The Journal of Systems and Software 146 (2018) 130–151

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

An anatomy of requirements engineering in software startups using multi-vocal literature and case survey

Nirnaya Tripathi[a,*], Eriks Klotins[c], Rafael Prikladnicki[b], Markku Oivo[a], Leandro Bento Pompermaier[b], Arun Sojan Kudakacheril[a], Michael Unterkalmsteiner[c], Kari Liukkunen[a], Tony Gorschek[c]

[a] M3S Research Group, University of Oulu, Oulu, Finland
[b] Computer Science School, PUCRS Pontifical Catholic University of Rio Grande do Sul, Brazil
[c] Software Engineering Research Lab, Blekinge Institute of Technology, Karlskrona, Sweden

ARTICLE INFO

Article history:
Received 8 February 2018
Revised 13 July 2018
Accepted 27 August 2018
Available online 28 August 2018

Keywords:
Requirements engineering
Software startups
Multi-vocal literature review
Case survey

ABSTRACT

Context: Software startups aim to develop innovative products, grow rapidly, and thus become important in the development of economy and jobs. Requirements engineering (RE) is a key process area in software development, but its effects on software startups are unclear.

Objective: The main objective of this study was to explore how RE (elicitation, documentation, prioritization and validation) is used in software startups.

Method: A multi-vocal literature review (MLR) was used to find scientific and gray literature. In addition, a case survey was employed to gather empirical data to reach this study's objective.

Results: In the MLR, 36 primary articles were selected out of 28,643 articles. In the case survey, 80 respondents provided information about software startup cases across the globe. Data analysis revealed that during RE processes, internal sources (e.g., for source), analyses of similar products (e.g., elicitation), uses of informal notes (e.g., for documentation), values to customers, products and stakeholders (e.g., for prioritization) and internal reviews/prototypes (e.g., for validation) were the most used techniques.

Conclusion: After an analysis of primary literature, it was concluded that research on this topic is still in early stages and more systematic research is needed. Furthermore, few topics were suggested for future research.

1. Introduction

Software startups are important for the economy due to their potential for creating jobs and adding value to domestic products on a global scale (Kane, 2010). However, due to their characteristics, including limited resources and staff with limited product development experience, product development occurs with irregular software engineering practices, and such practices often depend on startups' contexts (Tripathi et al., 2016). In software engineering, using unreliable RE techniques can cause software projects to fail. In RE, including relevant and experienced stakeholders and using standardized processes to collect requirements are vital elements (Pacheco and Garcia, 2012; Verner et al., 2007).

Small-scale companies face many challenges caused by substandard or basic RE practices, such as the needs to update various product components during required changes, communication and coordination issues, confusing project overviews, and writing difficulties regarding the understood requirements (Quispe et al., 2010; Tripathi et al., 2016). In the context of software startups, due to their small sizes and their staffs' limited software development experience, RE processes are basic (Paternoster et al., 2014; Klotins et al., 2015). These poor RE practices often make products unsuitable for their target markets (Paternoster et al., 2014; Mater and Subramanian, 2000).

RE processes are important for startup product development; however, comprehensive studies on RE in software startups are limited (Paternoster et al., 2014; Klotins et al., 2015). As a result, there is a lack of in-depth knowledge concerning RE in software startup contexts and RE's importance when it comes to startups' success. Thus, there is a research gap, and further studies are

* Corresponding author.
E-mail address: nirnaya.tripathi@oulu.fi (N. Tripathi).

https://doi.org/10.1016/j.jss.2018.08.059
0164-1212/

Table 1
Research questions and their rationale.

Research question	Rationale
RQ1. What are the sources of ideas (requirements) for software startup product/service features?	To find relevant requirements for software startups' products, it is important to seek suitable sources from which requirements can be taken (Pacheco and Garcia, 2012; Bourque and Fairley, 2014).
RQ2. What required elicitation methods are used by the startup?	Once sources are identified, methods of eliciting requirements from the sources should be applied. Using a suitable method is important for finding and defining relevant product requirements (Pacheco and Garcia, 2012; Bourque and Fairley, 2014).
RQ3. How are requirements documented in the software startup?	As requirements are elicited, it is important to document the requirements so that it is easy for stakeholders in company to assess them (Bourque and Fairley, 2014; Condori-Fernandez et al., 2009).
RQ4. How does the software startup prioritize and manage requirements?	When the documents for the requirements are created, it is important to prioritize the requirements to decide which requirements need to be implemented first (Bourque and Fairley, 2014; Achimugu et al., 2014).
RQ5. How does the software startup validate requirements?	Once the requirements are prioritized, it is essential that they be validated so stakeholders can clearly understand what the requirements mean before implementing them (Bourque and Fairley, 2014).

needed. Moreover, due to importance of software startups in economic growth and the roles of product developments in the software startups' success, it is important to quantify extant knowledge on RE in software startups and gather empirical data from real software startup cases. This study aimed to address these needs and achieve the following five objectives:

- Exploring sources used by software startups to gather requirements;
- Examining techniques used by software startups to elicit requirements from sources;
- Studying documentation methods used by software startups to store requirements;
- Exploring techniques used by software startups to prioritize requirements; and
- Examining methods applied by software startups to validate requirements.

To reach the objectives and address the research gaps, several research questions were formulated (Table 1). The decision was made to conduct a comprehensive study using an MLR (i.e., to examine all literature that highlights RE in software startups) (Garousi et al., 2016) and perform a case survey on the topic (i.e., to collect empirical data about RE in software startups) (Larsson, 1993). We opted to use the MLR, since it is an extended version of the systematic literature review that can help in identifying scientific literature and gray literature on the phenomenon. Since the research on software startups, especially in the area of RE, is still in the nascent stage, we need to incorporate practitioners' point of view, which is shared through internet channel in the form of webpages and white papers. Through MLR, we aim to establish state of art on the topic. In the MLR, 28,643 articles (9671 and 18,972 scientific and gray articles, respectively) were identified, from which 36 primary articles were selected that discussed specific aspects of RE in software startups. We selected a case survey, since this approach enables the researchers to collect empirical data in the qualitative and quantitative forms to analyze the case in depth. In the case survey, 80 respondents provided information about software startup cases, which provided empirical data on RE in software startups. Through information collected from the literature articles and the empirical data, the research questions were answered and the research objectives were reached.

The paper discusses these steps in six sections. Section 2 includes background information and related work on the topic. Section 3 discusses the two research methods (MLR and case survey), data extraction, and data synthesis. Following this, Section 4 presents the results of the two methods in detail. Section 5 discusses the research questions and connects them to the results presented in Section 4. It also discusses the study implications in terms of practice and research, as well as the validity of the study. Finally, the last section, Section 6, provides a conclusion and makes suggestions for future research.

2. Background and related work

This section discusses software startups and their product development processes. It then discusses RE and studies related to this paper's topic.

2.1. Software startup and product development

According to Giardino et al. (2016), "software startups refer to those organizations focused on the creation of high-tech and innovative products, with little or no operating history, aiming to aggressively grow their business in highly scalable markets." Steve Blank distinguished four types of startups based on products and market types (Blank, 2013) as product development is an integral part of software startups, and through product development, companies' customer frequencies increase.

According to Crowne (2002), product development involves four phases, as follows: startup, stabilization, growth, and evolution. During the startup and stabilization phases, product ideas are improved on and validated to develop target products. In the growth and evolution phases, startups' product development stages stabilize and can be shaped according to the market needs.

In a study by Wang et al. (2016) six product development stages were noted. These were concept, development, working prototype, functional product with limited users, functional product with high growth, and mature product. Moreover, our earlier studies on software startups simply explored the effects of competitor interactions on product development and product idea validation (Tripathi et al., 2017; Seppänen et al., 2017).

2.2. Requirements engineering

RE is an important phase in successful software development. If requirements are not collected properly, projects can fail (Pacheco and Garcia, 2012; Verner et al., 2007). RE contains several phases, such as elicitation, analysis, specification, and validation (Bourque and Fairley, 2014). Requirement elicitation is a significant aspect of RE, in which requirements are elicited from many different sources; hence, it is vital to recognize and assess all potentially useful sources (Pacheco and Garcia, 2012; Bourque and Fairley, 2014). Some of the techniques involved are interviews (can provide sufficient data from the participants in a specific time), questionnaires (can collect data from multiple stakeholders simultaneously), brainstorming (informal discussion with different stakeholders to gather several ideas), domain analysis (analysis of existing documents to gather domain related knowledge), and observation (collecting the data by observing the process without any direct influence) (Zowghi and Coulin, 2005).

In requirement specification, documents are created through which the requirements are thoroughly assessed in terms of understandability, correctness, completeness, and consistency (Bourque and Fairley, 2014; Condori-Fernandez et al., 2009). Then, once the documentation is created, it is important to analyze and prioritize the requirements in terms of values to stakeholders. The objective is to create features that are the most valuable to stakeholders first and then consider other aspects, such implementation costs and resource availabilities (Bourque and Fairley, 2014; Achimugu et al., 2014). Some prioritization techniques used in software engineering are numerical assignment (requirements are characterized in terms of high, medium, and low), the pairwise comparison technique (requirements are grouped in pairs and analyzed to obtained final ratings of the requirements), and the cost-value prioritization technique (requirements are mapped against development costs using graphs) (Achimugu et al., 2014). Once requirements are collected and analyzed through documentation, it is important to validate them to ensure that they align with company strategies and that stakeholders (especially developers) understand the requirements in terms of clarity and consistency. Some techniques for validation are prototyping (used to determine whether the developer understood the requirements), model validation (use of models like object models or formal analysis notations to establish communication and flow of information between objects) and acceptance tests (helps in validating the use of systems by end users) (Bourque and Fairley, 2014).

In the context of small software companies, Aranda et al. (2007) observed that small companies perform requirement gathering practices differently, and the companies' cultures affect requirement gathering. The researchers also found that requirement practices are influenced by customer types, team experience, founders' choices, and company sizes. Moreover, Quispe et al. (2010) observed that, with very small software companies, there are often two major problems during requirement gathering. First, there are communication and collaboration differences and gaps between customers and teams. Second, unstandardized RE practices are used.

In software startups, due to strict time constraints and limited resources, requirements are mostly self-invented, rarely documented, and validated only after product releases. Moreover, requirements are often market driven, as customers and users are not frequently recognized during the initial stages of product development (Paternoster et al., 2014). Thus, there may be a lack of external stakeholder presence in startups' RE processes (Mater and Subramanian, 2000). Including customers and users as stakeholders in elicitation processes and the use of adaptable techniques, such as user stories and estimations, can help software startups in rapidly handling changing market requirements (Paternoster et al., 2014; Coleman and O'Connor, 2008).

2.3. Related work

Few studies have conducted literature reviews on software startups. However, some have used mapping techniques to describe software engineering practices in software startups (e.g., Paternoster et al., 2014; Klotins et al., 2015). Concerning RE in software startups, two studies (Melegati et al., 2016; Gralha et al., 2018) have taken a grounded theory approach to understand the process. Melegati et al. (2016) proposed a theoretical model based on nine interviews with different Brazilian software startups to give an overview of the RE process and how elements in startups affect the process. Similarly, Gralha et al. (2018) collected data from 16 software startups to describe the evolution of requirements practices through six dimensions and the effects they can have on the products, staff, and company.

Concerning reviews of gray literature in software startups, Bajwa et al. (2017) systematically explored gray literature to assess pivots in software startups. Moreover, in relation to RE, some discussed software requirement prioritization and RE in software product lines (e.g., Achimugu et al., 2014; Alves et al., 2010). Babar et al. (2014) conducted a systematic literature review to explore past systematic literature studies on RE.

In addition to the studies above, two were similar to this study (see Holl et al., 2012; Rabiser et al., 2010). They combined systematic literature reviews and surveys to analyze abilities to support multi-product lines and requirements for product derivation support. However, comprehensive studies on RE in software startups that combine systematic literature reviews and empirical techniques are not present in the extant literature. To address this research gap, this study uses an MLR (Garousi et al., 2016) and a case survey (Larsson, 1993) on RE in software startups to answer the research questions and reach the study objectives (see Section 1). The next section discusses the research process in detail.

3. Research methodology

To address the research objectives and fulfill the research gap, research questions were formulated (Table 1). The research questions focused on understanding the phases of RE, including elicitation, analysis, specification, and validation, in the software startup context. The research questions were exploratory (i.e., to explore and understand the phenomena) and base rate (i.e., to discover natural patterns in the phenomena) (Easterbrook et al., 2008). Furthermore, we took pragmatism as our stance on the research, since we are interested in finding practical answers to the research questions through available research methods (using mixed-method approaches) (Easterbrook et al., 2008) for collecting qualitative and quantitative data to explore the topic. Qualitative data were found using the existing literature through an MLR (Garousi et al., 2016), while empirical qualitative and quantitative data were found using a case survey (Larsson, 1993). The MLR (Garousi et al., 2016) and case survey (Larsson, 1993) were used to comprehensively answer the research questions. Both research methods and the research process are outlined in Fig. 1. As can be seen in the figure, the research gap and questions were inputs for the research process. The inputs were utilized during two phases—phase one representing the MLR and phase two comprising the case survey. Two more phases (three and four) followed; data extraction and data analyses were done for the qualitative and quantitative data. Finally, the results of phase four were used as outputs to answer the research questions.

