenham & Charters (2007):

- “To summarize the existing evidence concerning a treatment or technology e.g. to summarize the empirical evidence of the benefits and limitations of a specific agile method”.
- “To identify any gaps in current research in order to suggest areas for the further investigation”.
- “To provide a framework or background for appropriately positioning of new research activities”.

In the literature review process the following aspects are covered (Kitchenham & Charters, 2007):

- One of the important elements in systematic review, review protocol is developed to conduct the systematic literature review. The review protocol specifies the research questions being answered and the methods to be used for undertaking a particular review.
- Search strategy is defined to conduct the review. The aim of the search strategy is for a complete and intensive search for primary studies and to identify the maximum possible number of relevant literatures.
- Selection criteria are defined to conduct the review. Inclusion and exclusion criteria are used to obtain potentially relevant primary studies that provide direct evidence about the research question.
- Quality assessment for inclusive studies is applied. The information needed to be extracted from primary studies are evaluated through quality criteria.
- Data extraction is done according to the research question. Data extraction forms or other reviewing tools are used to document the extracted information.
- The search strategy and results have been documented for the future reference of the reader.
- Summarizing and analyzing the data on study results of all the primary studies focusing on the research question.
- Reporting the results.

SLR involves three phases to be executed in order to obtain reliable and valid results. Figure 1, shows the three main phases of SLR which are based on guidelines of Kitchenham & Charters (2007).

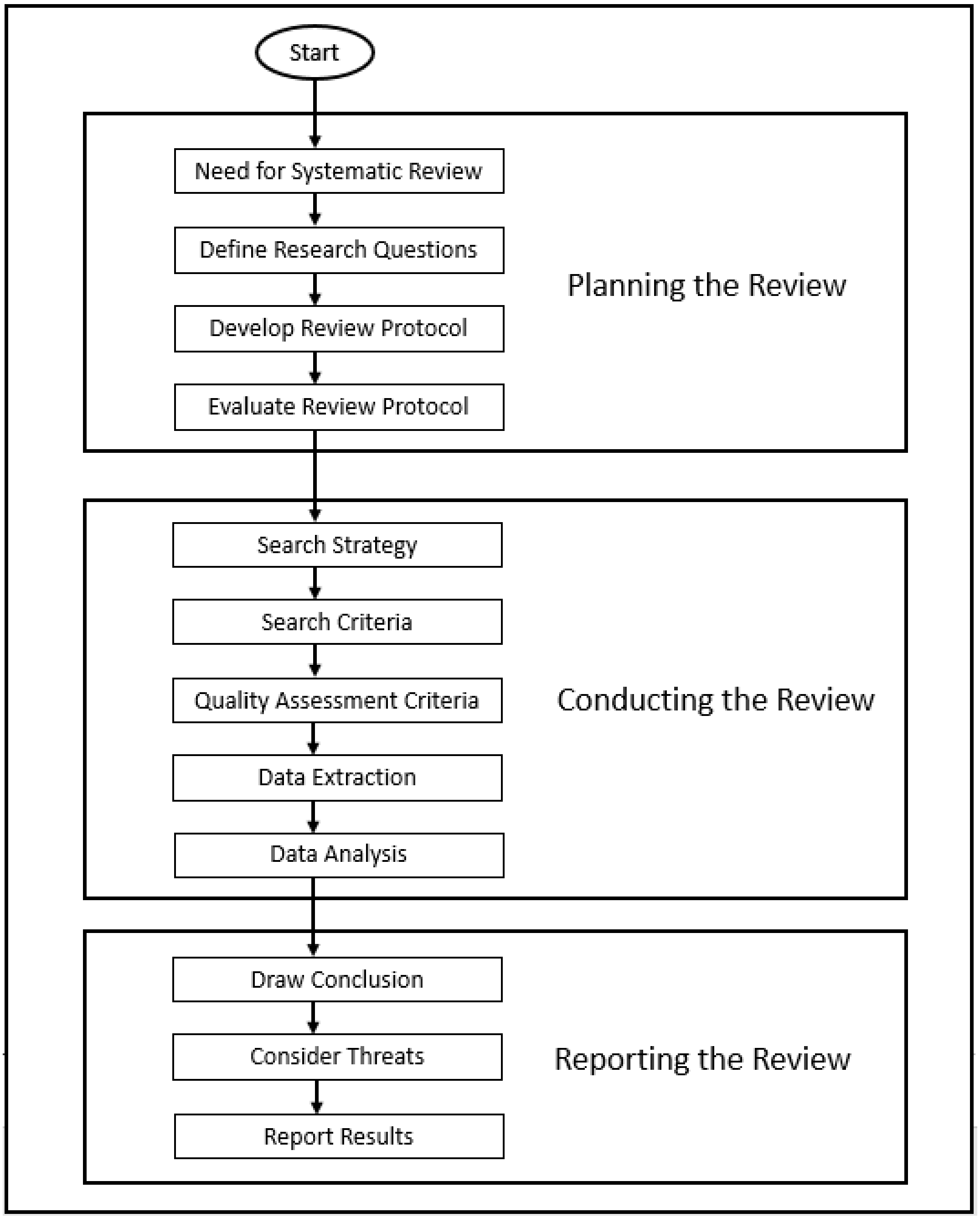

Figure 1. Steps that are followed for SLR (Kitchenham & Charter, 2007).

3.2 Planning the Review

The planning phase is comprised of a set of steps to follow during the review. The initial step in planning stage is the identification of the need for the review followed by specifying the research question(s). After research questions are formulated, a review protocol is developed by defining the basic review procedures to conduct the SLR. Lastly the review protocol is evaluated by the supervisor of this thesis. The stages associated with planning the review are presented in the following sections, adopted from Kitchenham & Charters (2007).

3.2.1 Identification of the Need for Systematic Review

The need for undertaking a systematic review arises from the interest to summarize existing scientific knowledge about software development methods and practices used in startups. As stated in chapter 1 of this thesis, there is an identified gaps in software development in startups. Further studies are needed to explore the state-of-art on software startup research. In order to fill this gap, this SLR is needed to extract and analyze software development methodologies and work practices used in startups.

3.2.2 Research Question

The following the research questions will be answered in this thesis:

RQ1. What is currently known about the software development methodologies in software startups?

RQ2. What are the software development practices used by software startups reported in the existing literature?

The research questions are formulated in such a way that methodologies and practices related to software development are studied based on literature. The purpose of the first research question (RQ1) is to identify and extract software development methodologies used by startup companies. Issues related to utilized development methodologies and their characteristics with respect to startup environment will be answered by RQ1. The second research question (RQ2) focuses on extracting different reported software development practice that are used by software startups. Relevant information related to the usage of development practices in startups are also extracted from each primary study.

3.2.3 Developing Review Protocol

The review protocol describes the set of procedures that will be followed to undertake a specific systematic review (Kitchenham & Charters, 2007). It is a detailed plan to obtain relevant study material to answer the given research questions. The predefined review protocol is needed to reduce the possible biases of the researchers. The protocol is developed based on the review process described in the guidelines for performing the SLR which is adapted from Kitchenham & Charters (2007):

- Background information of the SLR
- Formulate the research questions of the review.
- Defining the search strategy, search terms and databases used for search
- Details of selection criteria (inclusion and exclusion criteria) and selection procedures to be applied.
- Checklists and procedures for the quality assessment of the individual studies.
- Defining data extraction strategy to obtain required information from each primary study.
- Defining data analysis strategy- what technique is used to analyze and summarize the results of the primary studies.

3.2.4 Search Strategy

The aim of the search strategy is to find as many studies relating to the research questions as possible. In order to achieve this, a search strategy is developed and applied to several electronic databases. To get an overall idea about the quantity of the articles and later to formulate search strings, a pilot search is performed.

Pilot Search

Kitchenham & Charters (2007) has proposed to perform pilot searches for identifying the potential studies using the search strings and resources defined in the review protocol. To get an overall idea about the available number of the literatures, a pilot was performed on Google Scholar. Google scholar is chosen for the reason that it includes literatures from a diversified fields of study. The pilot search was applied on Google Scholar with default options using the input keyword 'software development in startup' without quotes.

A result of 189,000 were found that include articles, books, patents, citations, etc. Then the keyword was modified by placing quotes around the keyword and running the search again, after which 107 hits were found. This indicates that "software development in startup" as a single concept has brought considerably less interest than software, development and startup as distinct keywords. It also indicates that using "software development in startup" does not provide enough amount of articles for our search. Based on this, separate keywords are applied to Google Scholar. The keyword 'software startup' without quotes was applied for the search and 258,000 results were found.

To refine the search results, the word 'software development' without quotes are added along with 'software startup'. As a result, a total of 183,000 hits were found. At the next step, the word 'methods and practices' without quotes is added to the previously used string, and provided a result of 58,800 papers. Finally, on the removal of a duplication of the word 'software', 50,200 results were found. The outcome indicates that separate keywords provide more search results than a single term and the author learnt that there are considerable amount of literature on the topic.

Based on the results from Google Scholar pilot search, the strings are modified again by adding synonyms and related terms for each concepts and applied to the actual selected databases. Google Scholar searches across resources like articles, books, theses, and abstracts etc. (Andrey, Uolevi & Kari, 2011) which are not related to software engineering. As a result, search keywords are applied to various databases which are considered to contain most of the studies in the software engineering discipline to get more precise and relevant result. These selected database include IEEE, Scopus and Web of Science. While applying the search, refining at databases is applied to our pilot search to get more related studies to the topic. The refining criteria includes selection by year (articles published between Jan 2006 and Dec 2016), subject area (Engineering; Computer Science and Business, management & accounting) and language (English). The pilot search results found in each database are presented below in Table 2.

Table 2. Results from pilot search.

Search Keywords	Database	Result (First hit)	Result (After applying selection criteria)
(Startup* OR software startup OR software start-up OR high-tech venture OR high-tech start-up OR IT startup OR lean startup) AND (Software Development OR development model OR software engineering) AND (practice* OR method* OR process* OR strategy OR approach)	IEEE Xplore	369	240
	Scopus	395	261
	ISI Web of Science	500	151
Total		1264	652

The pilot search from the three databases indicates that the addition of synonyms and related terms to each keywords resulted a more relevant and related studies. Learning from the results of the pilot searches, additional search criteria was defined for the actual search strings. This includes addition of synonyms, similar concepts, related terms and alternative spellings for each core concept in defining the actual search strings. Advanced search mechanisms using search operators (such as OR, AND) are also applied to each selected databases. The actual search strings and databases are discussed below.

Search Strings

Kitchenham & Charters (2007) proposed to use PICOC (population, intervention, comparison, outcome and context) approach to frame search strings. Population refers to very specific population groups to be studied. In this study, the population is represented by 'software startups'. Intervention refers to software methodology, tool, technology, procedure that addresses a specific issue. Therefore, the intervention in our context is "software development". Comparison (referred to as the "control) is the "methodology" or "practice" which is addressed in the software development activities. Outcome represents the effects of using a specific method or technique, whereas context represents the condition in which the comparison takes place (e.g. academia or industry) (Kitchenham & Charters, 2007). However, the outcome and context aspects are excluded from the search string structure for the reason that the effect of using a methodology or practice is not covered in this thesis. Context is not also used for search string formulation since the conditions of the startups are not important to answer research questions.

Based on Rumsey's (2008) guideline, synonyms, broader or wider concepts, related terms, alternative spellings and different parts of speech for each core concept are used in defining the exact search keywords. The core concepts, representing population, intervention and comparison, are derived from our research questions. The following main search strings are used in selection process: "Software Startup" AND "Development" AND "Practices". To increase publication coverage, related keyword for each respective strings are used in the search process. The final used keywords in the search process are listed in Table 3.

Table 3. List of search strings.

Criteria	Core Concept	Related Words
Population	Software Startup	startup, software startup, software start-up, high-tech venture, high-tech start-up, IT startup, lean startup
Intervention	Software development	software development, engineering, implementation, build
Comparison	Practice	practice, methodology, method, process, strategy, approach, model

Databases

Databases are electronic sources that are used for acquiring relevant studies. The search string is applied to each database that is relevant to software development methods and practices in startups. Three scientific databases are selected considering their coverage and use in the domain of software engineering, and their ability to handle advanced queries, following the example of Paternoster (2008) and Barney, Petersen, Svahnberg, Aurum & Barney (2012). The selected scientific databases includes:

- IEEE Xplore
- Scopus
- ISI Web of Science

3.2.5 Selection Criteria

Selection criteria are the procedures used to identify and select the most appropriate and relevant primary study materials from the searched results. These selection criteria are aimed to be inclusive with respect to the number of retrieved papers, which are related to software development in startups. The process of selecting relevant studies is performed using the selection criteria which include the inclusion, exclusion and quality assessment criteria. These criteria are designed based upon the research question. The inclusion and exclusion criteria along with quality assessment criteria are presented as follows.

Inclusion Criteria

The inclusion criteria used was:

- The study should be written in English
- The study should be published between Jan 2006 and December 2016
- The study directly answers one or more of the research questions of this study
- The study should clearly state its focus on software startups
- The study should be available in full-text
- The study can be in the form of an experience report, applied engineering practices, development models or lessons learned.

Exclusion Criteria

The exclusion criteria used was:

- Duplicate articles
- Not written in English
- Papers written before 2006 and after Dec, 2016
- Studies not clearly focused on software startups
- Related to software startups but not software engineering perspective
- Studies related to established companies
- Studies related to non-software companies (biotech, manufacturing, electronics, etc.)
- Studies related to technicalities of startups (funding, algorithms, programming languages, etc.)
- Not peer-reviewed scientific papers (i.e. gray literature, books, presentations, blog posts, etc.)

3.2.6 Quality Assessment Criteria

In SLR, it is considered critical to assess the “quality” of primary studies besides the general inclusion/exclusion criteria. Quality assessment criteria is used as an instrument to provide more detailed inclusion/exclusion criteria and weight the importance of individual studies when results are being synthesized (Kitchenham & Charters, 2007). The quality assessments are measured based on checklists that need to be evaluated for each study. According to Kitchenham & Charters (2007), there is no consensus on the definition of quality. However, in order to assess the quality and to reduce any form of bias, the author of the thesis applied quality criteria checklist. Five questions were formulated to assess the quality of selected primary studies. The checklist of questions are structured to ensure that the selected papers address the research questions. Based on the questions below, all the primary studies addressing the questions (qualities) were chosen. All papers should satisfy the inclusion criteria and quality checklist to be considered in the review. The paper's full-text were studied and analyzed thoroughly to answer the questions. Questions to evaluate the quality of a paper is presented in Table 4.

Table 4. Checklists for the quality assessment of the individual studies (Adopted from Sheuly, S. (2013)).

Quality checklist question	Answer
1. Is the objective of the paper clearly mentioned?	1.YES 2. NO
2. Does a paper report a study in a startup company?	1.YES 2. NO
3. Does a paper address software development practice in startups?	1.YES 2. NO
4. Does the paper clearly describe the research methodology used?	1.YES 2. NO
5. Does the research paper define the study results? Are the results helpful to find the research questions?	1.YES 2. NO

3.2.7 Evaluating the Review protocol

The review protocol is used as an instruction for the actual implementation of the study, thus it is the most critical element of a systematic review. The evaluation process is essential to make the study transparent and of good quality. As a result, the review protocol was reviewed and verified by the thesis supervisor who is an experienced researcher in this field.

3.3 Conducting the Review

Following the acceptance of the review protocol, the actual literature review is conducted following each of the steps described in review protocol. The search strings and databases are identified from which the SLR is performed. Based on the strategy, the search strings are applied on the selected databases. Once potentially relevant primary studies have been obtained, selection criteria which include both the inclusion and exclusion criteria, are applied to check the relevance of the paper. The quality checks for identified primary studies are also conducted to check whether the quality criteria is met.

Following the quality assessment, data is extracted from the selected primary studies and stored in the defined data extraction forms. The data extraction strategy is followed to accurately record the information obtained from primary studies. The extracted data is then synthesized in order to report the results of the reviewed primary studies. This analysis of extracted data actually provide the answer to the proposed research question. While conducting the literature review, documenting the search process and the results is undertaken. The review is documented in sufficient detail for readers to be able to assess the thoroughness of the search and to be transparent, replicable and possible to reanalyze.

3.3.1 Selection Process

The selection process describes the actual implementation of selecting the literatures. The search strategy is applied according to the review protocol described in the previous sections. The selection process is a multistage process which involves different steps in selection to ensure the inclusion of any relevant papers. The process of selecting papers are presented in Figure 2.

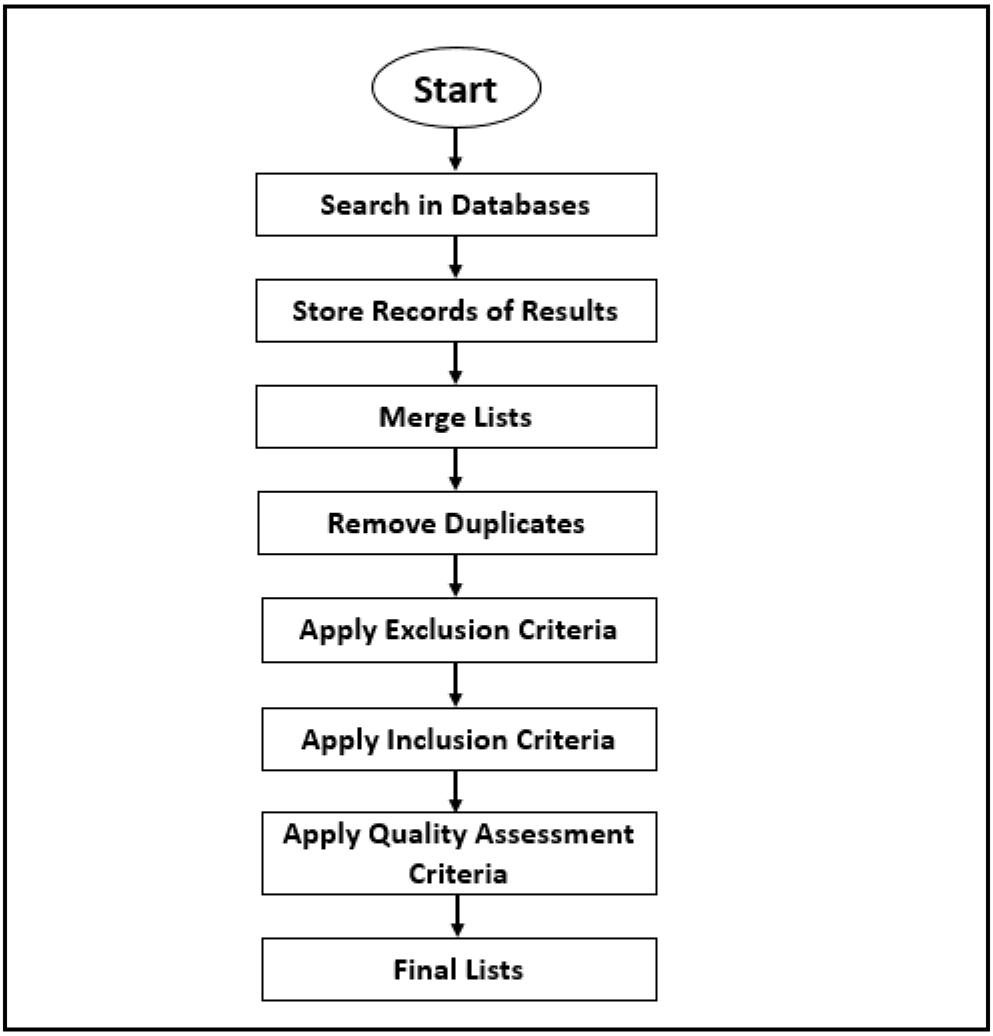

Figure 2. Workflow for selecting primary studies.

In order to select the papers we followed the workflow in Figure 2. The first step was applying search strings to each database that are selected for acquiring primary studies. The final list of search strings used in selection process are:

> ("early-stage firm" OR "early-stage company" OR "high-tech venture" OR "high-tech ventures" OR "start-up company" OR "start-up companies" OR "high-tech start-up" OR "high-tech start-ups" OR "high-tech startups" OR "high-tech startup" OR "startup company" OR "startup companies" OR "software startup" OR "lean startup" OR "lean start-up" OR "lean startups" OR "software startups" OR "IT start-ups" OR "IT start-up" OR "IT startup" OR "IT startups" OR "software start-up" OR "software start-ups" OR "software product startup" OR "software product startups" OR "software start up" OR "web startup" OR "web start-up" OR "web startups" OR "web start-ups" OR "internet startup" OR "internet start-up" OR "internet startups" OR "internet start-ups" OR "mobile startup" OR "mobile start-up" OR "mobile startups" OR "mobile start-ups") AND (develop* OR engineer* OR implement* OR build*) AND (software OR process OR model OR methodolog* OR method OR practice OR strategy)

The above tailored search strings are executed in the three databases depending on the syntax requested by each database. On applying the search string, a total of 5,151 articles were obtained from all the three databases (As of Oct 11, 2017). The number of publications retrieved from each selected scientific database are reported below in table 5.