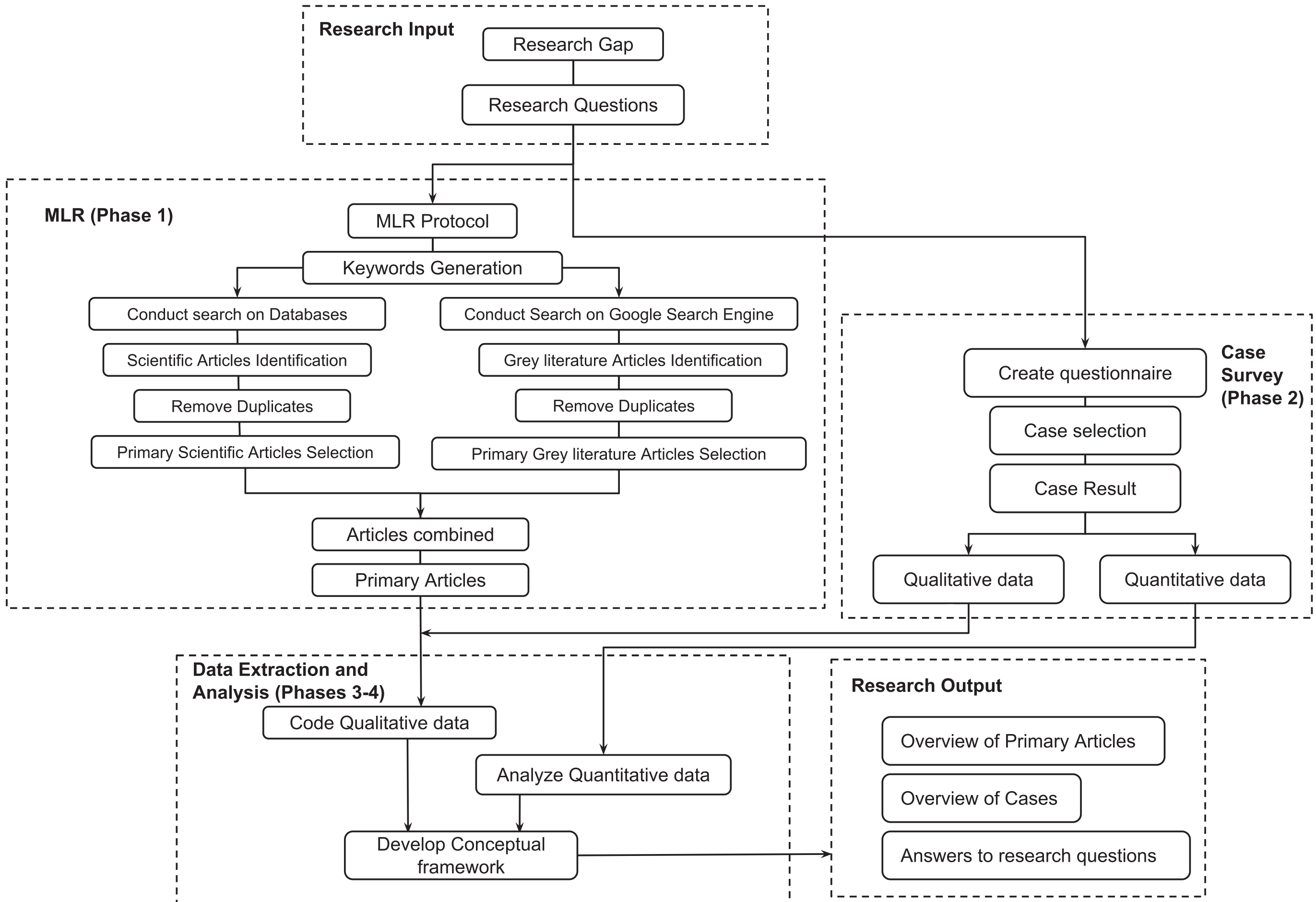

Fig. 1. Research process.

Table 2
Keywords.

	Population	Intervention
Key words	"startup" OR "start-up" OR "early-stage firm" OR "early stage firm" OR "early-stage company" OR "early stage company"	"requirement" OR "requirement engineering" OR "customer" OR "end user" OR "stakeholder" OR "elicitation" OR "elicit" OR "functional requirement" OR "non-functional requirement" OR "backlog" OR "user story" OR "effort estimation"

3.1. *Multi-vocal literature review (phase one)*

As literature on RE in software startups is limited (Paternoster et al., 2014; Klotins et al., 2015), the choice was made to extend the literature search to include both scientific and gray articles. Gray literature is useful for discovering practitioners' points of view. The MLR's (Garousi et al., 2016) steps, which were implemented to retrieve relevant articles for this study, are discussed below.

3.1.1. *Literature search strategy*

The MLR was divided into two steps. The first involved searching for articles in scientific databases, while the second was performed to search for gray literature on Google. To find the articles, keywords related to population and intervention were used (Table 2). Keywords used for population were taken from a previous study (Bajwa et al., 2017), while keywords for intervention were based on the learning gained during our previous work (Tripathi et al., 2017). In that study, we conducted an experiment on the RE process in software startups. The population and intervention keywords were used together to look for articles using scientific databases and Google. The search process started in August 2016 and ended in October 2016.

Two authors of this paper jointly executed the process. Specifically, the sixth author took the responsibility for searching the articles from the scientific database, while the first author executed the Google search process, since author had database search experienced from his previous literature review work (Tripathi et al., 2016). The objective was for the first author to guide and check the sixth author's process if a discrepancy emerged. For example, the IEEE database reported an immense number of articles during each search, and to resolve this issue, both authors concluded that would be better to export the top 200 results for each search from the IEEE database, since articles after the first 200 results appeared to be beyond the scope of our study.

As discussed above, the data sources were scientific databases and the Google search engine, as follows:

- ACM[1] (scientific database);

[1] http://dl.acm.org/

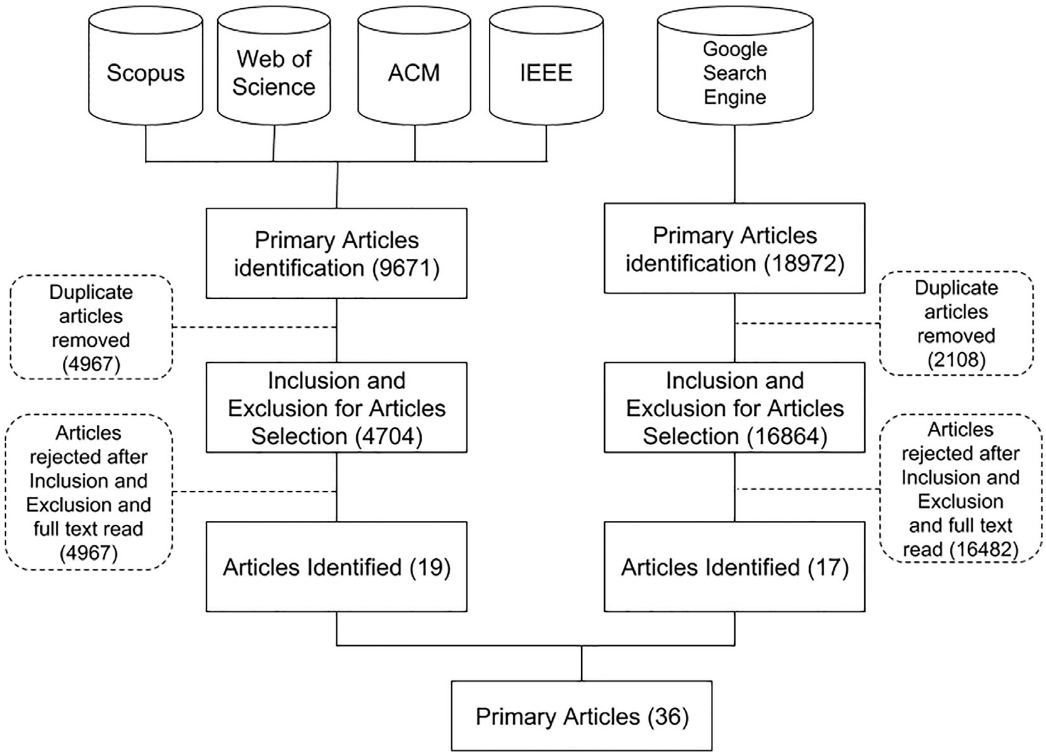

Fig. 2. Search and study selection procedure.

- IEEE[2] (scientific database);
- SCOPUS[3] (scientific database);
- Web of Science[4] (scientific database); and
- Google's web search engine[5] (for gray literature).

In addition, to systematically search scientific literature, the StArt tool[6] was used, since this is designed to conduct systematic literature reviews, as it incorporates all the important steps needed to perform the literature review process. The tool was also used by the coauthors in their previous study (Tripathi et al., 2016). To rigorously search for gray literature, the steps used by Bajwa et al. (2017) were included, and thus, the Google Chrome internet browser was used. These steps are as follows:

- Clear the browser's search history and cache, and sign out of Google;
- Disable Google's instant search predictions;
- Activate "100 results/links per page" in Google Chrome's settings; and
- Add the plugin SEOquake[7] to Google Chrome to transfer results to Microsoft Excel for analyses.

3.1.2. Literature selection procedure

After retrieving articles, the inclusion and exclusion criteria were applied. The first and sixth authors simultaneously conducted the screening process using the criteria mentioned below. If any issues arise regarding selecting an article, a meeting was set up in which both authors discussed the issue and tried to resolve it based on mutual agreement. One set of criteria was used for the scientific articles, and another was used for the gray articles. Fig. 2 shows the steps used for the two stages of study selection. For scientific articles, their titles, abstracts and keywords were analyzed based on following criteria:

- Include the article if it contains the keywords in Table 2;
- Include the article if it is easy to obtain and it is in English; and
- Exclude the article if it resembles a letter or an editorial.

The number of articles found using the scientific databases was 9,671. After removing duplicates with the help of the StArt tool, 4704 articles were left for consideration using the inclusion and exclusion criteria. The StArt tool helped in giving the information related to articles titles, abstracts, and keywords. The sixth author went through each article manually; the title was assessed first, and if it was relevant to the topic, then the abstract and keywords were checked. If the article title, abstract, and keywords reflected the population and intervention, it was included. Other criteria were also considered for including the articles. After applying the above criteria, 32 articles were selected for the full-text reading. Thirteen were rejected at this point because they were not within the scope of our study. Some examples of the rejected articles are Fagerholm et al. (2017), Large (2005) and Marion and Simpson (2009). After full-text reading, 19 scientific articles were selected.

The articles retrieved from Google had the following inclusion and exclusion criteria:

- Include the article if its URL works and it is in English;
- Include the article if it contains the keywords in Table 2;
- Include the article if it is a webpage with text. If the webpage is only videos, audio, or images without text, it should be excluded; and
- Exclude the article if it is from Quora, Slideshare, or LinkedIn. (This was done because, on Quora and LinkedIn, general discussion happens on the subject with lack of detail reflection. Similarly, Slideshare provides presentation material on the given topic and lacks detailed explanation.)

2 http://ieeexplore.ieee.org/Xplore/home.jsp
3 https://www.scopus.com/home.uri
4 https://apps.webofknowledge.com
5 http://www.google.com
6 http://lapes.dc.ufscar.br/resources-and-downloads/tools
7 https://www.seoquake.com/index.html

Table 3
Questions about requirement elicitation from the case survey.

Category	ID	Questions
Requirement sources	31	What are the sources for ideas (requirements) for product/service features? Options: Internal sources (e.g., invention, brainstorming), analysis of similar products, standards/laws/regulations, business goals, market trends, potential and existing customers, none of the above, I do not know, other (specify)
	32	How were these sources identified and selected?
Elicitation methods	33	What requirements elicitation (extraction) methods are used in the startup? Options: customer interviews, customer surveys- feedback forms in the product/service, observation, analysis of similar products, prototyping/mockups, dedicated brainstorming sessions, onsite customer, none of the above, I do not know, other (specify)
	34	How are the requirements elicitation methods you selected above used?

The number of articles retrieved with Google was 18,972. From those articles, 2106 were duplicates, which were identified through a Microsoft Excel feature by the first author. After removing the duplicates, 16,864 articles remained in the Excel sheet to be analyzed using the inclusion and exclusion criteria. If the title of the article contained keywords for the population and intervention, then the criteria listed above were applied. After the criteria were applied and the full texts were read, 17 articles (16 internet articles and 1 scientific article) remained that were relevant to this study. Then, the scientific database (19) and Google (17) articles were combined, for a total of 36 primary articles. The data extraction and analysis processes for these primary articles are discussed in Sections 3.3 and 3.4.

3.2. Case survey (phase two)

Empirical data (containing qualitative and quantitative data) were collected using a case survey (Larsson, 1993), a technique that helps researchers combine the benefits of case studies and surveys. Through case surveys, it is typically easy for researchers to explore qualitative data to conduct deep analyses of cases and analyze quantitative data to conduct statistical analyses. As software startups belong to different domains, and product development involves either pure software or embedded-based software products, the case survey approach was beneficial for this study. It allowed the researchers to improve the generalizability of the research and analyze software startups in different domains.

3.2.1. Case selection and results

To find the cases for this study, a questionnaire was created that aimed at collecting information regarding the use of software engineering methods and practices in software startups (Klotins, 2017). The questionnaire was inspired by the co-authors' earlier work (Klotins et al., 2018) on software engineering in startup companies, which was based on a multi-vocal exploratory study of 88 startup experience reports, and thus served as a foundation for our case survey. The questionnaire contained 10 sections that addressed different software engineering aspects of software startups and retrieved meta-information about the cases. One section focused on RE; a full list of the questions can be found elsewhere,[8] and sample questions are given in Table 3.

A workshop was also conducted to validate and refine the questions to ensure they were simple for the respondents to complete. In the workshop, the authors of this paper and other researchers from the startup research community[9] participated and provided feedback. Then, to ensure that the respondents understood the questions, a pilot study was conducted with four software startups to collect their views regarding the questions (through an online meeting).

The questionnaire was officially given to the respondents from December 1st, 2016 to April 15th, 2017. Convenience sampling was employed, in which four authors approached software startups (considering the aspects of software startups, such as a short operating history and innovative product development, as mentioned in the definition in Section 2.1) found through personal contacts. A total of 81 respondents completed the questionnaire. Information about them was extracted into a Microsoft Excel sheet for analyses. After examining the data in the sheet, it was found that one respondent did not answer all the questions. Hence, this respondent's information was removed from the sheet. Therefore, 80 cases were analyzed.

3.3. Data extraction classification (phase three)

A data extraction form was created that involved mining information from the primary literature articles and cases about RE in software startups. As can be seen from Fig. 1, qualitative data were extracted from the primary articles and case survey. Data from the questions in the case survey, including questions 5–7, 14, 24–25 and 31–43 (see footnote 8), were exported into a Microsoft Excel sheet[10] and split into two sub-sheets. The first sheet contained qualitative data (e.g., answers to questions 32 and 34 in Table 3), while the second sheet contained quantitative data (e.g., answers to questions 31 and 33 in Table 3). The primary articles (in PDF format) and data in the first sheet were imported into NVivo, and the data in the second sheet were imported to into SPSS. NVivo was used because it assists in analyzing qualitative data, enabling the creation of codes and themes from the extracted text. Similarly, SPSS was selected because it helps in interpreting quantitative data and contains several features for performing different statistical analyses. More information on the data extraction from the primary articles and case survey is provided below.

3.3.1. Primary article properties

Some of the properties that were extracted from the primary articles were as follows

- Article identifications (an identification number was assigned to each article);
- Article types (each article was classified based on its type, which was scientific, book, white paper, or internet article);
- Publication year (each primary article's publication date);
- Research classifications (each article was classified based on its research type using a classification system by Wieringa et al. (2006) that included evaluation [new knowledge was contributed using a suitable research method],

[8] RE questions: https://tinyurl.com/ya23u2g8

[9] Software startup research network. https://softwarestartups.org/about/

[10] Case survey data: https://tinyurl.com/ya56tafo

solution proposal [new solutions were provided without proper validation], philosophy [new perspective was provided about a certain area], opinion [authors reflected their opinions on phenomena], experience [authors reflected their experience based on observations of phenomena], and validation [solutions validated research methods]); and
- Pertinence (how much each article examined RE in software startups was assessed, and articles was classified as full [i.e., focusing on RE in software startups fully], partial [i.e., discussing certain aspects of the topic], or marginal [i.e., exhibiting a slight or narrow emphasis on the topic]).

3.3.2. Case properties

The properties that were extracted from each case were as follows:

- The case's identification number (an identification number was assigned to each case);
- The respondent's relationship with his or her startup (i.e., the type of relationship the respondent had with the case);
- The case's location (i.e., the location of the case's main office);
- The case's status (information on the situation of the startup was extracted based on whether the startup was operational [the team was working on products and/or services], paused [the team stopped working but there was an intention to resume], acquired by another company, or closed down [the team disbanded or was working on something else]);
- The case's size (the number of people working in the startup as a team);
- The respondent's experience (whether the respondent was a founder of the startup, was hired by the startup, was performing services for the startup but was not part of its core team, had invested in the startup, had studied the startup as part of a research project, or just happened to know the startup);
- The startup's software development team size (i.e., the number people in the case that worked primarily on software development); and
- The startup's software application products (i.e., the software applications that were used as part of the case's products and/or services, which were extracted in terms of data dominant software [e.g., websites and mobile apps with high degrees of data processing and storage], systems software [e.g., operating systems, networks, communications, drivers, and middleware], control-dominant software [e.g., embedded, hardware control, real-time, and process-control software], and computation-dominant software [e.g., operations research, artistic and scientific software]).

3.3.3. Requirements engineering in software startups

To address this study's research questions, 13 features were extracted, as follows:

- Requirement sources (from the primary articles and the case survey): Information related to requirement sources was extracted from the primary literature articles, and the requirement sources used in the cases were extracted in terms of internal sources; business goals; analyses of similar products; standards, laws, and regulations; market trends; and potential and existing customers;
- Source identifications (from the case survey): Evidence related to the source identification techniques used was extracted from the cases and coded;
- Elicitation techniques (from the primary articles and case survey): Techniques used for elicitation were extracted from the primary articles, and techniques used in the cases were extracted in terms of customer interviews, observations, onsite customer feedback, customer surveys, feedback forms for products and services, prototyping mockups, analyses of similar products, and dedicated brainstorming sessions;
- Elicitation techniques (from the case survey): The elicitation techniques that were used were extracted from the cases;
- Requirement documents (from the primary articles and case survey): Documents that mentioned the articles were explored, and the types of documents that were applied in the cases were extracted. Document types were extracted in terms of requirements that were not documented in any way; requirements that were documented with informal notes; requirements that were discussed and drawn on whiteboards or posters, which were then kept up to date; requirements that were documented in organized ways and kept updated; templates that were used to document requirements; requirements that were continuously updated; and RE that followed organized processes with formal specifications;
- Requirement document levels (from the case survey): The levels at which the documents were applied were extracted in terms of product levels (goals), function levels (functions/actions), feature levels (features), and component levels (details and solutions);
- Documentation steps (from the case survey): The steps used in the documentation processes of the cases were examined;
- Requirement selection criteria (from the case survey): The criteria used in the cases were analyzed regarding requirement prioritization in terms of implementation costs, time and effort, and value to customers, products, companies, and shareholders;
- Selection criteria (from the case survey): The procedures employed to select the criteria used in the cases were explored;
- Value definitions (from the case survey): If values were considered by cases when they selected requirements, the types of values were examined;
- Requirement management (from the primary articles and case survey): The requirement management processes used in the cases and primary articles were explored;
- Validation methods (from the primary articles and case survey): The methods used to validate the requirements of the sources and the cases were evaluated in terms of prototype demonstrations to customers, customer surveys, A/B tests, customer interviews internal reviews, and prototypes; and
- Validation implementations (from the case survey): The steps that cases used to implement validation methods were examined.

3.4. Data analyses and interpretations (phase four)

This section discusses how the extracted data were analyzed and interpreted.

3.4.1. Qualitative data analysis

To analyze the qualitative data, data extractions and syntheses were done using NVivo 11, and an integrated approach (deductive and inductive methods) was taken (Cruzes and Dyba, 2011). The approach allowed codes to be organized in the rational forms of start lists and precodes; moreover, allowed the codes to be employed using a ground-up method from which concepts emerged.

With a deductive approach, start lists and precodes were created to find important variables. The objective was incorporating concepts that were already discussed in studies. Precodes were created to extract information about various categories from the primary articles (specifically, requirement sources, elicitation techniques, requirement documents, requirement management, and validation methods). For example, consider text from the primary article [P18]: "The elicitation phase is clearly different depending on whether it is a client- or user-targeted startup. In the first case, the team just has to ask the client what he wants to be made. However, the latter is different: There is no such person as the client to ask for requirements." This quotation comes from a discussion of requirement sources; hence, it was labeled in the precode as "requirement sources." Similarly, the precodes' internal sources; business goals; analyses of similar products; standards, laws and regulations; market trends and potential and existing customers were created under requirement sources to extract information collected on the features from the case survey.

An inductive approach was also applied in which text was reviewed line by line so that any of the elements that appeared were assigned to the code. Furthermore, the codes were constantly compared to refine the existing codes and generate new ones. Specifically, the inductive approach was used for elements like source identification, elicitation techniques, value definitions, and requirement management to constantly compare the extracted text from cases and allow themes to emerge. For example, the data collected for Q31 in Table 3 were analyzed line by line so that any concepts or themes that existed in the data could be recognized in terms of how the requirement sources were identified in the startup cases. The results of this inductive approach are described in Section 4.3.3.

3.4.2. Quantitative data analysis

The primary articles' properties, in terms of type, publication year, research classification, contribution, and pertinence, and the cases' properties, such as respondents' relationships with startups and case locations, were statistically analyzed in relation to frequencies. Descriptive statistics were used to determine the frequency of each variable and recognize any patterns. Various features, such as requirement sources, elicitation techniques, requirement documents, requirement document levels, requirement selection criteria, and validation methods, were statistically analyzed according to frequency (Blaikie, 2003). In addition, to determine whether there were any relationships between variables, such as requirement sources and elicitation techniques, bivariate correlation was used. If there were several responses (e.g., to questions about software application products and requirement sources), multiple-response cross-tabulations and frequencies were used to analyze the variables (Blaikie, 2003). The results from the analyses of the qualitative and quantitative data are discussed in the next section.

4. Results

As mentioned above, a total of 36 articles and 80 cases were gathered to answer this study's research questions. Descriptions of the articles and cases are given in Tables 4 and 5. This section presents an overview of the primary articles and cases. To address the research questions, the different aspects of RE in software startups are discussed.

4.1. Overview of the primary articles

In this section, the types of articles that were found and their publication years, types of research contributions, and relevance to RE in software startups are provided. More information on the articles is given in Table 4, which contains various details, such as the articles' identification numbers, publication years, types (scientific or internet), research classifications, and relevance.

4.1.1. Types of articles

During this study's analyses, it was found that 20 (56%) of the articles were scientific and 16 (44%) were gray. Therefore, as can be seen in Fig. 3, the number of scientific articles was larger than the number of gray articles. This shows that researchers focus more on startups than practitioners do.

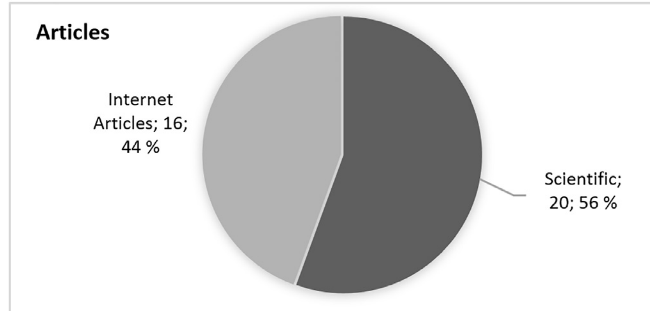

Fig. 3. Article proportions.

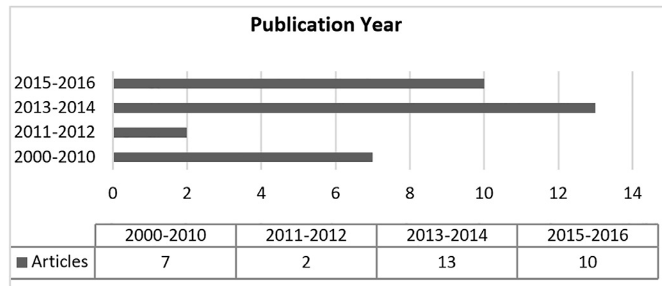

Fig. 4. Articles' publication years.

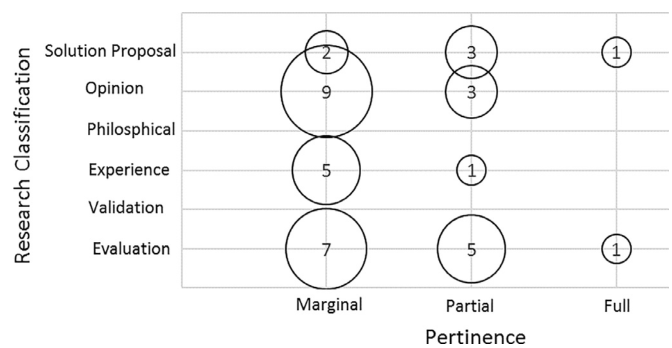

Fig. 5. Articles' pertinence and research classification.

4.1.2. Publication years

Concerning publication years, it was found that the number of articles about RE in software startups is continuously increasing. As Fig. 4 shows, seven articles from 2000–2010 were found that related to this study. From 2011–2014, there was an increase in the number of articles (eight), and the highest number of articles was from the period 2013–2014 (13). However, four internet articles were found that did not give publication dates, and the study's selection process ended in October 2016. Therefore, the total number of articles for 2015–2016 was 10, but the correct number of published articles for this period is open to interpretation. Nevertheless, it is clear that there was an increase in interest regarding software startups and RE.

4.1.3. Research classifications and pertinence

From the data in Fig. 5, one can see that 23 articles belonged to the marginal category, 11 belonged to the partial category, and 2 were fully relevant. This signifies that there were few articles that focused fully on RE in software startups. Similarly, in most of the articles, the research classifications (adopted from another study Wieringa et al., 2006) were opinion (12 articles) and evaluation (13 articles). The reason for the large number of opinion and evalua-

Table 4
Details of the primary articles.

ID	Year	Article type	Classification	Pertinence	ID	Year	Article type	Classification	Pertinence
[P1]	2013	Scientific	Evaluation	Marginal	[G2]	2009	Internet	Opinion	Marginal
[G3]	2014	Internet	Opinion	Marginal	[P4]	2013	Scientific	Evaluation	Partial
[P5]	2014	Scientific	Solution Proposal	Marginal	[P6]	2007	Scientific	Evaluation	Marginal
[G7]	2012	Internet	Opinion	Marginal	[P8]	2002	Scientific	Solution Proposal	Partial
[P9]	2015	Scientific	Evaluation	Partial	[P10]	2016	Scientific	Evaluation	Partial
[G11]	2015	Internet	Opinion	Partial	[P12]	2016	Scientific	Evaluation	Marginal
[P13]	2013	Scientific	Experience	Marginal	[G14]	2013	Internet	Opinion	Partial
[G15]	2014	Internet	Opinion	Marginal	[P16]	2012	Scientific	Experience	Marginal
[P17]	2014	Scientific	Evaluation	Marginal	[P18]	2016	Scientific	Evaluation	Full
[G19]	2015	Internet	Opinion	Marginal	[P20]	2014	Scientific	Evaluation	Marginal
[G21]	–	Internet	Opinion	Marginal	[P22]	2001	Scientific	Solution Proposal	Partial
[P23]	2013	Scientific	Solution Proposal	Full	[G24]	2010	Internet	Experience	Marginal
[P25]	2005	Scientific	Experience	Partial	[P26]	2015	Scientific	Evaluation	Partial
[G27]	2013	Internet	Opinion	Partial	[P28]	2016	Scientific	Evaluation	Marginal
[P29]	2015	Scientific	Solution Proposal	Marginal	[G30]	–	Internet	Experience	Marginal
[G31]	–	Internet	Opinion	Marginal	[P32]	2013	Scientific	Evaluation	Partial
[P33]	2015	Scientific	Evaluation	Marginal	[G34]	2011	Internet	Opinion	Marginal
[G35]	2009	Internet	Experience	Marginal	[G36]	–	Internet	Opinion	Marginal

Table 5
Details of the cases.

ID	Status	Size	Team	Software application	Region	ID	Status	Size	Team	Software application	Region
C1	Operational	4–8	1–3	Data, Control	S. America	C41	Operational	4–8	1–3	Data, System	Europe
C2	Operational	4–8	1–3	Data, Other	S. America	C42	Operational	9–12	4–8	Data, Other	Europe
C3	Acquired	4–8	4–8	Data, Other	Europe	C43	Operational	>20	4–8	System, Computation	Europe
C4	Operational	>20	>20	Systems, Computation	S. America	C44	Operational	9–12	4–8	Data, System	Europe
C5	Operational	16–20	9–12	Control	S. America	C45	Operational	1–3	1–3	Data, Other	Europe
C6	Operational	–	1–3	Data, Other	S. America	C46	Operational	1–3	1–3	Data, Other	Europe
C7	Operational	>20	4–8	Data, Other	Europe	C47	Operational	4–8	1–3	Data, System	Europe
C8	Closed down	4–8	1–3	Data, Other	Europe	C48	Operational	>20	9–12	Data, Other	Asia
C9	Acquired	13–15	9–12	Data, Other	Europe	C49	Operational	4–8	1–3	Data, Other	S. America
C10	Operational	>20	16–20	Data, Other	S. America	C50	Operational	1–3	1–3	Data, Computation	S. America
C11	Operational	>20	16–20	Data, Other	Europe	C51	Operational	4–8	1–3	Data, Computation	Europe
C12	Paused	1–3	1–3	–	Europe	C52	Operational	4–8	–	–	S. America
C13	Paused	4–8	1–3	Data, Other	S. America	C53	Operational	1–3	–	–	S. America
C14	Operational	>20	4–8	Data, System	S. America	C54	Operational	–	4–8	Data, Control	Asia
C15	Operational	>20	>20	Data, System, Other	S. America	C55	Closed down	1–3	1–3	Control	S. America
C16	Operational	4–8	4–8	Data, Other	Europe	C56	Operational	16–20	4–8	Computation	Asia
C17	Closed down	9–12	4–8	Data, Other	Asia	C57	Operational	1–3	1–3		S. America
C18	Operational	16–20	13–15	Data, Control, Other	Asia	C58	Paused	4–8	4–8	Data, Other	Asia
C19	Operational	1–3	–	–	Europe	C59	Closed down	4–8	4–8	System	Europe
C20	Operational	>20	13–15	–	Europe	C60	Operational	>20	>20	Data, Other	N. America
C21	Closed down	4–8	1–3	Data, Other	Europe	C61	Acquired	4–8	1–3	–	S. America
C22	Paused	1–3	1–3	Data, Control, Other	–	C62	Operational	4–8	4–8	Data, Computation	–
C23	Operational	4–8	1–3	Data, Other	Europe	C63	Closed down	4–8	1–3	–	S. America
C24	Closed down	4–8	1–3	Data, Computation	Europe	C64	–	4–8	1–3	Data, Other	Europe
C25	Closed down	1–3	1–3	Data, Other	Europe	C65	Closed down	4–8	4–8	Data, Other	Europe
C26	Paused	4–8	4–8	Data, Other	Europe	C66	Operational	4–8	1–3	Data, Other	S. America
C27	Operational	4–8	1–3	Computation	S. America	C67	Operational	4–8	1–3	Data, Other	S. America
C28	Operational	>20	>20	System, Computation	S. America	C68	Operational	1–3	1–3	Data, Other	S. America
C29	Operational	4–8	1–3	–	Europe	C69	Operational	1–3	1–3	Data, Other	S. America
C30	Operational	16–20	9–12	Data, System	S. America	C70	Operational	1–3	1–3	Data, Other	S. America
C31	Operational	4–8	4–8	Computation	S. America	C71	Operational	1–3	1–3	Data, Other	S. America
C32	Operational	>20	>20	Data, Other	N. America	C72	Operational	–	4–8	–	S. America
C33	Operational	1–3	1–3	Data, Other	S. America	C73	Operational	1–3	1–3	Systems, Computation	S. America
C34	Operational	9–12	4–8	Data, Control	Europe	C74	Operational	4–8	1–3	Data, Other	S. America
C35	Operational	9–12	4–8	Data, Other	Europe	C75	Paused	4–8	–	Data, Other	Asia
C36	Operational	4–8	1–3	Data, Other	Europe	C76	Operational	1–3	1–3	Computation	S. America
C37	Operational	4–8	1–3	Data, Computation	S. America	C77	Acquired	1–3	1–3	Systems	S. America
C38	Operational	4–8	4–8	Computation	Europe	C78	Acquired	9–12	9–12	–	Europe
C39	Operational	1–3	1–3	Data, Computation	Europe	C79	Operational	9–12	1–3	Data, Other	Europe
C40	Closed down	9–12	1–3	Data, Other	Asia	C80	Operational	4–8	1–3	Data, Other	Europe

tion articles is that such articles belonged to either articles on the internet or scientific articles in general.