Table 5. Selected databases and retrieved papers.

Database	Number of papers	Number of papers Excluded at databases	Remaining number of papers	Duplicates	Total
IEEE Xplore	190	64	126	100	1,249
Scopus	4562	3421	1141		
ISI Web of Science	399	317	82		
Total	5,151	3,802	1,349		

The next step was selection of papers which is performed in two steps, first at database level and then with the help of Refworks. Refworks is a web-based bibliography and reference management system that enables to create personal database by importing references from text files or online databases (www.refworks.com).

At database level, papers are filtered out using each database refining technique, such as filtering by subject area, publication year, document type and language. Based on the topic of this study, engineering and computer science subject areas are considered. Besides that the publication between Jan 2006 and Dec 2016 are included in our search. Document types like conference publications, journals & magazines, articles, book chapters, conference review and article in press are included. As a result, 3,802 papers are excluded and a total of 1,349 (126 from IEEE, 1141 from Scopus, 82 from Web of Science) papers are collected and considered for further selection.

After the execution of filtering at database level, list of results from each database are recorded and transferred to a reference management tool (Refworks.com) and the result lists from the each database are merged together with it. Then, removal of duplications is performed using the reference management tool which automatically detects duplicates based on metadata (author, publication year and title). A total of 100 duplicated items are removed from our list resulting in 1249 papers remaining in the list for further selection. The selection process and the results achieved at each stage are summarized in the figure 3 below.

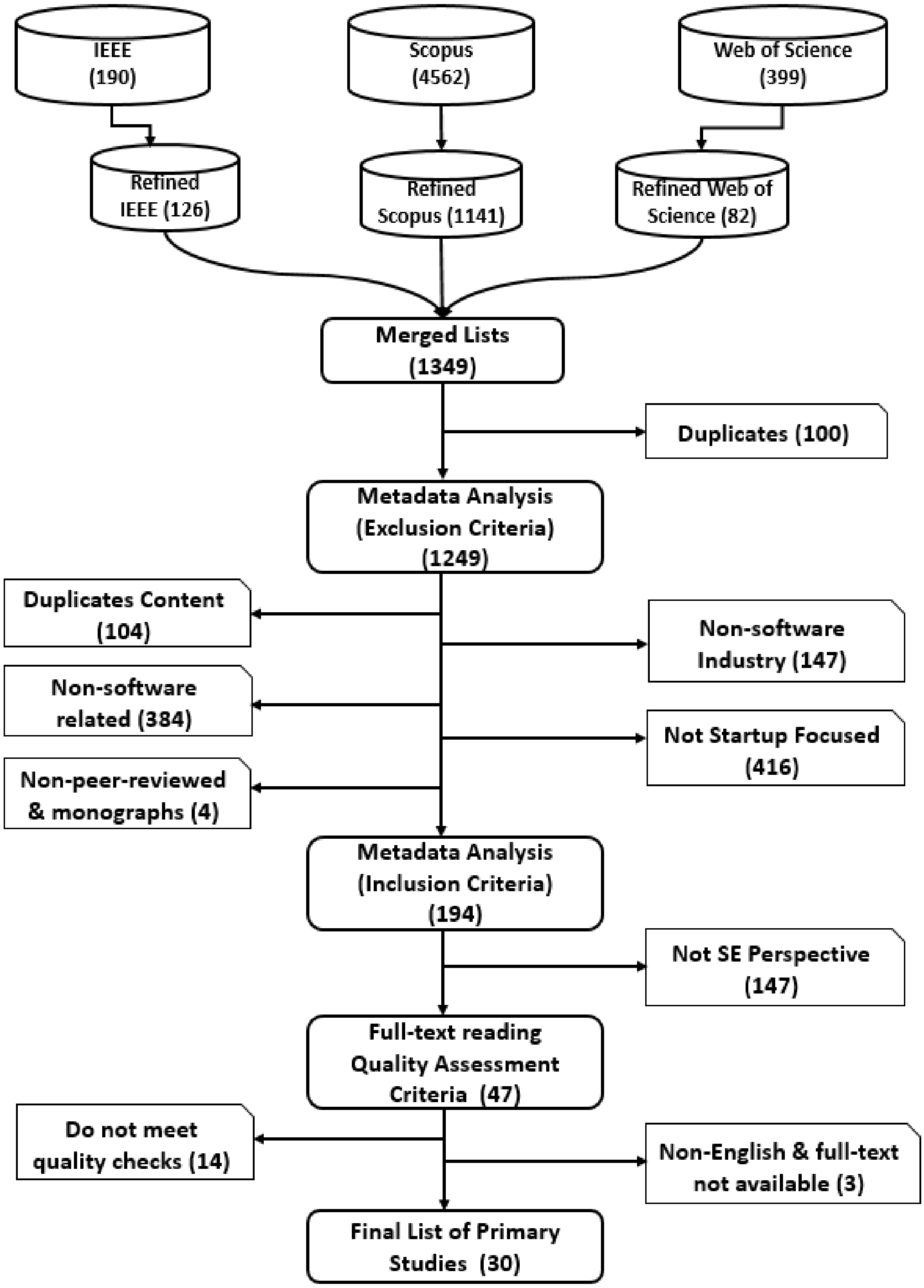

Figure 3. Selection process and the number of papers identified at each stage.

The next step was to apply the inclusion and exclusion criteria to remove irrelevant papers. The exclusion criteria is applied to the lists of papers. By analyzing the metadata (title, keywords, abstract) of each paper, irrelevant papers are identified that matched the exclusion criteria. A total of 1055 papers are excluded from our list due to matching the exclusion criteria, resulting in 194 remaining papers for further reading. Those excluded papers, for instance, are non-software industry, non-software related, not focused on startups. Non-peer reviewed papers are also excluded from the list.

A further selection was conducted to make sure that the selected articles are relevant to the goal of this research work. Therefore, inclusion criteria is applied. A more in-depth review of the abstract of each paper is performed to determine whether it matched our inclusion criteria. As a result, 47 papers are selected. The basis to include a paper is that it should clearly state its focus on software startups, and address software development in startups from software engineering perspective. For instance, papers with the focus on e-commerce, marketing, entrepreneurship, funding, business model, etc. are excluded at this stage. The rationale and amount of papers for exclusion are described below in table 6.

Table 6. Rationale for excluded papers.

Rationale	Amount
Duplicate at database level	100
Non-software industry	147
Not software related	384
Related to software but not startups (not focused on startups)	416
Related to software startups but not software engineering perspective	147
Not peer-reviewed	4
Do not meet quality checks	14
Full-text not available	2
Not in English language	1
Duplicates at title and abstract level	104
Total retrieved	1,349
Total excluded	1,319
Total included	30

The last step in the selection process was to apply quality assessment criteria on the papers to ensure that the selected papers are most relevant to answer the research questions. The quality assessment is conducted by reading the full-text of the paper. After reading the entire paper, relevant primary studies were chosen, eventually leading to the final set of 30 primary studies. As a result, these primary studies are considered for the final review.

3.3.2 Data Extraction Strategy

The data extraction strategy is formulated to collect all the information needed to address the research questions. Relevant data from each primary study were extracted by reading the full-text of the paper. The following data extraction process was conducted to extract and record the relevant data from primary studies. An Excel spreadsheet was prepared and data was recorded on the respective category from each primary paper.

The following data were extracted from the selected primary studies.

- Initially, general data including author(s), title, publication year and keywords are extracted from the primary study during the process of inclusion/exclusion criteria
- The research methodology used in the primary study
- The number of cases (startup companies) participated in the study
- The software development methodology used by the startup companies, to find answer for RQ1
- Any mentioned development practices used in each startup company, to find answer for RQ2

During the review, all relevant data was extracted and recorded using the data extraction spreadsheet. The extracted data was analyzed using appropriate data analysis approaches described below and finally report was produced.

3.3.3 Data Analysis

Data analysis involves examining and summarizing the results of extracted data from the included primary studies (Kitchenham & Charters, 2007). In this thesis, qualitative data analysis approach is used to analyze the extracted data. Qualitative data analysis involves integration of studies comprising natural language results and conclusions, where different researchers may have used terms and concepts with subtly (or grossly) different meanings (Kitchenham & Charters, 2007).Therefore, qualitative data analysis technique is used to answer the RQ1 and RQ2.

In the data extraction process, the main concepts related to software development methodologies and practices in software startups are identified from each primary study, using the original author's terms. Those main concepts were then organized in tabular form to enable comparisons across different studies so that findings will be extracted for the research questions. In summary, data analysis is achieved by:

- Identification of software development methodologies used by startups from each study
- Documentation of a set of reported development practices in startups from each primary study
- Elaboration of gaps on the applicability of working practices in startup contexts, if any.

In the next section, results related to identified development methods and work practices and their application in the startup context are presented.

3.4 Reporting the Review

The final phase of a SLR involves reporting the results of the review. The results of the SLR are carried out by writing up the findings of the systematic review. During the review, relevant primary studies are selected and relevant data were extracted from selected primary studies using the data extraction forms. The collected data was synthesized using appropriate data synthesis approaches and finally the results are reported in chapter 4.

4. Result

From an initial selection of 1249 studies, 30 primary studies passed the selection process and were included in the review. Each study was thoroughly reviewed and analyzed to answer the research questions of this thesis. Section 4.1 describes the overview of the studies. Then section 4.2 presents the analysis of the results.

4.1 Overview of Primary Studies

The following subsection discusses the overview of the selected primary studies with major classifications. The primary studies varies in their research approach, which includes empirical studies, case studies, evaluation research and experience reports. Each primary study has given a unique study ID (for example, P1, P2 …. Pn) for easy referencing in this SLR. List of primary studies is attached to the appendix B.

4.1.1 Publication Trend

We have considered the primary studies, which are published between years Jan 2006 to Dec 2016. Figure 4 shows the distribution of papers published on software development methods and practices in software startups in each year during 2006-2016.