4.2. Overview of cases

In this unit, we provide an overview of the respondents' background information, including their experience and relationship to the cases, and outline the locations of the cases. Table 5 presents additional data for each case, such as the identification number, present-day status, size (i.e., numbers of employees and members on the development team), and type of software application used in the product and region.

4.2.1. Respondents' backgrounds

It can be seen from the data in Fig. 6 that most of the respondents were the founders of a startup, 30 of whom had less than 6

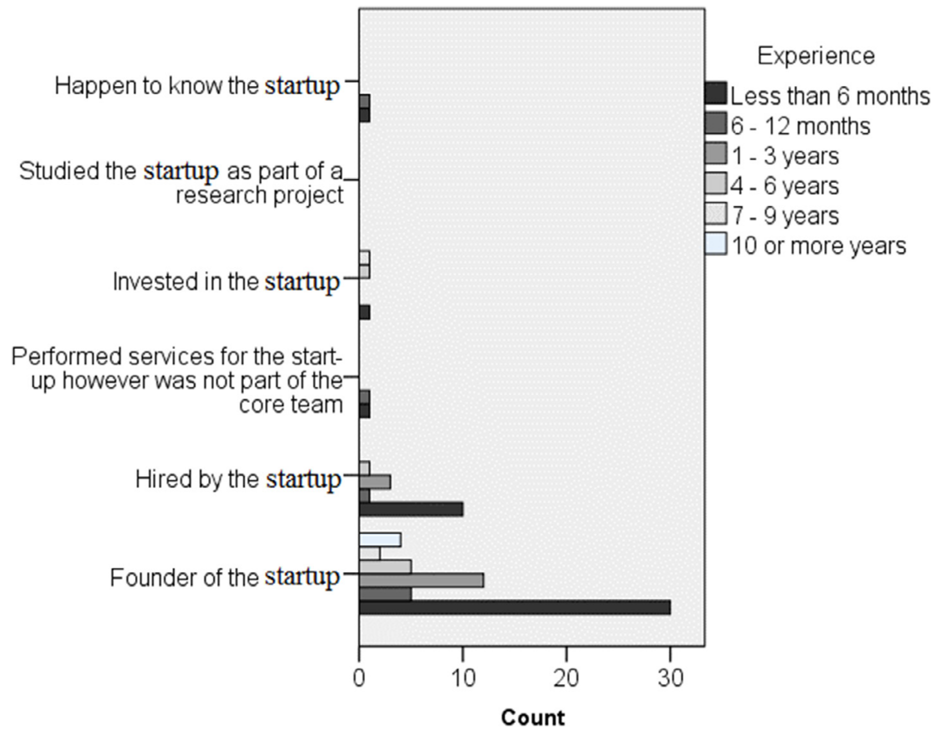

Fig. 6. Respondents background.

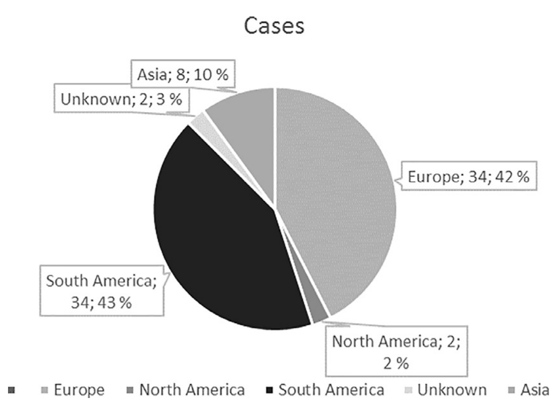

Fig. 7. Cases location.

months of experience with startups. The next largest group of respondents was hired by a startup, and 10 of these respondents had less than 6 months of experience. The rest of the respondents were in the range of 2–3. They performed services for a startup (but were not employed by it), invested in a startup, or simply knew about a startup. Therefore, most of the respondents had some level of knowledge (as founders) about software startup cases.

4.2.2. Case locations

The pie chart in Fig. 7 shows the locations of the cases. As illustrated, most of the cases were from Europe (34 cases, 42%) and South America (34 cases, 43%) because the researchers who conducted the case survey were primarily from these locations. Around eight of the cases were from Asia, while two were from North America. Thus, the cases came from various parts of the globe, largely South America and Europe, and the results obtained from these cases can be relevant for software startups.

4.3. Requirement sources and elicitation techniques

Software startups are faced with uncertainty about whether their product(s) can fit their target market and meet their customers' expectations. Large companies have a slight advantage in this regard based on their prior experience with product development and their intuition about which types of product can succeed. Therefore, startups need to continuously experiment, innovate, and iterate to create the best product/market fit [G21, G15]. Making continuous changes to a product can result in the loss of a market segment, so software startups need to build a product that reaches certain viability standards, fitting the product requirements (based on evidence) to address broad markets, as well as specific customer segments [G31]. However, interpreting product requirements can be a challenging process for software startups because they do not use traditional RE practices [P10, G11]. Instead, their RE is influenced by business actions and external factors [G11, P18].

Requirement elicitation, an important aspect of RE, involves collecting requirements from all potential sources using appropriate elicitation methods (Bourque and Fairley, 2014). Concerning software startups, Melegati et al. [P18] interviewed startup members and found that requirement elicitation techniques depend on the type of target (i.e., customer or user) in focus. In a single case study [P8], when the researchers observed the early stages of a startup, the product team or founder used elicitation techniques, and all the individuals in the startup contributed by providing the requirements. However, as that startup grew in size, the project manager reviewed and managed the elicitation process. Such a process has various advantages; for example, it enables stakeholders to collaborate and reveals product limitations [P26]. The use of game planning to analyze features was reported by [P6]; in contrast, if the users are unknown, persona and scenario techniques can help improve the elicitation processes [P10].

4.3.1. Requirement sources

In the case survey, the respondents were asked to indicate which sources they used to collect requirements (see Question 31

Table 6
The use of requirement sources and the frequency of use.

Sources	Responses		Percent of cases
	N	Percent	
Internal sources	71	19.8%	88.8%
Analysis of similar products	56	15.6%	70.0%
Standards/laws/regulations	22	6.1%	27.5%
Business goals	41	11.5%	51.2%
Market trend	36	10.1%	45.0%
Potential and existing customers	59	16.5%	73.8%
I do not know	2	0.6%	2.5%
Other	71	19.8%	88.8%
Total	358	100.0%	

in Table 3). Table 6 shows and compares the summary statistics for these requirement sources, which are discussed below (along with the MLR results). In the same table, the "Percentage" column represents the percentage of responses for each source, whereas the "Percentage of Cases" column represents the percentage of cases that used each source.

Internal sources. It is apparent that an internal source (88.8%, 71 cases), for example, invention or brainstorming, was the major source of requirements in the software startups. As startups are innovative in nature, they need to be creative when gathering requirements. Various activities, such as invention or brainstorming, seem to be a common approach for creative thinking (Maiden and Robertson, 2005). Early stage software startups have no customers, and finding customers is challenging for these startups [P26]; hence, the vision and requirements for a startup are outlined by its founder(s) and technical employees, such as developers [P6]. Similar cases were mentioned in [P9] and [P26], in which the founders, chief technical officer (CTO) and chief executive officer (CEO) provided minimum features through intuition and experience. In the case survey, a respondent from case C71 mentioned that internal sources were selected based on previous experience, whereas respondents from other cases, such as C3, C23, C58, and C72, stated that their teams conducted brainstorming sessions to collect requirements. The experience of internal members is crucial for requirement sources when users and customers are unknown [P28].

Potential and existing customers. The next most used source was potential and existing customers (74%, 59 cases). For software startups, a focus on customers is essential for business growth, so customer perception should be a priority [G15, G30]. For example, members of an internal startup in Microsoft, during their initial stage, created several product requirements, but later, when they talked to their customers, the product requirements changed completely [G30]. In the case survey, the teams in cases C2 and C10 met and spoke with customers about the requirements. Discussing requirements with customers (and asking them the right questions) can help make product features suitable for large market sections and reduce wasted time and resources during the initial feature-gathering stages [G31, G34]. Sometimes, customers have original ideas and provide their requirements (C29, C36). In addition, if a product contains features that hold perceived values for customers, then a pleased customer can endorse the product to others [P28].

Analyses of similar products. The next most used source observed in this study was the analysis of similar products (70%, 56 cases). Some startups aim to incorporate few features that are similar to successful rival products. Analyses of similar products can provide several possible solutions for problems that a startup product is trying to address, thereby making new products better than the existing ones [P25]. Another benefit was mentioned in (C74); while startup staff members were analyzing similar products (smartphones), they discovered a lack of similar products in Brazil, which revealed that the market for their current product was unexplored. Some startups want to avoid putting extra effort into understanding and solving problems. Therefore, through an analysis of similar products, some features of existing products can be adopted to reduce uncertainty and avoid the necessity of feature validation [P4].

Business goals, market trends, and standard laws and regulations. Business goals (51%, 41 cases) and market trends (45%, 36 cases) were the next most used sources in this study, whereas standard laws and regulations (27%, 22 cases) appeared to be the least used source. According to [P9], to produce a scalable B2B product, some startups consider business as needed to create requirements and solutions, and they do not focus on users' problems. In C33 and C46, a business plan and canvas were helpful requirement sources. Some studies ([P18] and [P26]) pointed out that markets are important factors for software startups, and a good understanding of a market can enable it to assess customer requirements and understand competitor products [P25]. In case C71, market trends were observed using online searches and observations. As for standard laws and regulations, one possible reason for their lack of sources used could be the strictness of these guidelines. For some cases, such as C17 and C46, startups were able to select relevant laws and regulations through their previous experiences with similar products. In addition, [P18] observed that startups focusing on financial and defense markets need to follow strict requirement processes, as this is an important customer requirement. Hence, collecting requirements from standard laws and regulations is compulsory if the domain of a target product requires compliance with these regulations.

4.3.2. Source selection based on software application domain

During the analyses, it was interesting to see whether there were any relationships between requirement sources and types of software applications used in products. Table 7 shows a cross-tabulation of multiple responses regarding the use of requirement sources in a specific software application. As shown in the table, the most used sources for various applications were as follows: for data-dominant software, internal sources (89%, 50 cases) and potential and existing customers (75%, 42 cases); for system software, internal sources (92%, 11 cases) and potential and existing customers (75%, 9 cases); and for control-dominant software, internal sources (92%, 11 cases) and potential and existing customers (69%, 13 cases). An interesting observation here is that products using data-dominant software, system software and control-dominant software frequently used internal (invention or brainstorming) and potential/existing customers as their sources. However, for computation-dominant software, analyses of similar products (88%, 14 cases) were frequently used sources, along with internal sources (100%, 16 cases) and customers (100%, 16 cases). The reason for the frequent selection of internal sources and customers as a source can be explained by the small sizes of software startups; if a startup has customers, considering customer requirements for a product is essential for the startup to establish itself in the market.

4.3.3. Factors leading to source identification and selection

In response to Question 32 (see Table 3), 80 respondent answers were extracted. During the analysis, four broad themes (as factors) emerged that resulted in source identification and selection, which are discussed below.

In-house decisions. In-house decisions often led to the identification and selection of sources. In 20 cases, requirement source identification and selection occurred during an in-house decision-making process. Decisions on source selection were made during brainstorming meetings with teams, team formation planning and the selection of organization practices (C2 and C23). Some respondents from cases C3, C23, C28, C33, and C56 pointed out that de-

Table 7
Types of sources and software applications developed.

			Sources						Total
			Internal sources	Analysis of similar products	Standards/laws/regulations	Business goals	Market trend	Potential and existing customers	
Software type	Data-dominant software	Count	50	39	14	31	26	42	55
		% of Total	74.6%	58.2%	20.9%	46.3%	38.8%	62.7%	82.1%
	Systems software	Count	11	7	1	7	5	9	12
		% of Total	16.4%	10.4%	1.5%	10.4%	7.5%	13.4%	17.9%
	Control-dominant software	Count	12	6	2	7	6	9	13
		% of Total	17.9%	9.0%	3.0%	10.4%	9.0%	13.4%	19.4%
	Computation-dominant software	Count	16	14	5	11	7	13	16
		% of Total	23.9%	20.9%	7.5%	16.4%	10.4%	19.4%	23.9%
Total		Count	61	46	18	37	31	52	67
		% of Total	91.0%	68.7%	26.9%	55.2%	46.3%	77.6%	100.0%

Table 8
Elicitation methods and their frequencies.

Elicitation methods	Responses		Percent of cases
	N	Percent	
Customer interviews	48	13.3%	60.0%
Customer surveys. feedback forms	35	9.7%	43.8%
Observation	49	13.6%	61.3%
Analysis of similar products	55	15.2%	68.8%
Prototyping. mock-ups	46	12.7%	57.5%
Dedicated brainstorming sessions	52	14.4%	65.0%
On-site customer	24	6.6%	30.0%
None of the above	1	0.3%	1.3%
I do not know	3	0.8%	3.8%
Other	48	13.3%	60.0%
Total	361	100.0%	

cisions on source selection occurred at internal brainstorming sessions. During an internal meeting, sources like potential and existing customers were selected to find flaws in the proposed requirements (C48) and improve customer loyalty (C21). Similarly, in case C38, analysis of a similar product was the chosen method for understanding other companies' products in a similar field and recognizing current trends.

Previous experience. Another theme identified from 19 cases was the startup members' previous experience with identifying and selecting sources for requirements. The respondents from cases C8, C16, C23, and C39 claimed that their previous experience in dealing with a similar customer segment helped them select suitable sources to extract the requirements for their products. Furthermore, in case C59, the founder tried to convert his undergraduate thesis into a product, so his source selection was his thesis. Similarly, cases C51 and C79 mentioned that previous job experience in a company with a similar product helped them identify suitable sources.