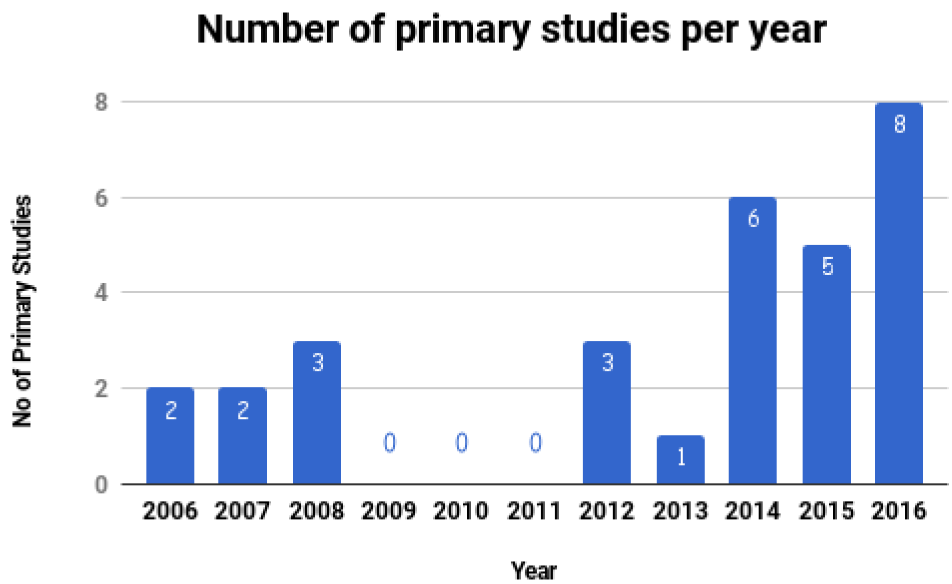

Figure 4. Publication distribution by Year.

From the distribution of studies, it was observed that most of the studies (77%) were published after year 2012. Therefore, it can be argued that there is a growing interest in the area of software startups. For some reason no publication from years 2009-2011 passed the screening process of the primary studies. The exception is that, one study (Taipale, 2010) was highly relevant to this SLR but excluded because the full-text is not available.

4.1.2 Research Methods

Thirty primary studies use various research methods and techniques i.e. case studies, surveys, design science research, grounded theory, systematic mapping study and mixed methods as shown in Table 7.

Table 7. Classification of primary studies based on research method.

Research Type	Number of Primary Studies	Percentage	Reference
Case Study	14	47 %	[P2] [P4] [P8] [P10] [P12] [P16] [P21] [P22] [P24] [P26] [P27] [P28] [P29] [P30]
Survey	6	20 %	[P1] [P3] [P11] [P15] [P19] [P20]
Design Science Research	1	3 %	[P5]
Grounded Theory	3	10 %	[P6] [P9] [P25]
Mixed Research (Survey + SLR) + (Case study + SMS)	2	7 %	[P7] [P14]
Systematic Mapping Study	4	13 %	[P13] [P17] [P18] [P23]

Fourteen primary studies (57%) were conducted as a case study which is the highest among the categories. Out of these studies, the author of one study [P22] considered the method as 'startup stories' rather than case study due to the little amount of empirical data collected. Six primary studies (20 %) are survey-based, three studies (10%) used grounded theory while one primary study (3 %) used design science research approach.

These primary studies conducted interviews and observations techniques for data collection (Corbis, J., Strauss, A. (1990)). Four primary studies (13%) used systematic mapping to review and analyze the research papers published earlier. The results from these SMS papers are used to review and analyze literatures that we are not included due to their publication year. Two primary studies (7%) used a mixed research approach. One of those studies used a mix of survey and literature review approach whereas the other paper used both case study and systematic mapping study.

The result shows that case study is dominating research approach. Even though the scientific relevance of a study is not determined by the research type, a case study with actual startups have a larger potential to provide useful results. A research method that facilitates investigation in realistic contexts generally contributes high relevance to the study (Klotins, 2015). Therefore, we can observe that high number of case studies in this review contributes high relevance to the results of the study.

4.1.3 Research Focus

The primary studies are categorized into three groups based on the research focus of the paper: as software development, process management, and managerial/organizational. The classification of research focus is based on the definitions and categories adopted from Paternoster et al. (2014). Number of primary studies to the corresponding category of research focus is shown in figure 5.

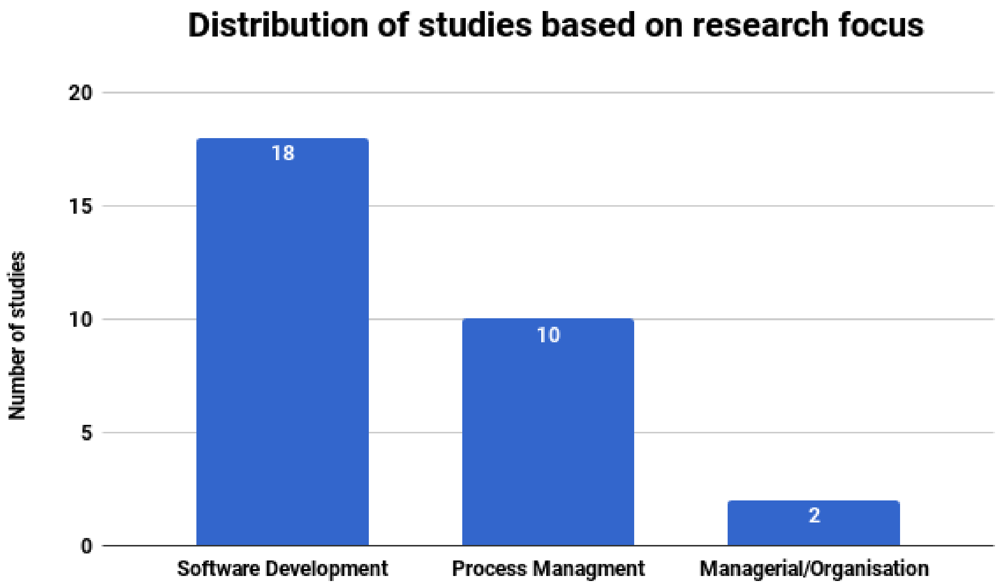

Figure 5. Distribution of primary studies based on focus areas.

Software development: The category includes engineering activities used to write and maintain the source code. 18 out of 30 primary studies indicated that their research area is focused into software development in startups. For instance, study [P6] discuss the software development strategies in startups. This study introduced a strategy to release the product as quickly as possible, speeding up the development through low-precision engineering activities in the startup context. Study [P28] has a focus on the adoption of agile methodology and Extreme Programming (XP) principles by a software startup, as a solution for problems related with traditional software developments.

Process management: The category includes engineering methods and techniques used to manage the development activities. 10 out of 30 primary studies has a research focus on process management that are used by startups in software production. Process management represents all the engineering activities used to manage product development in startups (Paternoster, 2014). Study [P3] has a focus on process management where it investigated the challenges of Lean Startup approach and customer development in building products and businesses in startups. Study [P9] has also focus on process management where it characterized the experiences of startup companies in developing processes to support their software development activity.

Managerial/Organizational: The category includes aspects that are related to software development, by means of resource management and organizational structure. Only 2 out of the total 30 primary studies are identified with a focus on managerial or/and organizational aspects of software development: [P7] and [P24]. Study [P7] identifies the critical factors for the success of startup companies in Israel. Whereas study [P24] proposes a method for personal capability assessment of each individual team member in a small agile teams using personal points. Individual primary studies along with their focus of research are attached in the Appendix E.

We can observe that majority of primary studies (28 out of 30 studies) are related to software development and process management in startups, which we considered fundamental for our research questions (software development methodologies and practices). This indicates that the selected primary studies have a strong contribution and high relevance to answer the research questions. Only 2 out of 30 papers have a focus on managerial or organizational practices, which are less relevance to answer research questions.

4.2 Analysis of Results

The following section discusses the results of SLR related to software development methodologies and practices used in software startups. These two aspects are most relevant aspects which are related to the research question.

4.2.1 Analysis of Development Methodologies

The list of identified software development methodologies and their distribution along with the references is given in table 8. Based on the data collected from the primary studies, our study indicates that agile and Lean Startup methodologies are the most widely used development methodologies across startup companies.

Table 8. Frequency of methodologies and primary studies.

Methodology	Frequency	Reference
Lean Startups	11	[P2][P3][P4][P6][P10][P11] [P12][P15][P19][P22][P27]
Agile Methodology	6	[P5] [P21] [P24] [P25] [P26] [P30]
Prototyping	2	[P1][P14]
XP	2	[P9][P28]
Ad hoc Management	2	[P10] [P16]
Model Driven Development	1	[P8]
Waterfall	1	[P9]
Rational Unified Process (RUP)	1	[P9]
Distributed Scrum	1	[P10]
Hyper-agility	1	[P20]
Iterative/incremental	1	[P25]
Tailored Agile	1	[P10]

Other methodologies which are identified from primary studies are prototyping, Model-Driven Development (MDD), Waterfall, RUP, and Ad hoc methodology. There are also a comparison studies [P9], [P10], [P12], [P25] among several startup cases which autonomously use different methodologies. Two of the studies [P7] and [P29] have not specified any of software development method, but the primary studies are included in review for the reported development practices (RQ2). Out of the 4 reviewed systematic mapping studies, all of the reviews [P13], [P17], [P18], [P23] favor agile and Lean Startup approaches for software startup companies. The list of primary studies along with their reported methodology is given in Appendix C.

In most of the cases, due to the flexible nature of startups, development processes do not strictly follow these specific methodology, but opportunistically select practices and customize according to their context [P5], [P9], [P16], and [P23]. Development methods are mostly tailored to the specific features that characterize each development context and startup behavior [P23]. For instance, [P5] describes that applying the principles of a specific methodology is difficult at early stage of a startup. As a solution, the author suggested a new model called Early Stage Software Startup Development Model (ESSSDM).

Coleman [P9] also explains that process tailoring is made by dropping some of the practices of a specific methodology and adding some new practices from other methods which reflected their own particular operating environment. This way of tailoring processes help the startup company to suit with the company's business strategy. For instance, the startup cases under the study of [P9] adopted three different methodologies; Waterfall, RUP, and XP. But all of these case startups tailored the methodology they were using, by excluding some of the practices contained within it and adding some new practices which fits to their particular context.

Lean Startup Methodology

Lean Startup is a methodology which supports the identification of the most risky parts of a software business and produce a Minimum Viable Product (MVP) to systematically test and plan the modified version of next iteration [P23]. Primary studies indicated that Lean Startup methodology is one of the most commonly used development methodology among software startups. A total of 11 primary studies have discussed Lean Startup approach in startup context ([P2-P4] [P6] [P10-P12] [P15] [P19] [P22] [P27]). In most of the primary studies, authors refer to Eric Ries (Ries, 2011) while explaining Lean Startups principles in general. All of the 11 primary studies who described Lean Startup method are published after year 2012. The concept of Lean Startup was first proposed by Eric Ries in 2011 (Ries, 2011). Lean Startup methodology is preferred among software startups for its promising principles of customer development [P2] [P3] [P6] [P22], development of Minimum Viable product (MVP) [P2] [P3] [P6] [P12], waste reduction [P6], its highly evolutionary development approach and team empowerment [P6].

Batova & Clark [P3] defines customer discovery as a process in which business owners develop hypotheses about their business models and then validate or invalidate those hypotheses by researching potential customers. This supports the idea of customer development which aims to come up with new viable products and services under conditions of extreme uncertainty. It involves the production of functioning prototype guided by customer feedback [P3] [P6]. With this tangible prototype, it helps to approach anticipated customers. Customer discovery leads into validation of the product idea or prototype and, if proven, creation of the product [P19] [P22]. Therefore, this results a strong customer development and relationship [P6].

Another principle of Lean Startup methodology is the concept of MVP (Ries, 2011). Batova & Clark, [P3], describes MVP as “a prototype of the product or service to be developed that has the smallest number of features for which customers would be willing to pay”. MVPs are developed for hypothesis thesis and after validation from the customer, it leads to creating the actual product. MVP is the most pared down version of a product that can still be released and used by customers. A MVP is a product which is developed with sufficient features to deploy the product for early adopters. The final, complete set of features is only designed and developed after considering feedback from the product's initial users (Ries, 2011). Primary studies [P2] [P3] [10] [P19] [P22] described that startups have practiced the MVP principles of the Lean methodology. Duc & Abrahamsson [10] made a conclusion that MVPs could be useful for a software startup as a design artifact, a boundary spanning artifact and a reusable artifact, as a result the startup the benefits in the development of final product, such as increase feedback quality and reachability.

Waste reduction principles were introduced by Ries (2011) which aim to reduce waste in production processes and making these processes more efficient by eliminating unnecessary research and development. Giardino [P6] also mentioned that Lean Startup methodology is used by startups for the benefit of waste reduction processes in Lean Startup methodology. Before developing products or even prototypes, startups should find out if they have actual customers and understand what these customers want [P3]. As a result startups avoid wasting time in building unwanted functionality and preventing exhaustion of resources for the startup companies [P2] [P3].

Agile Methodology

A total of 6 out of 30 primary studies in startups have used agile software development methodology. Agile methodology is an umbrella for other many methodologies which also took the attention of several authors. For instance, distributed scrum [P10], XP [P9] [P28], hyper-agility [P20], iterative & incremental [P25], and tailored agile [P10] are also practices explored in several startup cases. Primary studies [P5] [P21] [P24-P26] [P30] mentioned that startups adopt agile development methodology as in a general category. Agile methodology is described as a preferred methodology for startups for the reason of shorter development time [P21] [P25], their smaller company size and easiness to manage teams [P24]. Agile methodology provides satisfaction to the customer with early and continuous delivery of software [P25] and improves management of the development process with better customer relationship [P26]. The flexible principles of agile allow requirements and solutions to evolve through self-organizing collaboration between cross-functional teams [P30]. Agile methodology also promotes adaptive planning which enables continuous improvement or response to change [P30], and for being solution-focused [P5].

McCaffery et. al. [P21], states that startups are increasingly attracted to agile methodology with the promise of shorter development schedules and greater delivery flexibility and support to their business goals for small software companies. They also explain that agile methodology is preferred by startup companies to incorporate into any potential process assessment method. Taylor et al., [26], indicates that using agile methodology in startup companies improves management of the development process and customer relationships. The study also indicates that agile methodology can lead to more enthusiastic development teams. However, agile methodology do not suit to every problem domain in startup cases. Startups need to define the level of agility and tailor the agile process management for different software development contexts.

On the other hand, McCaffery et. al. [P21] explain that agile and more traditional approaches struggle to get used by software startups due to an excessive amount of uncertainty and high time-pressure. It is also mentioned that startup companies prefer lightweight methodologies that allow companies to pick and tailor practices as they promote reactiveness and allow rapid changes in the product. The author(s) conclude that development processes in startups are evolutionary in nature, and the product is obtained by iterating and updating an early prototype following the customer feedback (Paternoster, 2014). Another primary studies [P13] and [P10] argue that agile methodologies have been considered the most viable development methodology for software startups, for the reason that agile methodologies embrace changes rather than avoiding them, allowing development to follow the startups' business strategy. Another reason for startups to prefer agile approach is that it offers fast releases with an iterative and incremental approach, and therefore shortens the lead time from idea conception to production with fast deployment [P13].

Prototyping

Prototyping is another used agile methodology for software development in startups. Two primary studies [P1] [P14] has stated that evolutionary prototyping is a common way of development method among their cases. According to Giardino [P14], startups prefer to build an initial prototype and afterwards build the real product after validating the product in the market. This approach enables startups to find the proper product/market fit as fast as possible. Based on the prototype, developers uses the chance to get feedback from the users, which then helps to identify new functionalities and improvements. All of the case startups under [P14] adopted prototype-centric development approach. They were highly involved in developing a throw-away prototypes or evolutionary prototypes. Throw-away prototypes are rapidly constructed on paper/software to visualize the business idea whereas evolutionary prototypes are construction of the launching product through a detail and gradual planning [P14]. Therefore, the two prototype focused primary studies described that prototype development approach is used to obtain fast user responses and quick product validation, followed by building a functioning prototype and iterate it over time. According to our study, other development methodologies used by startups include Scrum [P10], Model-driven development [P8], waterfall [P9], hyper-agility [P20], Rational Unified Process [P9], Iterative/Incremental [P25], and tailored agile [P10], and ad hoc management [P10][P16] approach.

The term "prototype" is often used in startup context as an interchangeable term with MVP. From software engineering perspective, a revolutionary prototype shares a lot of characteristics with a MVP. But they also differ that a MVP has enough value that people are willing to use it or buy it initially and it demonstrates enough future benefit to retain early adopters (Ries, 2011). Whereas revolutionary prototype is a product build based on documents, which describes the requirements, and then iteratively revised with the customer to ensure that both parties share the same understanding of requirements and status of advancement (Sharp & Hall, 2016). But both products are used to facilitate product design, to bridge communication gaps and to facilitate cost-effective product development activities.

Extreme Programming (XP)

XP is one of the agile methodology practiced by software startup companies. Two primary studies [P9] and [P28] explains that their startup cases adopt XP methodology. Coleman [P9] states that XP is more flexible and developer-centered development method with the advocacy of self-empowered teams and shared ownership. It is also associated with an "embrace and empower" style of management in software startups. Tingling [P28] describes that XP is the most widely used agile development methodology, which has a generative set of guidelines that consisting of twelve inter-related principles. Some of these principle include 40 hour work week, coding standards, collective code ownership, continuous integration, pair programming and refactoring etc. According to Tingling [P28] the extent of adopting the principles of XP in startups is affected by temporal conditions and institutional maturity of the startup, and both management and developer cultures are also important determinants. Tingling concludes that all extreme programming principles were not adopted equally and were subjected to temporal conditions.

4.2.2 Analysis of Development Practices

From 30 primary studies, a total of 95 work practices are extracted. The work practices are categorized into three categories, those includes development practices, process management practices and managerial or organizational practices. The categorization of work practices is based on the aspect of their research focus, presented in section 4.1.3. In the rest of this section, the common and frequently used work practices are presented. The whole list of work practices identified in this study is attached in Appendix D.

Software Development Practices

Software development practices are defined as engineering activities used to write and maintaining the source code [P23]. Some of the common development practices among software startups include evolutionary prototyping and experimenting via existing components, continuous value delivery, use of easy-to-implement tools to facilitate product development [P13], develop only what is needed for now [P11], goal-driven rapid development of technology rather than process directed [P20], use of open source solutions, Refactoring, Use of key performance indicators to assess the consumer's demand [23], and pair programming [28]. Use of evolutionary prototype is the most common development practice among startups to focus on implementing a limited number of suitable functionalities (minimum viable product) and to launch the product on the market as quickly as possible [P1][P4][P6][P10][P13][P15][P18][P27].

Process Management Practices

According to the classification schema, engineering methods and techniques used to manage the development activities in the startup companies are categorized into Process management practices [P23]. The summarized and commonly used process management practices reported to be useful in startups are customer development and use of customer feedback [P1] [P3], daily scrum, kanban [P2], process tailoring [P1] [P9] [P26], performing incremental process improvement actions, adopting short and light software process improvement & release management process model [P21], pivoting practices [P1] [P22], and incremental and iterative establishment of software processes [P29].

Software process improvement actions and process tailoring practices are the most commonly used process management practices explored in this review. The author of [P16] argues that adoption of release management process helps to improve the workflow of a startup company. Study [P16] mentioned that ad hoc mechanism is used to manage development processes at the beginning, but after the adoption of process improvement actions, to the startup has improved the management problems. Coleman [P9] explains that all of the case startup companies tailored the used process model, by dropping some of the practices and adding the new ones which suits to their environment, and then indicating process tailoring is an influential factor for success in startups.

Managerial or Organizational Practices

Managerial or organizational practices include aspects that are related to software development, by means of resource management and organizational structure. These includes core team expertise, and diversified team management, careful selection of personnel [P7], managerial experience [P9], building small omni-functional teams with no functional boundaries [P20], focusing on team building [P22], empowerment of team responsibilities and their ability [P23], and using methods for personal capability assessment of each individual team member [P24].

5. Discussion

This section of the thesis discusses the findings of the SLR and answer the research questions defined in the earlier chapter.

RQ1. What is currently known about the software development methodologies in software startups?

Based on the analysis of primary studies, it is revealed that agile and Lean Startup methodologies are the popular software development methods used by software startups. Other methods used by startups include XP, Prototyping, Scrum, MDD, Waterfall, RUP, Ad hoc management, Iterative/incremental, and tailored agile method.

Software startups, particularly at their early stage, apply mixed development approaches and practices due to a rapidly changing environment. As the result of constraints in resources, personnel and time, startups change their development methodology in order to fit with the existing situation. The results indicated that the important thing for project managers in startup is to overcome the challenges of uncertainty and resource scarcity, and then getting the customer satisfaction. The project managers choose lightweight methodologies which allow more flexibility to adopt tailored practices. As a result, the development team is able to react fast in changing the product according to business strategies.

In startup context, where the environment is chaotic and unpredictable, it is difficult to accurately predict the risks and the required practices to develop a product. Similarly it's difficult to follow the principles of a specific development methodology. Instead, flexible and reactive methodologies which favor changes in making decisions, development processes, and learn from failures are most suitable in the startup companies. This is a key factor in choosing and adopting effectively a development methodology in a startup company. Developers should have the freedom to choose development practices quickly, and change or stop immediately when developments goes wrong, fix the approach and learn from previous failures. For this reason, methodologies tailored from agile and Lean Startup methodologies are considered better options to fit with the culture and needs of startup companies.