Background research. Apart from the above two themes, another theme that emerged from 18 cases was conducting research on the topic to familiarize oneself with the state of the art for a product idea (C1). Research and Internet searches were conducted to find literature on the user, market, and competitor (C12, C22, C41, C47, and C49). The aim was to develop an overview of reports related to the product idea to better examine the user, customer, competition, and market. Their research allowed them to choose the right sources for their requirements. In two cases (C29 and C30), it was pointed out that conducting market research on the hottest trends in the product domain enabled startup members to understand the status of their product idea in the current market and find suitable requirement sources. Similarly, in C72 and C74, the respondents conducted background research on similar products to find existing solutions for the problem(s) they were currently working on, and they considered the analysis of similar products as a source for their requirements.

Mentorship. Mentorship received during the early stages of a startup, whether from a mentor or personal acquaintance, provided direction in terms of choosing suitable sources for product requirements (C6). In cases C12, C47, and C77, experts, acquaintances, and other entrepreneurs provided relevant guidance, while in C1, C42, C66, and C75, sources were identified when startups consulted with companies developing products in a similar domain to obtain guidance.

4.3.4. Requirement elicitation techniques selected and applied

Once the sources are identified, it was important to elicit the requirements from those sources using a suitable method. In the context of software startups, Table 8 shows that analyses of similar products (68.8%, 55 cases) and dedicated brainstorming sessions (65%, 52 cases) are the most used elicitation methods. Next, customer interviews (60%, 48 cases), observation (61%, 49 cases) and prototyping (57%, 46 cases) are frequently applied methods of elicitation. Customer surveys (43%, 35 cases) and on-site customers (30%, 24 cases) appear to be the least used methods of requirement gathering. We also analyzed how these elicitation methods were applied, and the results are presented below.

Analyses of similar products/dedicated brainstorming sessions. Analyzing similar products was used as an elicitation method in 55 cases. In cases C12 and C80, this method was used to understand their competitors' products and the market, whereas for cases C23 and C65, it was used to find information about rival product features. In case C65, this technique was used to understand how certain features were developed. In terms of dedicated brainstorming sessions, this method was selected to experiment with different products and collect requirements from participants. In some cases, sessions were organized to obtain a solution for various challenges in relation to the product (C1 and C2), collect information concerning the product design and feature selection (C21), and obtain more information about the collected requirements so that better requirements could be selected (C54). Brainstorming sessions can be used when there is speculation regarding requirements to clarify what is needed and generate enhanced requirements [G11]. In C38 and C71, brainstorming sessions, which usually lasted an entire day, were held several times before any final decisions were made (C38). Similarly, in [P23], brainstorming sessions were held to identify the product goals and potential external stakeholders.

Customer interviews. Conducting customer interviews can help a startup to quickly grasp the accuracy of the product requirements (without wasting much time or resources on the guesswork) [G34]. Interviews are another good method for determining who the customers are and measuring their' perceived value [P17, G21]. Listening to customers during interview sessions can improve the validity of a problem and allow startups to gain knowledge regarding the product domain [G11]. In cases C12, C37, C73, and C77, customers were interviewed to obtain detailed requirements from the customer point of view, receive confirmation about whether the product features actually align with the targeted product, and identify any issues related to the requirements. In case (C67), customer interviews were conducted while visiting and contacting customers to collect their ideas. Customer interviews can be influenced by a founder's knowledge and background, analysis of existing products, and problems experienced by a founder [P4].

Customer surveys/feedback forms. In [P29], the authors stated that startups should conduct surveys carefully to collect information from a population sample that represents the target market. Startup personnel need to be familiar with research methods that can be used to collect and analyze responses insightfully. Because there can be differences in how requirements are written and spoken about, it is important to gather and validate requirements from various data points to evaluate the requirements in a quantifiable manner [G3]. In several cases, customer survey and feedback forms were selected to elicit feedback from customers about the product features (C13 and C28) and understand customer experience and satisfaction with the product (C23 and C40). A respondent from case C38 mentioned that feedback forms were used in conferences or events to gather feedback from users face to face. The objective was to obtain an overview of the product requirements.

Observations, prototypes, and mockups. Observation is a useful tool for understanding what people do with a product, in contrast to relying on what people say about the product features [G11]. Observations were conducted after receiving customer feedback in C12 and to create constraints for a new product in C31. In other cases, observations were conducted during conferences and events with respect to how users were actually using the product or service (C29), and observations were conducted on the use of rival products to generate requirements (C80). With regard to prototyping and mockups, these methods can help startups gather knowledge and reduce the risks associated with a product [P26]. In cases C2 and C5, prototypes were not only used to make sketches and generate ideas but also as a tool for minimizing inaccuracies. Prototypes are developed in house or with a small number of participants (C29). In case C65, mockups were developed using PowerPoint presentations to discuss a product's main features. According to the literature, such as [G24] and [G35], prototyping can help in the following scenarios: negotiating multiple ideas regarding a product, understanding user wishes, making a product technically feasible, and understanding which requirements have a significant effect from the customer perspective [G35].

On-site customers. As the least used elicitation method, on-site customers were observed in only 24 cases. This method is used when a startup is in an early stage and does not yet have customers. Thus, the CEO and management team can act as on-site customers and provide the requirements [P6]. In case (C36), the respondents mentioned the use of on-site customers as follows: "We have very close on-site or remote contact without customers at the moment and team is sitting in one office, so this is straightforward for us." In case C65, a business acted as an on-site customer. According to [P22], members from a marketing team can represent the customer for the development team and help them to make decisions about the requirements.

In our study, we noticed that some cases selected the above elicitation methods in a systematic way, whereas others used an ad hoc approach for selecting and using elicitation methods to capture their requirements. For example, in case C22, the approach for requirement elicitation was systematic, involving observation, custom interviews and surveys, analyses of similar products, brainstorming, and prototyping. Similarly, for case C31, their approaches were observation, analyses of similar products, brainstorming, and prototyping. In addition, case C48 used elicitation methods in the following sequence: customer feedback, brainstorming, analyses of similar products, and finally, prototyping. Case C57 began by selecting customer interviews and surveys and the analysis of similar products as elicitation methods, which they followed up with a brainstorming session, on-site customers, and prototyping. In contrast, cases C3 and C78 used an ad hoc approach most frequently, while cases C16 and C47 used a different combination of methods. For case C80, the method selection strategy changed over time.

4.3.5. Sources for requirements and use of elicitation methods

In our study, we were interested in determining whether a relationship existed between the requirement sources and the elicitation method used; we did find a correlation between these two factors, as shown in Table 9. When internal sources were selected as a requirement source, the statistically significant, two-tailed values ($\acute{i} = 0.05$) and Pearson correlation coefficients for elicitation methods were as follows: prototyping ($p = 0.023$, $r = 0.254$), dedicated brainstorming sessions ($p = 0.004$, $r = 0.319$), and observation ($p = 0.01$, $r = 0.285$). This indicates that these methods have a statistically significant linear relationship, but the magnitude of association is weak ($0.1 < r < 0.3$). A reason for this result could be that internal sources usually involve an internal team that works during an ideation session to find suitable requirements. Therefore, using analyses of similar products as an elicitation method, as well as using internal sources, will provide information on existing features of similar products. Likewise, brainstorming sessions will enable the extraction of requirements from internal sources.

When analyses of similar products were chosen as requirement sources, the statistically significant, two-tailed values and Pearson correlation coefficients for elicitation methods were as follows: analyses of similar products ($p = 0.00$, $r = 0.441$) and on-site customers ($p = 0.005$, $r = 0.310$). From this, we noticed a statistically significant linear relationship with a moderate association ($.3 < r <$

Table 9
The correlation between requirement sources and elicitation methods.

Customer interviews	Customer surveys, feedback forms	Observation	Analysis of similar products	Prototyping, mock-ups	Dedicated brainstorming sessions	On-site customer		
Internal sources	Pearson Correlation	0.113	−0.085	0.285*	0.101	0.254*	0.319**	0.147
	Sig. (2-tailed)	0.318	0.455	0.010	0.371	0.023	0.004	0.194
Analysis of similar products	Pearson Correlation	0.078	0.137	0.207	0.441**	0.155	0.092	0.310**
	Sig. (2-tailed)	0.492	0.224	0.065	0.000	0.171	0.420	0.005
Standards/laws/regulations	Pearson Correlation	0.160	0.021	0.088	0.234*	0.190	0.158	0.208
	Sig. (2-tailed)	0.156	0.852	0.440	0.037	0.092	0.160	0.064
Business goals	Pearson Correlation	0.071	0.104	0.251*	0.368**	0.274*	0.228*	0.093
	Sig. (2-tailed)	0.529	0.359	0.025	0.001	0.014	0.042	0.413
Market trend	Pearson Correlation	0.174	0.266*	0.255*	0.285*	0.269*	-.021	0.340**
	Sig. (2-tailed)	0.122	0.017	0.022	0.011	0.016	0.853	0.002
Potential and existing customers	Pearson Correlation	0.151	0.125	0.284*	0.149	0.177	0.158	0.267*
	Sig. (2-tailed)	0.182	0.268	0.011	0.186	0.117	0.162	0.017

* Correlation is significant at the 0.05 level (2-tailed). ** Correlation is significant at the 0.01 level (2-tailed).

0.5). Likewise, when standard laws and regulations were chosen as a source, a significant, two-tailed value was observed for analyses of similar products ($p = 0.037$, $r = 0.234$), showing a statistically significant linear relationship with a weak association. Since there are strict guidelines to be followed when standard laws and regulations are the requirement source, analyses of similar products will enable the startup to focus on strict guidelines to be included in the existing product.

When business goals were selected as a requirement source, analyses of similar products ($p = 0.001$, $r = 0.368$), observation ($p = 0.025$, $r = 0.251$), prototyping ($p = 0.014$, $r = 0.274$), and dedicated brainstorming sessions ($p = 0.042$, $r = 0.228$) indicated a weak to moderate relationship. With respect to market trends, analyses of similar products ($p = 0.011$, $r = 0.285$), observation ($p = 0.022$, $r = 0.255$), prototyping ($p = 0.016$, $r = 0.269$), and on-site customers ($p = 0.002$, $r = 0.340$) had the most statistically significant, two-tailed values with a weak to moderate association. In contrast, when customers were selected as a requirement source, there were statistically significant, two-tailed values for observation ($p = 0.011$, $r = 0.284$) and on-site customers ($p = 0.017$, $r = 0.267$), with a weak association.

4.4. Requirement specifications

According to [P18], startups do not have necessarily have a clear documentation process for collecting requirements, and they use tools like physical or electronic boards for documenting these requirements. However, startups with requirements involve following strict guidelines need an accurate documentation process. The authors in [P10] arrived at a similar conclusion; in this study, the startup followed informal specifications and used ticket-based tools, which contained user stories that described features in easy-to-understand language. Physical tools, such as Post-it notes and whiteboards, were also used to make product requirements visible to everyone. In [P8], the authors mentioned a startup that created a document, called a list, to collect the relevant features and time estimates given by the developer; the list was handled by the project manager, who was also responsible for generating and recording the features on the list. In [P4], the authors emphasized that, to compare product features, documentation in a proper format is essential. In situations where startups are working on multiple products, for example, documentation can help in handling these products. In contrast, [P22] highlighted that some startups do not have to create requirement documents because of a strict deadline to deliver the product to the market; therefore, their main focus is implementing the product.

4.4.1. Requirement documentation

In this case survey, we aimed to find out the documentation methods that were used during requirement collection; Table 10 displays the frequency of methods used in the cases. As evident in the table, the most commonly used method was documentation with informal notes (37%, 29 cases). Examples of this method include the creation of tickets in a ticket-tracking tool (C13) and the use of Google Drive for documentation (C29). The objective of using informal notes for documentation is improving learning concerning the requirements (C71), and it can be used on multiple levels of development (C12 and C66). The next most used approach was recording requirements on a whiteboard or poster after a discussion (22%, 17 cases): For example, in (C29), the startup used a whiteboard during their design meetings. Similarly, case C65 used a whiteboard to record high-level user stories, and in case C67, a surface similar to a Kanban board was used to document and manage features. Documentation that is organized and kept up to date is a method that was seen in 14 cases (18% of all cases). In this regard, case C15 used backlog tools, such as TFS, to organize

Table 10
The documentation methods and their frequencies.

Documentation methods	Frequency	Percent	Valid percent	Cumulative percent
No documentation	5	6.3	6.4	6.4
With informal notes	29	36.3	37.2	43.6
Discussed and drawn on whiteboard/ poster (kept up-to-date)	17	21.3	21.8	65.4
Documentation in an organized way and kept updated	14	17.5	17.9	83.3
Use of template (continuously updated and requirements engineering follows)	6	7.5	7.7	91.0
With a formal specification	2	2.5	2.6	93.6
I do not know	5	6.3	6.4	100.0
Total	78	97.5	100.0	
Missing System	2	2.5		
Total	80	100.0		

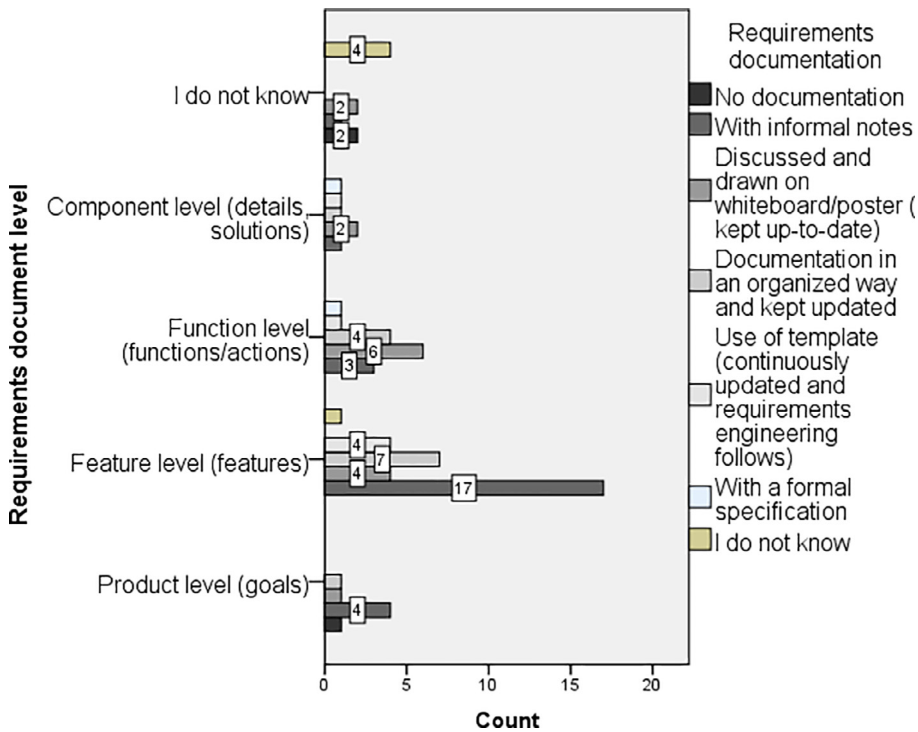

Fig. 8. Levels and types of documentation used for requirements.

the documentation, whereas case C34 used Microsoft Word documents, which were managed by version control and stored in a project management tool. Once development starts, requirements in the Word document are broken down into stories and entered into Jira. In case C36, Confluence and Jira were used for documentation, and in case C65, Trello was used, in which requirements are stored as cards (written in the form of user stories). The use of templates (8%, six cases) and formal specifications (3%, two cases) were the least often observed methods, although a template was used in case C5, and ISO-13485 steps were followed for project documentation. Respondents from five cases mentioned that they did not use any kind of documentation. For example, in case C45, documentation was not considered important, as the startup was small and held requirement validation as a priority.

4.4.2. Requirement documentation level

Another striking observation was the level at which the requirements were documented (shown in Fig. 8). In the figure, we can see that the majority of documentation occurred at the feature level (in 33 cases), followed by documentation at the function level (in 15 cases). Interestingly, documentation was carried out the least at the product (seven cases) and component (six cases) levels. Notably (see the Fig. 8), documentation with informal notes was done most frequently at the feature level (17 cases). Next, the most frequent use of the feature level occurred when the startup's documentation was organized and kept up to date (seven cases). However, the use of a whiteboard was mostly at the function level (six cases), as was the case for well-organized documentation (four cases). Regarding product and component levels, the product level mostly applied when documentation was done with informal notes (four cases). The component level applied when information was recorded on a whiteboard (two cases).

4.5. Requirement prioritization

Once documentation is created for the requirements, it is important to prioritize the requirements in terms of value. The objective is to make the features that add value to the product first. The author in [P5] mentioned that startups work in highly uncertain conditions, and due to the scarcity of resources in a startup, managing these changing requirements is not easy. Article [P29] urged the importance of feature prioritization, which should be done continuously, and stressed that prioritized features should be implemented as quickly as possible, before other features. Furthermore, [P8] argued that the startup project manager is responsible for managing the requirements based on scenarios (i.e., effort estimates and priorities). A game plan should be followed to manage the requirements and assign value to the features, as re-

Table 11
Criteria to decide which requirements to implement next.

Prioritization criteria	Responses		Percent of cases
	N	Percent	
Cost to implement	22	13.5%	29.3%
Time to implement	29	17.8%	38.7%
Effort to implement	22	13.5%	29.3%
Value to customer/product/company/shareholders	63	38.7%	84.0%
I do not know	5	3.1%	6.7%
Other	22	13.5%	29.3%
Total	163	100.0%	

ported in [P6]. In the startup case discussed in [P8], they created a backlog for the features (handled by the project manager), and the startup's priorities were set based on the business value. However, in [P18], the startup created a layered prioritization scheme in which the founders or senior managers determined the priorities of a long-term plan. The authors in [P18] also pointed out that the market has an effect on prioritization, and if a startup is customer driven, then the customers set the priorities for product features.

4.5.1. Criteria for deciding which requirements to implement next

In terms of the criteria that are used to decide which features should be implemented next (Table 11 shows the frequency of use for these criteria), it appears that the value to the customer/product/company/shareholders is foremost (84%, 63 cases). The next most important aspect is the time required to implement the feature (39%, 29 cases), while the effort (29%, 22 cases) and cost (29%, 22 cases) to implement are the least important criteria.

Values to customers, products, companies, and shareholders. Values to customers, products, companies and shareholders were the most-used criteria (they were used in 84% of the cases). Details about these values are as follows:

- Customer value: Customers' requirements were of utmost importance, and features that provided value to customers and markets were prioritized (C30). In [P32], customer value was defined as "a customer's perceived preference for and evaluation of those product attributes, attribute performances, and consequences arising from use that facilitate (or block) achieving the customer's goals and purposes in use situations." In C32, features that had significant effects on customers were considered. In other cases, the startups' objectives were to build to features that made product use easier for customers (C37) and increase numbers of customers (C9, C10). Article [P17] pointed out that, to improve customer value, understanding how one's company perceives customer value, interviewing important customers, and receiving feedback on product features are important. Moreover, it was pointed out that investors are interested in startups that develop products with customer value. The authors in [P9] and [P28] noted that various practices, such as customer development and lean startup methods, can add customer value. As stated in [P28], *"These practices emphasize that startups should concentrate on producing customer value and avoid wasteful activities, i.e., non-value adding activities"*;

- User value: Some startups prioritized user value. Their objective was to ensure that users were loyal to their products and not others (C7, C22). Another objective was to make applications easier to use and increase sales (C20, C21). For products in the game industry, users' preferences (C29) and encouraging users to engage with products more (C32) were priorities;

- Business value: This was considered in C3, C6, and C15, in which prioritization was done according to business goals. Startups' objectives were to bring value to the company to reduce their workloads, improve their operations, reduce their mistakes, and increase their profits (C59, C65). Article [P17] mentioned that 80% of business value can be generated by 20% of software modules. Likewise, [P9] noted that creating products that provide businesses with value by indulging users and stakeholders is a challenge;

- Product value: In [P32], product value was described as follows: "Product value is related to the product price and it changes depending on the competitive products. For C1 and C4, product value was an important criterion in terms of improving products' computational performance and evaluating products' performance in markets. Some startups aimed to have error-free products to launch to users (C40).

Implementation time. A total of 29 cases considered implementation time and when their teams were small and had limited abilities (C22). For some cases, time was considered during analyses of features' feasibility (C30), and specifically, the time it would take to implement their most valuable features (C72). Some studies, such as [P22], notes that, in their early stages, startups need to reduce the time it takes them to reach their markets; hence, they need to consider the time it will take to implement products.

Implementation effort and cost. A total of 22 cases (29.3%) considered implementation costs. In C2, a focus was placed on short-term costs to provide competitive features. However, C38 emphasized features that quickly result in profits (C38). In fact, in some cases, urgency and cost were considered more than other constraints were (C66). Similarly, in C9, implementation effort was considered in terms of technical dependency, and in (C22), which involved a startup that had a small team with limited abilities, effort was also considered. This was considered by startups so they could be realistic about feature implementation (C32).

4.5.2. Changing requirements management

One observation was how startups managed changing requirements. Several aspects were considered during such management; they are discussed below.

Increments. Incrementally working on projects was considered beneficial in terms of changing requirements management. For example, C2 worked in increments and avoided planning details in its initial stages. Doing so allowed it (and other startups) to use adaptable processes (C15, C47). Especially, using Scrum, with its practices like Sprint, helped many startups manage requirements (C35, C42). For example, Sprint was used retrospectively by C70 to improve users' feedback and manage product requirements. In addition, C17 used iterations to develop products due to contract requirements, according to which, customers could make changes two developments ahead.

Use-case updates. Some cases mentioned that they continuously updated documents to manage their requirements. For example, C3 updated use cases; C9 discussed the effects of changing requirements, then made changes as required. Similarly, C44 regularly reviewed documents and worked to keep them updated; this was also done by C57 and C69.

Regular meetings. A few startups held regular meetings to manage requirements and update documents. For example, C4 had regular meetings to review product backlogs. Similarly, C20's employees, developer, product manager, CTO, and CEO met in person on

Table 12
Validation methods and their frequencies.

Validation methods	Responses		Percent of cases
	N	Percent	
Prototype demonstrations to customers	35	18.5%	48.6%
Customer surveys	21	11.1%	29.2%
Customer interviews	31	16.4%	43.1%
Internal reviews/prototypes	42	22.2%	58.3%
A/B testing	19	10.1%	26.4%
I do not know	6	3.2%	8.3%
Other	35	18.5%	48.6%
Total	189	100.0%	

short notice to discuss required changes. Cases C32 and C37 held management meetings for updating documents.

Some startups did not have specific processes to manage requirement changes, but they were quite flexible in their approaches (C21 and C7). However, some were not; C22 had a fixed approach, as its requirements were mostly fixed. In addition, C77 did not typically need to adapt to requirement changes, as the organization's team was small.

4.6. Requirement validation

For startups, [P18] noted that requirement validation is an important RE process. Requirements can be tested regardless of whether customers' and users' needs are included. MVP validation can assist development teams in continuously learning about markets and help management teams in decision making regarding the business values of products [G3]. The authors in [P33] noted that validation is important in reducing uncertainties and problems. Moreover, validation can help with checking assumptions, limiting unnecessary resources used during product development, interacting with users and customers, and pivoting [G27].

4.6.1. Validation methods

The frequencies of the validation methods startups used for requirements are shown in Table 12. Internal reviews and prototypes were the most common (used in 58%, or 42, of the cases). The second most common was prototype demonstrations to customers (used in 49%, or 35, of the cases), and the third most common was customer interviews (used in 43%, or 31, of the cases). The fourth and fifth most common methods were customer surveys and A/B tests (used in 29% and 26%, or 21 and 19, of the cases, respectively).

Prototype demonstrations to customers. Prototype demonstrations to customers were used to validate product types (C2). For example, C25 installed applications in front of customers for demonstrations and asked customers' opinions about them. Some startups' main objective was to receive customer feedback (C23).

As [G24] mentioned, demonstrations helped startups in evaluating products' behaviors and technical risks. The same benefit was reported by [G35]; product feasibility can be evaluated with demonstrations to customers. In fact, this method is useful for B2B firms with limited numbers of customers. However, actual prototypes (not just screenshots) need to be shown to customers so that reliable customer feedback can be incorporated. In some startups' initial stages, prototypes' limited functionalities were demonstrated to customers (C38). As [G36] noted, mockup seven with only a few features can allow customers to provide feedback. Often, a startup's objective is to test which features are appealing and useful to potential users (C50). However, [G21] found a limitation with prototype demonstration, as it cannot be used at enterprise levels.

Internal reviews and prototypes. In [G24], it was pointed out that internal reviews and prototypes can help when startups have multiple requirements. Furthermore, it can be beneficial if prototypes are evaluated early and regularly, which can be used as tools for team learning [G7]. Case C10 made the same observation: The startup used prototypes to receive team feedback. Moreover, some startups, such as C2 and C4, mentioned that they validated prototypes internally, and this helped them determine risks and implementation costs. Others implemented internal reviews and prototypes before conducting A/B tests (C32). Similarly, as noted by [P6], development teams can give prototype demonstrations for management teams and work closely with such teams to gather instantaneous feedback. Case C45 quickly validated requirements without expending unnecessary effort on documents by using mockups.

Customer interviews and surveys. In [G34] and [G3], it was mentioned that customer interviews are crucial because they provide startups with necessary knowledge for business improvements, especially during the early stages. In addition, discussions can be used to test product functionality (C10). Startups' main objectives are to gather feedback (C12) and consider directions for further improvement (C22).

Concerning game development, game players are often interviewed during prototype tests (C29). Interviews help startups in understanding the behaviors of their customers and users (C71). The authors in [P17] noted that interviews are suitable methods for measuring customers' perceptions of value. However, with respect to customer surveys, [G3] stated that simply conducting interviews is not sufficient for understanding and solving problems. It is also important to quantify and measure surveys by collecting various data points. For example, C2 launched an online survey to check user engagement and to receive feedback using an online NPS tool; however, C34 only sent a survey to customers to receive product feedback.

A/B testing. The authors in [G11] pointed out that one benefit of A/B tests is determining which options are best when several selections are available to improve products. Accordingly, C2 used such tests to validate hypotheses with Google Analytics. Similarly, C4 and C13 used A/B tests for validating products before customers examined features and evaluating technical changes. A/B tests were also conducted by C32, but alongside on-site customer interviews. Case C50 analyzed potential users and products' aesthetics with A/B tests. Another benefit of A/B tests, as [P2] noted, is that development-team projects can be examined by small groups of users to evaluate whether products perform as expected.

5. Discussion

This section discusses the results, implications of the results for practitioners and academics, and validity of the study.

5.1. Answers to the research questions

As mentioned above, software startups have the potential to create innovative products. In software engineering, RE is important for success in software development. Therefore, it is important for startups' product development, but comprehensive studies on RE in software startups are limited. To address this gap, research questions (Table 1) were created, and an MLR and case survey were conducted. These approaches allowed the researchers to assess extant studies on RE in software startups to address this study's research questions, as discussed below.

5.1.1. Answer to RQ1 (requirement sources)

The first research question aimed to discover what sources are used by software startups to collect requirements for products or services. From the MLR, it was found that elicitation processes are important in terms of stakeholder collaborations, and they often depend on the types of targets (either customers or users) startups set for products (see [P18], [P8], and [P26]). The results of the case survey revealed that internal sources are the most used sources for gathering requirements (they were used in 89% of the cases). Apart from this, potential and existing customers (used in 74% of the cases) and analyses of similar products (used in 70% of the cases) are other frequently used sources of requirements.

Internal sources are often employed because, during the initial stages of development, customers and users of products are unknown; hence, startups' best choice is collecting requirements from internal sources by inventing or brainstorming. Furthermore, concerning sources identified and selected during decision processes, most sources emerge via in-house decisions or using staff members' experiences with similar product domains. Therefore, during internal decisions or when building on past experiences in similar product development domains, startups favor internal resources, potential and existing customers, and analyses of similar products.

5.1.2. Answer to RQ2 (elicitation methods)

One objective of the study (in the second research question) was identifying elicitation methods startups use to collect requirements from selected sources. The data in Table 8 show that analyzing similar products (done in 68.8% of the cases), conducting dedicated brainstorming session (done in 65% of the cases), and conducting customer interviews (done in 60% of the cases) are significant elicitation methods used to extract requirements. These methods are employed in different ways. For example, analyses of similar products are used to search for information about how certain features of rival products are developed, while dedicated brainstorming sessions are used to experiment with different products and elicit innovative ideas from participants. In addition, customer interviews are used to confirm which features meet customers' expectations. In some cases, the above elicitation methods were applied systematically, whereas in others, ad hoc approaches were employed to determine the requirements.

5.1.3. Answer to RQ3 (requirements documentation)

The third question was, "How are requirements documented in software startups?" The case survey results (Table 10) showed that many startups (37% of the cases) had internal documents in the form of informal notes, that is, tickets formed in ticket tracking and other tools, such as Google Drive. Another widely used technique (22% of the cases) for documentation is drawing on whiteboards and posters. Whiteboards are used to draw high-level user stories, write, and manage features. For example, some studies ([P18] and [P10]) stated that startups do not have clear documentation processes and use various tools, such as physical and electronic boards with sticky notes, to document requirements and make product requirements visible to all staff members. It should be noted that this study also found that many startups document requirements at the feature and function levels (33 and 15 of the cases did so, respectively).

5.1.4. Answer to RQ4 (requirements prioritization)

The fourth research question was about requirement prioritization and management in software startups. From the case survey, in terms of the criteria used to decide which requirements should be implemented in the future, it was found in most of the startups (84% of the cases) that values to customers, products, companies, and shareholders were given the highest priority regarding feature implementation (Table 11). Moreover, to manage requirements, most startups used incremental work processes. Some studies ([P8] and [P6]) stated that requirements are prioritized and managed using scenarios (effort and priority estimates) and planning games.