B) RQ2. What are the development practices in startups which have been addressed by the existing literature?

The primary study shows that there are 95 different software development practices used by software startups. The most common practices among startups include customer development, process tailoring, use of prototypes or/and minimum viable product, continuous value delivery, and use of easy-to-implement tools to facilitate product development. The SLR also reveals that software development practices are not fully adopted and documented in startups. Instead startup companies pick development practices which suits to their startup environment and most of the process are tailored. Empowerment of team members are considered productive managerial/organizational mechanism for better responsibilities and improved expertise.

Customer development is vital activity in startup companies. In order to survive, startups need to be customer focused. Studies indicate that discovering customers is one of the top priority in startups to keep them in business. Besides that ensuring customer satisfaction is another important factor, since this creates future possibilities to work with the same customer and/or recommendations with others. Similarly, understanding the customer needs is crucial. Before starting to work with the customer, it is important that the development team understand requirements of the software product.

6. Conclusion

The results shows that agile and Lean Startup are the most common development methodologies used by software startups. The reason for this is that they allow flexibility in the process, adaptability to frequently changing environment, easy way of communication with the customer and fast reachable to the market with low cost and fast delivery. But it should be noted that startups do not strictly follow methodology principles, due to limited resource and time pressure. Instead they tailor processes according to their situation by dropping some of the features in the existing methodology and adding new practices from other methods.

In some startups, it is even difficult to adopt a certain methodology in a structured way, rather they apply ad hoc mechanism to manage development processes. It is also noted that adoption of a software development method and practices are mostly dependent on the experience of the owners or/and project managements, the maturity level of the startups, team size, availability of resources. In general, software development methodologies in startups are informal, tailored and very light.

The result shows that there is an increasing interest in software development methods and practices in startup companies. The publication trend indicates that publications on the topic are increasing in recent years. The research method categorization proved that the highest research approach used to study software startups is case study. The research focus classification also indicates that practitioners are more interested in studying the software development and process management aspects of a startups rather than managerial or organizational structure of a startup companies.

Studies have shown that there is scarce of studies on software development practices. The engineering practices in startups are not fully explored. The practices are not also documented in startups because of less focus on the processes and documentation. Klotins (2015) explains that existing literatures on engineering practices in startups are not transferable to practitioners. Paternoster (2014) has also proved that the existing state-of-art is scarce.

6.1 Limitations & Validity Threats

This SLR was conducted in a systematic manner to cover all possible studies relating to software development methods and practices from startup context. The main limitation of this SLR is regarding to conducting the search. The following are the list of limitations of the SLR that should be taken into consideration:

- The review did not include books about software development in startups.
- The review only included papers that were accessible in the three digital databases: IEEE, Scopus and Web of Science.
- The review only included articles that were available in full-texts.

Validity threats are one of the major factors that influence the accuracy of research negatively. For this reason, it is necessary to identify and handle these threats to make the review results reliable and transferable to others. This study has several threats to validity, which are categorized into the following four categories: Investigation bias and publication bias, Threats to study selection, and Threats to data extraction process and its results.

Investigator bias

Since the review was conducted by an individual person, there is a higher potential of threats to validity in comparison with review conducted by several researchers. In order to reduce this bias, the author execute some tasks twice to ensure the quality of the work. For instance, reading the title and abstracts was conducted twice to minimize possible mistakes by the author.

Publication bias

Publication bias is a common bias in systematic reviews which is related to the problem that positive results are more likely to be published than negative results (Kitchenham & Charters, 2007). This bias can be observed also in this SLR which includes only a few papers on failed startup cases. Most of the studies are performed in collaboration with successful startups. However, this is not considered a major threat as this bias matches to the study aim, identifying the development practices in startup: rather than analyzing work practices with regard to their performance. The search keywords used in this SLR cover a wide range of startup synonyms. Some papers has used different names to refer startups. For example, Very small companies as in primary study [P25]. In order to mitigate this problem, the author has used the definition of Ries (Ries, 2011) for startups. Primary studies are included for the review when case startup has shared the characteristics of startups defined by Ries.

Threats to study selection

In order to minimize the threats to the selection of primary studies, the search strategy including inclusion/exclusion criteria was defined in the review protocol and used to cover as many studies as possible; a pilot search was performed to formulate the search strings which was then followed by the actual search; search string was applied in the most well-known databases in the field of software engineering. In addition, the titles and the abstracts were read several times to include the right studies. Applying these methods helped to minimize the threats to study selection. The risk of excluding relevant primary studies is further reduced by the use of a wide range of search strings on multiple databases, which cover the majority of scientific publications in the field. However, it was not possible to retrieve the full-text of 2 papers: neither available in online catalogs, nor get response on request from the authors. The selection of relevant primary studies can be further biased by the personal opinion of the author executing the process. To mitigate this threat, the study selection procedure was defined and documented as a rigid protocol for the study selection. In addition to that the exclusion rationale for each paper was explicitly recorded, providing clear evidence from the paper to support the decision.

Threats to data extraction process and its results

Another validity threat is related to data extraction phase. The author designed data extraction form and data extraction process while designing the review protocol and followed that process to record relevant information from primary studies. This procedure helped to minimize the bias related to data extraction process.

6.2 Future Work

In this SLR, software development methods and practices in startup companies are identified based on existing literatures. More empirical studies are needed to strengthen the results of this study. Due to the scarcity of studies, and inadequate level of reporting rigor in the primary studies, we believe a further study with high rigor research approach and with the focus on software development methods and practices is recommended to transfer the state-of-art to other startup practitioners.

Moreover, it is needed to evaluate the effectiveness and suitability of each development methodology and practice for different software startup situations (i.e. early stage, or established startups) and environments. Startups change their development process due to a rapidly changing environment. This hints that a research topic what process is suitable for different startup situations. It is also needed to evaluate the effectiveness of each methodologies and practice on the performance of startups. The explored practices are the commonly used practices, not the best or bad practices. For that reason, it is needed to evaluate the impact of each practices on the startups and categorize them as ‘good’ or ‘bad’ practices in the startup context.

References

Awad, M. A. (2005). A comparison between agile and traditional software development methodologies. *University of Western Australia*.

Atkinson, S., & Benefield, G. (2013). Software development: why the traditional contract model is not fit for purpose. *46th Hawaii International Conference,* USA, 4842-4851.

Barney, S., Petersen, K., Svahnberg, M., Aurum, A., & Barney, H. (2012). Software quality trade-offs: A systematic map. *Information and Software Technology, 54*(7), 651-662.

Batova, T., Clark, D., & Card, D. (2016). Challenges of lean customer discovery as invention. *IEEE ProComm, International Professional Communication Conference*, Austin, USA. doi: 10.1109/IPCC.2016.7740514

Berrocal, J., Garcia-Alonso, J., & Murillo, J. M. (2016). The Fall and Rise of NimBees. *42th Euromicro Conference on Software Engineering and Advanced Applications*, Limassol, Cyprus. doi: 10.1109/SEAA.2016.27

Björk, J., Ljungblad, J., & Bosch, J. (2013). Lean product development in early stage startups. *Chalmers University of Technology - Software Engineering Division*, Göteborg, Sweden.

Blank, S. (2006). The four steps to the epiphany: Successful strategies for products that win. *Publisher: Cafepress.com.*

Blank, S. (2011). Embrace failure to start up success. *Nature News, 477*(7363), 133-133.

Budgen, D. (2015). What do we know and how well do we know it? Current knowledge about software engineering practices. *Software Technologies (ICSOFT), 10th International Joint Conference.* Colmar, France.

Carmel, E. (1994). Time-to-completion in software package startups. *Proceedings 27th Hawaii International Conference on System Sciences (HICSS),* 498–507.

Celar, S., Turic, M., Vickovic, L. (2014). Method for personal capability assessment in agile teams using personal points. *22nd Telecommunications forum TELFOR,* 1134-1137, IEEE.

Chapman, J. R. (2004). Software Development Methodology. Retrieved November 15, 2017, from http://www.hyperthot.com/pm_sdm.htm

Clarke, P., & O'Connor, R. V. (2012). The situational factors that affect the software development process: Towards a comprehensive reference framework. *Information and Software Technology, 54*(5), 433–447.

Clark, T., & Muller, P. A. (2012). Exploiting model driven technology: A tale of two startups; *Software & Systems Modeling 11* (4), 481-493.

Chorev, S., & Anderson, A.R. (2006). Success in Israeli high-tech start-ups; critical factors and process. *Technovation, 26*(2), 162–174.

Coleman, G., & O'Connor, R.V. (2008). An investigation into software development process formation in software start-ups. *Journal of Enterprise Information Management, 21*(6), 633 - 648.

Crowne, M. (2002). Why software product startups fail and what to do about it. *Engineering Management conference*, Cambridge, UK. pp 338-343.

Dande, A., Eloranta, V. P., Kovalainen, A. J., Lehtonen, T., Leppänen, M., Salmimaa, T., ... & Koskimies, K. (2014). Software startup patterns - an empirical study. *Tampere University of Technology. Department of Pervasive Computing. Report; 4.*

Duc, A. N., & Abrahamsson, P. (2016). Minimum viable product or multiple facet product? The Role of MVP in software startups. *International Conference on Agile Software Development*, 118-130, Springer, Cham.

Eloranta, V. P. (2014). Towards a pattern language for software start-ups. *Proceedings of 19th European Conference on Pattern Languages of Programs*, Germany. Article No. 24.

Fuggetta, A. (2000). Software process: a roadmap. *Proceedings of the Conference on the Future of Software Engineering (ICSE).* pp. 25–34.

Giardino, C., Paternoster, N., Unterkalmsteiner, M., Gorschek, T., & Abrahamsson, P. (2016). Software Development in Startup Companies: The Greenfield Startup Model. *IEEE Transactions on Software Engineering, 42*(6), 585-604.

Giardino, C., Unterkalmsteiner, M., Paternoster, N., Gorschek, T., & Abrahamsson, P. (2014). What do we know about software development in startups? *IEEE Software, 31*(5), 28-32.

Giardino, C., Wang, X., & Abrahamsson, P. (2014). Why early-stage software startups fail: A behavioral framework. *International Conference of Software Business*, 27-41. Springer, Cham.

Ghezzi, A., Cavallaro, A., Rangone, A., & Balocco, R. (2015). A comparative study on the impact of business model design & lean startup approach versus traditional business plan on mobile startups performance. *Department of Management, Economics and Industrial Engineering, Politecnico di Milano, Milan, Italy*.

Hokkanen, L., & Leppänen, M. (2015). Three patterns for user involvement in startups. *Proceedings of 20th European Conference on Pattern Languages of Programs*, ACM. DOI: 10.1145/2855321.2855373

IEEE Computer Society (2004). Guide to the Software Engineering Body of Knowledge (SWEBOK-2004 Version). *IEEE Computer Society*. Los Alamitos, CA.

Kajko-Mattsson, M., & Nikitina, N. (2008). From knowing nothing to knowing a little: experiences gained from process improvement in a start-up company. *Proceedings of the International Conference on Computer Science and Software Engineering (CSSE)*. 617–621.

Kitchenham, B. A., Dyba, T., & Jorgensen, M. (2004). Evidence-based software engineering. *Proceedings of the 26th International Conference on Software Engineering (ICSE)*. 273-281.

Kitchenham, B., & Charters, B. (2007). Systematic literature reviews in software engineering – A systematic literature review. *Tech. Rep. EBSE, Keele University and Durham University Joint Report,* Staffordshire, UK.

Klotins, E., Unterkalmsteiner, M., & Gorschek, T. (2015). Software Engineering Knowledge Areas in Startup Companies: A Mapping Study. *Blekinge Institute of Technology.*

Lenarduzzi, V., & Taibi, D. (2016). MVP Explained: A Systematic Mapping Study on the Definitions of Minimal Viable Product. *42th Euromicro Conference on Software Engineering and Advanced Applications*, Limassol, Cyprus. doi: 10.1109/SEAA.2016.56

Leppänen, M. (2014). Patterns for starting up a software startup company. *Tampere University of Technology.*

Marion, T., Dunlap, D., & Friar, J. (2012). Instilling the entrepreneurial spirit in your R&D team: What large firms can learn from successful startups? *IEEE Transaction on Engineering Management, 59*(2), 323-337.

May, B. (2012). Applying Lean Startup: An Experience Report -- Lean & Lean UX by a UX Veteran: Lessons Learned in Creating & Launching a Complex Consumer App. *Agile Conference (AGILE)*. 141-147.

McCaffery, F., Taylor, P. S., & Coleman, G. (2007). Adept: A unified assessment method for small software companies. *IEEE Computer Society*.

Mc Caffery, F., Taylor, P. S., & Coleman, G. (2007). Adept: A unified assessment method for small software companies. *IEEE software, 24*(1), 24-31.

Meng, J. L., & Yao, L. (2008). The status and development trend of the development model based on XP method. *Wireless Communications, Networking and Mobile Computing 4th International Conference*. Dalian, China. doi: 10.1109/WiCom.2008.1319

Nguyen-Duc, A., Seppänen, P., & Abrahamsson, P. (2015). Hunter-gatherer cycle: A conceptual model of the evolution of software startups. *Proceedings of the 2015 International Conference on Software and System Process*, 199-203. ACM.

Nguyen-Duc, A., Shah, S.M.A., & Abrahamsson, P. (2016). Towards an Early Stage Software Startups Evolution Model. *42th Euromicro Conference on Software Engineering and Advanced Applications*. Limassol, Cyprus. doi: 10.1109/SEAA.2016.21

Paternoster, N., Giardino, C., Unterkalmsteiner, M., Gorschek, T., & Abrahamsson, P. (2014). Software development in startup companies: A systematic mapping study. *Information and Software Technology, 56*(10), 1200-1218.

Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2017). Agile software development methods: Review and analysis. *arXiv preprint arXiv:1709.08439*.

Refworks. Retrieved October 12, 2017 from http://refworks.com

Ries, E. (2011). *The lean startup: How today's entrepreneurs use continuous innovation to create radically successful businesses*. Crown Books.

Rumsey, S. (2008). *How to find information: a guide for researchers*. UK, McGraw-Hill Education.

Sánchez-Gordón, M. L., & O'Connor, R. V. (2016). Understanding the gap between software process practices and actual practice in very small companies. *Software Quality Journal, 24*(3), 549-570.

Seppänen, P., Tripathi, N., Oivo, M., & Liukkunen, K. (2017). How Are Product Ideas Validated? *International Conference of Software Business*, 3-17. Springer, Cham.

Sharp, H., & Hall, T. (2016). *Agile Processes in Software Engineering and Extreme Programming*. SpringerOpen.

Sheuly, S. (2013). A systematic literature review on agile project management. Lappeenranta University of Technology.

Sutton, S. M. (2000). The role of process in software start-up. *IEEE software, 17*(4), 33-39.

Taipale, M. (2010). Huitale – a story of a Finnish lean startup. *Lean Enterprise Software and Systems,* 111-114, Springer, Berlin, Heidelberg.

Taylor, P. S., Greer, D., Coleman, G., McDaid, K., & Keenan, F. (2008). Preparing small software companies for tailored agile method adoption: Minimally intrusive risk assessment. *Software Process: Improvement and Practice, 13*(5), 421-437.

Terho, H., Suonsyrjä, S., Jaaksi, A., Mikkonen, T., Kazman, R., & Chen, H. M. (2015). Lean startup meets software product lines: Survival of the fittest or letting products bloom? *Proceedings of the 14th Symposium on Programming Languages and Software Tools (SPLST'15)*. 134-148.

Tingling, P., & Saeed, A. (2007). Extreme programming in action: A longitudinal case study. *International Conference on Human-Computer Interaction. Berlin, Heidelberg*. 242-251.

Unterkalmsteiner, M., Abrahamsson, P., Wang, X., Nguyen-Duc, A., Shah, S., Bajwa, S. S., … & Edison, H. (2016). Software startups–a research agenda. *e-Informatica Software Engineering Journal, 10*(1), 89-123.

Wangenheim, C. G. v., Weber, S., Hauck, J. C. R., & Trentin, G. (2006). Experiences on establishing software processes in small companies. *Information and software technology, 48*(9), 890-900.

Wu, H. Y., & Callaghan, V. (2016). From Imagination to Innovation: A Creative Development Process. *Intelligent Environments (Workshops*), 514-523.

Appendix A: Search Results

IEEE

Group 1

Keywords	Result	Result after screening
("early-stage firm" OR "early-stage company" OR "high-tech venture" OR "high-tech ventures" OR "start-up company") AND (develop* OR engineer* OR implement* OR build*) AND (software OR process OR model OR methodolog* OR method OR practice OR strategy)	38	28

Group 2

Keywords	Result	Result after screening
("start-up companies" OR "high-tech start-up" OR "high-tech start-ups" OR "high-tech startups" OR "high-tech startup") AND (develop* OR engineer* OR implement* OR build*) AND (software OR process OR model OR methodolog* OR method OR practice OR strategy)	39	23

Group 3

Keywords	Result	Result after screening
("startup company" OR "startup companies" OR "software startup" OR "lean startup" OR "lean start-up") AND (develop* OR engineer* OR implement* OR build*) AND (software OR process OR model OR methodolog* OR method OR practice OR strategy)	70	54

Group 4

Keywords	Result	Result after screening
("lean startups" OR "software startups" OR "IT start-ups" OR "IT start-up" OR "IT startup") AND (develop* OR engineer* OR implement* OR build*) AND (software OR process OR model OR methodolog* OR method OR practice OR strategy)	19	11

Group 5

Keywords	Result	Result after screening
("IT startups" OR "software start-up" OR "software start-ups" OR "software product startup" OR "software product startups") AND (develop* OR engineer* OR implement* OR build*) AND (software OR process OR model OR methodolog* OR method OR practice OR strategy)	12	8

Group 6

Keywords	Result	Result after screening
("software start up" OR "web startup" OR "web start-up" OR "web startups" OR "web start-ups") AND (develop* OR engineer* OR implement* OR build*) AND (software OR process OR model OR methodolog* OR method OR practice OR strategy)	8	5

Group 7

Keywords	Result	Result after screening
("internet startup" OR "internet start-up" OR "internet startups" OR "internet start-ups" OR "mobile startup") AND (develop* OR engineer* OR implement* OR build*) AND (software OR process OR model OR methodolog* OR method OR practice OR strategy)	4	2

Group 8

Keywords	Result	Result after screening
("mobile start-up" OR "mobile startups" OR "mobile start-ups" OR "software package start-up" OR "software package start-ups") AND (develop* OR engineer* OR implement* OR build*) AND (software OR process OR model OR methodolog* OR method OR practice OR strategy)	0	0

Total result before screening = 190
Total result after screening = 131

Scopus

Keywords	Result before screening	Result after screening
("early-stage firm" OR "early-stage company" OR "high-tech venture" OR "high-tech ventures" OR "start-up company" OR "start-up companies" OR "high-tech start-up" OR "high-tech start-ups" OR "high-tech startups" OR "high-tech startup" OR "startup company" OR "startup companies" OR "software startup" OR "lean startup" OR "lean start-up" OR "lean startups" OR "software startups" OR "IT start-ups" OR "IT start-up" OR "IT startup" OR "IT startups" OR "software start-up" OR "software start-ups" OR "software product startup" OR "software product startups" OR "software start up" OR "web startup" OR "web start-up" OR "web startups" OR "web start-ups" OR "internet startup" OR "internet start-up" OR "internet startups" OR "internet start-ups" OR "mobile startup" OR "mobile start-up" OR "mobile startups" OR "mobile start-ups") AND (develop* OR engineer* OR implement* OR build*) AND (software OR process OR model OR methodolog* OR method OR practice OR strategy)	4562	1141

Web of Science

Keywords	Result before screening	Result after screening
("early-stage firm" OR "early-stage company" OR "high-tech venture" OR "high-tech ventures" OR "start-up company" OR "start-up companies" OR "high-tech start-up" OR "high-tech start-ups" OR "high-tech startups" OR "high-tech startup" OR "startup company" OR "startup companies" OR "software startup" OR "lean startup" OR "lean start-up" OR "lean startups" OR "software startups" OR "IT start-ups" OR "IT start-up" OR "IT startup" OR "IT startups" OR "software start-up" OR "software start-ups" OR "software product startup" OR "software product startups" OR "software start up" OR "web startup" OR "web start-up" OR "web startups" OR "web start-ups" OR "internet startup" OR "internet start-up" OR "internet startups" OR "internet start-ups" OR "mobile startup" OR "mobile start-up" OR "mobile startups" OR "mobile start-ups") AND (develop* OR engineer* OR implement* OR build*) AND (software OR process OR model OR methodolog* OR method OR practice OR strategy)	399	82