5.1.5. Answer to RQ5 (requirements validation)

The final research question was about requirement validation. By analyzing the empirical data (Table 12), it was found that internal reviews of prototypes are employed by most startups (58% of the cases) to validate product requirements. Such reviews are used before products are launched to customers to ensure there are no unknown risks or costs before implementing the products. Demonstrations to customers (used in 49% of the cases) and customer interviews (used in 43% of the cases) are the second- and third-most common methods used for validation. Product demonstrations confirm customers' and users' opinions and increase the product segments. Customer interviews validate products' functionality and collect customer feedback. In fact, in some articles ([P18] and [P33]), it was noted that requirement validation is an important process in terms of reducing uncertainties and unknown problems, improving interactions with users and customers, and building support [P27].

5.2. Implications for research and practice

To ensure that this study's results are useful for academics, the MLR was used; with its results, researchers and practitioners can consider both viewpoints that were not peer reviewed (gray literature) and results that were (some used empirical methods). Moreover, researchers can use the findings by considering the contributions and relevance of extant literature on the topic (Fig. 5), which indicates that rigorous empirical studies fully relevant to RE in software startups is limited, and more research on the topic is required.

In terms of practice, the results can be used in five key ways. First, gray literature was employed to discover practitioners' points of view and thoughts concerning requirement gathering. Practitioners interested in extant literature on the topic can see the information given in Table 4. Second, the data collected from the case survey were from different regions around the globe. Therefore, the results and their applicability to the software startup context are credible, and startups that are developing software-based products or services can use the results during their requirement gathering processes. Third, this study focused on five main aspects, as follows: requirement sources, elicitation methods, requirement documents, requirement prioritization and management, and requirement validation. Data on these factors can be used by software startup personnel to improve the requirements. Fourth, practitioners can use the cases in Table 5 to analyze characteristics related to environments. Finally, the requirement processes elaborated in the results section can provide the following five benefits:

- The results regarding requirement sources (Table 6) can help practitioners know which sources can be suitable in specific contexts;
- The results regarding elicitation methods (Table 8) can be utilized by practitioners to identify and apply certain elicitation methods to specific settings;

- The data regarding documentation methods (Table 10) can assist software startup personnel in selecting suitable documentation techniques;
- The findings regarding requirement prioritization and management (Table 11 and Section 4.5.2) can be used by practitioners based on specific situations; and
- The data regarding validation methods (Table 12) can be used by startup employees to validate requirements.

5.3. Key contributions to the literature

In Section 2.3, we highlighted that only limited studies, such as (Paternoster et al., 2014; Klotins et al., 2015), have conducted literature reviews and discussed some aspects of RE in startups. Paternoster et al. (2014) mentioned that RE in software startups, especially in terms of eliciting the description and managing the requirements, is challenging. They also pointed out that customers and users are unknown, and requirements are often market driven. Furthermore, requirement specification is challenging, and requirements often change rapidly. Similarly, Klotins et al. (2015) mentioned that requirement engineering in the research is still lacking, and therefore, knowledge on the topic is limited.

Comparing the two studies mentioned above with ours, we can state that our work contributes to the literature in the following ways: The results from our research support the view of Klotins et al. (2015) that research on RE is still in the early stage, since we found only two research articles that had full pertinence to the topic. We also observed that the practices used during the RE phases are often based on the context and are not clearly defined. Moreover, we found that, if customers and users are unknown, requirements are captured through brainstorming and analyses of similar products.

Studies like Melegati et al. (2016) and Gralha et al. (2018) discussed RE in software startups through empirical means. Melegati et al. (2016) highlighted that the product team and business model influence elicitation and ideas coming from all the personnel. In terms of documentation, there are no clear documentation processes, but the uses of physical or electronic boards and project management tools (in the large team) are common. Similarly, prioritization is affected by the founders, market, and product team. Validation depends on the market (the product is market driven), customer (customer driven), and experimentation (user driven). Similarly, Gralha et al. (2018) mentioned that startups do not follow proper RE. From the data, they observed that most of the recorded practices take place at a basic level, avoiding documentation and using earlier experience regarding the requirement validation. A product is developed in an iterative manner, and it is improved based on the customer feedback, employer's experience, and market need. However, these researchers also pointed out that, as a company matures, requirement practices also evolve, where customers are given more preference during elicitation and documentation.

Concerning the above studies, the findings from our study support the views mentioned in Melegati et al. (2016) and Gralha et al. (2018) that there are no strict RE phases. The elicitation process is influenced by the internal members of the startups (highlighted similar in Melegati et al. (2016)) in terms of collecting the requirements through internal sources and brainstorming sessions. Furthermore, in the documentation, a similar observation was made in our study, as most of the documentation was carried out at basic level via informal notes or the use of whiteboards or posters. Regarding the prioritization, most of the primary prioritization was affected in terms of bringing value to the customer, product, company, or shareholders. Finally, validation is performed through internal reviews, prototyping, or prototype demonstration to the customers.

5.4. Validity discussion

To discuss the validity of our study, we use validity criteria, including the external, conclusion, internal, and construct validity to evaluate the research, as described in Zhou et al. (2016) and Runeson and Höst (2009).

5.4.1. External validity

External validity involves how much results of a study are applicable to other contexts. The data used to address this study's research questions came from an MLR and case survey that focused on RE in software startups. To enhance the generalization scope, the MLR incorporated all relevant extant literature using keywords (based on our earlier study and other researchers' study on the topic) that reflected RE in software startups. One threat to the external validity can be the literature's restricted time interval; this was overcome by not using a time filter in our search, and thus, the literature retrieved was from 2000–2016. Another threat can be related to insufficient research evidence in the primary articles. This threat is valid in our study, since several primary articles (scientific and gray literature) had a limited focus on the studied topic. Moreover, most startups in the case survey belonged to various domains, but they largely from South America and Europe; thus, there is a limitation in terms of generalization of the results and their applicability in a software startup context.

5.4.2. Conclusion validity

Conclusion validity involves whether a proper procedure was followed to come to credible conclusions and if the same results would be obtained if the procedure were repeated. Concerning the literature review, a systematic process was used to identify scientific and gray articles that reflected the requirements in software startups. One threat could be related to bias in the study selection. This threat was overcome by two authors jointly working on the study search and selection procedure. Furthermore, to make the process concrete, tools were used to perform the literature review, with a process that has been proven to be effective in various studies. Another threat to conclusion validity could be the improper classification of the primary articles. This threat was reduced by using the guidelines in Wieringa et al. (2006) to evaluate the articles in terms of research classification. One more threat could be related to the primary study replication; this was overcome with the use of instruments like StArt and Microsoft Excel tools to automate the process so that duplicate articles would be disregarded. Similarly, concerning the case survey, a design based on experiences shared by startup practitioners was used, and questions regarding RE were based on the co-authors' previous study. In addition, a software startup research community reviewed the questions, and the survey was pilot tested by a few startups before it was made available to the public. One threat could be related to volunteer bias. To address this, the participants (researchers and practitioners) that joined the workshop/case survey were also interested in the case survey results for use in their research or startups. To meet their interest, the results of the case survey were made available to them for further use.

5.4.3. Internal validity

Internal validity involves determining whether there is a causal relationship between the two factors and whether the third factor has any influence on the investigated one. Concerning the MLR, one threat could be an incorrect search method. To overcome this threat, the recognized scientific databases were used in the search method and for accessing the gray literature Google's search engine was employed. Similarly, during execution of the keyword search strings, the two researchers worked simultaneously. The two authors checked each other's results during the search method. Another threat could be culture bias. However, this threat was overcome because the two authors who executed the MLR were from the same country. Another threat could be related to publication bias, and this was overcome in our study by focusing on finding both peer-reviewed and non-peer-reviewed (unpublished) literature on the subject. Similarly, the case survey used guidelines proposed in other studies. In its early development stage, the case survey was cross-checked by multiple researchers in a workshop. Therefore, it is possible for the other researchers to conduct MLRs and case surveys on topics similar to this study's topic.

5.4.4. Construct validity

Construct validity involves finding the suitable operational measures for the concepts under study, as well as determining whether operational measures represent the viewpoint of the researcher and the research aligns with the research questions. For the MLR, to address the RQs, articles with related keywords regarding RE and software startups were selected. One threat related to construct validity could be the imprecise description of the MLR setting. To overcome this threat, a review protocol was followed, using the guidelines from the existing literature, to search for the scientific and gray literature separately. In addition, tools assisted in the clear description of the MLR settings. Another threat could be the use of incorrect keywords. To address this, the keywords were adopted from the earlier studies on the phenomena to find suitable literature to answer the RQs. Similarly, for the case survey, the survey questions were designed in such a way that they could address the RQs. Furthermore, the questions were pilot tested with some software startups, and therefore, other respondents found them to be appropriate before they were launched officially.

6. Conclusion and future work

Software startups are innovative in nature and have the potential to create products that scale rapidly in the market. Since software startups are new and inexperienced organizations, the use of standardized practices during software development are inconsistent (as reported in the previous literature). In the software development process, RE has a key role to play for the success of the project. Therefore, it is important to understand RE in the software startup context to help these companies to create requirements for developing successful products. In our study, we explored this topic in depth by collecting literature through a multi-vocal literature review and empirical data through a case survey method. During our multi-vocal literature review, we found 36 articles that related to RE in the context of software startups. In our case survey, we found 80 cases across the globe developing products in different domains, and we investigated how they were performing RE in their context. By analyzing the literature and case survey data, we presented a detailed overview of the requirements process in software startups. Our study contributes to the literature in two ways. First, we conducted a comprehensive literature review on the requirement gathering process in software startups and case survey on software startups. Therefore, the results obtained can be applicable in most software startup contexts. The second contribution relates to RE in software startups, as we gave a detailed description on use of requirements sources, elicitation methods, documentation process, prioritization of the requirements, and requirement validation.

6.1. Future work

In our future work, we will consider the case of Oulu city (in Finland), in which we will attempt to evaluate the software startups to understand the requirement process in this context and how the elements in the startup ecosystem affect the requirement gathering process. Other work would involve finding the course that software startups had to follow from the product conception stage until they performed their RE process.

Acknowledgement

The authors would also like to sincerely thank the members of Software Startups Global Research Network (https://softwarestartups.org/) for their assistance in the case survey workshop. In addition, we would like to thank all the respondents who participated in the survey and provided the data. Finally, the authors are grateful to all the reviewers for their high-quality feedback, which helped to improve this research article.

Appendix A. Primary articles

[P1]	J. Kasurinen, J. Strandn, K. Smolander, What do game developers expect from development and design tools?, 17th Int. Conf. Eval. Assess. Softw. Eng. (2013) 3641. doi:10.1145/2460999.2461004.
[P4]	J. Bjrk, Jens and Ljungblad, Jens and Bosch, Lean Product Development in Early Stage Startups., in: IW-LCSP@ ICSOB, 2013: pp. 19–32. doi:10.13140/RG.2.1.3785.7368.
[P5]	V. Eloranta, Towards a Pattern Language for Software Start-Ups, 19th Eur. Conf. Pattern Lang. Programs. (2014) 111. doi:10.1145/2721956.2721965.
[P6]	P. Tingling, A. Saeed, Extreme programming in action: a longitudinal case study, in: Int. Conf. Human-Computer Interact., 2007: pp. 242251.
[P8]	J.A. Blotner, Agile Techniques to Avoid Firefighting at a Start-up, OOPSLA 2002 Pract. Reports. (2002) 1–ff. doi:10.1145/604251.604253.
[P9]	P.K. Chilana, A.J. Ko, J. Wobbrock, From User-Centered to Adoption-Centered Design, Proc. 33rd Annu. ACM Conf. Hum. Factors Comput. Syst. - CHI 15. (2015) 17491758. doi:10.1145/2702123.2702412.
[P10]	N. Paternoster, G. Carmine, Software Development in Startup Companies: The Greenfield Startup Model, IEEE Trans. Softw. Eng. 0 (2016) 233. doi:10.1109/TSE.2015.2509970.
[P12]	X. Wang, H. Edison, S.S. Bajwa, C. Giardino, P. Abrahamsson, Key Challenges in Software Startups Across Life Cycle Stages, in: XP 2016, Springer International Publishing, Cham, 2016: pp. 169182. doi:10.1007/978-3-319-33515-5_14.
[P13]	B. Cherry, Blended startups: Combining IT with social, mobile, instant communities, Proc. Annu. Hawaii Int. Conf. Syst. Sci. (2013) 42774285. doi:10.1109/HICSS.2013.118.
[P16]	B. May, Applying lean startup: An experience report - Lean & lean UX by a UX veteran: Lessons learned in creating & launching a complex consumer app, Proc. - 2012 Agil. Conf. Agil. 2012. (2012) 141147. doi:10.1109/Agile.2012.18.
[P17]	S. Marciuska, C. Gencel, P. Abrahamsson, Feature usage as a value indicator for decision making, Proc. Aust. Softw. Eng. Conf. ASWEC. (2014) 124131. doi:10.1109/ASWEC.2014.16.
[P18]	J. Melegati, A. Goldman, Requirements Engineering in Software Startups: a Grounded Theory Approach, in: Proc. 22nd ICE/IEEE Int. Technol. Manag. Conf., IEEE, 2016: pp. 448454.
[P20]	C. Giardino, X. Wang, P. Abrahamsson, Why early-stage software startups fail: a behavioral framework, in: Softw. Business. Towar. Contin. Value Deliv., Springer, 2014: pp. 2741.
[P22]	J. Zettel, F. Maurer, J. Mnch, L. Wong, LIPE: a lightweight process for e-business startup companies based on extreme programming, Prod. Focus. Softw. Process Improv. (2001) 255270. doi:10.1007/3-540-44813-6_23.
[P23]	B. Callele, David and Boyer, Aubrie and Brown, Kent and Wnuk, Krzysztof and Penzenstadler, Requirements Engineering as a Surrogate for Business Case Analysis in a Mobile Applications Startup Context., in: IW-LCSP@ ICSOB, 2013: pp. 33–46. doi:10.13140/RG.2.1.3785.7368.
[P25]	A.M. Davis, A.S. Zweig, The rise and fall of a Software Startup, J. Inf. Technol. Theory Case Appl. Res. 2 (2005) 3148. doi:10.1126/science.1059746.
[P26]	C. Giardino, S.S. Bajwa, X. Wang, P. Abrahamsson, Key challenges in early-stage software startups, in: Int. Conf. Agil. Softw. Dev., 2015: pp. 5263.
[P28]	K. Hokkanen, Laura and Kuusinen, Kati and Vnnen, Minimum viable user experience: A framework for supporting product design in startups, in: Int. Conf. Agil. Softw. Dev., 2016: pp. 66–78. doi:10.1007/978-3-642-13054-0.

(*continued on next page*)

[P29]	L. Hokkanen, M. Leppnen, Three patterns for user involvement in startups, Proc. 20th Eur. Conf. Pattern Lang. Programs - Eur. 15. (2015) 18. doi:10.1145/2855321.2855373.
[P32]	S. Marciuska, C. Gencel, P. Abrahamsson, Exploring how feature usage relates to customer perceived value: A case study in a startup company, Lect. Notes Bus. Inf. Process. 150 LNBIP (2013) 16677. doi:10.1007/978-3-642-39336-5.
[P33]	A. Bekhradi, B. Yannou, In vivo in situ experimentations projects by innovative cleantech start-ups in Paris To cite this version:, in: ASME 2015 Int. Des. Eng. Tech. Conf. Comput. Inf. Engineering Conf. -, 2015.
[G2]	S.G. Blank, The Leading Cause of Startup Death Part 1: The Product Development Diagram, Steveblank.Com. https://steveblank.com/2009/08/27/the-leading-cause-of-startup-death-the-product-development-diagram, 2009 (accessed November 1, 2016).
[G3]	Luis Montes, Ten Lessons I Learned From a Failed B2B Software Startup, http://www.sramanamitra.com/2014/09/08/ten-lessons-i-learned-from-a-failed-b2b-software-startup, 2009 (accessed November 1, 2016).
[G7]	T. Sharon, Lean Startup is Great UX Packaging, Smashing Mag. https://www.smashingmagazine.com/2012/10/lean-startup-is-great-ux-packaging, 2012 (accessed November 1, 2016).
[G11]	E. Kretschmann, Lean StartUp, (2015). http://blog.zuehlke.com/en/tag/requirements-engineering-en/%0A, 2015 (accessed November 1, 2016).
[G14]	What the Heck are Non Functional Requirements, {(2013)}. http://accelerateddevelopment.ca/blog/what-the-heck-are-non-functional-requirements/%0A, {2013} (accessed November 1, 2016).
[G15]	S. Culture, A. Default, 7 Startup Habits Worth Keeping No Matter How Big Your Business Grows. https://www.entrepreneur.com/article/239534%0A, {2014} (accessed November 1, 2016).
[G19]	J. Geshwiler, Four Tough Questions Startup CEOs Need to Ask to Scale. http://www.xconomy.com/boston/2015/05/19/four-tough-questions-startup-ceos-need-to-ask-to-scale, 2015 (accessed November 1, 2016).
[G21]	How to be a successful corporate startup. https://brianmoelich.com/2015/04/17/how-to-be-a-successful-corporate-startup/%0A, (accessed November 1, 2016).
[G24]	E. Ries, R. Mannak, Case Study: Rapid iteration with hardware. http://www.startuplessonslearned.com/2010/10/case-study-rapid-iteration-with.html, 2010 (accessed November 1, 2016).
[G27]	K. Punjabi, Validate or Die: Using Validation to Build the Right Product - MindTheProduct. http://www.mindtheproduct.com/2013/09/validate-or-die-using-validation-to-build-the-right-product, 2013 (accessed November 1, 2016).
[G30]	J. Siegel, A startup within Microsoft searches for (and finds) what customers want. https://news.microsoft.com/features/a-startup-within-microsoft-searches-for-and-finds-what-customers-want/%0A,(accessed November 1, 2016).
[G31]	How to Validate Product Market Fit for your Startup. http://www.thestarta.com/articles/scaling/validate-product-market-fit-startup/%0A,(accessed November 1, 2016).
[G34]	C. Hann, Finding Customers Ahead of a Startup Launch. https://www.entrepreneur.com/article/220018, 2011,(accessed November 1, 2016).
[G35]	E. Ries, Case Study: Using an LOI to get customer feedback on a minimum viable product. http://www.startuplessonslearned.com/2009/10/case-study-using-loi-to-get-customer.html%0A, {2009}, (accessed November 1, 2016).
[G36]	A. Maurya, Customer Development Checklist for My Web Startup Part 1 Customer Discovery: What should I build. https://leanstack.com/customer-development-checklist-for-my-web-startup-part-1, (accessed November 1, 2016).

References

Achimugu, P., Selamat, A., Ibrahim, R., Mahrin, M.N., 2014. A systematic literature review of software requirements prioritization research. Inf. Softw. Technol. 56 (6), 568–585.

Alves, V., Niu, N., Alves, C., Valena, G., 2010. Requirements engineering for software product lines: a systematic literature review. Inf. Softw. Technol. 52 (8), 806–820.

Aranda, J., Easterbrook, S., Wilson, G., 2007. Requirements in the wild: how small companies do it. In: Proceedings of the 15th IEEE International Requirements Engineering Conference, RE'07. IEEE, pp. 39–48.

Babar, M.I., Ghazali, M., Jawawi, D.N., 2014. Systematic reviews in requirements engineering: a systematic review. In: Proceedings of the 8th Malaysian Software Engineering Conference (MySEC). IEEE, pp. 43–48.

Bajwa, S.S., Wang, X., Duc, A.N., Abrahamsson, P., 2017. “Failures” to be celebrated: an analysis of major pivots of software startups. Empir. Softw. Eng. 22 (5), 2373–2408.

Blaikie, N., 2003. Analyzing Quantitative Data: From Description to Explanation. Sage.

Blank, S., 2013. The Four Steps to the Epiphany: Successful Strategies for Products that Win. BookBaby.

Bourque, P., Fairley, R.E., 2014. Guide to the Software Engineering Body of Knowledge (SWEBOK (R)): Version 3.0. IEEE Computer Society Press.

Coleman, G., O'Connor, R.V., 2008. An investigation into software development process formation in software start-ups. J. Enterp. Inf. Manag. 21 (6), 633–648.

Condori-Fernandez, N., Daneva, M., Sikkel, K., Wieringa, R., Dieste, O., Pastor, O., 2009. A systematic mapping study on empirical evaluation of software requirements specifications techniques. In: Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement. IEEE Computer Society, pp. 502–505.

Crowne, M., 2002. Why software product startups fail and what to do about it. evolution of software product development in startup companies. In: Proceedings of the IEEE International Engineering Management Conference IEMC'02. IEEE, pp. 338–343.

Cruzes, D.S., Dyba, T., 2011. Recommended steps for thematic synthesis in software engineering. In: Proceedings of the International Symposium on Empirical Software Engineering and Measurement (ESEM). IEEE, pp. 275–284.

Easterbrook, S., Singer, J., Storey, M.-A., Damian, D., 2008. Selecting empirical methods for software engineering research. In: Guide to Advanced Empirical Software Engineering, 285–311.

Fagerholm, F., Guinea, A.S., Mäenpää, H., Münch, J., 2017. The right model for continuous experimentation. J. Syst. Softw. 123, 292–305.

Garousi, V., Felderer, M., Mika, M.V., 2016. The need for multivocal literature reviews in software engineering: complementing systematic literature reviews with grey literature. In: Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering. ACM, p. 26.

Giardino, C., Paternoster, N., Unterkalmsteiner, M., Gorschek, T., Abrahamsson, 2016. Software development in startup companies: the greenfield startup model. IEEE Trans. Softw. Eng. 42 (6), 585–604.

Gralha, C., Damian, D., Wasserman, A., Goulão, M., Araújo, J., 2018. The evolution of requirements practices in software startups. In: Proceedings of the 40th International Conference on Software Engineering. ACM.

Holl, G., Grünbacher, P., Rabiser, R., 2012. A systematic review and an expert survey on capabilities supporting multi product lines. Inf. Softw. Technol. 54 (8), 828–852.

Kane, T. J., 2010. The Importance of Startups in Job Creation and Job Destruction. doi:10.2139/ssrn.1646934.

Klotins, E., 2017. Using the case survey method to explore engineering practices in software start-ups. In: Proceedings of the 1st International Workshop on Software Engineering for Startups. IEEE Press, pp. 24–26.

Klotins, E., Unterkalmsteiner, M., Gorschek, T., 2015. Software engineering knowledge areas in startup companies: a mapping study. In: Proceedings of the International Conference of Software Business. Springer, pp. 245–257.

Klotins, E., Unterkalmsteiner, M., Gorschek, T., 2018. Software engineering in start-up companies: an analysis of 88 experience reports. Empir. Softw. Eng. 1–35.

Large, D.W., 2005. Best marketing and sales practices for technology start-ups: a review and fresh evidence. In: Proceedings of the IEEE International Engineering Management Conference, 1. IEEE, pp. 339–343.

Larsson, R., 1993. Case survey methodology: quantitative analysis of patterns across case studies. Acad. Manag. J. 36 (6), 1515–1546.

Maiden, N., Robertson, S., 2005. Integrating creativity into requirements processes: experiences with an air traffic management system. In: Proceedings of the 13th IEEE International Conference on Requirements Engineering. IEEE, pp. 105–114.

Marion, T.J., Simpson, T.W., 2009. New product development practice application to an early-stage firm: the case of the paperpro® stackmaster™. Des. Studies 30 (5), 561–587.

Mater, J. L., Subramanian, B., 2000. Solving the Software Quality Management Problem in Internet Startups. Keynote Address October 17.

Melegati, J., Goldman, A., Paulo, S., 2016. Requirements engineering in software startups: a grounded theory approach. In: Proceedings of the 2nd International Startups Workshop on Software StartupsTrondheim, Norway.

Pacheco, C., Garcia, I., 2012. A systematic literature review of stakeholder identification methods in requirements elicitation. J. Syst. Softw. 85 (9), 2171–2181.

Paternoster, N., Giardino, C., Unterkalmsteiner, M., Gorschek, T., Abrahamsson, P., 2014. Software development in startup companies: a systematic mapping study. Inf. Softw. Technol. 56 (10), 1200–1218.

Quispe, A., Marques, M., Silvestre, L., Ochoa, S.F., Robbes, R., 2010. Requirements engineering practices in very small software enterprises: a diagnostic study. In: Proceedings of the XXIX International Conference of the Chilean Computer Science Society (SCCC). IEEE, pp. 81–87.

Rabiser, R., Grnbacher, P., Dhungana, D., 2010. Requirements for product derivation support: results from a systematic literature review and an expert survey. Inf. Softw. Technol. 52 (3), 324–346.

Runeson, P., Höst, M., 2009. Guidelines for conducting and reporting case study research in software engineering. Empir. Softw. Eng. 14 (2), 131.
Seppänen, P., Tripathi, N., Oivo, M., Liukkunen, K., 2017. How are product ideas validated? In: Proceedings of the International Conference of Software Business. Springer, pp. 3–17.
Tripathi, N., Annanper, E., Oivo, M., Liukkunen, K., 2016. Exploring processes in small software companies: a systematic review. In: Proceedings of the International Conference on Software Process Improvement and Capability Determination. Springer, pp. 150–165.
Tripathi, N., Seppänen, P., Oivo, M., Similä, J., Liukkunen, K., 2017. The effect of competitor interaction on startups product development. In: Proceedings of the 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA). IEEE, pp. 125–132.
Verner, J.M., Evanco, W.M., Cerpa, N., 2007. State of the practice: an exploratory analysis of schedule estimation and software project success prediction. Inf. Softw. Technol. 49 (2), 181–193.
Wang, X., Edison, H., Bajwa, S.S., Giardino, C., Abrahamsson, P., 2016. Key challenges in software startups across life cycle stages. In: Proceedings of the International Conference on Agile Software Development. Springer, pp. 169–182.
Wieringa, R., Maiden, N., Mead, N., Rolland, C., 2006. Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. Requir. Eng. 11 (1), 102–107.
Zhou, X., Jin, Y., Zhang, H., Li, S., Huang, X., 2016. A map of threats to validity of systematic literature reviews in software engineering. In: Proceedings of the 23rd Asia-Pacific Software Engineering Conference (APSEC). IEEE, pp. 153–160.
Zowghi, D., Coulin, C., 2005. Requirements elicitation: a survey of techniques, approaches, and tools. In: Engineering and Managing Software Requirements. Springer, pp. 19–46.

Nirnaya Tripathi is a doctoral student at M3S research unit, University of Oulu since 2014. He received his Masters degree in Information Processing Science at University of Oulu in 2012. His research interest includes entrepreneurship, startup ecosystem, software engineering in startup, large-scale lean and agile software development and global software development. He is currently doing research on startup ecosystem and requirements engineering in software startups.

Eriks Klotins is a Ph.D. student of Software Engineering at Blekinge Institute of Technology (BTH). The focus of his thesis is software engineering practices in startups. He has over nine years of experience in managing software development projects ranging from large government IT systems to small start-ups.

Rafael Prikladnicki is an associate professor at the Computer Science School and the director of the Science and Technology Park (Tecnopuc) at the Pontifical Catholic University of Rio Grande do Sul, Brazil, where he also leads the MuNDDoS research group. He is the chair of IEEE Software's advisory board.

Markku Oivo (Ph.D., eMBA) is full professor of software engineering at the University of Oulu (Finland) since 2002. He is head of the M3S research unit. During 2000–2002 he was Vice President and director at Solid Co. He held several positions at VTT 1986–2000. He had visiting positions at University of Maryland (1990-91), Schlumberger Ltd. (Paris 1994-95), Fraunhofer IESE (1999–2000), University of Bolzano (2014–2015) and Technical University of Madrid (2015). He worked at Kone Co. (1982-86). He has initiated and managed 100+ research projects with tens of millions of euros funding. He has over 100+ publications. His research interests include empirical software engineering, agile, lean, quality & process improvement, and startups.

Leandro Pompermaier is an assistant professor at the School of Technology at the Pontifical Catholic University of Rio Grande do Sul (PUCRS)- Brazil. In addition, he is the leader of the startup area of the PUCRS Science and Technology Park (TECNOPUC). He holds a master degree in computer science from the Federal University of Rio Grande do Sul and is currently a Ph.D. student at PUCRS. He is also an entrepreneur and angel-investor in some IT companies and software startups.

Arun Sojan is a Ph.D. student working at the M3S research unit, University of Oulu. His interest is in Automotive software security.

Michael Unterkalmsteiner is a senior lecturer in the Blekinge Institute of Technology's Software Engineering Research Laboratory. His research interests include data-driven software engineering, software measurement and testing, process improvement, and requirements engineering. He primarily conducts empirical research in close collaboration with industry partners. Michael received a Ph.D. in software engineering from the Blekinge Institute of Technology. Contact him at michael.unterkalmsteiner@bth.se; https://www.lmsteiner.com.

Kari Liukkunen (Ph.D., M.Ed.) has 20 years of management and R&D experience in industry and academy. He has worked at University of Oulu since 2001. Now he is working in a Director position in the HILLA program. HILLA is at the moment the largest research program in Finland with its 50 million euro funding. He is also the head of the Software Factory laboratory and has worked as a visiting researcher in Fraunhofer IESE, Germany. He has published over 60 papers in international conferences and journals and served as chair and committee member in organizing numerous international conferences.

Tony Gorschek is a Professor of Software Engineering at Blekinge Institute of Technology – where he works as a research scientist in close collaboration with industrial partners. Dr. Gorschek has over fifteen years industrial experience as a CTO, senior executive consultant and engineer. In addition he is a serial entrepreneur with five startups in fields ranging from logistics to internet based services and database register optimisation. At present he works as a research leader and in several research projects developing scalable, efficient and effective solutions in the areas of Requirements Engineering, Product Management, Value based product development, and Real Agile™ and Lean product development and evolution. https://www.gorschek.com, tony.gorschek@bth.se.