Appendix B: List of Primary Studies and Study ID

Author(s)	Title of paper	Study ID
A. Nguyen-Duc;S. M. A. Shah;P. Ambrahamsson, (2016)	Towards an Early Stage Software Startups Evolution Model	[P1]
B. May, (2012)	Applying Lean Startup: An Experience Report -- Lean & Lean UX by a UX Veteran: Lessons Learned in Creating & Launching a Complex Consumer App	[P2]
Batova,T.;Clark,D.;Card, D. (2016)	Challenges of lean customer discovery as invention	[P3]
Berrocal,J.;Garcia-Alonso,J.;Murillo,J. M. (2016)	The Fall and Rise of NimBees	[P4]
Björk,J.;Ljungblad,J.; Bosch,J. (2013)	Lean product development in early stage startups	[P5]
C. Giardino;N. Paternoster;M. Unterkalmsteiner, T. Gorschek, P. Abrahamsson (2016)	Software Development in Startup Companies: The Greenfield Startup Model	[P6]
Chorev, S.; Anderson, A. R. (2006)	Success in Israeli high-tech start-ups; Critical factors and process	[P7]
Clark,T.;Muller,P. -A (2012)	Exploiting model driven technology: A tale of two startups	[P8]
Coleman, G.;O'Connor,R. V. (2008)	An investigation into software development process formation in software startups	[P9]
Duc,A. N.;Abrahamsson,P. (2016)	Minimum viable product or multiple facet product? The role of MVP in software startups	[P10]
Eloranta,V. P. (2014)	Towards a pattern language for software startups	[P11]
Ghezzi,A.; Cavallaro,A.; Rangone,A.;Balocco,R. (2015)	A comparative study on the impact of business model design & lean startup approach versus traditional business plan on mobile startups performance	[P12]
Giardino,C.;Unterkalmsteiner,M.;Paternoster,N.;Gorschek,T.;Abrahamsson,P. (2014)	What do we know about software development in startups?	[P13]
Giardino,C.;Wang, X.; Abrahamsson, P. (2014)	Why early-stage software startups fail: A behavioral framework	[P14]
Hokkanen,L.;Leppänen,M. (2015)	Three patterns for user involvement in startups	[P15]
Kajko-Mattsson,M.;Nikitina,N. (2008)	From knowing nothing to knowing a little: Experiences gained from process improvement in a startup company	[P16]

Klotins,E.;Unterkalmsteiner,M.;Gorschek,T. (2015)	Software engineering knowledge areas in startup companies: A mapping study	[P17]
Lenarduzzi,V.;Taibi,D. (2016)	MVP Explained: A Systematic Mapping Study on the Definitions of Minimal Viable Product	[P18]
Leppänen,M. (2014)	Patterns for starting up a software startup company	[P19]
Marion,T.;Dunlap,D.;Friar,J. (2012)	Instilling the entrepreneurial spirit in your R&D team: What large firms can learn from successful startups	[P20]
McCaffery,F.;Taylor, P. S.;Coleman,G. (2007)	Adept: A unified assessment method for small software companies	[P21]
Nguyen-Duc,A.;Seppänen,P.;Abrahamsson,P. (2015)	Hunter-gatherer cycle: A conceptual model of the evolution of software startups	[P22]
Paternoster,Nicolo;Giardino,Carmine;Unterkalmsteiner,Michael;Gorschek,Tony; Abrahamsson,Pekka (2014)	Software development in startup companies: A systematic mapping study	[P23]
S. Celar;M. Turic;L. Vickovic (2014)	Method for personal capability assessment in agile teams using personal points	[P24]
Sánchez-Gordón,M. -L;O'Connor,R. V. (2016)	Understanding the gap between software process practices and actual practice in very small companies	[P25]
Taylor,P. S.;Greer,D.; Coleman,G.;McDaid,K.;Keenan, F. (2008)	Preparing small software companies for tailored agile method adoption: Minimally intrusive risk assessment	[P26]
Terho,H.;Suonsyrjä,S.;Jaaksi,A.;Mikkonen,T.;Kazman,R.;Chen,H. -M (2015)	Lean startup meets software product lines: Survival of the fittest or letting products bloom?	[P27]
Tingling,P.;Saeed, A. (2007)	Extreme programming in action: A longitudinal case study	[P28]
Wangenheim,C. G. v.; Weber, S.;Hauck,J. C. R.; Trentin,G. (2006)	Experiences on establishing software processes in small companies	[P29]
Wu,Hsuan-Yi;Callaghan,Vic (2016)	From Imagination to Innovation: A Creative Development Process	[P30]

Appendix C: Primary Studies and Software Development Methodologies

Study ID	Software Development methodology used in the startup
[P1]	Prototyping
[P2]	Lean Startup
[P3]	Lean Startup
[P4]	Lean Startup
[P5]	Agile method
[P6]	Lean Startup
[P7]	-
[P8]	Model Driven Development
[P9]	Waterfall, Rational Unified Process (RUP), XP
[P10]	Lean Startup, Tailored Agile, Ad Hoc, Distributed Scrum
[P11]	Lean Startup
[P12]	Lean Startup + Business Model Design (BMD) = Lean Startup
[P13]	-
[P14]	Prototyping (Evolutionary)
[P15]	Lean Startup
[P16]	Ad Hoc management
[P17]	-
[P18]	-
[P19]	Lean Startup
[P20]	Hyper-agility
[P21]	Agile Method
[P22]	Lean Startup
[P23]	-
[P24]	Agile method
[P25]	Agile method, Iterative/Incremental
[P26]	Agile method
[P27]	Lean Startup
[P28]	XP
[P29]	-
[P30]	Agile method

Appendix D: Primary Studies and Software Development Practices

Work Practice	**Work Practice Category** **SD- Software Development** **PM- Process Management** **O/M- Organizational/Management**	**Primary Study**
Usage of co-evolution patterns of products and customer knowledge	SD	[P1]
Prototype-centric development and scale-up Agile	SD	[P1]
Customer feedback	SD	[P1]
Tailoring agile practices	PM	[P1]
Pivoting practices	SD	[P1]
Daily scrums	PM	[P2]
Release once a week	SD	[P2]
Customer development	PM	[P3]
MVP	SD	[P4]
Lean Principles	PM	[P5]
Agile practices	PM	[P5]
Scrum	PM	[P5]
Kanban	PM	[P5]
ESSSDM	SD	[P5]
Customer-focused development	SD	[P5]
Working closely with customers	SD	[P5]
Pivoting towards product/market fit.	PM	[P5]
Highly evolutionary development approach	SD	[P6]
Customer development	PM	[P6]
Prototyping	SD	[P6]
Core Team expertise	M/O	[P7]
Market driven and customer focused New Product Process	M/O	[P7]
Diversified team management	M/O	[P7]

Careful selection of personnel	M/O	[P7]
Using UML	M	[P8]
Process tailoring	PM	[P9]
Managerial experience	M/O	[P9]
Prototyping	SD	[P10]
MVP	SD	[P10]
Develop only what is needed for now	SD	[P11]
Serve single customer segment first	PM	[P11]
Develop the development culture before processes	PM	[P11]
Combined application of Business Model Design and Lean Startup Approach	SD	[P12]
Strategic Entrepreneurship	PM	[P12]
Use of well-known frameworks	SD	[P13]
Use of evolutionary prototyping	SD	[P13]
Experimentations via existing components	SD	[P13]
Continuous value delivery	SD	[P13]
Focusing on core functionalities that engage paying customers	SD	[P13]
Empowerment of teams to influence final outcomes and	M/O	[P13]
Handle fast-paced, changing information	PM	[P13]
Use of metrics to quickly learn from consumers' feedback and demand	SD	[P13]
Use of easy-to-implement tools to facilitate product development	SD	[P13]
Use of evolutionary prototypes	SD	[P14]
Building MVP	SD	[P15]
Prototyping	SD	[P15]
Implementing UX	SD	[P15]
Performing incremental process improvement actions	PM	[P16]
Process Improvement Method	PM	[P16]
Adopting Release Management Process Model	PM	[P16]

Adopting Release management process	PM	[P16]
Documentation	PM	[P16]
System testing	SD	[P16]
Release development	SD	[P16]
Market-driven requirements engineering	SD	[P17]
Minimum Viable Product (MVP)	SD	[P18]
Being lean in product validation	SD	[P19]
Customer discovery	PM	[P19]
Building Small omni-functional teams with no functional boundaries	O/M	[P20]
Goal-driven rapid development	SD	[P20]
Instinctive exploration of market potential rather than quantitative analysis	PM	[P20]
Plan-driven process assessment	PM	[P21]
Unified Assessment Method	PM	[P21]
Software Process Improvement (SPI)	PM	[P21]
Adoption of MVP and pivoting	PM	[P22]
Hunting and gathering activities	PM	[P22]
Focusing on idea refinement, team building and funding domains	O/M	[P22]
Refactoring	SD	[P23]
Use of existing components(code)	SD	[P23]
Outsourcing tests	SD	[P23]
Empowerment of team responsibilities and their ability	O/M	[P23]
Use of key performance indicators	SD	[P23]
Plan of short-medium term objectives	PM	[P23]
Measuring development cycle time to find areas of improvement	SD	[P23]
Use of easy-to-implement tools	SD	[P23]
Use of open source solutions.	SD	[P23]
Using methods for personal capability assessment of each individual team member	M/O	[P24]

Use of own software process preferable to their own particular environment	SD	[P25]
Minimally intrusive assessment approach	PM	[P26]
Tailoring agile methods	PM	[P26]
Adapting ADA (agility/discipline assessment)	PM	[P26]
MVP / build-measure-learn based iterative method with MVPs	SD	[P27]
Continuous Integration	SD	[P28]
Continuous testing	SD	[P28]
Refactoring	SD	[P28]
40 Hour Work Week	SD	[P28]
Planning Game	SD	[P28]
Simple Design	SD	[P28]
Small Releases	SD	[P28]
System Metaphor	SD	[P28]
Code Standards	SD	[P28]
Pair programing	SD	[P28]
On-site customer	SD	[P28]
Incremental and iterative establishment of software processes	PM	[P29]
Creative Innovation Development (CID)	SD	[P30]

Appendix E: Primary Studies and Research Focus

Study ID	Focus
[P1]	Process Management
[P2]	Software Development
[P3]	Process Management
[P4]	Software Development
[P5]	Process Management
[P6]	Software Development
[P7]	Managerial/Organization
[P8]	Process Management
[P9]	Process Management
[P10]	Software Development
[P11]	Software Development
[P12]	Process Management
[P13]	Software Development
[P14]	Software Development
[P15]	Software Development
[P16]	Process Management
[P17]	Software Development
[P18]	Software Development
[P19]	Software Development
[P20]	Software Development
[P21]	Process Management
[P22]	Software Development
[P23]	Software Development
[P24]	Managerial/organization
[P25]	Software Development
[P26]	Software Development
[P27]	Software Development
[P28]	Software Development
[P29]	Process Management
[P30]	Process Management